# Une ingénierie participative des exigences pour les systèmes interactifs complexes en aéronautique

Hélène Uninski


# THÈSE

En vue de l'obtention du

## Doctorat de l'Université de Toulouse

Délivré par *l'Université Toulouse III - Paul Sabatier*
Discipline ou spécialité : *Réseaux, Telecom, Système et Architecture*

---

Présentée et soutenue par *Hélène UNINSKI*

Le *20 décembre 2017*

Titre : *Une ingénierie participative des exigences pour les systèmes interactifs complexes en aéronautique*

---

JURY

Mackay Wendy E., INRIA Saclay, Rapportrice
Amyot Daniel, Université d'Ottawa, Rapporteur
Hascoet Mountaz, Université de Montpellier, Examinatrice
Palanque Philippe, Université de Toulouse III, Président
Lassis Erick, Direction Générale de l'Aviation Civile, Membre invité
Conversy Stéphane, ENAC, Directeur de thèse

---

École doctorale : MITT
*Mathématiques, Informatique et Télécommunications de Toulouse*

Unité de recherche : *UMR 5505 IRIT*
*Institut de Recherche en Informatique de Toulouse*

Directeur de Thèse : *Stéphane Conversy*

## Remerciements

Positionner en premier et rédiger en dernier, les remerciements constituent un exercice délicat : est-il possible de les adresser sans oublier quelqu'un qui aurait contribué aux quatre années de travail présentées dans cette thèse ? Vu mon âge avancé, suis-je obligée de remonter aux origines ? Une solution serait de remercier le maximum de monde, mais les remerciements perdraient alors leur sens véritable : mettre en évidence les personnes sans lesquelles je n'aurais pu mener la thèse jusqu'au bout. Par conséquent, je tiens à remercier plus particulièrement :

- Les ingénieurs en aéronautique ayant accepté de participer aux travaux de la thèse : leur disponibilité et leur capacité à prendre du recul sur leur propre activité ont été primordiaux. Leur anonymat, condition nécessaire à la richesse des échanges, reste préservé.
- Les personnes ayant contribué au développement des visualisations interactives des exigences : Pierre Chevalier, Gabriel Doucet et Thomas Lemee.
- Wendy MacKay : ses travaux en conception participative m'ont donné envie de faire de la recherche en IHM, sous l'impulsion de Stéphane Chatty, il y a vingt ans ;
- Daniel Amyot : ses connaissances en ingénierie des exigences m'ont aidée à progresser dans ce domaine et à affiner le positionnement de mes travaux ;
- Denis Louviot, pour la confiance qu'il m'a accordée sur le projet E-Fan, et qu'il a renouvelée sur le nouvel instrument de vol ;
- Erick Lassis, pour le temps sanctuarisé à la rédaction de la thèse ;
- Philippe Palanque, pour l'hébergement administratif et sa présidence dynamique et efficace du jury ;
- Jérémie Garcia, pour sa relecture accompagnée de judicieux conseils de rédaction ;
- Stéphane Conversy, pour sa direction de thèse stoïque face aux doutes continuels.

Ingénieure dans un monde de chercheurs depuis dix ans, je deviens chercheuse dans un monde d'ingénieurs. Le chassé-croisé peut surprendre, mais il a été nécessaire pour aboutir au présent manuscrit, que je dédie à mes enfants, Lucie et Antoine.

## Table des matières

## Table des figures et des tableaux

# Introduction

*'New Systems Mean New Problems'.*
Systemantics: How Systems Work and Especially How They Fail (Gall 1977)

> *The complexity of software is an essential property, not an accidental one. Hence descriptions of a software entity that abstract away its complexity often abstract away its essence. Mathematics and the physical sciences made great strides for three centuries by constructing simplified models of complex phenomena, deriving properties from the models, and verifying those properties experimentally. This worked because the complexities ignored in the models were not the essential properties of the phenomena. It does not work when the complexities are the essence. Many of the classical problems of developing software products derived from this essential complexity and its nonlinear increased with size. From the complexity comes the difficulty of communication among team members, which leads to product flaws, cost overruns, schedule delays. From the complexity comes the difficulty of enumerating, much less understanding, all the possible states of the program, and from that comes the unreliability. From the complexity of the functions comes the difficulty of invoking those functions, which makes programs hard to use. From complexity of structure comes the difficulty of extending programs to new functions without creating side effects. From the complexity of structure comes the unvisualized state that constitutes security trapdoors. Not only technical problems but management problems as well come from the complexity. This complexity makes overview hard, thus impeding conceptual integrity. It makes it hard to find and control all the loose ends. It creates the tremendous learning and understanding burden that makes personnel turnover a disaster.*
>
> No Silver Bullet: Essence and Accident in Software Engineering (Brooks 1987)

Dans notre imaginaire, l'ingénierie se réfère à une démarche rationnelle, permettant de résoudre un problème bien posé à l'aide d'outils mathématiques. Ainsi, l'ingénieur structure va calculer les dimensions d'une poutre en précisant les efforts dans une formule mathématique. Pour preuve, les futurs ingénieurs sont sélectionnés en classe préparatoire par leurs performances en mathématiques.

Pourtant, la signification étymologique d'ingénieur dépasse largement l'application de connaissances mathématiques pour résoudre un problème donné. Le mot « ingénieur » vient de ***Ingenium*** (*in-geno, **gigno***), qui se rapporte à l'engendrement et à la naissance. Ingenium désigne dans son sens premier les qualités innées d'une chose. En second lieu, il s'applique aux êtres humains et à leurs dispositions naturelles. Puis il exprime, parmi les dispositions naturelles de l'être humain, l'intelligence, l'habileté, l'inventivité, *l'esprit* selon Voltaire dans l'Encyclopédie (Diderot et D'Alembert 1751a). Enfin, par extension, il désigne les êtres humains qui sont particulièrement doués de cette faculté (Pons s. d.).

Ainsi, quand on se réfère à l'être humain, l'ingenium est celui qui crée et invente, avec la capacité de dépasser l'existant et de le transformer, qu'il s'agisse des idées, des arts, des innovations techniques, des organisations sociales. Par ailleurs, la poutre est un exemple simpliste et réducteur : depuis la

révolution industrielle, les systèmes conçus par les ingénieurs, des premières machines à vapeur à la station spatiale internationale, sont de plus en plus *complexes*.

L'étymologie du terme complexité renvoie au latin *complexus* (co, « ensemble », et plexus, « tissé »), ce qui est tissé ensemble (Benkirane 2006) : le principe de complexité est un principe qui consiste à relier les objets, pas seulement à les distinguer. Pour mieux comprendre ce principe de complexité, on peut le confronter à d'autres principes, afin d'en dégager les dimensions.

Le principe de simplification se fonde sur la séparation en différents domaines de connaissance, en isolant les objets de connaissance de leur contexte. Par cette disjonction, on réduit la connaissance d'un tout à la connaissance des objets disjoints constituant ce tout, et en dehors de leur environnement. En faisant cela, on perd la connaissance des qualités du tout qui ne se trouvent pas dans les objets disjoints (« p28: sur les principes de la pensée complexe (par E. Morin) », s. d.)).

Certains proposent de voir dans la complexité deux principes : un principe arithmétique, le nombre, et un principe topologique, le pli. Ainsi, plutôt de dire d'un tout qu'il est complexe, on dira qu'il y a un grand nombre d'objets et un grand nombre de figures. La complexité se rapproche alors de la combinatoire de Leibniz (« p371: sur la complexité: à la fois nombre et pli », s. d.)).

Dans le livre fondateur sur la théorie générale des systèmes, von Bertalanffy (1973) se réfère à la complexité en distinguant le nombre des éléments, les relations entre ces éléments, mais aussi la nature de ces éléments. La définition de système qu'il propose introduit une dimension supplémentaire, l'environnement :

*A system may be defined as a set of elements standing in interrelation among themselves and with an environment.*

On perçoit désormais l'objet à connaître comme un système *ouvert*, c'est-à-dire une partie insérée et active dans un plus grand tout, qui est l'environnement. Et la compréhension de cet environnement va participer à la connaissance de l'objet.

Le Moigne (Le Moigne 2006) introduit la dimension de la finalité : il ne s'agit plus d'analyser la réalité de l'objet, mais de concevoir le modèle de l'objet par rapport à des finalités définies par le modélisateur. La décomposition de l'objet en éléments devient une dimension parmi d'autres, que le modélisateur est appelé à ne pas détailler dans un premier temps, pour se concentrer sur ce que *fait* l'objet.

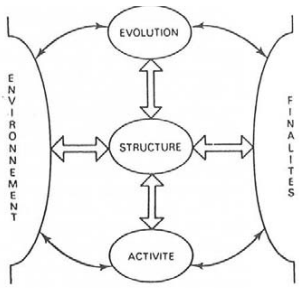



Figure 1: la vision système selon JL Le Moigne

La complexité d'un système se définit alors par toutes ces dimensions : finalités, environnement, activité, structure, évolution, ainsi que les relations entre ces dimensions (les flèches à double sens sur la Figure 1). La complexité provient des quantités de ces dimensions : grand nombre d'unités composant la structure, diversité d'activités, grand nombre d'évolutions, grande quantité d'interactions entre les différentes dimensions.

Mais, comme le souligne Edgar Morin (Morin 2005), la complexité provient également des incertitudes, des indéterminations, des aléas : la complexité comme « incertitude au sein de systèmes richement organisés ». C'est pour faire face à l'incertitude que la pensée systémique préconise de considérer l'objet à définir globalement dans sa relation avec son environnement et d'interpréter l'objet par son comportement, sans se soucier de suite d'établir une image de sa structure interne (Le Moigne 2006). C'est le passage de l'analyse à la conception, que l'on peut résumer par l'opposition entre *Cerner pour diviser et séparer* de Descartes (Descartes et Poisson 1724) et *Discerner pour relier et conjoindre*, l'Ingenium de Cicéron (Le Moigne 2006).

L'instrument de la pensée systémique est la **modélisation** : au lieu d'expliquer l'objet à connaître, il s'agit de concevoir un *modèle* de l'objet, c'est-à-dire une représentation de l'objet qui permet « de rendre compte d'une part de toutes les observations faites, et d'autre part de prévoir le comportement du système considéré dans des conditions plus variées que celles qui ont donné naissance aux observations » (Le Moigne 2006)p73). Ainsi, l'objectif de la modélisation n'est pas de simplifier mais de permettre l'intelligibilité du système sans en éliminer sa complexité.

Au-delà de la construction de système, le paradigme systémique propose une approche scientifique de construction des connaissances, dans laquelle les systèmes ne sont pas *dans* la nature, mais sont des constructions intellectuelles et artificielles permettant de représenter les objets *de* la nature que le scientifique veut connaître. Avec cette approche, le chercheur devient *concepteur* de modèles, c'est-à-dire *ingénieur.* Est-ce à dire que l'approche systémique conduit à une unification de l'activité entre chercheur et ingénieur ?

Le chercheur conçoit des modèles pour enrichir et construire des connaissances, l'ingénieur pour construire un objet technique. Ainsi, la question de la fidélité du modèle par rapport à la réalité que le modélisateur veut décrire se pose pour le chercheur, car il cherche à interpréter, comprendre et prévoir un phénomène existant : la réalité précède la modélisation. La fidélité ou pertinence du modèle est évaluée à travers sa capacité à prévoir des faits non encore observés. En effet, l'intérêt pour le chercheur est de combler des vides dans la connaissance, donc de prévoir des phénomènes sans possibilité immédiate de vérification ou d'expérimentation (Barel 1971).

La question ne se pose pas pour l'ingénieur, du moins dans un premier temps, car il cherche à construire une nouvelle réalité à travers la conception de modèles. Dans L'Encyclopédie (Diderot et D'Alembert 1751b), la définition de **modèle** précise que le mot :

> *est en particulier en usage dans les bâtiments, & il signifie un patron artificiel, qu'on fait de bois, de pierre, de plâtre, ou autre matière, avec toutes ses proportions, afin de conduire plus surement l'exécution d'un grand ouvrage, & de donner une idée de l'effet qu'il fera en grand.*

Pour l'ingénieur, la modélisation précède la réalité. En reprenant la définition de la Figure 1, l'ingénieur, en tant que concepteur de système technique, va produire des modèles de ce que le système fait (activité), ce que le système est (structure), et ce que le système devient (évolution), par rapport à des finalités qui sont définies par lui ou par une autre entité (client). Le rôle du modèle est de communiquer avec d'autres ingénieurs pour la construction du système technique.

Dans un premier temps, la question de la fidélité du modèle par rapport à la réalité est inversée : l'ingénieur va tester si la réalité, c'est-à-dire le système technique qu'il réceptionne, correspond bien au modèle. Cependant, dans un deuxième temps, dans les cas où il ne peut pas reproduire toutes les conditions d'opérations du système (comme dans l'exemple de la station spatiale internationale), il va pouvoir utiliser le modèle du système pour faire des prévisions sur le comportement du système en opérations.

Ainsi, le modèle, en tant que « système de *symboles »,* qu'ils soient graphiques, discursifs, mathématiques, iconiques, est l'outil central de l'ingénieur pour exprimer les différents aspects du système à venir. En tant que « patron artificiel » ou représentation du système à venir, le modèle d'un système complexe est-il lui aussi nécessairement complexe ?

# Présentation de la thèse

Dans cette thèse, nous faisons l'hypothèse qu'il est possible de décorréler la complexité du modèle de la complexité du système en se concentrant sur l'utilisabilité des outils de modélisation. La finalité de nos travaux est d'étendre la capacité intellectuelle des ingénieurs à gérer la complexité d'un système, en spécifiant au mieux son comportement. L'utilisabilité des représentations du système, en termes de visualisations et d'interactions, peut constituer une voie particulièrement adaptée pour appréhender la complexité d'un système pour les ingénieurs.

Nous proposons une vision *située* de l'ingénierie des exigences, dans laquelle nous mettons en valeur les parties prenantes (client, utilisateurs, ingénieur en exigences, fournisseurs) et leur champ de travail commun : le système, en utilisation et en définition. Dans cette vision située, les exigences sont un mécanisme de coordination entre les parties prenantes, mais ne constituent pas la véritable base du travail collaboratif entre parties prenantes. Seuls les ingénieurs en exigences utilisent les exigences comme base de travail collaboratif entre eux, dans les activités de vérification.

Nous proposons d'instrumenter ces activités avec des *visualisations interactives de texte structuré* : elles fournissent aux ingénieurs en exigences les moyens d'allier la souplesse nécessaire à l'acte de conception pendant le processus avec la rigueur requise par la certification en sortie du processus. Cependant, nous pensons que les outils de l'ingénieur ne peuvent pas répondre totalement à la maitrise de la complexité liée aux incertitudes de l'environnement : l'ingénieur doit se résoudre à ne pas pouvoir tout prédire dans le comportement du système. Nous proposons une nouvelle approche, l'*ingénierie participative des exigences*, articulant des techniques utilisées en conception participative pour impliquer les utilisateurs, et des techniques d'abstraction et de formalisation des exigences, afin de mieux informer la définition du système.

Le chapitre 1 présente l'ingénierie système prescrite par les normes. Il introduit le vocabulaire de l'ingénierie système et les activités prescrites de l'ingénieur. L'analyse des activités prescrites nous permet de dégager notre problématique de recherche relative aux tâches et outils utilisés en ingénierie des exigences.

La chapitre 2 présente un état de l'art sur l'ingénierie des exigences, en termes de processus, de pratiques industrielles et d'outils. Il nous permet de dégager la problématique de l'utilisabilité des outils d'ingénierie des exigences, déclinée en quatre questions de recherche :

> RQ1 : Quelles sont les activités d'ingénierie des exigences réalisées par les ingénieurs en aéronautique ?
>
> RQ2 : Comment les outils supportent-ils ces activités ?
>
> RQ3 : Comment améliorer l'utilisabilité des outils d'ingénierie des exigences, en exploitant les principes de la visualisation d'information ?
>
> RQ4 : Comment faire participer les utilisateurs à la spécification du système futur à partir de leurs usages du système actuel ?

Le chapitre 3 présente notre méthode, mêlant étude de cas et utilisation de prototypes pour comprendre les activités réalisées par les praticiens industriels en aéronautique et leur support outillé.

Le chapitre 4 présente notre analyse des pratiques industrielles en aéronautique. Il présente les contextes d'utilisation observés, les scénarios d'utilisation et les problèmes d'utilisabilité relevés. Nous en déduisons une vision située de l'ingénierie des exigences et formulons des premières

exigences d'utilisabilité pour les outils d'ingénierie des exigences (Hélène Gaspard-Boulinc et Conversy 2014).

Le chapitre 5 présente des propositions de visualisations interactives de texte structuré pour instrumenter les activités de raffinement et vérification des exigences. L'évaluation de ces visualisations par des ingénieurs nous permet de consolider notre vision située de l'ingénierie des exigences et d'enrichir les exigences d'utilisabilité sur les outils (H. Gaspard-Boulinc et Conversy 2017).

Au-delà des outils, le chapitre 6 présente une nouvelle approche : l'ingénierie participative des exigences. Sa finalité est une production d'exigences matures spécifiant dans le système futur la prise en compte de situations non prévues dans le système actuel, mais gérées par les utilisateurs en contexte opérationnel. Nous illustrons l'application de l'ingénierie participative des exigences sur quatre projets : contrôle aérien et collaboration (Conversy et al. 2011), cockpit de l'avion-école électrique E-Fan 2.0 (Pujos et al. 2016)(Hélène Gaspard-Boulinc, Conversy, et al. 2016), analyse du rapport d'accident du vol 447 Rio-Paris (Conversy et al. 2014) et nouvel instrument de vol (Louviot, Gaspard-Boulinc, et Conversy 2017).

# Chapitre I. Définitions et normes de l'ingénierie système

Après la deuxième guerre mondiale, la complexité des systèmes conçus par les ingénieurs a fait émerger le besoin d'un cadre améliorant la communication et la coopération parmi les parties qui créent et gèrent les composants d'un système. Par exemple, le projet ATLAS de missile balistique intercontinental dans les années 50 aux Etats-Unis a impliqué 18 000 scientifiques, ingénieurs, experts techniques dans les universités et les industries. 70 000 personnes, de l'administration jusqu'à l'usine, réparties dans 22 entreprises, ont participé activement au projet. Ces chiffres montrent le nombre important d'acteurs mais également leur hétérogénéité, et donc la nécessité d'un cadre de coordination entre ces acteurs. La réponse est l'ingénierie système, définie comme :

*une approche interdisciplinaire pilotant l'ensemble des efforts techniques et managériaux nécessaires à la transformation de besoins, attentes et contraintes de parties prenantes en une solution, et de maintenir cette solution pendant toute sa durée de vie*.

Il existe trois normes de référence en ingénierie système : ISO/IEC 15288, ANSI/EIA 632 et IEEE 1220. Les deux premières sont issues du même standard américain EIA 632. Nous n'étudierons ici que la norme internationale ISO 15288, la norme ANSI/EIA 632 étant un standard américain (ANSI/EIA 632 - Processes for Engineering a System).

La norme ISO 15288 (ISO/IEC/IEEE 2015) référence abondamment la norme ISO 12207 (ISO/IEC 2008), qui existait précédemment pour l'ingénierie des logiciels. Ces deux normes ISO évoluent régulièrement et de façon coordonnée pour une mise en cohérence progressive.

La norme IEEE 1220 (IEEE 2005), également ISO 26702 (ISO/IEC 2007), se positionne par rapport à la norme ISO 15288 dans une annexe informative dédiée, alors que la norme ISO 15288 ne la cite pas. Cela peut être expliqué par l'existence historique de deux branches depuis les années 90 (Figure 2) (Spitzer 2000).

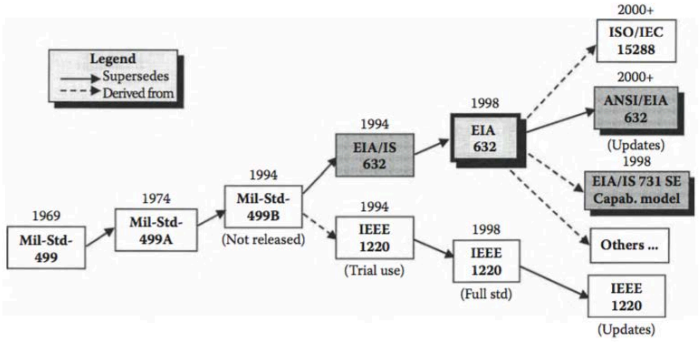



Figure 2: une vision chronologique des standards IS (Spitzer 2000)

Par conséquent, nous proposons de retenir la norme ISO 15288 et la norme IEEE 1220 pour notre analyse. Dans ce chapitre, nous proposons une analyse de ces deux principaux standards définissant l'ingénierie système, afin d'identifier le cadre de référence et les activités prescrites de l'ingénieur système.

# I.1. La définition de « système »

La norme ISO 15288 définit un système par :

> *a combination of interacting elements organized to achieve one or more stated purposes*

La norme IEEE 1220 intègre la définition des éléments dans celle de **système** :

> *A set or arrangement of elements [people, products (hardware and software) and processes (facilities, equipment, material, and procedures)] that are related, and whose behavior satisfies operational needs and provides for the life cycle sustainment of the products.*

La première définition est celle retenue par la société savante en ingénierie système INCOSE (INCOSE 2015). Nous proposons de comparer les deux définitions pour conduire une analyse détaillée et une mise en perspective par rapport aux définitions vues en introduction. L'analyse des définitions nous permet d'introduire les principaux concepts d'ingénierie système : un élément, une exigence, une spécification, un système de soutien, l'environnement du système, l'utilisateur et l'opérateur.

## I.1.1. Une combinaison ou un ensemble ?

Il est intéressant de retrouver le mot « combinaison » (a combination of elements), au lieu du mot « ensemble » (a set of elements) pour exprimer une interrelation entre les éléments, qui est une des sources de la complexité comme nous l'avons vu précédemment avec la combinatoire de Leibniz (Serres 2015). Cependant, la représentation graphique d'un système (cf Figure 3) montre une décomposition hiérarchique qui fait penser à un ensemble. Le rappel de la définition, intégré dans le schéma, utilise d'ailleurs le terme « a set ». Cette représentation hiérarchique ne traduit pas la combinaison des éléments, voire au contraire les isole par une décomposition analytique.

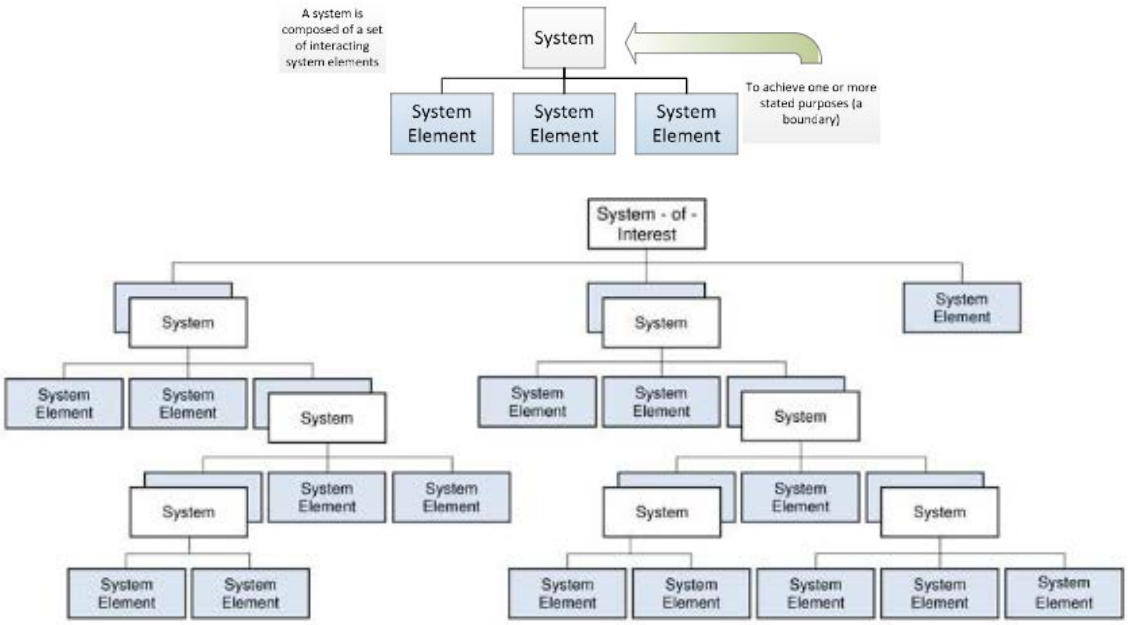



Figure 3: représentation hiérarchique d'un système (norme ISO 15288- p12)

Pourtant, c'est la seule représentation graphique ou modèle du système qui est proposée par les normes, alors que la norme ISO 15288 liste en annexe F les différents modèles d'un système (fonctionnel, comportemental, temporel, structurel, de masse, de disposition, de réseau) sans en donner un seul exemple. Par conséquent, les normes de référence, en proposant la décomposition hiérarchique comme seule représentation graphique d'un système peuvent transmettre une vision erronée de l'ingénierie système, la résumant à de la décomposition analytique.

De plus, comme rappelé dans (Le Moigne 2006), entre deux éléments ayant une entrée et une sortie, il n'existe pas une relation, mais dix-huit relations possibles (Figure 4). Si les éléments ont plusieurs entrées-sorties, le nombre des relations possibles peut rapidement augmenter. En supposant que cela soit le cas pour les nombreux éléments du système considéré, se pose la question de l'existence et de la lisibilité d'autres représentations graphiques que celles proposées par les normes.

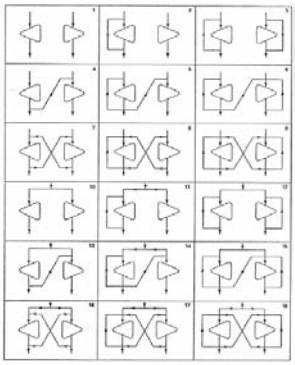


Figure 4: dix-huit relations possibles entre deux éléments à une entrée/une sortie (Le Moigne, Jean-Louis 2006)

## I.1.2. D'éléments ou de processus ?

Un élément du système est défini par la norme ISO 15288 par :

> *Member of a set of elements that constitutes a system.*

Cette seule définition d'élément d'un système est tautologique par rapport à la définition de système, et remplace de surcroit la notion d'ensemble à celle de combinaison. Les exemples fournis ne sont pas très précis:

> *A system element can be hardware, software, data, humans, processes (e.g. processes for providing service to users), procedures (e.g., operator instructions), facilities, materials, and naturally occurring entities (e.g., water, organisms, minerals), or any combination.*

Une note à la définition donne une information supplémentaire :

> *A system element is a discrete part of a system that can be implemented to fulfill specified requirements.*

La norme IEEE 1220 parle *d'éléments* pour désigner les êtres humains et les produits (matériel et logiciel), et de *processus* pour désigner toutes les activités de soutien au cycle de vie : développement/test, construction, distribution et maintenance, opérations/formation et retrait (cf Figure 6). Elle définit une hiérarchie d'éléments, en prenant comme focus le système à développer :

un système est composé de produits, eux-mêmes composés de sous-systèmes, composés d'assemblages (Figure 5).

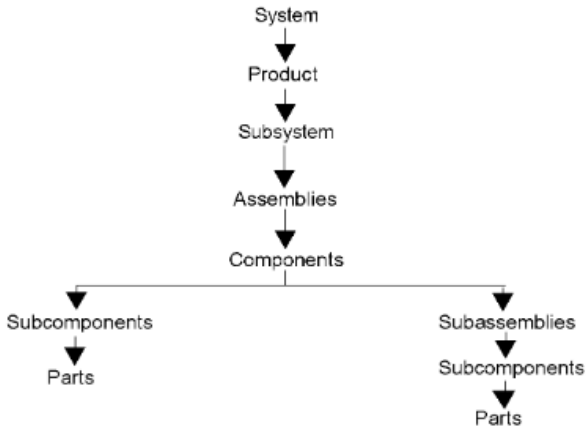



Figure 5: la hiérarchie des éléments d'un système selon IEEE 1220

Les deux normes insistent sur le fait que dans un système, les éléments constitutifs du système sont parfois eux-mêmes des systèmes, soit parce qu'ils sont complexes c'est-à-dire composés de matériel, de logiciel et d'humain, soit parce que le fournisseur peut le considérer à son niveau comme un système. Tout est question de point de vue : la norme ISO 15288 renomme ainsi les éléments complexes en « système » et nomme le système à développer « système d'intérêt » (cf Figure 3). Pour IEEE 1220, même si un des éléments peut être qualifié de système dans d'autres contextes, il est nommé « sous-système » dans le contexte du système à développer.

La question finalement est de savoir où s'arrête la décomposition pour isoler les éléments constitutifs du système. La norme IEEE 1220 semble étalonner la taille d'un composant au fait qu'il soit géré par une seule équipe de développement, la décomposition sous le composant ne servant alors qu'à répartir le travail au sein d'une même équipe. Mais ce n'est pas complètement explicite.

La note à la définition proposée par ISO 15288 est plus explicite en se référant à la possibilité d'une implémentation à partir *d'*exigences *spécifiées*. Il s'agit clairement d'un composant dont la spécification rend possible le développement.

## I.1.3. Spécifiés par des exigences

Par conséquent, les notions d'exigence (requirement en anglais) et de spécification se trouvent au cœur de la définition même d'un élément du système. La norme ISO 15288 définit une **exigence** comme:

> *Stakeholder requirements describe the needs, wants, desires, expectations and perceived constraints of identified stakeholders. They are expressed in terms of a model that may be textual or formal, that concentrates on system purpose and behaviour, and that is described in the context of the operational environment and conditions.*

Le mot « requirement » est un des plus utilisés dans le texte de la norme, mais aucun exemple de ce que peut être une exigence n'est fourni par l'ISO 15288. De plus, le terme de *spécification* n'est pas défini parmi les termes de référence de la norme.

Nous devons nous référer à IEEE 1220 pour la définition de **spécification** :

> *A document that fully describes a design element or its interfaces in terms of requirements (functional, performance, constraints, and design characteristics) and the qualification conditions and procedures for each requirement*

La spécification est donc un ensemble d'exigences décrivant de façon complète les éléments du système et ses interfaces. Cette définition nous permet de mieux approcher le concept d'exigence, en distinguant quatre types d'exigence : fonctionnel, de performance, de contraintes et de conception. Enfin, la définition d'exigence proposée par IEEE 1220:

> *A statement that identifies a product or process operational, functional, or design characteristic or constraint, which is unambiguous, testable or measurable, and necessary for product or process acceptability (by consumers or internal quality assurance guidelines)*

permet d'identifier des caractéristiques sur une exigence : non ambiguë, testable ou mesurable, et nécessaire à l'acceptabilité du produit. Mais cette définition soulève une nouvelle question : le caractère nécessaire de l'exigence est lié à l'acceptabilité du produit par un consommateur, qui n'est à son tour pas défini dans les normes. Il est seulement cité en exemple de client, au même titre que l'utilisateur final (end user), fournisseur, bénéficiaire et acheteur.

Nous retenons que le fait de pouvoir identifier un ensemble d'exigences qui peut donner lieu à une réalisation tangible (implémentation) permet d'identifier un élément du système.

## I.1.4. Environnement et frontière

La notion d'environnement et d'interaction avec l'environnement n'apparaît pas dans la définition de *système*, alors que c'était un élément introduit par la théorie générale des systèmes ((von Bertalanffy 1973) et Figure 1: la vision système selon JL Le Moigne).

De plus, la frontière du système n'est pas identique selon les normes : les activités de soutien sont intégrées dans le système pour IEEE 1220 sous forme de processus (Figure 6), alors qu'ils constituent des systèmes extérieurs pour ISO 15288 (Figure 7). Pour la norme IEEE 1220, c'est « le *projet* qui décide quels sont les éléments dont la conception est sous contrôle » : le critère est de savoir s'il y a un *effort de développement à produire* sur l'élément et s'il est *sous contrôle*. Ces deux critères expliquent pourquoi les activités de soutien au système font partie du système.

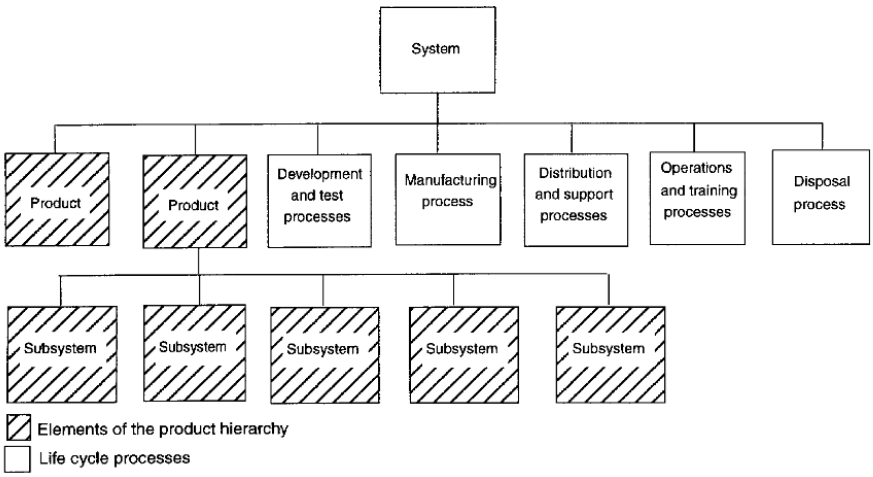



Figure 6: le système inclut les processus de soutien selon la norme IEEE 1220 (blocs non hachurés)

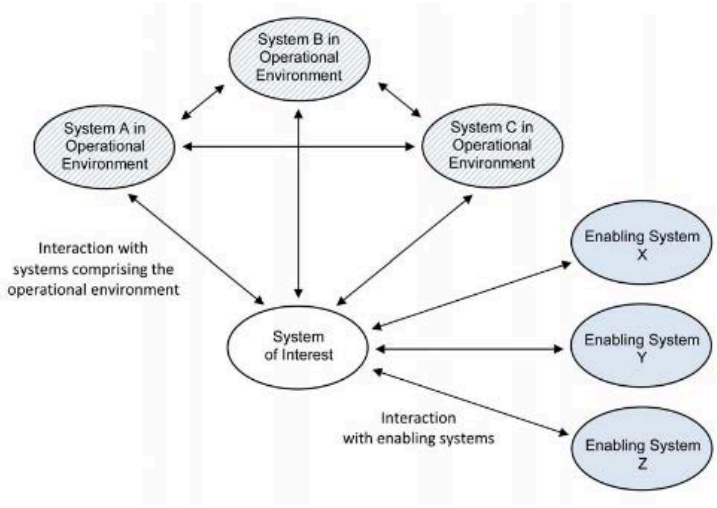



Figure 7: les systèmes de soutien sont en interface avec le système selon la norme ISO 15288 (bulles bleues à droite)

La norme ISO 15288 s'étend peu sur la définition des frontières du système. Mais la notion de projet semble également déterminante dans la délimitation du système, de façon implicite : le projet qui est responsable du système d'intérêt peut être responsable de la création d'un système de soutien, les deux systèmes étant alors vus comme un seul système. Ou au contraire, la création d'un système de soutien peut être vue comme un projet séparé, et donc être un autre système. La définition du **système de soutien** peut nous donner des indications :

> *system that supports a system-of-interest during its life cycle stages but does not necessarily contribute directly to its function during operation*

La distinction entre le système d'intérêt et le système de soutien est basée sur une notion de contribution directe au fonctionnement opérationnel du système. Cependant, la norme précise que « s'ils sont interdépendants, alors ils peuvent être vus comme un seul système ». La notion de projet mise en avant par IEEE 1220, en termes de contrôle sur la conception et d'efforts de développement, s'interprète à travers le rôle donné par (Le Moigne 2006)-p127 au modélisateur dans la délimitation du système (il parle de « guider la main du découpeur ») : ce sont les *intentions du modélisateur*, de conception et de répartition du travail pour la norme IEEE 1220, qui permettent de déterminer explicitement les frontières du système.

La norme ISO 15288 a peut-être du mal à exprimer ce que sont les frontières du système d'intérêt parce qu'elle revendique de couvrir l'ensemble du cycle de vie du système, de la conception du système à son retrait. Or les intentions vont peut-être être différentes selon l'étape de la vie du système. La norme ISO 15288 ne précise finalement pas de règles permettant d'aider à repérer des frontières entre le système et son environnement, mis à part la « contribution directe au fonctionnement du système », ce qui reste vague : l'air contribue directement au fonctionnement du système avion, par le phénomène de portance, tout en étant considéré comme faisant partie de l'environnement de l'avion.

## I.1.5. L'humain est-il un élément du système ?

Alors que les normes restent peu précises sur la délimitation des frontières du système, elles accordent un point particulier à l'humain. Elles s'accordent pour faire la différence entre l'humain *utilisateur* et l'humain *opérateur*.

> *humans can be viewed as both users external to a system and as system elements (i.e., operators) within a system. (ISO 15288- p11)*

L'humain opérateur est un élément du système qui contribue au fonctionnement du système, ce qui est cohérent avec les autres éléments du système. Pour contribuer à ce fonctionnement, il maitrise des connaissances, des savoir-faire et des procédures. C'est à la condition de cette maitrise qu'il est considéré comme élément du système. L'humain utilisateur est extérieur au système : il bénéficie du système en utilisation, selon IEEE 1220. Pour ISO 15288, l'humain utilisateur peut également *interagir* avec le système. La dichotomie actif/passif par rapport au système ne permet donc pas de distinguer les deux rôles attribués à l'humain. Par contre, la maitrise de connaissances, savoir-faire et procédure n'est pas exigée pour l'humain utilisateur. C'est ce qui différencierait l'utilisateur de l'opérateur.

Cependant, les deux normes précisent que les deux rôles, opérateur et utilisateur, peuvent être simultanément ou séquentiellement, investis par le même individu. Nous pensons que cette précision apporte plus de confusion que de clarification. D'un point de vue *simultané*, comment un même individu pourrait-il être à la fois considéré comme faisant partie du système et extérieur au système ? Comment distinguer les connaissances, savoir-faire et procédure qu'il mobiliserait en tant qu'opérateur, et qui ne lui seraient pas utiles en tant qu'utilisateur ? D'un point de vue *séquentiel*, si un individu est dans un premier temps opérateur *dans* le système, puis dans un deuxième temps utilisateur *à l'extérieur* du système, cela implique une frontière du système mouvante dans le temps à l'échelle de l'individu. Cette possibilité de frontière mouvante va à l'encontre de ce qui peut être dit par les mêmes normes sur un choix de frontières à faire pour délimiter le système d'intérêt. Cette incertitude sur l'appartenance ou non du même individu au système pose la question de la définition de la frontière entre l'individu et le système.

## I.1.6. Synthèse des questions posées par la définition de système

Les normes de référence, en proposant la décomposition hiérarchique comme seule représentation graphique d'un système, peuvent transmettre une vision erronée de l'ingénierie système, la résumant à de la décomposition analytique. Si les éléments du système ont plusieurs entrées-sorties, le nombre des relations possibles peut rapidement augmenter. L'introduction de nombreux automatismes dans les systèmes actuels allant dans ce sens, se pose la question de l'existence et de la lisibilité d'autres représentations graphiques que celles proposées par les normes.

Le fait de pouvoir identifier un ensemble d'exigences qui peut donner lieu à une réalisation tangible permet d'identifier un élément du système. Les exigences sont au cœur de la définition d'un système.

La notion d'environnement et d'interaction avec l'environnement n'apparaît pas dans la définition de *système*, alors qu'elle est cruciale dans la théorie du système général (von Bertalanffy 1973)(Le Moigne 2006) : ce sont les *intentions du modélisateur* qui permettent de déterminer explicitement les frontières du système. En revendiquant de couvrir l'ensemble du cycle de vie du système, de sa conception à son retrait, les normes ne font-elles pas face à des intentions différentes selon l'étape de vie du système qui impliqueraient des frontières différentes ?

Alors que les normes restent peu précises sur la délimitation des frontières du système avec son environnement, elles accordent un point particulier à l'humain. Elles s'accordent pour faire la différence entre l'humain *utilisateur,* qui serait à l'extérieur du système, et l'humain *opérateur* qui serait un élément du système. Les deux normes, en précisant que les rôles, opérateur et utilisateur, peuvent être simultanément ou séquentiellement investis par le même individu, sous-entendent une incertitude à l'échelle d'un individu sur son appartenance ou non au système, et pose la question de l'existence de la frontière entre l'individu et le système.

# I.2. Les processus d'ingénierie système prescrits

Nous proposons dans cette partie une description normative des processus d'ingénierie système, à partir de l'analyse des normes de référence ISO 15288 et IEEE 1220. Avec cette analyse nous poursuivons la mise en perspective des normes par rapport aux définitions vues en introduction, et sur la place des modèles dans les processus prescrits d'ingénierie système. Nous continuons à présenter les concepts d'ingénierie système tels que le référentiel d'exigences, l'architecture et la traçabilité.

## I.2.1. Perspective « orientée résultats » des processus

Les deux normes sont très différentes en termes d'organisation du contenu et de présentation. IEEE 1220 est centrée sur les processus techniques de l'ingénierie système, alors que ISO 15288 propose en complément aux processus techniques trois autres groupes de processus : contractuels, management et organisationnel. De plus, IEEE 1220 est didactique : elle propose un processus global d'ingénierie système (Figure 8), montrant les relations des huit processus techniques, ainsi qu'un diagramme pour chaque processus technique, montrant les entrées, les différentes activités à mener, et les sorties. Enfin, elle propose des plans de documents produits par les processus techniques. Au contraire, ISO 15288 ne propose aucun diagramme illustrant les relations entre les processus, seule la cartographie des vingt-cinq processus de la Figure 9 est proposée. Les liens entre les processus sont indiqués sous forme de note additive dans la description textuelle des processus. L'absence de diagrammes rend la norme difficile à comprendre.

Cependant, il est possible d'identifier une correspondance entre les processus des deux normes. En adoptant une perspective orientée résultats, nous proposons en Tableau 1 un regroupement des processus selon 4 catégories : ingénierie des exigences, conception du système, analyse système et management technique.

Il convient de noter les problèmes récurrents de vocabulaire, que ce soit en français et en anglais sur les termes spécification et design. En anglais, la spécification désigne le résultat des activités de « Requirements Engineering ». En français, la spécification peut désigner soit les activités, soit le résultat de ces activités, selon le contexte. Il n'y a pas véritablement de traduction de Requirements Engineering. Nous utiliserons le terme Ingénierie des Exigences. En anglais, c'est le terme « design » qui comporte cette ambiguïté, pouvant désigner tour à tour le processus de conception, et le résultat de ce processus (la solution). En français, l'ambiguïté est parfois moins forte, avec l'utilisation du terme « conception » pour décrire le processus, et l'utilisation du terme « design » pour le résultat.

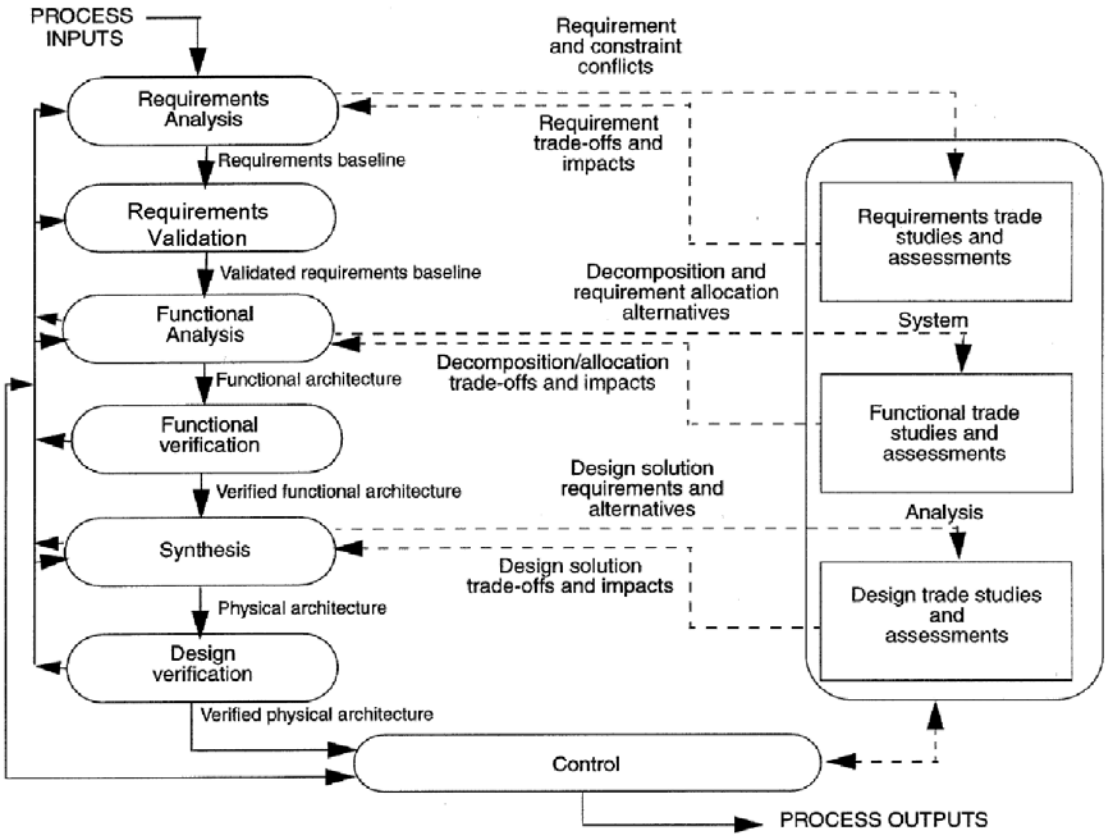



Figure 8: le processus global proposé par IEEE 1220

| | **IEEE 1220** | **ISO 15288** | **Résultats** |
|---|---|---|---|
| Ingénierie des exigences (Spécifier les exigences) | Requirements Analysis<br>Requirements validation<br>Functional Analysis | Business or Mission Analysis<br>Stakeholder Needs & Requirements Definition<br>System requirements definition | Référentiel des exigences (requirements baseline) |
| Conception du système (Concevoir le système) | Functional Analysis<br>Functional Verification<br>Synthesis<br>Design verification | Architecture definition<br>Design definition | Architecture fonctionnelle<br>Architecture physique= éléments système + interfaces |
| Analyse système (Mener des analyses système) | Requirements trade studies and assessments<br>Functional trade studies and assessments<br>Design trade studies and assessments | System Analysis | Résultats d'analyse pour justifier des choix |
| Management technique (Codnuire le management technique) | Control | Project Assessment and control Process<br>Risk management<br>Configuration Management<br>Information management<br>Measurement process | Répertoire intégré des données, avec contrôle des changements |

Tableau 1 : correspondance entre processus IEEE 1220 et ISO 15288

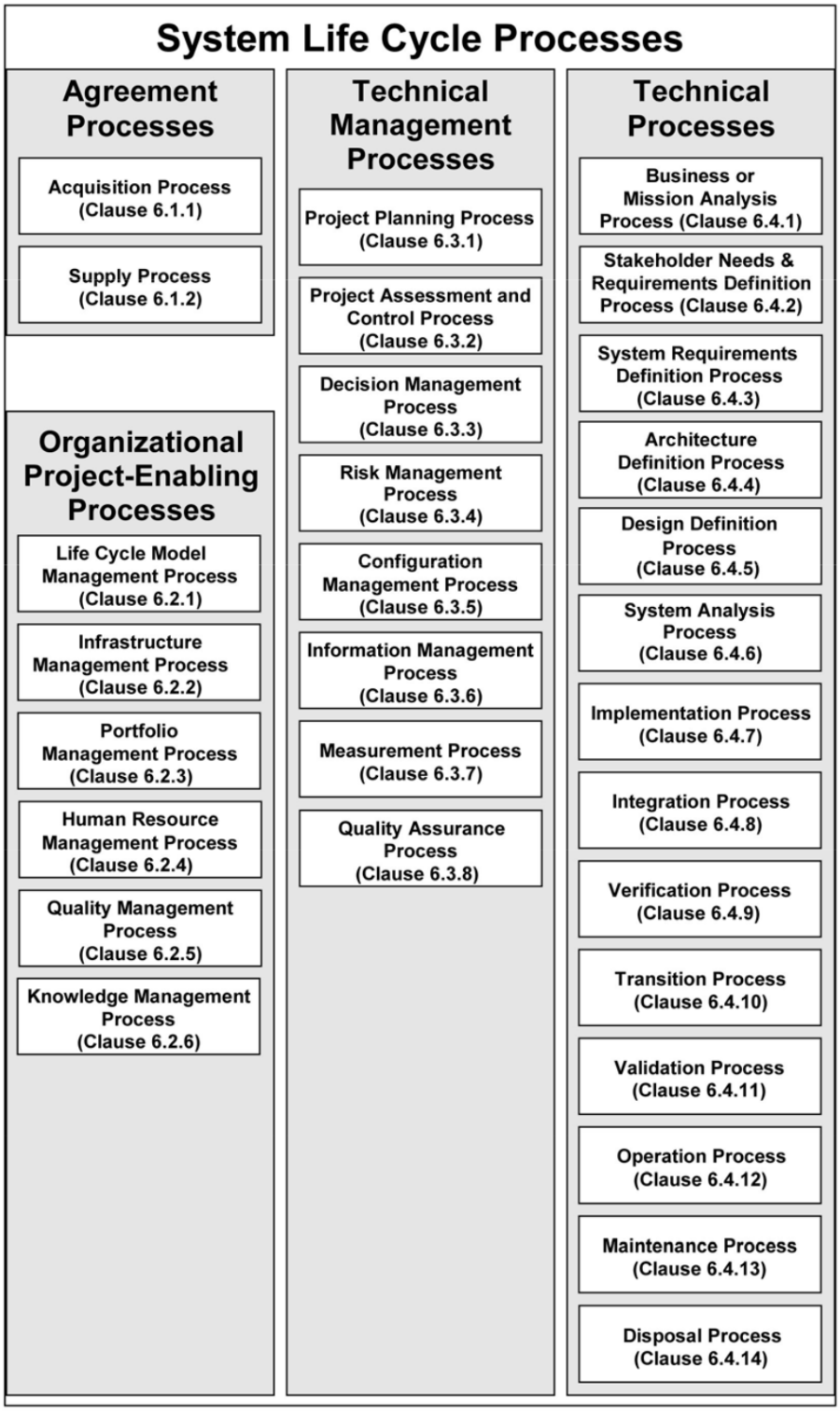



Figure 9: les processus d'ingénierie système selon la norme ISO15288:2015

Nous partons de la catégorisation définie en Tableau 1 (ingénierie des exigences, conception du système, analyse système, management technique) pour détailler les activités prescrites par les normes. Nous étudions notamment la place des modèles et des outils de modélisation dans les activités prescrites.

## I.2.2. Ingénierie des exigences

### I.2.2.1. Objectif

L'objectif des processus que nous avons regroupés sous l'appellation « ingénierie des exigences » est de définir les missions du système, les besoins des parties prenantes et de les traduire sous forme d'exigences système. Les besoins (ou *stakeholder requirements*) sont une vue orientée utilisateur des capacités désirées du système, qui vont être transformés en exigences système (*system requirements*), correspondant à une vue technique et orientée fournisseur des caractéristiques du système (p54- ISO 15288). Les deux normes parlent également d'une définition de l'espace problème. Le résultat est le référentiel d'exigences**,** défini comme:

> *A specification or product that has been formally reviewed and agreed upon, that thereafter serves as the basis for further development, and that can be changed only through formal change control procedures.(ISO/IEC 2007)*

Le référentiel représente « la définition du problème à résoudre ». La définition de référentiel insiste sur l'aspect formalisé, provenant d'une revue valant accord, et dont tout changement ne peut avoir lieu qu'à travers des procédures de contrôle.

### I.2.2.2. Activités prescrites

Les activités prescrites sont représentées par IEEE 1220 dans le diagramme de la Figure 10: il s'agit d'activités de définition des attentes des parties prenantes, des contraintes du projet, des contraintes externes, des scénarios opérationnels, des frontières du système, des interfaces, des environnements d'utilisation, des exigences fonctionnelles, des exigences de performances, des modes d'opérations, des caractéristiques de conception, les facteurs humains (IEEE 1220) ou les interactions entre l'utilisateur et le système (ISO 15288).

Les branchements proposés vers les autres processus décrivent que les exigences fonctionnelles identifiées doivent par la suite être décomposées dans le processus d'analyse fonctionnelle, et que les exigences de performance doivent servir de critères dans le processus d'analyse système. On retrouve les mêmes activités de définition ou d'identification dans les processus ISO 15288, avec la même précision sur le fait de nourrir le processus d'analyse système en critères de performance, et d'évaluer les exigences en termes de coûts et de faisabilité technique.

La norme ISO 15288 intègre en supplément une activité de caractérisation de l'espace solution dans le processus *Business/Mission analysis*. En note, il est précisé que l'espace solution peut « inclure l'identification de systèmes, produits ou services existants qui peuvent répondre au besoin de modifications opérationnelles ou fonctionnelles ». Cette analyse peut faire appel au processus de Définition d'Architecture avec des « vues d'architecture » (architecture views). Une activité supplémentaire est d'identifier les classes de solution, de les évaluer afin d'en sélectionner une. Les critères de choix sont définis par la « stratégie de l'entreprise ». Que notre lecteur ne se méprenne pas : nous relevons ce point et le mettons à son attention, mais il ne constitue pas un point central du processus, l'ensemble de ces éléments étant rédigés sous forme de note additive au texte principal de la norme.

Comme indiqué précédemment, les normes ne fournissent aucun exemple de besoins ou d'exigences. La norme ISO 15288 se réfère plusieurs fois à la norme ISO 29148 (ISO/IEC/IEEE 2011), que nous analyserons dans notre état de l'art.

En termes de modèle, IEEE 1220 préconise l'utilisation de modèles en ces termes, dans la branche « functional context analysis » :

> *Analyses are conducted to **understand** the functional behavior of the system under various conditions and to assess the integrity of the functional architecture. Analyses should involve the **simulation** or stimulation of functional models utilizing operational scenarios that expose the models to a variety of stressful and nonstressful situations that reflect **anticipated** operational usage and environments.*

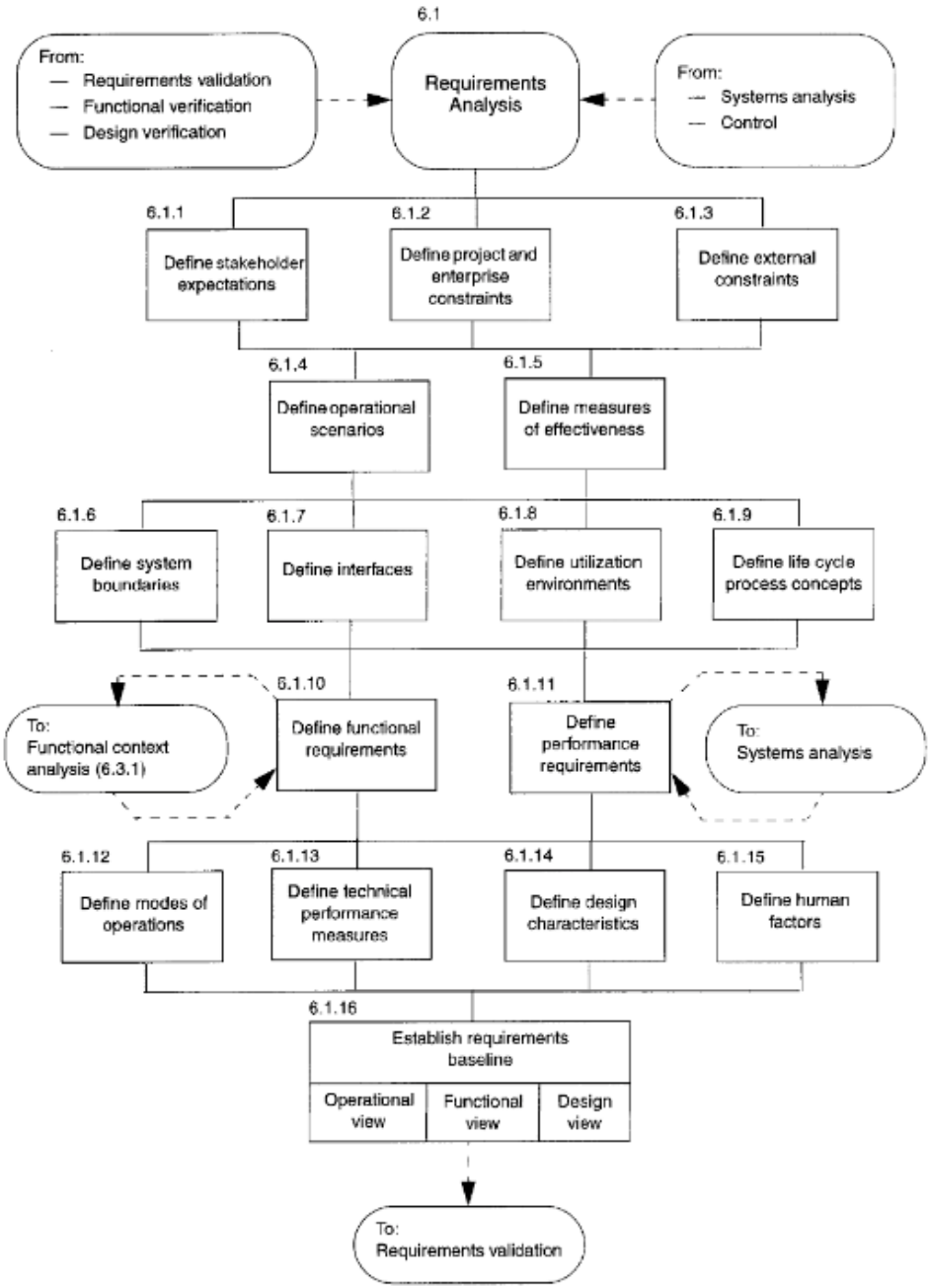



Figure 10: les activités prescrites par IEEE 1220 sur l'ingénierie des exigences

Ainsi, les modèles sont utilisés ici pour comprendre le comportement fonctionnel du système et pour le prédire dans diverses situations. La conception du modèle, à savoir le travail de modélisation, n'est

pas évoqué dans les processus d'ingénierie des exigences. En fin de processus, il y a une activité de validation des exigences par les parties prenantes, afin de s'assurer que les besoins ont été capturés et bien exprimés : cela fait l'objet d'un deuxième diagramme pour IEEE 1220 (Figure 11). L'identification d'un écart ou d'un conflit entraine une nouvelle itération sur les exigences (branchement identifié sur Figure 11) jusqu'à l'obtention d'un référentiel validé. La norme précise que ce n'est qu'une fois que le référentiel est validé qu'il est possible de passer au processus suivant, introduisant une notion d'ordre dans l'application des processus.

En résumé, dans la norme IEEE 1220, l'itération est préconisée en interne aux processus de spécification de système, jusqu'à obtenir un référentiel validé, mais n'est pas autorisée entre les processus de spécification et les processus de conception.

La gestion de l'itération est différente dans la norme ISO 15288 : en effet, elle précise que les processus de spécification et de conception sont parfois menées de façon itérative (p57):

> *Iteration of the Architecture Definition process with the Business or Mission Analysis process, System Requirements Definition process, Design Definition process, and Stakeholder Needs and Requirements Definition process is often employed so that there is a negotiated understanding of the problem to be solved and a satisfactory solution is identified.*

Ainsi, elle ne conditionne pas le passage de la spécification à la conception par une validation d'un référentiel d'exigences, mais par une *confirmation* que les exigences sont une réponse nécessaire et suffisante à des besoins, et une entrée suffisante à la conception.

> *Feedback helps ensure that the specified system requirements have been adequately captured and expressed. Confirmation is made that they are a necessary and sufficient response to stakeholder requirements and a necessary and sufficient input to other processes, in particular architecture and design.*

Cependant, elle exige que le référentiel soit géré en configuration, et que tous les changements résultant de la conception soient enregistrés, en faisant appel systématiquement aux processus de management technique.

> *Maintaining system requirements includes defining, recording, and controlling the baseline, generally under formal configuration management, along with managing any changes resulting from the application of other life cycle processes such as architecture or design.*

En résumé, dans la norme ISO 15288, l'itération entre les processus de spécification et de conception est soumise à un gestion de configuration rigoureuse enregistrant tout changement d'exigence et maintenant une traçabilité entre les différentes données d'ingénierie, à travers les processus de management technique. La norme IEEE 1220 précise que le référentiel d'exigences doit comporter trois « vues » (view) : opérationnelle, fonctionnelle et de design. Les termes de « modèle » et de « diagramme » n'apparaissent pas dans la description du référentiel d'exigences. Un seul élément pourrait ressembler à un modèle, mais est identifié comme « picture » : *Operational sequences/scenarios (best portrayed in pictures).*

La traçabilité entre les exigences système et les besoins des parties prenantes doit être réalisée afin d'assurer que tous les besoins sont remplis par une ou plusieurs exigences système, et que toutes les exigences système remplissent ou contribuent à remplir un besoin utilisateur. Un *dépôt de données* (integrated data repository) est conseillé pour assurer la traçabilité entre besoins et

exigences. ISO 15288 (p56) précise que les exigences système doivent être enregistrées *dans une forme adapté à la gestion des exigences à travers le cycle de vie. Ces enregistrements incluent la justification, hypothèse et décision associées, bases pour la traçabilité.*

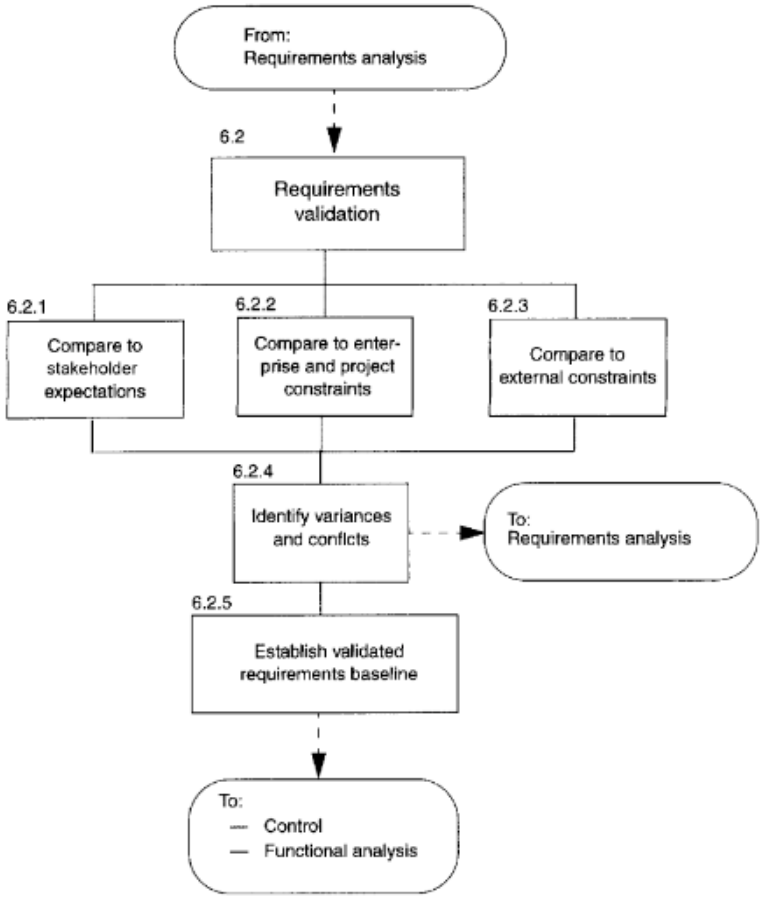



Figure 11: validation des exigences selon IEEE 1220

## I.2.3. Conception du système

### I.2.3.1. Objectif

L'objectif des processus que nous regroupons sous « Conception du système » est de décomposer les exigences système en fonctions, jusqu'à pouvoir identifier les éléments du système sur lesquels allouer ces fonctions. Le référentiel d'exigences système (validé ou géré en configuration) est en entrée des processus de conception.

Le résultat de ces processus est double, avec une architecture fonctionnelle définie comme :

> *An arrangement of functions and their subfunctions and interfaces (internal and external) that defines the execution sequencing, conditions for control or data flow, and the performance requirements to satisfy the requirements baseline.*

Et une architecture physique ou organique (design architecture) définie comme :

> *an arrangement of system elements, their decomposition, interfaces (internal and external), and design constraints.*

A l'issue de ces processus, il doit y avoir assez d'information sur le système et ses éléments pour atteindre un niveau d'implémentation de la solution. La distinction entre les deux architectures n'est pas évidente : nous proposons de l'expliquer à partir de la description des activités prescrites par les processus, en séparant la conception fonctionnelle de la conception organique.

### I.2.3.2. Activités prescrites pour la conception fonctionnelle

Les activités prescrites sont résumées par IEEE 1220 dans le diagramme de la Figure 12 : il s'agit, pour les exigences système validées, de conduire une décomposition fonctionnelle, en définissant les sous-fonctions, les comportements et séquencements des sous-fonctions, les échanges de données entre les sous-fonctions, les pannes et les fonctions de supervision. Cette décomposition est préconisée pour chaque scénario opérationnel identifié dans le processus d'ingénierie des exigences.

La norme ISO 15288 insiste sur le fait que ce processus doit générer plusieurs architectures fonctionnelles alternatives. Les activités à mener sont le développement de modèles et de vues d'architectures candidates, et une évaluation par rapport aux contraintes et exigences, à l'aide du processus « Analyse Système ». L'évaluation de ces architectures fonctionnelles alternatives doit permettre de sélectionner une architecture fonctionnelle, qui va être gérée en configuration, afin de gérer les évolutions, et maintenir la traçabilité bi-directionnelle entre les éléments d'architecture (modèles et vues) et les exigences.

Les notions de vues et de modèles sont largement utilisées dans la description des activités. La norme ISO 15288 indique que l'usage de modèles logiques et physiques dans la définition d'architecture est courant, et fournit des informations supplémentaires en annexe F sur les modèles fonctionnel, comportemental, temporel, structurel, de masse, de disposition, de réseau, de fiabilité.

L'architecture doit être autant que possible indépendante du design (ie de la solution) afin de permettre le plus de flexibilité possible dans l'espace de conception. Enfin, dans le cycle de vie d'un système, la norme précise que l'architecture fonctionnelle doit rester constante alors que la solution va changer en suivant les évolutions technologiques.

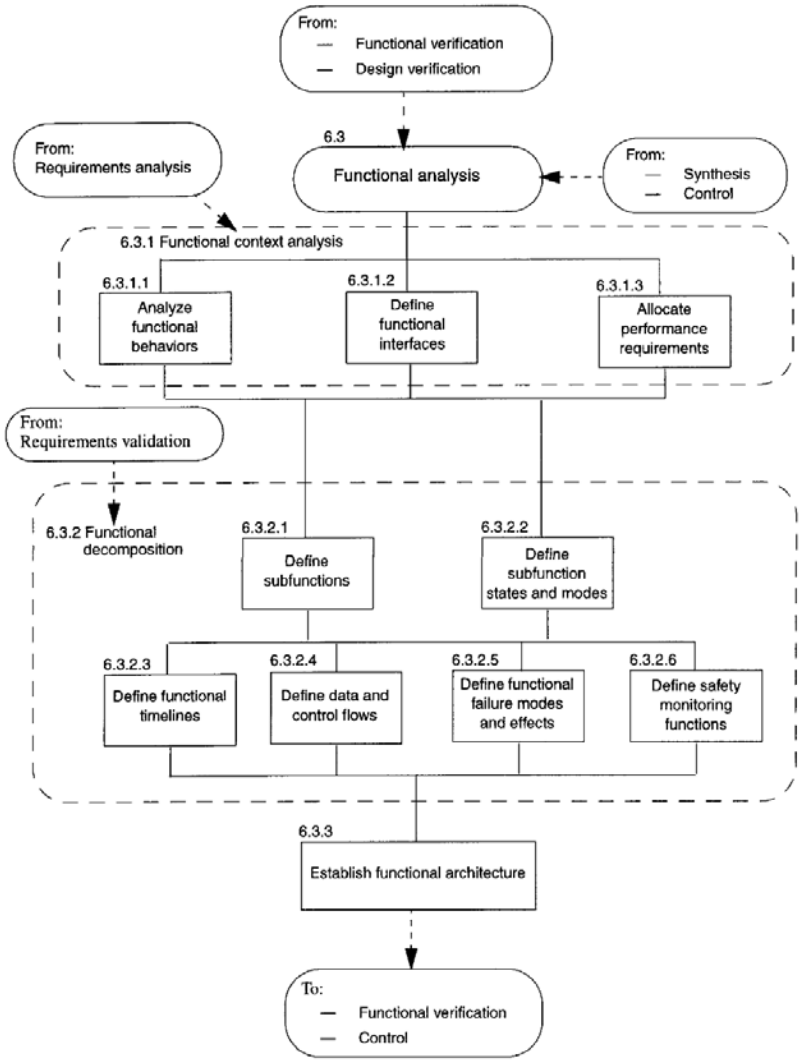



Figure 12: activités de conception fonctionnelle selon IEEE 1220

### I.2.3.3. Activités prescrites pour la conception organique

Les activités prescrites sont résumées dans la Figure 13 issue la norme IEEE 1220. Les activités sont orientées technologie et considèrent la faisabilité de construction et d'intégration : elles doivent permettre de transformer les éléments d'architecture en caractéristiques de la solution. Différentes solutions sont examinées et évaluées, les interfaces entre les éléments du système et avec l'environnement sont définies ou raffinées à partir de l'architecture fonctionnelle. Les alternatives pour obtenir les éléments du système sont évaluées, entre l'utilisation de produit sur étagère, la réutilisation d'une précédente solution, ou une nouvelle solution à développer. L'analyse de ces alternatives renvoie vers le processus « Analyse Système ».

Le développement de modèles et de prototypes est largement conseillé également dans ce processus, comme indiqué dans le diagramme de la Figure 13 (item 6.5.11) ou comme par exemple p63- iso 15288 :

> *This task transforms each architectural characteristic related to architectural entities assigned to the system element into design characteristics (dimensions, shapes, materials, critical quality*

*characteristics, data processing structures, etc.) using adequate representation such as drawings, diagrams, models, architectures, tables of metrics and their values, etc.; every data is associated with detailed acceptable margins for implementation).*

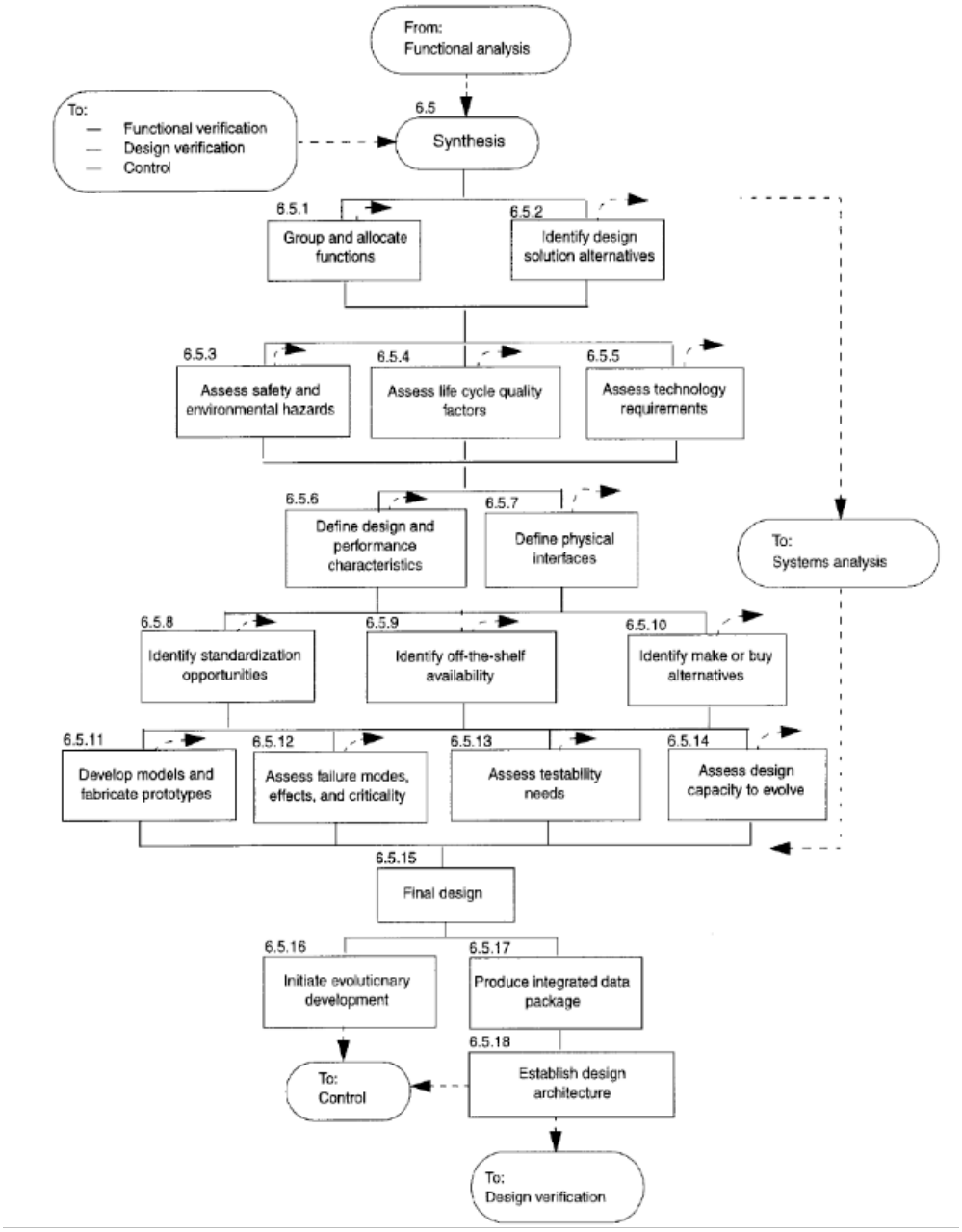



Figure 13: le processus de synthèse IEEE 1220

Enfin, la traçabilité est exigée (renvoi vers le processus Control dans la Figure 13) entre les caractéristiques de la solution et les éléments d'architecture, les interfaces, les résultats d'analyse et de justification des choix.

## I.2.4. Analyse système

### I.2.4.1. Objectif

L'objectif de ce processus est de fournir des informations rigoureuses pour faire un choix chaque fois qu'une question est soulevée. Le degré de rigueur de l'analyse dépend de la criticité de l'information. Les questions peuvent concerner n'importe quelle donnée d'ingénierie : des exigences, de l'architecture fonctionnelle, de l'architecture organique. C'est pour cela que ce processus est convoqué dans la description des activités d'ingénierie des exigences et de conception du système, et qu'il apparaît en parallèle des processus de spécification et de conception, à droite dans le processus global d'ingénierie système IEEE 1220 (Figure 8).

### I.2.4.2. Activités prescrites

Les activités prescrites, représentées par le diagramme de la Figure 14 dans la norme IEEE 1220, consistent à identifier des conflits ou des alternatives (problème ou question dans la norme ISO 15288), de définir le périmètre de l'analyse, notamment le degré de précision de la réponse, qui va permettre de choisir la méthode d'analyse. Les méthodes d'analyse incluent le jugement d'expert, le calcul « à la louche » (« back of the envelope » en anglais), des calculs, des simulations, des modèles mathématiques et du prototypage. C'est dans ce processus que nous retrouvons l'activité de l'ingénieur que nous évoquons en introduction, utilisant un modèle mathématique pour dimensionner une poutre par rapport à des contraintes. Les résultats de l'analyse permettent de répondre à la question et de faire un choix justifié. Ces résultats sont enregistrés et une traçabilité bidirectionnelle entre les résultats d'analyse et la donnée d'ingénierie objet de l'analyse doit être maintenue (il s'agit de la sortie vers le processus Control de la Figure 14). La norme ISO 15288 précise à nouveau que cette traçabilité peut être facilitée par un répertoire de données approprié, sans plus de détails.

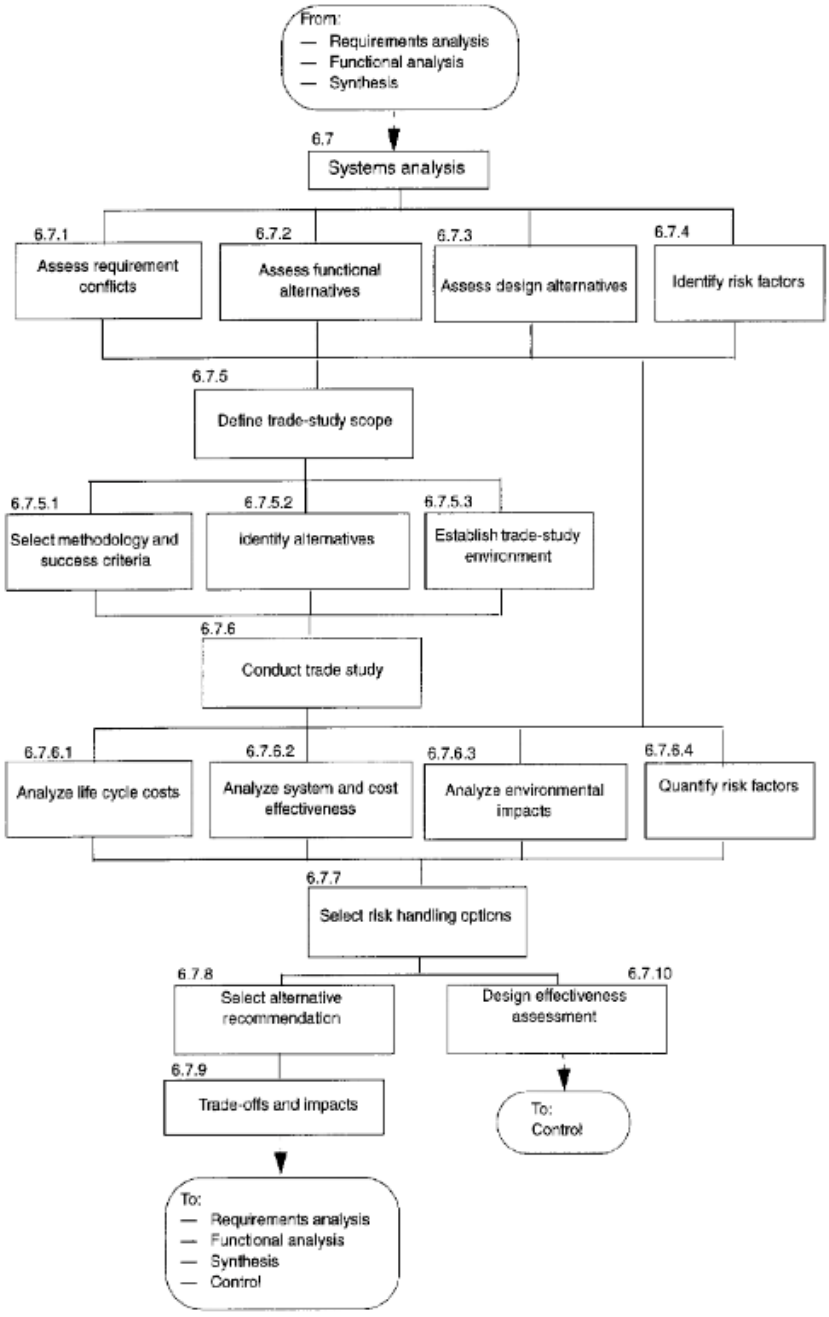



Figure 14: processus d'analyse système de IEEE 1220

## I.2.5. Management technique

### I.2.5.1. Objectif

L'objectif de ces processus de management technique est de documenter les activités des processus techniques, avec une gestion des données d'ingénierie et un contrôle des changements sur ces données. La norme ISO 15288 place tous ces processus dans une catégorie distincte des processus techniques (voir la cartographie de la Figure 9), ce qui rend l'articulation entre les processus difficile à comprendre. Le résultat est un répertoire intégré de données.

### I.2.5.2. Activités prescrites

Les activités prescrites sont relatives à la gestion des données, la gestion de configuration, la gestion des interfaces, la gestion des risques et la mesure de l'avancement. Comme le fait apparaître le diagramme des activités de la Figure 15, ces activités sont centrées sur la surveillance de données, dont le changement est soumis à approbation et entraine une mise à jour ordonnée de référentiels et de documents (plans d'ingénierie et plans techniques). Il s'agit de formaliser les exigences, l'architecture fonctionnelle et l'architecture organique dans des référentiels gérés en configuration. Ces documents vont servir de base contractuelle avec le fournisseur. Toute demande de changement ou d'écart est soumis à une procédure stricte d'enregistrement et d'évaluation pour approbation. Les changements approuvés sont traités dans les processus techniques. Alors que la description des activités est très précise sur la gestion de configuration et des changements (trois pages pour iso 15288), la description des activités est succincte sur la gestion de l'information, sur la nature des informations et leur forme. Les activités de gestion des données (ou de l'information pour ISO 15288) visent à mettre en place les répertoires appropriés et les procédures pour enregistrer et mettre à disposition des différentes parties prenantes les informations pertinentes issues des autres processus, mais ces informations ne sont pas clairement listées. De plus, ces activités incluent de rédiger, illustrer et transformer les données en « information utilisable pour les parties prenantes » (p43 - ISO 15288), sans d'autres précisions. Cependant, les informations fournies incluent la documentation officielle pour toute certification ou accréditation.

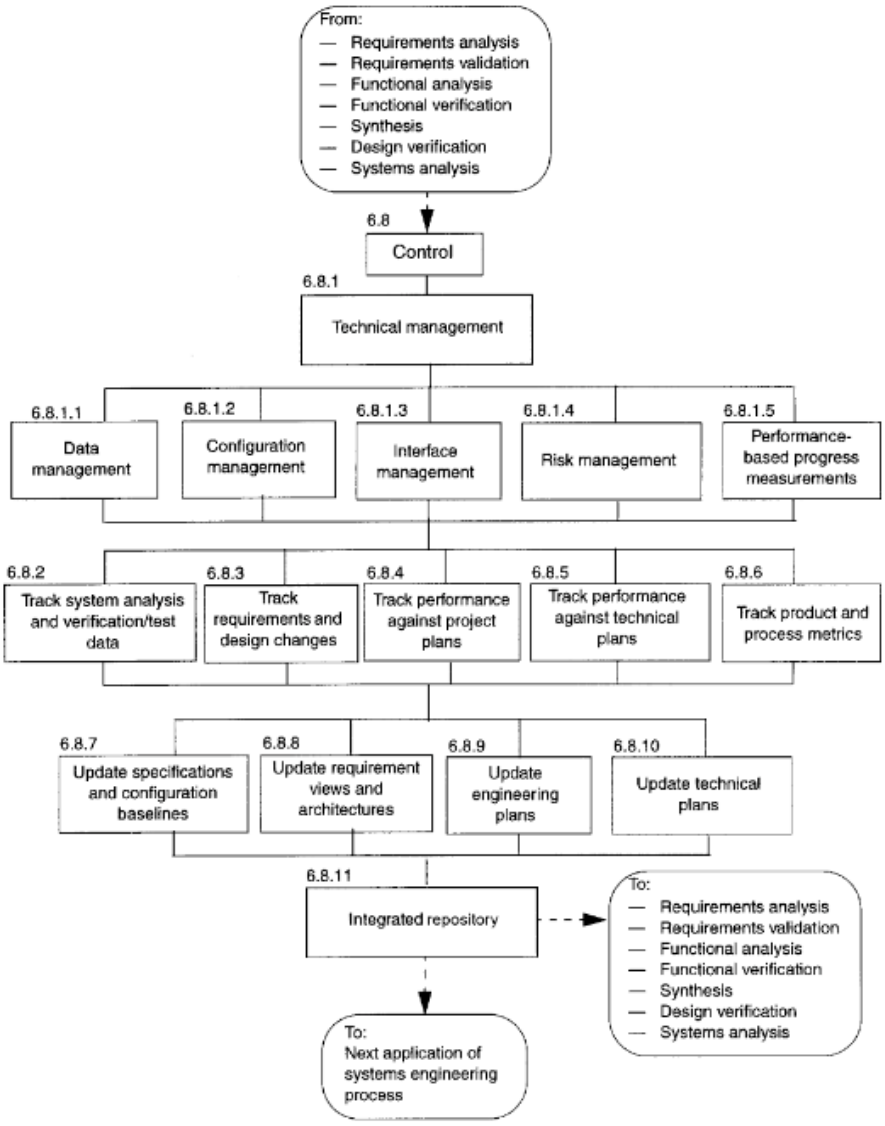



Figure 15: le processus de management technique de IEEE 1220

## I.2.6. Synthèse

L'application des processus permet des transformations progressives des données d'ingénierie, que nous nommerons artefacts d'ingénierie dans la suite de la thèse : besoins, exigences, fonctions, interfaces, éléments et liens. L'objectif principal est de réduire les ruptures entre la demande et la solution, afin de produire une solution vérifiée et justifiée répondant à la demande. On peut ainsi renverser le schéma proposé par IEEE 1220 (Figure 8), en l'orientant « résultats des processus », montrant la transformation successive du contexte du problème en besoins et scénarios, eux-mêmes transformés en exigences et contraintes, puis en fonctions et interfaces, et finalement en éléments et liens (Faisandier 2014).

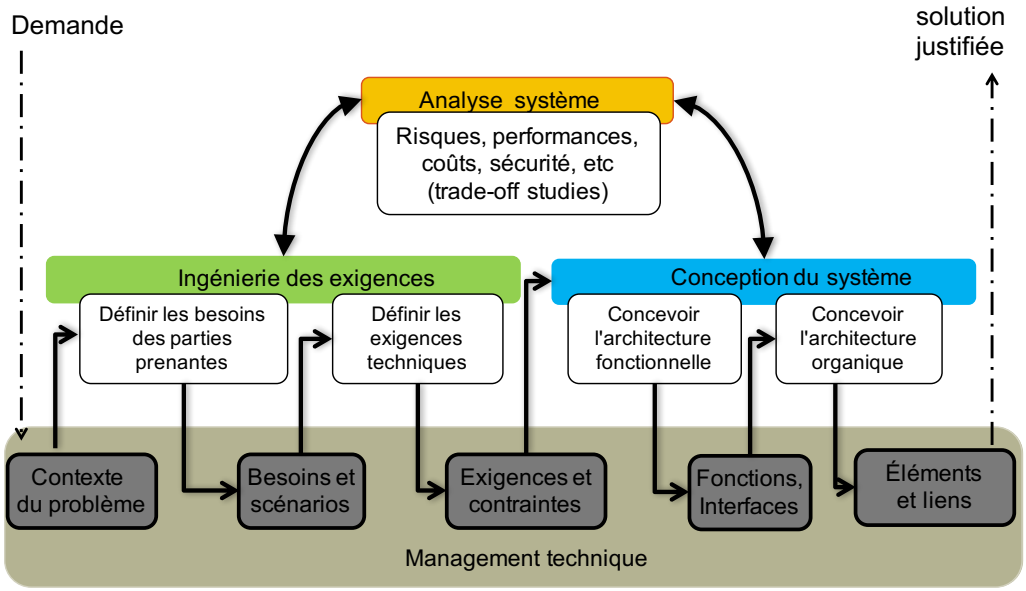



Figure 16: une perspective orientée résultats des processus IS

## I.3. Processus d'ingénierie système et cycle de vie du système

Les processus que nous venons de présenter peuvent être appliqués dans les différentes étapes du cycle de vie du système, tel qu'il est proposé par la norme IEEE 1220 (Figure 17) : pendant l'étape de définition du système, pendant l'étape de réalisation du système, pendant l'étape d'utilisation du système.

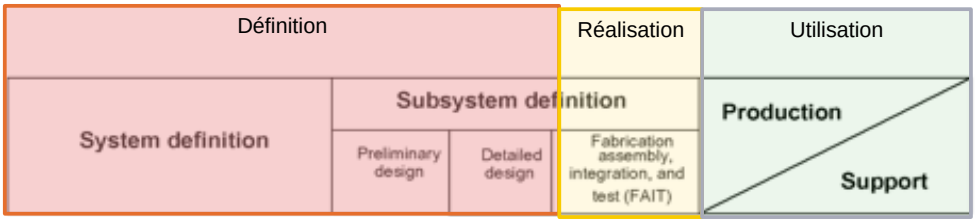



Figure 17: cycle de vie d'un système selon IEEE 1220, sur lequel nous identifions les étapes de définition, réalisation et utilisation du système

### I.3.1. Une application récursive pendant la définition du système

Pendant la définition du système, les normes précisent que les processus doivent être menés de façon récursive, en s'appliquant à chaque niveau de décomposition du système (selon la Figure 3), jusqu'à permettre le développement des éléments du système. Ainsi, les processus d'ingénierie des exigences au niveau système résultent en un référentiel d'exigences, qui est en entrée de la conception du système dans lequel les exigences sont décomposées et allouées au niveau inférieur aux sous-systèmes et aux interfaces entre sous-systèmes. C'est l'étape de définition système de la Figure 17. L'allocation d'exigences à un sous-système initie une application des processus d'ingénierie au niveau inférieur (étape de design préliminaire), jusqu'à atteindre une spécification des éléments du système (detailed design).

## I.3.2. Une application sur les anomalies détectées pendant la réalisation du système

Les processus de réalisation (fabrication, assemblage, intégration et tests) peuvent avoir lieu à partir du moment où les éléments du système et les interfaces entre éléments sont spécifiés (Figure 17). La norme ISO 15288 incluent les processus de réalisation dans sa cartographie des processus d'ingénierie système (Figure 9), sous les termes d'implémentation, intégration, vérification, transition et validation. Les exigences sont au cœur des processus de réalisation : ce sont les exigences des éléments du système qui sont en entrée de l'implémentation. De plus, la stratégie d'intégration des éléments réalisée est orientée par les priorités des exigences système. La qualification du système consiste à produire une couverture de tests par rapport aux exigences système et à tester la conformité de l'implémentation à ces exigences système. Le processus de vérification fournit une preuve que le système ou un élément du système remplit les exigences spécifiées. Il permet d'identifier les anomalies et les informations nécessaires à leur résolution. Le processus de validation fournit quant à lui une preuve que le système en utilisation remplit les exigences des parties prenantes (stakeholder requirements) dans les conditions opérationnelles attendues. La validation est ratifiée par les parties prenantes.

Les processus d'ingénierie système sont déroulées pour la correction d'anomalies détectées lors des processus de vérification et de validation : dans la Figure 16, la *demande* est constituée des anomalies détectées lors des tests. Nous notons que le maintien de la traçabilité apparaît également dans les processus de réalisation : il s'agit de maintenir le lien entre les éléments intégrés et vérifiés et les exigences correspondantes.

## I.3.3. Une application sur les évolutions demandées et les anomalies pendant l'utilisation du système

La norme IEEE 1200 présente les activités à mener lors de la phase d'utilisation du système dans la Figure 18, en distinguant Production et Support. La même distinction est faite entre deux processus, Opération et Maintenance dans la norme ISO 15288.

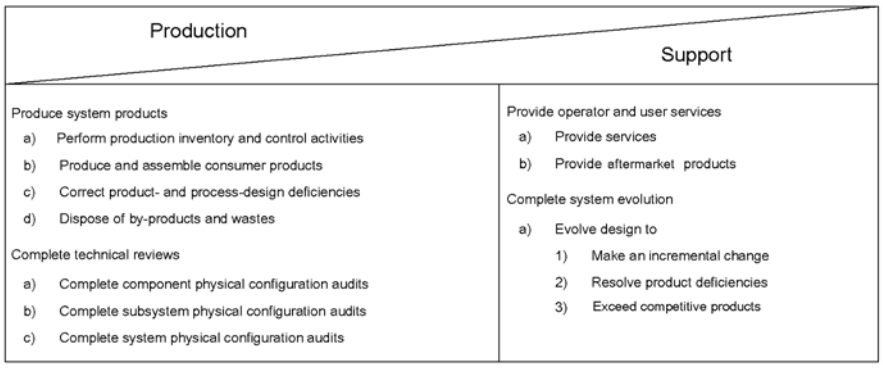



Figure 18: activités à mener lors des phases d'utilisation du système

Dans ces phases d'opération et de maintenance, les processus d'ingénierie système vont s'appliquer pour corriger les anomalies (maintenance corrective pour ISO 15288), pour faire un changement incrémental du système, ou pour faire des évolutions (maintenance évolutive – adaptive ou perfective pour ISO 15288). La traçabilité entre les éléments de maintenance et les éléments du système est également exigée.

# I.4. Problématique de recherche

La notion d'exigence est structurante pour la définition de système et constitue la brique de base de la communication entre les différentes parties prenantes du système : client, utilisateurs, ingénieur système, fournisseurs. Par conséquent, l'ingénierie des exigences est un enjeu crucial, que ce soit dans l'étape de définition de système, de réalisation du système, ou d'utilisation.

Cet enjeu est encore plus crucial pour les systèmes critiques, tels que les systèmes aéronautiques, car le processus de certification est basé sur la démonstration de la conformité du système avec les exigences spécifiées. Le maintien de la traçabilité entre les données d'ingénierie participe à garantir un niveau de sécurité du système (RTCA SC-167 et WG-12 EUROCAE 1992) (Kennedy et Towhidnejad 2017).

Cependant, les normes d'ingénierie système insistent sur le fait que les processus techniques doivent être menés de façon *itérative* pour affiner progressivement les résultats des processus. Ainsi, nous relevons que les normes demandent à la fois de la souplesse dans l'application itérative des processus, et une extrême rigueur dans la gestion des résultats de ces mêmes processus, en termes de traçabilité et de contrôle des changements.

De plus, l'application *récursive* des processus, c'est-à-dire à chaque niveau de décomposition du système, entraine une multiplication des exigences, chaque exigence d'un niveau n étant décomposée en plusieurs exigences pour le niveau n+1. En considérant un système complexe, constitué d'une multitude de composants hétérogènes, le nombre final d'exigences explose. Des travaux récents confirment que la gestion des exigences est un des enjeux posés par le développement des méthodes agiles dans les systèmes critiques (Wils et al. 2006)(Doss et Kelly 2016)(Hanssen, Wedzinga, et Stuip 2017).

Enfin, face aux incertitudes de l'environnement, malgré tous les efforts de spécification des exigences, il existe des « accidents système », c'est-à-dire un accident arrive alors que le système fonctionne tel qu'il a été spécifié, mais de façon inapproprié à un environnement particulier (Leveson 2002).

Notre question est de savoir comment l'ingénieur fait face :

(1) à cette double nécessité de souplesse et de rigueur dans l'application des processus d'ingénierie des exigences,

(2) au volume d'exigences et de liens entre éléments du système et entre artefacts d'ingénierie,

(3) au risque d'un comportement inapproprié du système dans un environnement particulier.

# Chapitre II. Etat de l'art

*' Tout ce que je sais, c'est que je ne sais rien.'*
Socrate

L'enjeu de l'ingénierie des exigences est d'atteindre une spécification complète et sans ambiguïté avec un double objectif : rendre compte des besoins du client et des utilisateurs, et servir de cadre contractuel avec les fournisseurs pour le développement des composants du système répondant à ces besoins. L'ingénierie des exigences étant un domaine de recherche appliqué, nous pouvons discerner deux contextes dans les travaux : un contexte académique et un contexte industriel. La méthodologie de recherche utilisée peut être corrélée au contexte. Ainsi, les méthodologies utilisées dans un contexte industriel relèvent d'une approche empirique à base d'études de terrain et d'études de cas. Des méthodologies à base d'expérimentation, de revue systématique de littérature et d'état de l'art sont utilisées en contexte académique (Svahnberg et al. 2015).

Nous présentons notre état de l'art selon ces deux perspectives (1) les études de terrain reportant les pratiques industrielles, (2) les travaux académiques sur l'utilisabilité et les visualisations en ingénierie des exigences. Nous discutons en synthèse le lien entre les deux perspectives et les opportunités de recherche.

# II.1. Introduction à l'ingénierie des exigences

## II.1.1. Les problèmes essentiels et le processus d'ingénierie des exigences

Les travaux précurseurs en ingénierie des exigences (Pohl 1994) (Sawyer, Sommerville, et Viller 1997) pointent les problèmes *par essence* de l'ingénierie des systèmes, dans la lignée de Brooks (Brooks 1987):

- La difficulté de découvrir les exigences : le client n'a pas une vue claire de ce dont il a besoin. Les besoins naissent de l'interaction entre les parties prenantes (client et utilisateurs) et l'ingénieur. La multiplicité des parties prenantes peut entrainer des conflits sur les exigences (*conformity* chez Brooks)
- La nature changeante des exigences du fait d'un environnement évolutif en termes de métiers, de réglementation et de compétition (*changeability* chez Brooks)
- La difficulté de représenter un système logiciel, qui n'a pas d'existence physique dans l'espace (*invisibility*).

Ils déduisent de ces problèmes essentiels trois axes structurants pour l'ingénierie des exigences, qui sont nommés *specification*, *agreement* et *representation* par Pohl (Pohl 1994), et *requirements document*, *requirement problems* et *draft statement of requirements* par Sawyer (Sawyer, Sommerville, et Viller 1997).

Le processus d'ingénierie des exigences est représenté comme une progression itérative selon ces trois axes, partant d'une vision opaque, conflictuelle et informelle des besoins pour atteindre une spécification complète, commune et formelle des exigences (voir Figure 19).

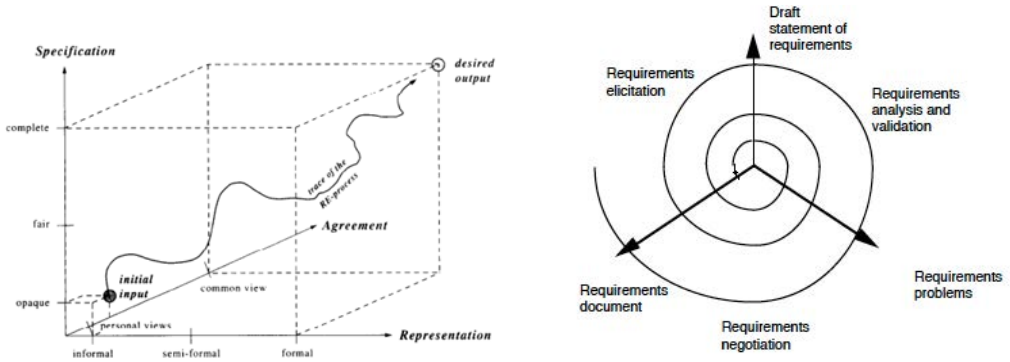



Figure 19: le processus d'ingénierie des exigences selon (Pohl 1994) à gauche, et (Sawyer, Sommerville, et Viller 1997) à droite

Au sein du processus d'ingénierie des exigences, trois à cinq types d'activités à mener sont identifiées, selon les groupements réalisés (Axel van Lamsweerde 2000) (Aurum et Wohlin 2005)(Cheng et Atlee 2007): élicitation, modélisation, analyse, vérification et validation, management.

Les activités d'élicitation des exigences concernent l'identification des parties prenantes, et la collecte auprès des parties prenantes des problèmes et défauts de l'existant, ainsi que les opportunités. Des premiers modèles du système sous forme de scénarios ou de prototypes peuvent être élaborés comme support à la discussion avec les parties prenantes. Les buts et objectifs de haut niveau du système à venir sont explicités, ainsi que les frontières entre le système et l'environnement

Les activités de modélisation et analyse concernent l'élaboration d'artefacts qui illustrent l'atteinte par le système des buts identifiés. Les modèles initiaux du système tels que des scénarios, des cas d'utilisation, des descriptions d'interactions avec le système sont précisés. Ces artefacts vont être soumis à l'analyse des parties prenantes afin d'être formulés de façon plus précise et classés par ordre de priorité. Une analyse de risques par les ingénieurs est également réalisée donnant lieu à des compromis sur le contenu des exigences ou sur leur priorité.

Les activités de vérification concernent la recherche et la correction d'inexactitude, d'incohérence, ou d'incomplétude. Les activités de validation permettent de s'assurer que le résultat permet de satisfaire les buts attendus par les parties prenantes.

Les activités de management concernent la documentation des différentes décisions prises pendant le processus, avec leur justification et hypothèses, la gestion des exigences, la traçabilité entre les exigences.

En résumé, la modélisation apparaît au centre du processus d'ingénierie des exigences (Axel van Lamsweerde 2000) : d'une part les modèles sont le résultat des activités d'élicitation, d'analyse et de négociation, et d'autre part les modèles orientent les activités ultérieures d'élicitation, analyse et négociation. Les modèles fournissent également la base de la documentation et des évolutions. La nature de ces modèles est multiple, les exigences pouvant être exprimées sous plusieurs formes : description textuelle, scénarios et cas d'utilisation, modèles basées sur les buts, machines à état et de graphes de flux, description du domaine, ou équations à base de langage formel (Cheng et Atlee 2007).

## II.1.2. Les différentes expressions des exigences

Nous présentons les différentes expressions des exigences développées par la recherche académique en ingénierie des exigences, en précisant le principe, un exemple, et les avantages et limites identifiés.

### II.1.2.1. Une description textuelle des exigences

#### II.1.2.1.1. Principes et exemples

La norme ISO 29148:2011 spécifique à l'ingénierie des exigences (ISO/IEC/IEEE 2011) précise les caractéristiques qu'une exigence bien exprimée doit remplir. Elle doit être :

- Identifiable : l'exigence possède une référence unique et identifiable.
- Utile : l'exigence a une valeur technique.
- Concise : la phrase est courte et simple à appréhender.
- Complète : l'exigence contient toutes les informations nécessaires à sa compréhension.
- Non ambiguë : il n'existe qu'une seule interprétation possible de l'exigence.
- Simple : l'exigence est facile à comprendre : sa rédaction est simple et les mots utilisés sont courants.
- Communicable : les destinataires peuvent la comprendre à l'aide de leurs connaissances.
- Unique : l'exigence n'exprime qu'une seule idée.
- Justifiable : l'exigence a une origine et une raison d'être.
- Vérifiable : l'exigence peut être vérifiée.
- Correctement rédigée : l'exigence respecte les règles de grammaire et les formalismes imposés.

Ainsi, pour exprimer une exigence bien formulée en langage naturel, il est recommandé d'adopter des règles de syntaxe telles que [Condition] [Sujet] [Verbe] [Complément] [Contrainte], où le sujet est le système d'intérêt (Figure 20) (ISO/IEC/IEEE 2011).

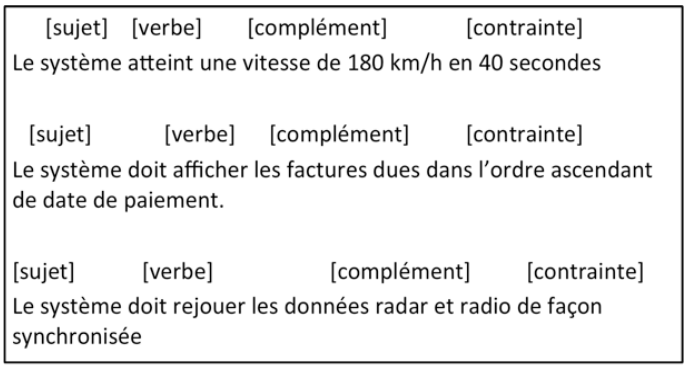



Figure 20: exemples d'exigences textuelles selon la norme ISO 29148

Une exigence possède des attributs tels qu'un identifiant unique, une priorité, la partie prenante intéressée par l'exigence (source), la justification de son existence. Elle possède aussi des attributs de gestion (date de création, date de dernière modification, auteur, version du système implémentant l'exigence).

La norme ISO 29148:2011 définit également des bonnes propriétés pour le référentiel des exigences :

- Clair : Les exigences sont aisément identifiables.
- Non redondant : Une exigence est située à un seul endroit.
- Maîtrisé : La version et la configuration de l'ensemble sont identifiables et maîtrisées.
- Complet : L'ensemble synthétise le point de vue de toutes les parties prenantes identifiées.
- Exhaustif : Le référentiel couvre l'ensemble des besoins, des attentes et des contraintes de chaque partie prenante.
- Homogène : Les exigences exprimées sont de même granularité.
- Cohérent : Les exigences ne sont pas contradictoires entre elles.

L'objectif de ces règles est de fournir un cadre pour la vérification d'une exigence et du référentiel des exigences.

#### II.1.2.1.2. Avantages et limites du langage naturel

Utiliser le langage naturel pour exprimer les exigences comportent plusieurs avantages. Premièrement, tout le monde peut écrire une exigence, il n'y a pas besoin de maitriser une notation particulière. Deuxièmement, les exigences sont compréhensibles par les parties prenantes non techniques, tels que le client et les utilisateurs du système, ce qui permet de faciliter la lecture et la validation des exigences par ces parties prenantes (D. M. Berry et Kamsties 2004) (Cheng et Atlee 2007). Enfin, le caractère informel et ambigu du langage peut représenter un avantage pour l'ingénierie des exigences, en facilitant une évolution graduelle des exigences, sans forcer à résoudre trop tôt les conflits et les ambiguïtés qui se posent au début du travail de spécification. Le langage naturel peut même permettre une résolution diplomatique de conflits à travers le choix pesé des mots et une construction délibérément ambiguë (Goguen 1994).

Cependant, l'ambiguïté des exigences peut conduire à plusieurs interprétations possibles d'un même texte par les développeurs du système, et donc fait courir le risque de réaliser un système ne

répondant pas aux besoins du client et des utilisateurs. L'utilisation de règles syntaxiques telles que celles énoncées par la norme ISO 29148 est une façon d'augmenter la précision du langage naturel, en le contraignant. De plus, il existe des guides de bonnes pratiques pour la rédaction d'exigences, faits par INCOSE (Fuentes et al. 2016) ou par des industriels (US Air Force 2009), allant jusqu'à la définition de langage naturel contrôlé (Condamines et Warnier 2017). L'augmentation de l'exigence avec des informations complémentaires, telles que la justification (ou rationale), la source (partie prenante concernée), le test ou le critère de satisfaction est une façon de diminuer l'ambiguïté d'une exigence (Ian Sommerville et Sawyer 1997) (S. Robertson et Robertson 2006). La mise en place d'un glossaire ou modèle du domaine permet de désigner les termes pertinents de l'environnement et raffiner la signification des exigences (Zave et Jackson 1997). Ce glossaire est construit à partir du langage utilisé par le client et les utilisateurs.

L'information de traçabilité entre les exigences peut également aider à lever une ambiguïté sur une exigence (D. M. Berry et Kamsties 2004). Ainsi, en établissant un lien de dépendance entre deux exigences e1 et e2, on indique que l'exigence e2 dépend de l'existence de l'exigence e1. La sémantique des liens entre exigences peut être de différentes catégories (Spanoudakis et Zisman 2004) : dépendance, raffinement, évolution, satisfaisabilité, recouvrement, conflit, rationalisation.

La contextualisation pour guider l'interprétation d'une exigence par rapport à d'autres exigences peut aller jusqu'à un ordre de lecture des exigences, l'ordre de narration montrant une séquence temporelle des exigences (Goguen 1994). Cependant, la sémantique du temps en termes de simultanéité et de point de synchronisation entre exigences reste difficile à exprimer avec du texte seulement, d'où les propositions de l'analyse structurée.

## II.1.2.2. L'analyse structurée (ou les analyses structurées)

### II.1.2.2.1. Principes et exemples

Les objectifs de l'analyse structurée (ou SADT pour Structured Analysis and Design Technique) sont de fournir des méthodes pour penser de façon structurée les problèmes complexes mais aussi pour travailler en équipe avec une division efficace du travail et des rôles pour l'effort de coordination, communiquer les résultats des interviews, analyses et conception avec une notation précise et claire, documenter les résultats et les décisions, contrôler la précision et la complétude à travers des revues fréquentes et des procédures d'approbation, et enfin planifier et évaluer le travail de l'équipe (Ross et Schoman 1977). L'analyse structurée combine un langage graphique et du texte, avec des noms et des verbes, pour proposer une description hiérarchique et graduelle sous la forme d'un modèle. La description est exprimée dans une décomposition de données et une décomposition d'activités (voir Figure 22), les deux étant constitués des mêmes éléments de construction graphiques : la boîte, pour représenter une partie, et les flèches représentant des entrées, des sorties, des contrôles et mécanismes entre les différentes parties. L'analyse structurée propose trois vues : l'analyse du contexte (répondant à la question Pourquoi), la spécification fonctionnelle (répondant la question Quoi), et les contraintes sur la solution (répondant à la question Comment). Le processus est de décomposer selon ces trois questions en descendant dans le niveau de détails. Un modèle SADT est une séquence organisée de diagrammes, chacun étant accompagné d'un texte concis. Un diagramme de haut niveau représente le système complet, et chaque diagramme de plus bas niveau montre une quantité limitée de détails sur un sujet bien contraint. Chaque diagramme est connecté à des portions de plus haut niveau du modèle afin de préserver les relations logiques de chaque composant dans le système entier (voir Figure 21).

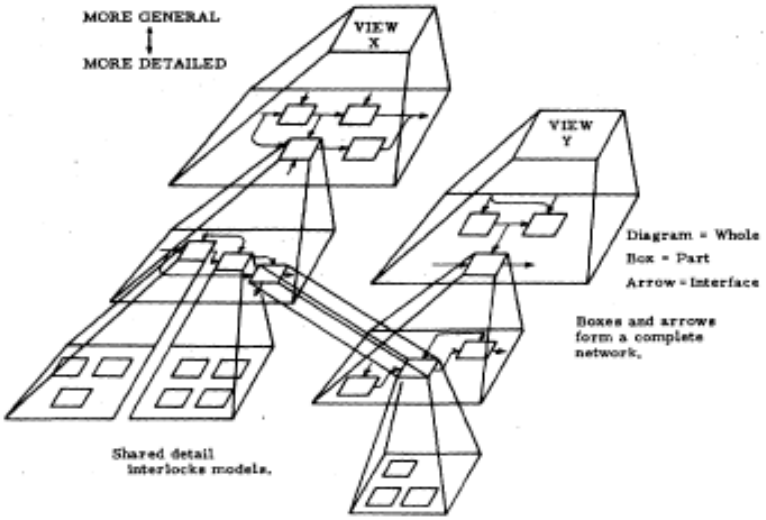



Figure 21: décomposition hiérarchique selon l'analyse structurée, issue de (Ross et Schoman 1977)

Les motivations de Ross sont centrées sur la communication des idées (Ross 1977) : la décomposition permet des vues adaptées aux modes de pensée et de compréhension de différentes audiences, que ce soit des personnes non-techniques telles que les managers ou les utilisateurs du système, ou des personnes techniques telles que les développeurs des composants du système. La maxime est:

> *Everything worth saying about anything worth saying something about must be expressed in six or fewer pieces."p18- (Ross 1977)*

Ainsi, la finalité du modèle est déterminée par rapport à son audience et à ce qui est intéressant à décrire, chaque niveau de décomposition devant être compréhensible. Le chiffre de 6 est justifié par la limitation cognitive de traitement de l'information de l'être humain (Miller 1956).

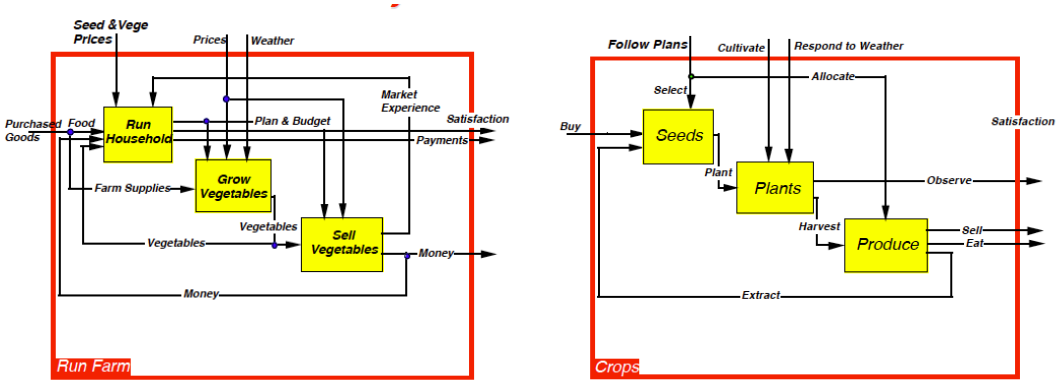



Figure 22: un diagramme activités et un diagramme données SADT

Ross reprend les principes de SADT pour développer la méthode de modélisation fonctionnelle IDEF0 sur demande de l'US Air Force qui souhaite améliorer la productivité des personnes dans l'analyse et la communication. (« Information technology – Modeling Languages–Part 1: Syntax and Semantics for IDEF0 » 2012).

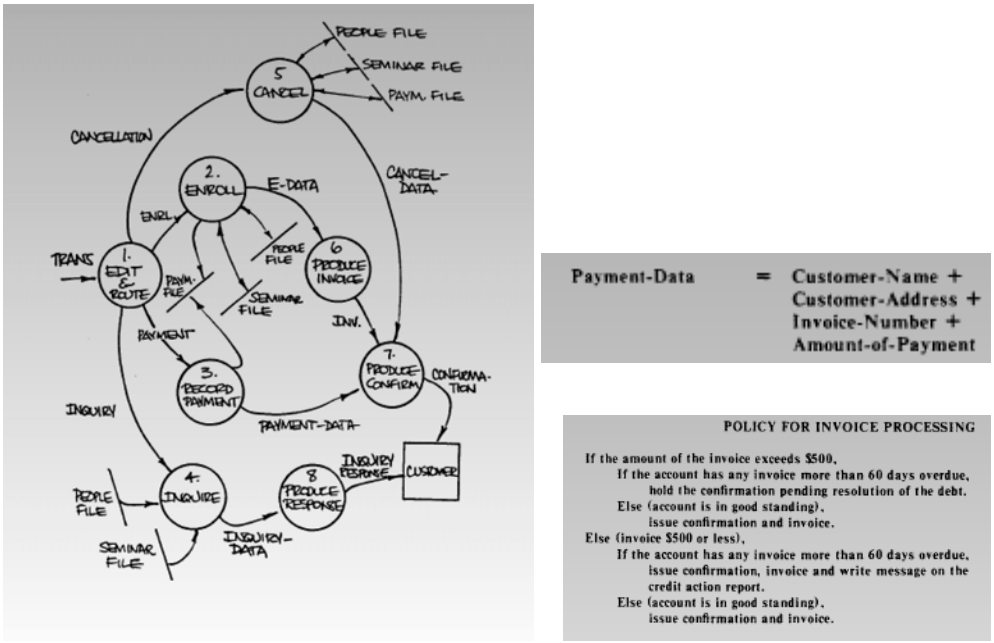



Figure 23: diagramme de flot de données, extrait du dictionnaire de données et mini-spécifications issus de (DeMarco 1979)

L'approche de DeMarco (DeMarco 1979) est similaire en termes de structuration hiérarchique, mais met davantage l'emphase sur la maintenabilité de la spécification dans le temps (la capacité de pouvoir la comprendre et la mettre à jour) : il promeut les diagrammes de flot de données (dataflow diagram) pour partitionner la spécification, un dictionnaire de données pour décrire les interfaces avec l'environnement, et des mini-spécifications formelles au plus bas niveau de la décomposition pour éviter le texte narratif (sous forme de table, d'arbre de décision ou de texte structuré- voir Figure 23). Un cercle représente un processus qui utilise une donnée et produit une nouvelle donnée en résultat. Le processus contient du texte qui suggère comment la donnée est transformée, et doit être numéroté. Dans les diagrammes de flot de données, la décomposition est réalisée par un raffinement des processus. Le comportement du système est modélisé en détaillant chaque processus en un diagramme de flot de données, en gérant des niveaux d'abstraction à partir de la numérotation des processus. Cela permet d'éviter un diagramme surchargé sur lequel tout serait à plat (voir Figure 24).

Yourdon propose une méthode structurée pour la modélisation des exigences, à partir de l'approche de DeMarco basée sur les diagrammes de flot de données, en ajoutant l'expression de comportements temps-réel, avec les diagrammes état-transition (Yourdon 1989). Il préconise d'utiliser des rectangles pour représenter les états, plutôt que des cercles, afin d'éviter la confusion avec les diagrammes de flot de données. (Yourdon 2006) (voir Figure 25).

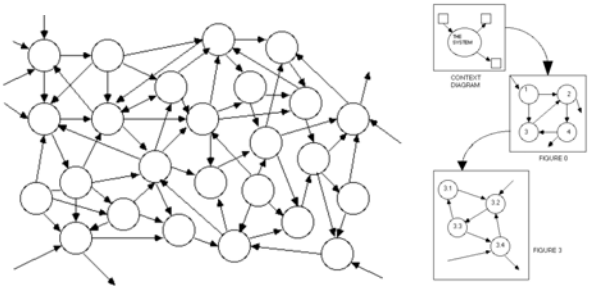


Figure 24: les niveaux d'abstraction des diagrammes de flot de données (à droite) permet d'éviter un diagramme surchargé (à gauche) (Yourdon 2006)

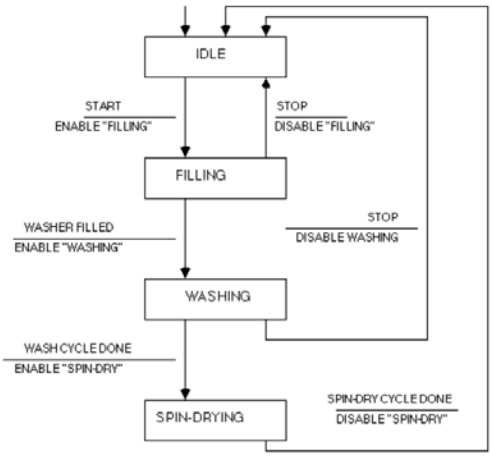



Figure 25: diagramme état-transition issu de (Yourdon 2006)

Dans la plupart des cas, un diagramme état-transition représente la spécification d'un processus (bulle) d'un diagramme de flot de données (voir Figure 26) : les événements de réalisation du diagramme état-transition sont fixées par les flux entrants du processus (X et Y), et les actions du diagramme état-transition correspondant aux flux de sortie du processus. Les liens sont réalisés à travers le respect des noms (X, Y, bubble 2 et bubble 3 sur la Figure 26).

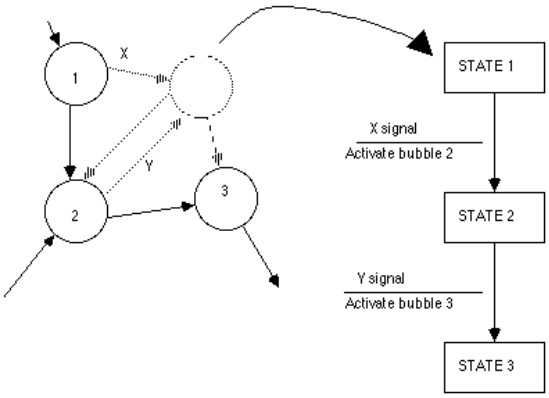



Figure 26: liens entre diagrammes de flot de données et état-transition issus de (Yourdon 2006)

### II.1.2.2.2. Avantages et limites

L'avantage des analyses structurées pour la modélisation des exigences est de proposer une notation graphique simple permettant de se dégager du style « victorien » des documents de spécification (DeMarco 2002). Cependant, certains analysent la simplicité des diagrammes comme une faiblesse (Woodman 1988) : la simplicité suggère un manque de pouvoir d'expression qui encourage des extensions de la notation gênant l'interprétation des diagrammes et leur traitement automatisé.

D'autre part, comme relevé par Yourdon (2006), la révision et maintenance des modèles graphiques posent problème. Créer les diagrammes prend du temps avant d'atteindre un accord avec le client ou l'utilisateur, tout en apportant une satisfaction en tant qu'activité intellectuelle et créative. Mais les modifications sont coûteuses en temps, avec de nombreux diagrammes à redessiner. Les analystes système ne prennent pas le temps de faire ces modifications, ou s'arrêtent à un niveau de décomposition. La conséquence est une spécification obsolète ou erronée. De plus, alors que la justification des techniques d'analyse structurée est de pouvoir exprimer des exigences sur des systèmes complexes, en prônant une décomposition systématique en niveaux d'abstraction, il n'y a

pas de continuité assurée avec le développement logiciel : la conception structurée proposée par Yourdon et Constantine (Yourdon et Constantine 1979) est adaptée à des programmes élémentaires fonctionnant sur un seul ordinateur. Le transfert des spécifications formulées avec de l'analyse structurée vers les développeurs reste compliqué.

Enfin, les utilisateurs et client du système peuvent avoir du mal à travailler avec les diagrammes des analyses structurées, avec des discussions abstraites sur les fonctions et les données, notamment pour les systèmes interactifs (Yourdon 2006) ou réactifs (Harel 1987), avec des caractéristiques orthogonales aux fonctions et aux données.

### II.1.2.3. Scénarios et cas d'utilisation

#### II.1.2.3.1. Principes et exemples

Le scénario est une histoire concrète d'utilisation du système (voir des exemples en Figure 27). Plutôt que de lister des exigences et des fonctions, l'idée est de se concentrer sur les activités que les utilisateurs réalisent avec le système, et de les décrire en racontant une histoire (Carroll 2000).

Les cas d'utilisation, promus par l'approche orientée objet (Jacobson 1992) et le processus unifié de développement (Jacobson, Booch, et Rumbaugh 1999), rejoignent cette perspective orientée utilisateurs du système: un cas d'utilisation est une séquence d'actions que le système réalise et qui conduit à un résultat observable ayant de la valeur pour un utilisateur particulier.

Les cas d'utilisation ont pour objectif d'être plus précis dans la description du système, notamment avec une structuration en préconditions, post-conditions et une exploration systématique d'alternatives à chaque étape. La cas d'utilisation sont synthétisés dans un diagramme de cas d'utilisation qui offre une vision globale du système (voir exemple en Figure 28) et permet d'organiser le développement du système par tranches (Jacobson, Spence, et Kerr 2016).

**A. Science Fiction Club in a Web forum**

After three years at Virginia Tech, Sharon has learned to take advantage of her free time in-between classes. In her hour between her morning classes, she stops by the computer lab to visit the science fiction club. She has been meaning to do this for a few days because she knows she'll miss the next meeting later this week. As she opens a Web browser , she realizes that this computer will not have her bookmarks stored, so she starts at the homepage of the Blacksburg Electronic Village. She sees local news and links to categories of community resources (businesses, town government, civic organizations). She selects "Organizations", and sees an alphabetical list of community groups. She is attracted by a new one, the Orchid Society, so she quickly examines their Web page before going back to select the Science Fiction Club page. When she gets to the club page, she sees that there are two new comments in the discussion on Asimov's *Robots and Empire*, one from Bill and one from Sara. She browses each comment in turn, then submits a reply to Bill's comment, arguing that he has the wrong date associated with discovery of the Zeroth Law.

**B. Science Fiction Club in a Community MOO**

After three years at Virginia Tech, Sharon has learned to take advantage of her free time in-between classes. In her hour between her morning classes, she stops by the computer lab to visit the science fiction club. She has been meaning to do this for a few days because she knows she'll miss the next meeting later this week. As she starts up the Blacksburg community MOO, she can see that the last person using this computer must have been interested in orchids, because the welcoming text describes her location as an orchid garden, along with Penny and Alicia, who are discussing some new exotic varieties. The text description mentions an exit to Main Street, so she leaves the garden and starts moving south. Along the street she runs into George, who is working on a banner for the fair. She gives him a quick hello, and continues southward until she sees an eastward exit will take her to Eastenders Pub; this is where the Science Fiction Club meets. She enters the room and is told that Bill and Sara are already there, along with a pitcher of Newcastle Brown. She can tell from their current comments that they have been discussing the timeline from Asimov's *Robots and Empire*.

**C. Science Fiction Club in a Collaborative Virtual Environment**

After three years at Virginia Tech, Sharon has learned to take advantage of her free time in-between classes. In her hour between her morning classes, she stops by the computer lab to visit the science fiction club. She has been meaning to do this for a few days because she knows she'll miss the next meeting later this week. When she tries to start up the online collaborative environment, she finds that this computer does not have the client, so she waits for a minute or two while it is automatically downloaded and installed. After she logs in, she is taken back to her previous visit location, and sees the familiar panoramic view of her livingroom, her to-do lists and sketchpad, and the interactive map of Blacksburg. She positions and zooms in on the map until she can see downtown buildings She enters the Eastenders Pub subspace, where the science fiction club usually meets. She sees a panoramic image of bar, faces that show Bill and Sara are here, a food and drink menu, and various standard tools. The map updates to show a floorplan of the Pub—the dining room, the darts room, the office, and the main bar. Bill and Sara are using a chat tool and a shared whiteboard to sketch an event timeline for Asimov's *Robots and Empire*. Joining Bill and Sara in the chat tool, she types "Based on the Zeroth Law, I'm afraid I must drink some of your beer".

Figure 27: Trois scénarios alternatifs d'une réunion de club en ligne pour une étudiante, extraits de (Rosson et Carroll 2009)

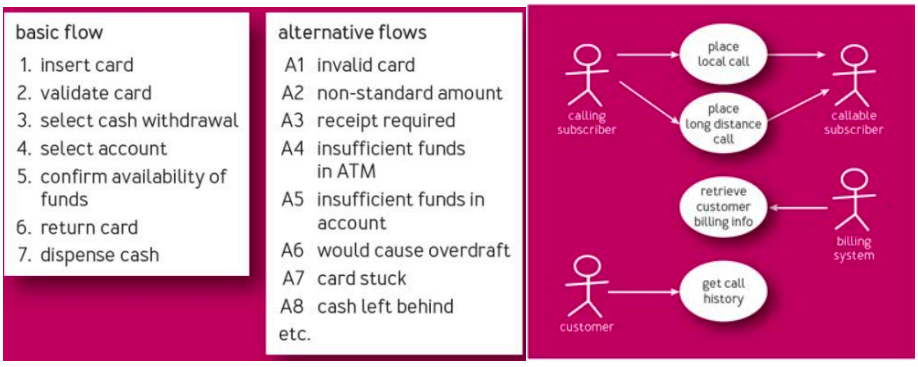



Figure 28: exemples de cas d'utilisation et de diagramme de cas d'utilisation, issus de (Jacobson, Spence, et Kerr 2016)

Enfin, l'approche orientée-objet est accompagnée par le développement de la notation UML (OMG 2015) qui reprend à son compte deux formalismes visuels, les diagrammes de séquence de message de Zave (Zave 1985) et les statecharts de Harel (Harel 1987), pour exprimer le comportement du système (voir Figure 29).

Les critères de lisibilité et de compréhension de la spécification sont un des moteurs des approches de Harel et Zave dans l'élaboration de leur formalisme visuel. La gestion de la complexité se fait par l'organisation de vues multiples sous un filtre commun pour Zave, chaque vue ayant son diagramme de séquence de message, et à l'aide de clusters pour Harel.

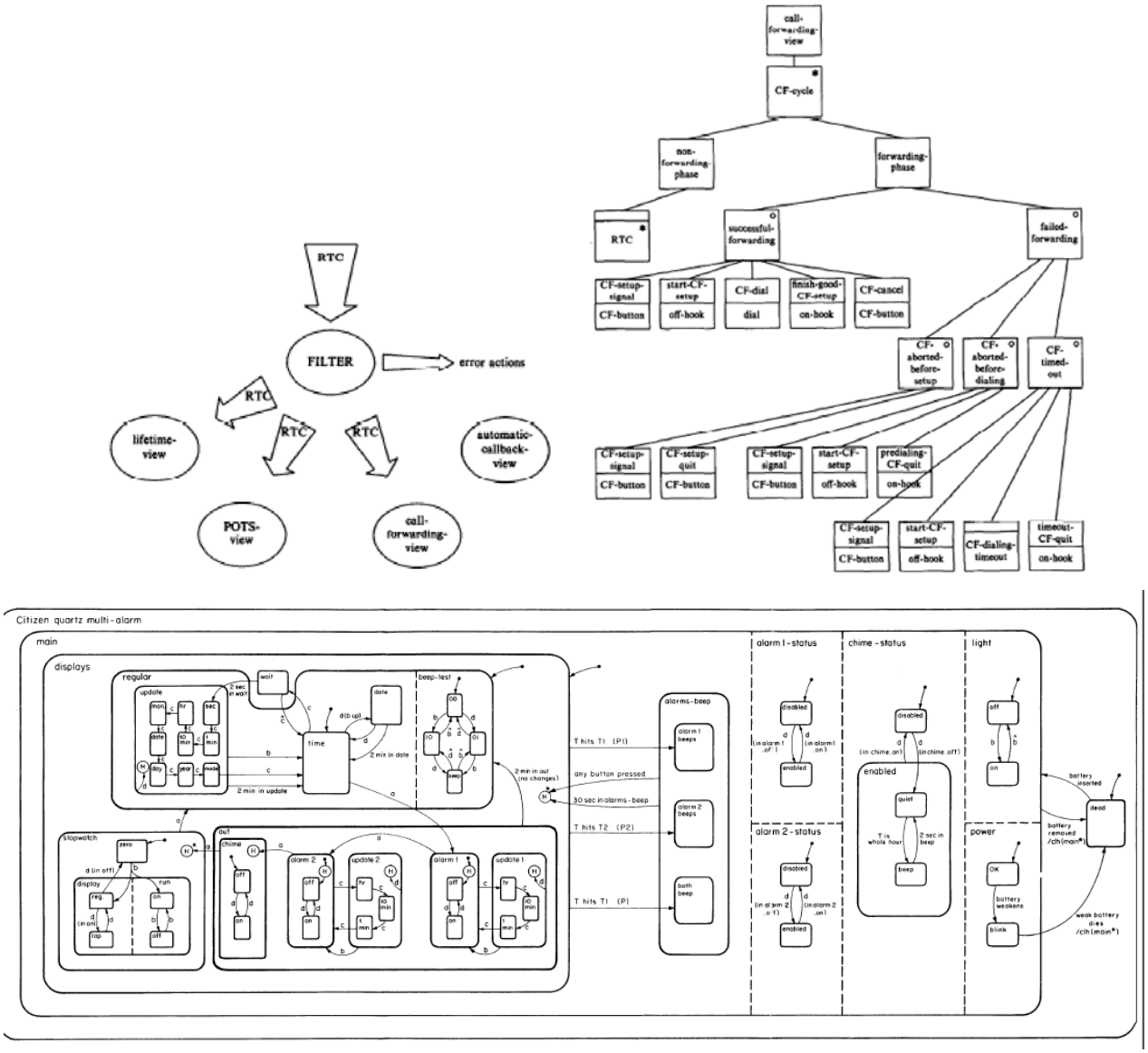



Figure 29: diagramme de séquence de message selon Zave (Zave 1985) et statechart selon Harel (Harel 1987) permettent de préciser le comportement du système

Les cas d'utilisation revendiquent d'être le chainon manquant entre les analystes et les développeurs : une modélisation des cas d'utilisation en diagrammes de séquence de message peut être transformée en diagrammes de classe représentant les objets et les services à implémenter grâce à des langages orientés-objet.

La notation de *Use Case Map* (Buhr 1998) propose de représenter des scénarios du système par le tracé de lignes à main levée représentant le chemin d'appel des composants (Figure 30), et faisant abstraction du détail des messages échangés entre composants.

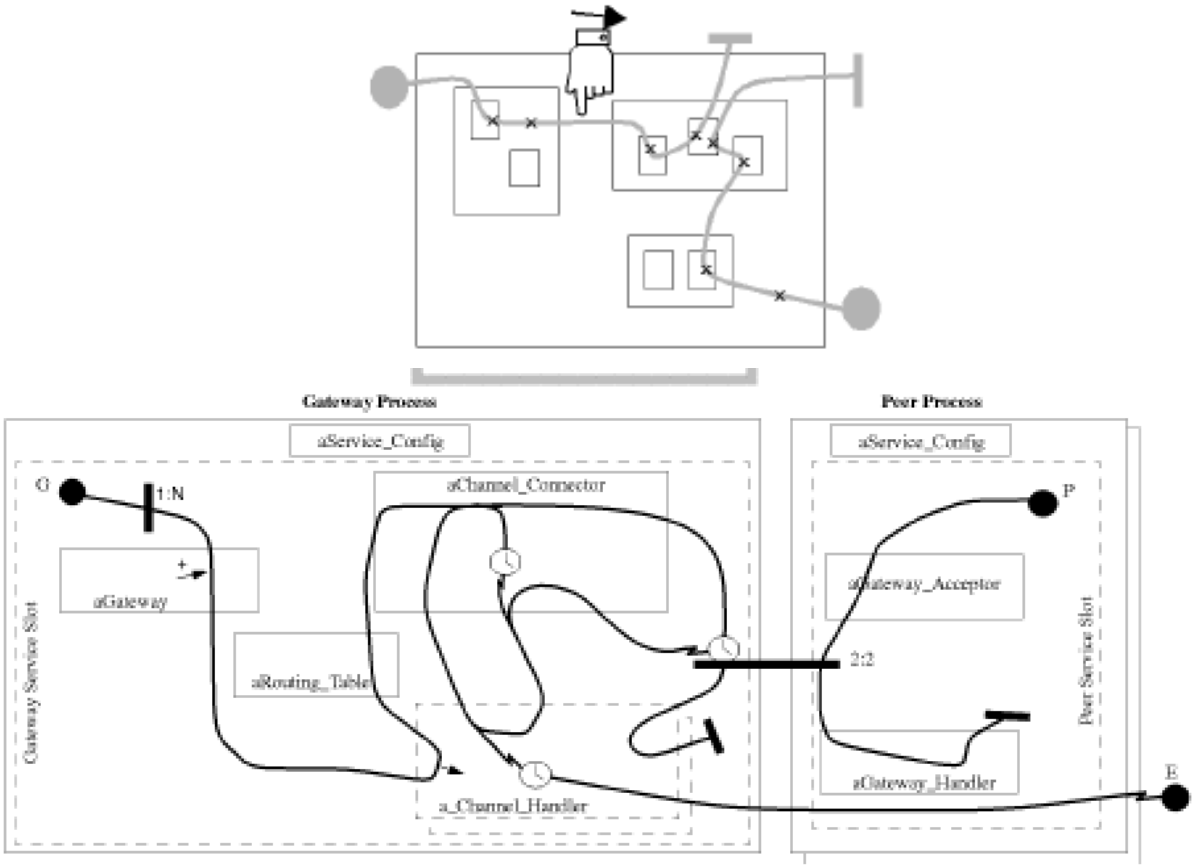



Figure 30: principe de la notation de Use Case Map et exemple (Buhr 1998)

Constantine et Lockwood (1999) proposent la notion de cas d'utilisation essentiels, qui est intermédiaire entre les scénarios et les cas d'utilisation. La description des cas d'utilisation essentiels est narrative et basée sur le vocabulaire du domaine, afin de permettre aux utilisateurs du système de les comprendre et les valider. La description est organisée en deux colonnes, l'une indiquant les intentions de l'utilisateur, l'autre la responsabilité du système (exemple en Figure 31).

| User Intention | System Responsibility |
|---|---|
| identify self | |
| | verify identity offer choices |
| choose | |
| | dispense cash |
| take cash | |

Figure 31: exemple simple d'un cas d'utilisation essentiel (Larry L. Constantine et Lockwood 1999)

### II.1.2.4. Avantages et limites

Les scénarios, en étant concrets et ancrés dans les problèmes des utilisateurs, agissent comme une « prothèse cognitive » : ils permettent de susciter le raisonnement et la créativité sur le système à concevoir (Carroll 2000). Les scénarios sont un bon outil pour impliquer les utilisateurs dans un processus itératif et participatif de conception, et pour aider les concepteurs à prendre du recul sur leur solution en usage (S. Bødker 2000).

En plus d'être concrets, les scénarios sont flexibles : ils sont délibérément incomplets et faciles à élaborer et modifier. Un autre avantage est de pouvoir intégrer les intentions de l'utilisateur dans la description textuelle, ainsi que des éléments de justification de choix issus de discussions avec les utilisateurs. Enfin, les scénarios peuvent être utilisés pour valider les exigences, en tant que données de pratiques observables d'usage du système à partir desquelles le système peut être simulé pour validation (Sutcliffe et al. 1998).

Dans les scénarios exprimés en langage naturel, nous retrouvons les limites citées précédemment sur les exigences textuelles, relatives à l'ambiguïté inhérente du langage naturel et l'évaluation de la qualité des scénarios. Des propositions d'atelier de rédaction de scénarios ont été faites pour éliminer ces ambiguïtés (Achour et Rolland 1997). Diaper (Diaper 2002) et Vicente (Kim J. Vicente 1999a) considèrent que les détails des scénarios constituent un risque de se concentrer sur des problèmes partiels et non représentatifs de la situation globale, n'ayant aucune assurance qu'un faible nombre de scénarios couvre tous les aspects du problème. Ils proposent les modèles de tâches, basés sur des décompositions hiérarchiques pour raisonner sur l'allocation de fonctions.

En UML, l'utilisation des formalismes visuels de diagramme de séquence de message et de statecharts permet de bénéficier de vues plus abstraites des scénarios. Cependant, Damn et Harel (Damm et Harel 1999) relèvent des problèmes sur les diagrammes de séquence liés au pouvoir d'expression du langage : ce dernier permet certes d'exprimer l'ordre des messages, mais ne permet pas d'exprimer la différence entre comportement possible et obligatoire, le comportement du système dans des conditions dégradées ou des scénarios interdits. Ils proposent une extension des diagrammes de séquence, les *live sequence chart,* pour contrer ces limites en proposant la spécification d'anti-scénarios (interdits) et en renforçant la structuration en sous-diagramme, branchement et itération. Amyot et al. (D. Amyot et al. 1999) proposent une approche rigoureuse basée sur les scénarios, en complétant les scénarios UCM par des spécifications formelles permettant de détecter des interactions entre scénarios.

Enfin, une limite du scénario vient d'un engagement prématuré dans la distribution des responsabilités entre agents du système (Damas, Lambeau, et van Lamsweerde 2006) : des modèles à base de buts et d'obstacles sont proposés pour élever le niveau d'abstraction des scénarios et évaluer des alternatives.

### II.1.2.5. Les modèles orientés buts

#### II.1.2.5.1. Principes et exemples

L'objectif des modèles orientés buts est d'élever le niveau d'abstraction et de guider l'élicitation et le raffinement des exigences à partir des buts. Un but est un objectif que le système considéré doit atteindre (A. van Lamsweerde 2001). Il permet d'exprimer les raisons pour lesquelles le système doit exister (Yu 1997). Les deux cadres conceptuels, i* (Yu 1997) et KAOS (Dardenne, Fickas, et van Lamsweerde 1991), proposent des langages graphiques permettant d'exprimer les liens entre buts, et acteur(i*)/ agent (KAOS) : un modèle de buts (Figure 32 et Figure 33) est un graphe ET/OU annoté

montrant comment les buts de haut niveau sont satisfaits par des buts de plus bas niveau et inversement comment les buts de plus bas niveau contribuent à la satisfaction des buts de plus haut niveau. Les buts sont décomposés en sous-buts jusqu'à atteindre une bonne compréhension du système. Dans KAOS, les buts sont opérationnalisés par l'expression de contraintes formalisées, telles que : $\forall$ elevator: elevator.state='moving' $\Rightarrow$ door.state='closed'. .

Dans i*, les exigences non fonctionnelles, telles que la sécurité ou l'utilisabilité, sont exprimées sous forme de *soft goals* (Mylopoulos, Chung, et Yu 1999). GRL est le langage standardisé par l'Union internationale des Télécommunications, sur la base d'un sous-ensemble de i*. Ce langage, couplé avec la notation des cartes de cas d'utilisation (Use Case Map), constitue la notation pour les Exigences Utilisateur (User Requirements Notation) (Daniel Amyot et al. 2009).

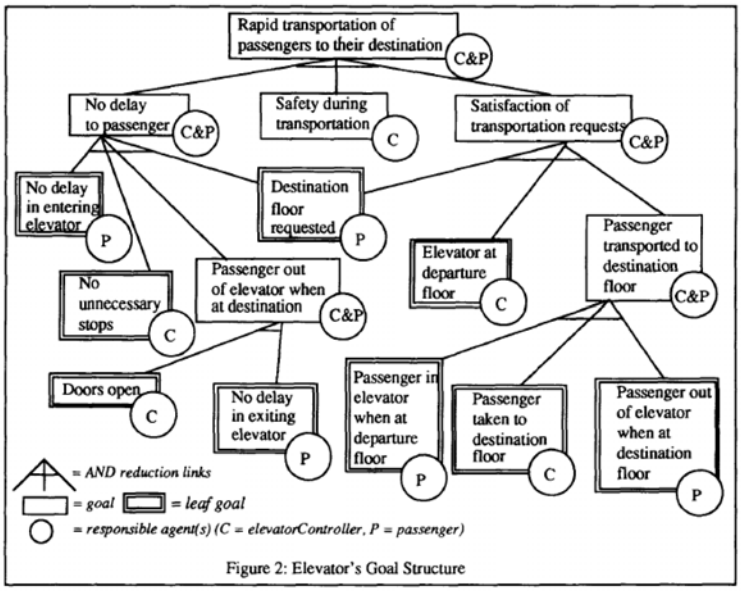



Figure 32: un modèle KAOS issu de (Dardenne, Fickas, et van Lamsweerde 1991)

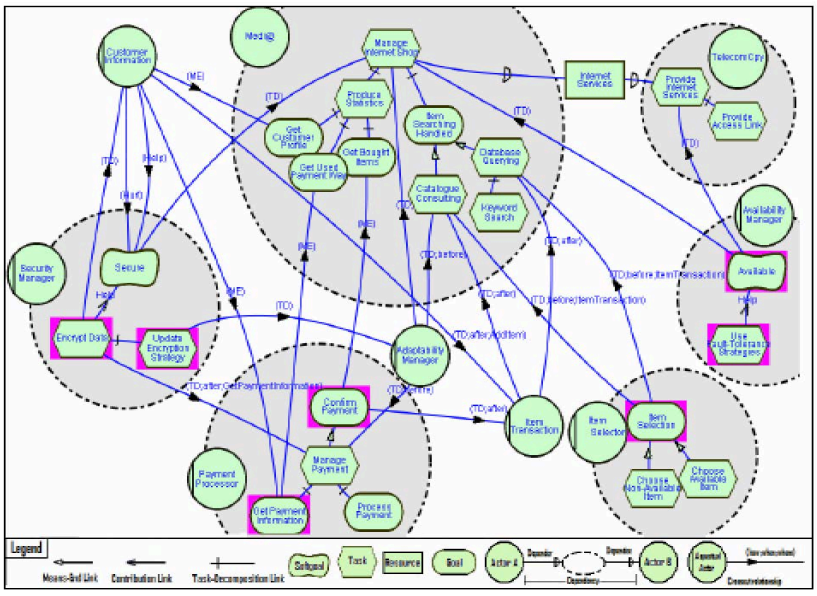



Figure 33: un modèle de buts utilisant la notation i* issu de (Castro et al. 2009)

#### II.1.2.5.2. Avantages et limites

Un avantage attendu des modèles orientés but est de pouvoir supporter le raisonnement sur des alternatives avec des branches OU au moment de l'élaboration du modèle (Yu 1997) (Axel van Lamsweerde 2009), se rapprochant des notations de design rationale, tels que QOC (MacLean et al. 1991). Un autre avantage est de vérifier la complétude des exigences spécifiées par rapport aux buts attendus, soit de façon informelle, soit de façon formelle si les buts et les contraintes sont formalisées en logique temporelle (Dardenne, Fickas, et van Lamsweerde 1991). La détection de conflits entre buts contribue à la vérification de la cohérence des exigences. Des techniques formelles sont également proposées pour détecter et résoudre les conflits entre buts (Dardenne, Fickas, et van Lamsweerde 1991). Les buts fournissent également une base pour la découverte d'exceptions de haut niveau ou d'obstacles, qui peuvent donner lieu à de nouvelles exigences pour les systèmes critiques. Des techniques formelles permettent de générer des obstacles à partir d'un modèle de buts et de résoudre automatiquement ces obstacles par substitution de but, substitution d'agent, reformulation du but ou prévention de l'obstacle (A. van Lamsweerde 2001). Les buts sont parfois évalués comme trop abstraits par le client et les utilisateurs, les comportements du système étant implicites. De plus, les buts peuvent être difficiles à éliciter de façon précise (Damas, Lambeau, et van Lamsweerde 2006).

### II.1.2.6. Les méthodes formelles de spécification

#### II.1.2.6.1. Principes et exemples

L'usage de méthodes formelles en ingénierie des exigences vient de la difficulté de vérifier des propriétés par inspection manuelle (D. Jackson 2006). Les méthodes formelles de spécification utilisent des notations mathématiques pour décrire de façon précise les propriétés que le système doit avoir, sans contraindre la façon dont ses propriétés sont réalisées. Le langage de spécification Z (Spivey 1989) propose de décomposer la spécification en éléments appelés schémas, qui décrivent les aspects statiques (les états possibles, les relations invariantes) et dynamiques du système (les opérations possibles, les relations entre entrées et sorties), les changements d'état (voir des exemples en Figure 34: exemples de spécification Z issus de (Spivey 1989).

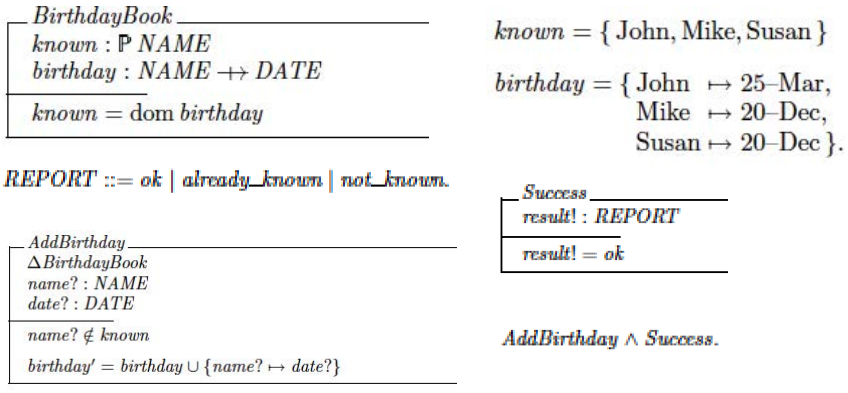



Figure 34: exemples de spécification Z issus de (Spivey 1989)

Une fois la description faite en Z, l'idée est de renforcer la description, en spécifiant les erreurs qui peuvent être détectées et les réponses à ces erreurs. Des opérateurs de calcul permettent de combiner la description initiale avec la gestion des erreurs (un exemple simple à droite en Figure 34). D'autres méthodes de spécification formelle ont été proposées à partir de Z, avec des outils d'analyse,

telles que Alloy (D. Jackson, Shlyakhter, et Sridharan 2001), B (J.-R. Abrial 1996)(Lano 1996), Event-B (Poppleton 2007).

#### II.1.2.6.2. Avantages et limites

Les méthodes formelles de spécification mettent l'accent sur la fiabilité du logiciel : le typage et la vérification de propriétés doivent permettre d'éliminer des erreurs et des comportements inattendus (Woodcock et Davies 1996). L'usage de méthode formelle peut également permettre d'automatiser la génération du code exécutable à partir de la spécification, garantissant le respect des propriétés dans le logiciel final et allégeant par conséquent les tâches de tests unitaires et d'intégration (Jean-Raymond Abrial 2006). Abrial rapporte cependant des difficultés dans l'utilisation de B dans les projets de la ligne 14 du métro de Paris et de la navette automatique desservant les terminaux de Roissy-CDG (Jean-Raymond Abrial 2006) : la construction du modèle abstrait à partir du document de spécification est problématique pour les ingénieurs, notamment dans la façon d'organiser les étapes entre les exigences fonctionnelles et les exigences de sécurité et de panne.

## II.1.3. Elaboration et vérification des modèles : automatisation vs interaction

Nous venons de présenter les différentes expressions des exigences que la recherche académique propose, chacune avec leurs avantages et leurs limites. Les exigences textuelles et les scénarios facilitent la communication avec le client et les utilisateurs, et leur flexibilité permet de faire évoluer facilement les exigences au fur et à mesure de leur compréhension.

L'élimination d'ambiguïtés, la recherche de précision, d'abstraction, et l'apparition de nouveaux types de systèmes (transactionnels, réactifs, distribués) sont évoquées pour proposer des langages plus formels, visuels ou mathématiques, accompagnés de méthodes structurées pour spécifier le système.

Une gestion de la complexité est proposée par les différents formalismes à travers des principes de décomposition hiérarchique, de niveaux d'abstraction, de clusters de vues ou de schémas. Les modèles obtenus permettent de conduire des raisonnements formels et de vérifier des propriétés sur le système (M. Jackson 2009): *Formal reasoning cannot be based on an informal model.* La cohérence du modèle peut être vérifiée automatiquement avec la mise en place d'algorithmes et des théorèmes de preuve, à condition que l'utilisation des formalismes soit rigoureuse. Les différents formalismes sont complémentaires et permettent d'exprimer les différentes dimensions du système : intentionnelle, comportementale et structurelle (Damas, Lambeau, et van Lamsweerde 2006).

Cependant, l'augmentation du niveau de formalisme dans l'expression des exigences (la lisibilité par une machine) dégrade la lisibilité des modèles par les êtres humains (Davis 1982). Nous proposons dans la Figure 35 de positionner les différentes expressions des exigences selon ces deux axes (inspirés par Davis), d'après notre état de l'art.

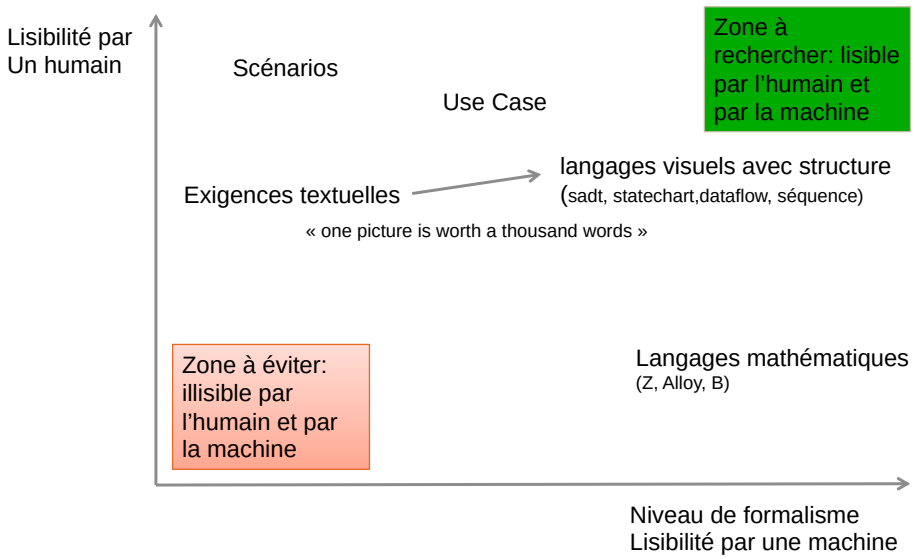



Figure 35: lisibilité par l'humain ou par la machine : faut-il choisir ?

Pour des systèmes complexes, l'élaboration et la vérification de ces modèles sont des tâches coûteuses pour les ingénieurs. De plus il faut assurer les liens entre les différents moyens d'expression (texte, scénarios, diagrammes, expressions régulières) et les différents niveaux d'abstraction : c'est l'enjeu de la traçabilité, définie par Gotel (Gotel et Finkelstein 1994) comme :

> *the ability to describe and follow the life of a requirement, in both a forwards and backwards direction (i.e., from its origins, through its development and specification, to its subsequent deployment and use, and through periods of on-going refinement and iteration in any of these phases).*

Deux approches existent pour alléger des tâches utilisateur : l'automatisation et l'interaction humain-machine. De nombreux travaux en ingénierie des exigences parient sur l'automatisation, en faisant l'hypothèse que transformer automatiquement un modèle en un autre modèle permet d'obtenir une spécification plus complète et cohérente, avec des liens entre modèles obtenus comme sous-produit de la transformation automatique (Stefan Winkler et Pilgrim 2010). Ainsi, la mise au point d'algorithme de traitement du langage naturel permet d'envisager la transformation automatique de scénarios textuels en diagrammes de séquences (Achour et Rolland 1997). Un modèle d'états-transition du système peut être synthétisé à partir de diagrammes de séquences de message (Harel et Kugler 2000)(Uchitel, Kramer, et Magee 2003), des diagrammes d'état sont générés à partir de scénarios (Whittle et Schumann 2000), ou des buts exprimés en logique temporelle linéaire peuvent être inférés de diagrammes de séquence de message (Damas, Lambeau, et van Lamsweerde 2006). Enfin, la génération de documentation à partir des traces d'exécution du logiciel est explorée pour diminuer le coût de création et de maintenance de la documentation (Braun, Amyot, et Lethbridge 2015).

Nous adoptons une approche différente de l'automatisation, centrée sur l'interaction humain-machine pour l'élaboration et la vérification de modèles des exigences. En suivant cette approche centrée utilisateur, notre première question est de comprendre quelles sont les pratiques industrielles en termes d'expression des exigences et les problèmes rencontrés en ingénierie des exigences par les ingénieurs. Dans cet objectif, nous réalisons un état de l'art des études de terrain.

# II.2. Les études de terrain relatives aux pratiques industrielles

L'objectif de cette partie est de présenter et analyser les pratiques industrielles en ingénierie des exigences, les problèmes rencontrés par les ingénieurs dans leurs pratiques, et les recommandations suggérées par les travaux antérieurs en termes d'outils et de méthodes.

Nous analysons les études de terrain en utilisant la grille de lecture suivante :

- Nombre d'entreprises et nombre de projets analysés ;
- Nombre de participants ;
- Méthode d'analyse (questionnaire, interviews, observations, action research, think aloud) ;
- Pratiques et problèmes relevés ;
- Recommandations en termes d'outils et méthodes.

Les trois premiers éléments de notre grille permettent de fournir une indication sur la validité et la généralisabilité de l'étude. Nous dégageons en synthèse des opportunités de recherche. Cet état de l'art des pratiques industrielles sera également un outil pour la généralisation de notre propre travail de recherche. En effet, la généralisabilité d'une étude peut être étayée par la cohérence de ses résultats avec les travaux antérieurs (Seddon et Scheepers 2012). Nous adoptons une présentation chronologique des études de terrain, des années 1990 jusqu'à aujourd'hui, afin d'évaluer l'impact de l'informatisation sur les pratiques des ingénieurs dans l'élicitation et la formalisation des exigences, ainsi que l'adoption des différentes méthodes et langages graphiques que nous avons présentés dans la partie précédente.

## II.2.1. Les études des années 1990

L'étude de référence de Lubars (Lubars, Potts, et Richter 1993) concerne 10 entreprises américaines, un total de 23 projets dans des domaines industriels variés, avec 35 interviews réalisées auprès de 87 développeurs. Tous les participants identifient la nature évolutive des exigences comme le problème spécifique de l'ingénierie des exigences. Les projets observés gèrent les changements de façon ordonnée, en suivant des procédures formalisées la plupart du temps. L'étude rapporte cependant des problèmes dans la gestion d'une documentation très volumineuse, la communication avec le client, et la mise en priorité des exigences. Le langage naturel est largement utilisé pour exprimer les exigences. Les opinions collectées sur les représentations d'analyse structurée sont mitigées selon les projets :

> *While some projects lauded it, others said that structured analysis does not work or produces* ***unintelligible*** *specifications.*

L'article ne fournit pas assez de détails pour expliquer cette différence de points de vue. Aucune utilisation de méthode formelle n'a été relevée. Les analyses orientée-objet étaient à leur début et ne sont pas encore très répandues. Cependant, le problème le plus fréquemment cité est l'absence de moyens pour modulariser les volumineuses spécifications orientées-objet, et l'explication à des non-spécialistes. Les participants ayant utilisé des diagrammes d'état-transition se plaignent car ils trouvent que les scénarios sont plus informatifs. L'étude rapporte une préférence pour les outils à vocation générale, incluant Excel, Word avec des macros, des bases de données, Hypercard, des outils de présentation et la messagerie électronique.

*Spreadsheets are adequate for most engineers' purposes in doing quick performance budgeting calculations. Database management systems and spreadsheets are used for requirements traceability. Hypercard and presentation tools are used to prototype user interfaces and demonstrate ideas, usually in preference to specific user interface design tools. Electronic mail is used to coordinate review meetings and other actions and to alert developers of changes.*

Les outils spécifiques sont utilisés par un tiers des projets seulement, et une insatisfaction est exprimée sur ces outils. Les auteurs expliquent la faible adoption des outils spécifiques par le fait qu'ils soient associés à une méthode, et qu'il y ait une réticence à former les développeurs d'une méthode à l'autre. Les recommandations sur les outils sont formulées sur :

- un support à la documentation :

*capacity for large volumes of documentation; interconnection of multiple, related documents; internal and external document references and navigation; representation of multiple levels of formality and authoritativeness; demonstrable traceability; change management at various levels of formality; configuration management of composite documents; and coexistence of uninterpreted text or pictures with language editing and analysis.*

- Des recommandations pour les outils spécifiques, alors même que les auteurs concèdent que les fonctionnalités existent déjà :

*The generation of prototype code, viewing of the model through multiple filters, and the simulation of a model are some of the features most requested. Even though some commercial tools provide these functions today, many users still resorted to separate tools, tools such as presentation programs and spreadsheets that are not usually considered CASE tools, to support this functionality.*

L'étendue de l'échantillon et la méthode de collecte de données en font une étude fiable. Cependant, la date de sa réalisation (1993) laisse supposer une évolution dans les pratiques industrielles et dans les outils.

Une étude longitudinale de trois ans, dans une seule entreprise orientée marché, réalisant des applications interactives à dialogue vocal, fait le même constat de l'utilisation par les concepteurs d'outils à vocation générale (Sumner 1995). Elle observe que les concepteurs utilisent différents moyens d'expression des exigences : des diagrammes de flux, accompagnés d'un tableau pour chaque entité du diagramme, et des plans de test formulés textuellement. Les outils à vocation générale permettent aux utilisateurs l'utilisation de ces différents moyens d'expression et de les faire évoluer dans le temps, selon leurs propres besoins. Les deux recommandations sur les outils sont liées à la possibilité de pouvoir paramétrer les outils, et de pouvoir créer des liens automatiques entre les données manipulées par les outils, à partir d'abstraction.

L'étude de grande envergure réalisée par Gotel et Finkelstein (Gotel et Finkelstein 1994) vise à déconstruire le problème de la traçabilité des exigences, qui semble perdurer malgré la présence accrue d'outils spécialisés censés offrir un support. Elle analyse les pratiques de plus de 100 participants dans 5 entreprises du Royaume-Uni, à travers des focus groups (37 participants), de questionnaires suivis de sessions d'interview (56 participants + 33 participants), et des observations d'exercices. Un manque de définition commune de la traçabilité est un premier facteur explicatif du problème persistent : les auteurs proposent de distinguer la pré-traçabilité, qui concerne les aspects de l'exigence précédent son inclusion dans la spécification (sa production), de la post-traçabilité, qui

concerne les aspects d'une exigence résultant de son inclusion dans la spécification (son déploiement). Cette séparation entre les deux types de traçabilité leur permet de distinguer les supports. Les auteurs éliminent le problème de la post-traçabilité en préconisant l'usage de méthodes formelles qui permettent de transformer automatiquement une spécification en exécutable, assurant par construction une post-traçabilité. Par contre, la pré-traçabilité n'est pas automatisable, du fait de la nature dynamique des sources des exigences et de l'environnement, d'où une concentration sur les problèmes et améliorations à la pré-traçabilité. La pré-traçabilité est souvent une connaissance tacite, difficile à obtenir, à documenter et à mettre à jour dans une organisation dirigée par les livrables et la production. Les bénéfices ne sont pas immédiats, et peu de support outillé existe pour assurer une mise à jour face à des exigences utilisateur changeantes. De plus, l'utilisation de la pré-traçabilité est difficile à définir et anticiper, les contextes sont variés, les personnes potentiellement concernées très différentes, posant un problème d'accès et de présentation de l'information. Les pratiques observées sont d'aller interroger les personnes impliquées dans le projet quand l'information n'est pas documentée ou difficilement localisable. Enfin, la spécification est parfois obsolète, du fait que des évolutions du système soient faites alors que les auteurs de la spécification ne sont plus impliqués. Certaines caractéristiques de projet semblent favoriser l'occurrence des problèmes : la séparation des individus dans plusieurs équipes entraine des problèmes de communication, d'implication, de responsabilité et partage d'information, qui sont atténués dans des projets fonctionnant en équipe réduite. L'utilisation d'outils à vocation générale est évoquée et comparée avec des environnements dédiés, mais sans précision sur les tâches précises ou artefacts produits.

Les auteurs proposent plusieurs solutions. Pour l'enregistrement de la pré-traçabilité, les propositions sont de créer des rôles dédiés à la documentation de la pré-traçabilité, et de favoriser l'obtention et l'enregistrement des informations en travaillant sur la conception d'environnement spécifique. Dans cet objectif, les auteurs préconisent d'utiliser des méthodes ethnographiques pour comprendre la production et l'usage d'une spécification. Pour faire face aux itérations et changements, les auteurs préconisent des travaux relatifs à la gestion d'une information informelle et instable. Enfin, pour faire face à un contexte d'utilisation changeant, des efforts doivent être consacrés à l'accès et visualisation des données, citant les travaux en recherche d'information, intelligence artificielle et interaction humain-machine.

Ramesh (Ramesh 1998) étudie les facteurs influençant les pratiques de traçabilité dans 26 entreprises américaines, auprès de 138 participants, sur quatre années. Le principal résultat est la distinction entre deux types d'utilisateurs de la traçabilité, bas niveau (*low-end*) et haut niveau (*high-end).* Les utilisateurs bas-niveau utilisent les relations de traçabilité dans la décomposition des exigences, l'allocation d'exigences aux composants du système, la vérification de conformité et la gestion des changements sur le système. Cependant, ils ne considèrent pas la traçabilité comme une tâche importante dans leur processus de développement et ne sont pas conscients d'aspects formels dans leurs pratiques de traçabilité. Ils utilisent principalement des matrices statiques, dont l'information est rapidement obsolète, du fait de changements non répercutés. Les utilisateurs de haut niveau sont concernés par la traçabilité et ont tendance à l'identifier comme un outil primaire dans les processus de développement de système, pour démontrer auprès du client que le système répond à leurs besoins, et pour la maintenance du système. Réalisant qu'il n'y a pas de méthodologie bien définie sur l'information à enregistrer, ils définissent des « schémas de traçabilité » et des modèles dans les outils spécialisés. Les outils disponibles pour capturer et utiliser les informations de traçabilité sont les mêmes pour les deux types d'utilisateurs, des bases de données relationnelles ou DOORS. Les utilisateurs de haut-niveau ont tendance à paramétrer les outils pour un meilleur support, ou développer leurs propres outils. Les auteurs concluent qu'il faut prévoir une

automatisation dans la création et la maintenance de l'information de traçabilité, et de la sémantique sur les liens pour prévoir du raisonnement automatisé.

L'étude réalisée par Chatzoglou (Chatzoglou 1997) est centrée sur les facteurs affectant le processus d'ingénierie des exigences dans les entreprises. Elle concerne 107 projets réparties sur 74 entreprises de différents domaines et de différents types au Royaume-Uni. Trois catégories de facteurs ayant un impact sur le processus d'ingénierie des exigences sont étudiées : les facteurs humains (expérience et attitude des membres de l'équipe, participation des utilisateurs et communication avec les membres de l'équipe), les facteurs techniques (techniques et outils utilisés pendant le processus), et enfin le management (allocation des ressources et style). Les caractéristiques du projet sont examinées en facteur modérateur. La méthode utilisée est à base de questionnaire. L'étude montre que le processus d'ingénierie des exigences est un processus itératif, 50% des projets réalisant au moins trois itérations. Le nombre d'itérations prévues initialement était différent pour 44% des projets, montrant la difficulté à évaluer le travail de collecte et d'analyse des exigences. Les projets qui mettent plus de ressources et de moyens dans la première itération ont moins de dérive de planning. Dans 47% des projets, aucune méthodologie n'est utilisée, et cela s'élève à trois quarts sur les projets portés par l'industrie. L'auteur explique cette différence par le fait que les personnes travaillant dans l'industrie ne sont pas au courant des nouvelles méthodologies et de leur utilisation, et que ces méthodologies ne sont pas faciles à mettre en pratique car mal documentées ou exigeant une formation importante. L'industrie a une préférence pour le prototypage et les méthodologies maison.

Les critères de choix d'une méthodologie donnés par les participants de l'étude sont : la facilité d'utilisation, le support, et les sorties (outputs). Le résultat le plus important, que l'auteur considère alarmant, est que la méthodologie n'est pas choisie par rapport à des caractéristiques objectives mais parce qu'elle est pratique et avantageuse pour les développeurs. La conclusion de l'étude est que la qualité des outils et des techniques n'est pas adéquate.

> *the quality of tools and techniques initially employed in the RCA process is not adequate either. This came as no surprise at all after the results of the examination of the use of methods/tools/techniques during the project development and the RCA process, have shown that almost half of the projects do not use any methodology during the development process or during the RCA process.*

La recommandation est de porter plus d'attention à la qualité des outils. L'auteur préconise l'usage de méthodologie qui permettra une réduction du nombre d'itérations dans les projets, le nombre important d'itérations actuel étant un signe d'absence de méthodologie et de mauvaise gestion des ressources humaines sur les projets.

A l'opposé de ces études avec de grands échantillons (Lubars, Potts, et Richter 1993)(Gotel et Finkelstein 1994)(Chatzoglou 1997), nous retenons l'étude réalisée par Guindon (R. Guindon et Curtis 1988) (Raymonde Guindon 1990) avec seulement 3 participants, dont l'originalité est de se centrer sur les activités cognitives du concepteur. La méthode est un protocole de verbalisation (think aloud) pendant 2 heures, afin d'identifier les activités et les processus cognitifs à l'œuvre dans la conception d'un système d'ascenseur.

Les résultats détaillés de l'étude mettent en évidence une démarche opportuniste de conception, passant du domaine du problème au domaine de la solution de façon itérative et non ordonnée. Selon les auteurs, le processus de conception se rapproche d'un processus de découverte de connaissances, ponctué d'inférences et d'ajouts de nouveaux objectifs qui réduisent l'incomplétude

et l'ambiguïté de la spécification. Ces inférences et ajouts sont provoqués par plusieurs sources de connaissance : d'autres exigences, des solutions partielles et des scénarios opérationnels. Les concepteurs tendent alors immédiatement à développer des solutions partielles pour ces exigences inférées ou ajoutées, essayant de tirer le plus possible parti de cette soudaine découverte. Les recommandations sur l'environnement de travail préconisent les éléments suivants :

- Ne pas intégrer une méthode qui contraindrait les concepteurs à un ordre strict d'activités, ce qui empêcherait la démarche opportuniste nécessaire à la découverte de la décomposition de la solution ;
- Supporter un accès et un basculement rapides entre plusieurs outils pour représenter et manipuler différentes sortes d'objets et leurs représentations. Certains objets sont des exigences informelles, des informations sur le problème du domaine, des problèmes et critères sur le système et sur le processus, des décisions de conception exprimées dans une notation formelle ou semi-formelle.
- Supporter une navigation facile entre ces objets, sans imposer un ordre prédéterminé des activités, tout en supportant un agenda des activités du concepteur ;
- Supporter une édition facile et une réorganisation des exigences, des problèmes de conception, des décisions ;
- Supporter l'identification de l'origine des exigences : initialement présentes, contraintes inférées et ajoutées.
- Supporter la représentation d'objets intermédiaires ou partiels dans la décomposition de la solution.

Enfin, les langages de représentation doivent supporter une progression souple d'exigences exprimées informellement jusqu'à des décisions de conception exprimées formellement et le code.

L'étude menée par Visser sur l'activité d'un ingénieur pendant trois semaines fait un constat similaire (Visser 1990) : l'ingénieur établit un plan de travail, fondé sur une décomposition hiérarchique du problème en vue de la spécification de l'installation d'une machine-outil. L'ingénieur affirme qu'il suit ce plan, mais les observations réalisées montrent qu'il dévie de ce plan. Visser étudie de façon précise la façon de travailler de l'ingénieur pour la spécification de l'installation, et met en valeur des allers-retours entre les différents niveaux de décomposition, avec des descriptions de composants réalisées dans une première itération de façon partielle, qui sont complétées et/ou modifiées dans des itérations ultérieures. Visser recherche les causes de ces déviations, en adoptant le modèle de tableau noir utilisé en intelligence artificielle (Hayes-Roth 1985), et en s'intéressant aux mécanismes de contrôle : comment l'ingénieur décide-t-il des actions à conduire et de quand les conduire ? Les critères de sélection pour conduire une action sont la disponibilité de l'information nécessaire à l'action, la difficulté relative de l'action par rapport au plan prévu, l'importance de l'action, et l'importance de l'objet de l'action. Les éléments qui déclenchent une déviation du plan sont relatifs à l'activation de la représentation d'un autre composant à partir d'un composant de départ, que ce soit par analogie, mais aussi parce que les deux composants sont en interface, ou que les conditions d'entrée d'une fonction dépendent des conditions de sortie d'une autre fonction. L'ingénieur peut également changer de plan parce que l'action prévue est trop coûteuse à réaliser à ce moment-là (le client n'a pas décidé ou l'information nécessaire est connue par un collègue absent), ou trop difficile. L'étude conclut que la rédaction de la spécification est une activité opportuniste organisée : l'ingénieur a un plan mais se permet d'en dévier si des actions moins coûteuses se présentent à lui. Il revient à ce plan ensuite. Une recommandation est émise sur les outils : permettre d'abandonner la solution du problème à un certain niveau, afin de traiter des éléments de solution à un autre niveau, et pouvoir reprendre l'état de la solution au niveau abandonné.

## II.2.2. Les études dans des années 2000

L'étude réalisée par Hall et al. (Hall, Beecham, et Rainer 2002) est centrée sur l'identification des problèmes dans le développement de systèmes logiciels. Elle est basée sur 45 groupes de travail, ayant au total impliqué 200 personnes. Les exigences concernent 48% des problèmes rapportés, que les auteurs classifient en deux catégories, organisationnelle et liée au processus. Les problèmes de communication entre les développeurs, des compétences et des ressources inappropriées sont les principaux problèmes organisationnels détectés. Des exigences initialement vagues, un processus indéfini, une croissance des exigences et la complexité de l'application sont les principaux problèmes relevés sur le processus. L'étude ne fournit aucune indication sur les outils utilisés. Elle conclut sur l'intérêt de proposer un modèle de maturité du processus d'ingénierie des exigences, et d'impliquer les développeurs dans l'amélioration des processus.

Alexander et al. (Alexander, Robertson, et Maiden 2005) cherchent à comprendre et identifient les facteurs influençant le processus d'ingénierie des exigences en industrie. Dans cet objectif, ils ont réalisé une enquête sur la base d'un questionnaire listant 29 facteurs potentiels, et ont obtenu 152 réponses de la part d'industriels de domaines variés. Les réponses montrent que ce sont d'abord la formation, les standards propres à l'entreprise, les outils, le régulateur, les principes élémentaires et les collègues expérimentés qui ont le plus d'influence sur le processus. Le régulateur est un facteur de plus grande influence sur les projets critiques par rapport à la sécurité, pour lesquels les projets sont de plus grande durée. Cette importance du régulateur explique, selon les auteurs, une attitude conservatrice par rapport au processus. Les réponses venant de l'aéronautique suggèrent que l'expérience des personnes est le facteur le plus influant sur le processus. Concernant les outils, 53 participants mentionnent les outils utilisés, DOORS pour 23 d'entre eux. L'usage de Word et Visio est mentionné, mais le nombre de réponses n'est pas précisé. Les outils sont notés comme ayant plus d'influence sur le processus par les participants utilisant des outils spécialisés de gestion des exigences, notamment dans le secteur de la Défense. Ce résultat suggère que les outils contraindraient le processus. Les auteurs rappellent que les outils doivent suivre le processus, et non le contraire. Enfin, les auteurs recommandent un enseignement des différents processus dans les universités, faisant le constat que les ingénieurs en fonction sont difficiles à atteindre.

A l'inverse de ces études basées sur des échantillons de grande ampleur, l'étude réalisée par Nguyen et Swatman (Nguyen et Swatman 2000) se concentre sur un projet logiciel impliquant 3 concepteurs, et utilise une méthodologie de recherche active (action research) pour analyser l'évolution des modèles dans le processus d'ingénierie des exigences, en faisant référence aux études de Guindon (Raymonde Guindon 1990) et Visser (Visser 1990). L'analyse conforte la nature opportuniste du processus d'ingénierie des exigences et montre une dynamique en dents de scie dans la complexité des modèles dans le temps. Les auteurs proposent une explication en distinguant la complexité essentielle du modèle de sa complexité accidentelle (*incidental* dans (Nguyen et Swatman 2000) et *accidental* dans (Nguyen, Carroll, et Swatman 2000)). La complexité essentielle du modèle est liée à la compréhension inhérente de l'espace problème, alors que la complexité accidentelle surgit de la faible adéquation entre la structure du modèle et la structure de la réalité que le modèle vise à représenter. Au départ d'un cycle, la découverte de nouvelles informations entraine une meilleure compréhension du problème (l'augmentation de la complexité essentielle) et une structuration du modèle pour représenter ce problème (la complexité accidentelle). Mais la complexité accidentelle est parfois réduite en adoptant une nouvelle perspective ou par réflexion critique, permettant une modélisation plus simple d'une réalité toujours plus complexe (Figure 36).

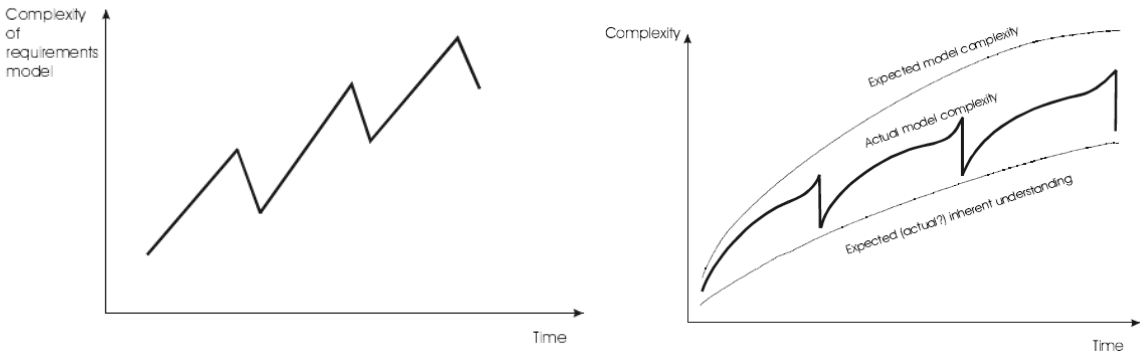



Figure 36: l'évolution de la complexité des modèles, observée par (Nguyen et Swatman 2000)

Les mêmes auteurs proposent d'utiliser les notations de design rationale IBIS et QOC pour enregistrer les décisions et les choix de modélisation pendant le processus, afin de mieux comprendre ces cycles et les points critiques de restructuration/simplification des modèles (Nguyen, Carroll, et Swatman 2000) : l'utilisation de ces notations permet de documenter le processus de conception et d'expliquer a posteriori les modifications sur les modèles, mais ne permettent pas de générer des modifications. Le chemin suivi par les concepteurs apparaît opportuniste et ne suit pas de plans prédéterminés liés à une décomposition systématique du problème en sous-problèmes. Le chemin dépend beaucoup de la créativité et de l'expérience des concepteurs. La conclusion est qu'une flexibilité dans le processus est nécessaire pour permettre ces oscillations bénéfiques, allant vers une simplification des modèles. En termes d'outils, les auteurs recommandent des outils pour surveiller la dynamique de la complexité des modèles pour les managers. De plus, les outils doivent assister les concepteurs dans l'enregistrement de justification et l'édition des modèles.

> *Tools for monitoring the dynamics of the complexity of requirements models are needed. Moreover, there is also the need for an integrated environment CASE tool to assist designers in recording rationale, and editing requirements models, as prerequisites for managing the process.*

Une enquête réalisée en ligne sur la base de 22 questions et ayant obtenu 194 réponses d'une multitude d'entreprises américaines (Neill et Laplante 2003) montre que le cycle en cascade est rapporté comme celui étant le plus utilisé. Les méthodes formelles sont rarement utilisées. 30% des participants répondent qu'ils n'utilisent pas de méthode de modélisation des exigences : la gestion du changement est cruciale dans les entreprises pour l'adoption de nouvelles méthodes (Laplante et Neill 2004). Les données collectées ne fournissent pas d'informations sur les outils utilisés et les problèmes rencontrés.

L'étude réalisée par Karlsson et al. (Karlsson et al. 2007) examine les pratiques industrielles dans 8 entreprises de développement de logiciels orientés marché. Elle est basée sur l'utilisation d'interviews et d'un groupe de travail, impliquant 14 personnes. L'étude vise à élargir la compréhension de l'ingénierie des exigences pour les entreprises orientées marché, sans focus initialement identifié. La nature fluctuante des exigences est confirmée, l'étude parlant de volatilité des exigences. Le flot constant d'exigences provenant des clients pose des problèmes de gestion de l'information (duplicatas) et de gestion des priorités. Fournir des retours aux clients sur leurs suggestions d'évolutions est coûteux, mais reste crucial pour conserver leur implication. Pendant le focus group, un manque d'outils supportant un processus flexible a émergé, alors que cette catégorie de problème n'avait pas été détectée par les chercheurs lors du codage et de l'analyse des interviews. Il n'y a pas

de recommandations formulées sur les outils, mais une volonté de prolonger l'étude en posant des questions spécifiques sur les outils et la façon de gérer la traçabilité.

L'étude réalisée par Graaf et al. (Graaf, Lormans, et Toetenel 2003) examine les pratiques de 8 entreprises de logiciels embarqués, à partir de données de 36 interviews, pour comprendre pour identifier les besoins spécifiques à l'ingénierie des systèmes embarqués. L'étude n'est pas centrée sur l'ingénierie des exigences, mais sur le processus global d'ingénierie des logiciels embarqués. Elle rapporte une séparation non évidente entre la phase de spécification et la phase de conception.

Les entreprises observées expriment les exigences en langage naturel et utilisent des éditeurs de texte ordinaires, avec des modèles de document. Les auteurs soulignent que les exigences non fonctionnelles n'ont pas de traitement spécifique, alors qu'ils s'attendaient à un traitement particulier lié aux exigences de l'embarqué. Des diagrammes sont utilisés en complément, dans un style qui ressemble à UML ou d'autres notations. L'utilisation d'outils de dessin à vocation générale est relevée comme un problème, du fait de l'absence d'une sémantique correcte induisant une mauvaise interprétation des diagrammes par les autres membres du projet, notamment ceux d'autres disciplines habituées à d'autres notations. Les cas d'utilisation et les diagrammes de classe pour la modélisation du domaine sont les formalismes UML les plus utilisés. A un niveau plus bas d'abstraction, les préconditions et post-conditions sont utilisées pour spécifier les exigences, en langage naturel ou langage de programmation (langage C). Les projets utilisent rarement les spécifications formelles, car difficiles à utiliser et à comprendre. Des difficultés sont rapportées sur la gestion des exigences, notamment sur la traçabilité entre exigences et entre exigences et architecture. Les auteurs expliquent que les relations sont difficiles à spécifier manuellement, et que les outils disponibles ne résolvent pas le problème : l'explication proposée est que les utilisateurs ne peuvent pas préciser la signification du lien entre les exigences, et plus spécialement la justification derrière le lien.

L'usage de feuilles de calcul est rapporté pour documenter les liens entre exigences. Certaines entreprises utilisent des outils avancés tels que RequisitePro (Rational), RTM (Integrated Chipware), et DOORS (Telelogic), mais sans succès : les auteurs expliquent que les utilisateurs n'avaient pas les compétences ou qu'ils ne pouvaient gérer que des relations triviales entre exigences, solution et tests. Les auteurs concluent en préconisant de la flexibilité dans les outils pour pouvoir paramétrer selon les besoins des différentes spécialités. Un des auteurs a développé une approche de reconstruction automatique de liens en utilisant une technique d'indexation sémantique latente de documents (M. Lormans et Deursen 2006). L'approche est implémentée dans un outil, ReqAnalyst (Marco Lormans, Deursen, et Gross 2008), qui traite la documentation projet et génère des matrices de traçabilité. L'évaluation sur un cas d'étude montre que l'outil est capable de générer des liens corrects, mais pas avec 100% de précision : des liens existants ne sont pas détectés et aucun moyen n'est proposé pour identifier ces liens manquants.

## II.2.3. Les études des années 2010

Sikora et al. (Sikora, Tenbergen, et Pohl 2011) ont conduit une étude qualitative dans 7 grandes entreprises en systèmes embarqués dans différents domaines, à l'aide d'interview et de questionnaire auprès de 17 participants. L'objectif est de déterminer les besoins en support méthodologique du domaine du système embarqué, selon quatre thèmes : l'utilisation du langage naturel vs des modèles, le support à une grande complexité des systèmes ; l'assurance qualité des exigences, et les interrelations entre spécification et conception. Les résultats de l'étude rapportent essentiellement les résultats du questionnaire et ne font pas systématiquement la différence entre les méthodes et les outils supportant ces méthodes.

Le langage naturel est la forme dominante de documentation des exigences utilisée par les participants, les modèles n'étant pas considérés comme des moyens adéquats pour documenter les exigences règlementaires. Les auteurs notent que les modèles ne sont pas inclus dans les documents à destination des fournisseurs. Les modèles sont considérés comme un moyen de communication et de collaboration lors de l'analyse des exigences, non comme un moyen de spécification. Les participants indiquent qu'un des obstacles majeurs est l'absence de méthodologie et de support outillé, en lien avec la satisfaction des exigences règlementaires de type DO 178-C. Les auteurs préconisent de fournir aux industriels une aide méthodologique leur permettant d'utiliser les modèles pour la satisfaction des standards de sécurité. Cela favoriserait l'utilisation des modèles et permettait aux industriels de bénéficier ainsi d'un niveau d'automatisation plus grand. L'utilisation de couches d'abstraction est identifiée comme un moyen important pour gérer la complexité d'un système. Une confusion des participants dans le contenu des différentes couches d'abstraction est relevée. De plus, les participants évaluent la traçabilité des exigences entre les différentes couches d'abstraction difficile à maintenir. Les outils existants de gestion des exigences sont considérés comme non satisfaisants par les participants pour traiter les exigences à différents niveaux d'abstraction. La conclusion des auteurs est d'ordre méthodologique, avec des recommandations sur le développement des exigences et le concept de niveau d'abstraction.

La vérification de la qualité des exigences, notamment en regard d'exigences réglementaires liées à la sécurité, est réalisée manuellement, à travers des inspections. Les critères de cohérence, traçabilité et testabilité sont identifiées comme candidats à un support méthodologique amélioré. Beaucoup de participants notent le temps important qu'ils passent pour assurer la traçabilité et la cohérence des exigences. Les auteurs proposent à nouveau de mettre en place des recommandations méthodologiques pour traiter de façon plus systématique les critères qualité.

Les participants conçoivent la séparation entre spécification et conception comme une distinction entre définition du problème et conception de la solution. Ils expriment un besoin important de traçabilité entre la spécification et la conception. Certains expriment le besoin d'avoir des règles permettant de développer des modèles d'exigences indépendants des solutions. Les auteurs proposent de prendre en compte ces besoins pour un support à la traçabilité, en visant un haut niveau d'automatisation pour réduire les efforts de création et de maintenance des liens.

L'étude des pratiques proposée par Nair et al. (Nair et al. 2015) fournit plus de détails sur la gestion des preuves de conformité à des exigences de sécurité dans les systèmes critiques. Un questionnaire en ligne a permis de collecter les réponses de 52 participants de 11 domaines critiques. Les questions de recherche sont relatives aux types d'informations et d'artefacts utilisés comme preuve, à la gestion du changement de la preuve, aux techniques de structuration de la preuve, aux techniques d'évaluation de la preuve, aux problèmes rencontrés par les industriels pour fournir la preuve, aux écarts entre l'état de l'art et les pratiques industrielles.

Les résultats montrent que les plans de Vérification et Validation, les résultats de tests et les spécifications de cas de tests sont les preuves les plus utilisées. La vérification par les modèles et par preuve de théorème est la plus faiblement rapportée. Les spécifications d'exigences et de solution sont largement utilisées comme preuve. Les participants rapportent des techniques de vérification manuelle sur la complétude de la preuve et l'analyse d'impact d'une modification. Les auteurs en déduisent un manque de support outillé pour l'évaluation de la complétude et l'analyse d'impact. La structuration des preuves est le plus souvent basée sur des modèles de documents textuels, et non graphiques. La majorité des participants rapportent l'utilisation de matrices entre les différentes preuves pour réaliser la traçabilité. Un quart des participants indique l'usage de modèles, des liens hypertexte et des conventions de nommage. L'évaluation des preuves se fait manuellement par

l'utilisation de check-lists et jugement d'expert. L'utilisation d'outils tels que DOORS n'est pas suffisante pour la provision de preuves liées à la sécurité. Les auteurs concluent que la provision de preuve est un challenge important et qu'il est nécessaire de mener une étude plus approfondie pour savoir quels outils sont adaptés aux activités relatives à la sécurité et à quel moment posent-ils des problèmes.

L'étude exploratoire de Ghazi (Ghazi et Glinz 2016) est centrée sur les artefacts d'ingénierie et les problèmes que leur taille peut poser en termes d'interaction utilisateur. Elle se base sur la réalisation de 29 interviews, réalisées à distance par vidéoconférence dans 29 entreprises différentes, sauf pour 3 d'entre elles. Elle aborde la problématique de l'affichage de beaucoup de fenêtres sur un ou plusieurs écrans, et les techniques utilisées par les participants pour faire face à cette problématique. Les résultats montrent que plus de la moitié des participants utilisent simultanément plus de 4 artefacts différents sur plusieurs écrans, et que cela entraine les problèmes suivants, par ordre de priorité : passage entre artefacts, travail sur une fenêtre trop petite, changement de focus, connaissance des relations entre artefacts, recherche de la bonne fenêtre, arrangement des fenêtres, mémorisation, recherche du focus courant. Les solutions apportées par les participants à ces problèmes sont un effort de mémorisation, des techniques classiques de zoom et de scroll, l'usage de plusieurs écrans associés à des impressions papier.

Cette étude confirme que les participants ont des problèmes d'utilisabilité, actuellement non résolus par les outils utilisés. Cependant, l'étude ne donne aucune indication sur la nature des artefacts, sur les outils utilisés, ou sur les intentions des utilisateurs dans l'affichage de ces artefacts en lien avec l'ingénierie des exigences.

## II.2.4. Synthèse

Les études de terrain en ingénierie des exigences mettent en évidence la nature changeante et volatile des exigences, le nombre sans cesse croissant des exigences et des liens - Nguyen et Swatman (Nguyen et Swatman 2000) parle d'entropie, et l'importance de la communication, que ce soit avec le client (Lubars, Potts, et Richter 1993) (Sumner 1995)(Karlsson et al. 2007), ou avec les développeurs (Hall, Beecham, et Rainer 2002)(Graaf, Lormans, et Toetenel 2003).

La gestion de la complexité du système se fait par l'utilisation de niveaux d'abstraction (R. Guindon et Curtis 1988) (Graaf, Lormans, et Toetenel 2003)(Sikora, Tenbergen, et Pohl 2011). Cependant, les études rapportent des difficultés sur l'usage de ces niveaux d'abstraction, avec des confusions sur le contenu et des problèmes de gestion de liens entre les différents niveaux d'abstraction (Sumner 1995) (Graaf, Lormans, et Toetenel 2003)(Sikora, Tenbergen, et Pohl 2011).

Le problème de la traçabilité des exigences est décomposé en pré-traçabilité et post-traçabilité d'une exigence par rapport à son intégration dans la spécification (Gotel et Finkelstein 1994). De plus, deux types d'utilisateurs de la traçabilité sont identifiés (Ramesh 1998) : les utilisateurs de « bas-niveau », qui ont un faible intérêt pour l'information mais en sont les principaux producteurs, et les utilisateurs de « haut niveau » qui ont un grand intérêt dans la traçabilité pour une vue complète sur le produit et le cycle de vie. Le problème de la post-traçabilité, c'est-à-dire les traces vers les aspects résultant d'une exigence, doit être réglé par l'usage de méthodes formelles qui permettent de transformer automatiquement une spécification en exécutable, assurant par construction une post-traçabilité (Gotel et Finkelstein 1994), et par des environnements supportant tout le processus de développement (Ramesh 1998).

Or, un résultat constant dans le temps, malgré l'apparition de nouveaux langages visuels et formels de modélisation, est que les exigences sont exprimées textuellement en langage naturel dans les

documents de spécification (Lubars, Potts, et Richter 1993)(Graaf, Lormans, et Toetenel 2003)(Neill et Laplante 2003)(Karlsson et al. 2007)(Sikora, Tenbergen, et Pohl 2011)(Nair et al. 2015). L'utilisation de diagrammes et graphiques pendant le processus pour exprimer le comportement du système est rapportée par plusieurs études (Lubars, Potts, et Richter 1993)(Sumner 1995)(R. Guindon et Curtis 1988)(Graaf, Lormans, et Toetenel 2003)(Sikora, Tenbergen, et Pohl 2011), mais plus sous forme de dessins qu'en suivant un langage strict. Ce non-respect de formalisme pose problème en compromettant la vérification automatique de modèles (Graaf, Lormans, et Toetenel 2003)(Sikora, Tenbergen, et Pohl 2011). En effet, la vérification de la complétude et cohérence des exigences et la traçabilité entre les exigences de différents niveaux d'abstraction sont des tâches réalisées « manuellement » par les ingénieurs et prennent du temps (Lubars, Potts, et Richter 1993)(Gotel et Finkelstein 1994)(Sumner 1995)(Ramesh 1998)(Graaf, Lormans, et Toetenel 2003)(Sikora, Tenbergen, et Pohl 2011)(Nair et al. 2015), dans tous les domaines d'activité y compris les systèmes embarqués et critiques alors que la traçabilité est une preuve de conformité à produire pour la certification des systèmes.

L'absence de méthodologie précise est un point relevé par plusieurs études de grande ampleur (Chatzoglou 1997)(Hall, Beecham, et Rainer 2002)(Neill et Laplante 2003) et confirmé par des études de moyenne ampleur dans les systèmes embarqués (Graaf, Lormans, et Toetenel 2003)(Sikora, Tenbergen, et Pohl 2011). Certains auteurs (Chatzoglou 1997)(Hall, Beecham, et Rainer 2002)(Neill et Laplante 2003)(Alexander, Robertson, et Maiden 2005) expliquent l'absence de méthodologie par une méconnaissance des industriels des méthodes issues de la recherche académique et le coût d'adoption pour un projet d'une nouvelle méthode en termes de formation et d'outillage. La revue de littérature de Ivarsson et Gorschek (Ivarsson et Gorschek 2009) analyse le problème du transfert technologique des travaux de recherche et complète ce point de vue: les travaux de recherche ne fournissent pas assez d'arguments de décision pour adopter la méthode proposée, les évaluations étant souvent réalisées par les chercheurs eux-mêmes ou sur des exemples jouets. Les freins à l'adoption d'une nouvelle méthode sont donc présents des deux côtés, industriel et académique. Pour les systèmes critiques, une explication supplémentaire est une attitude conservatrice vis-à-vis des standards de sécurité, pour lesquels il n'existe pas d'antécédents dans l'usage de nouvelles méthodes pour démonter la conformité au standard (Alexander, Robertson, et Maiden 2005) (Sikora, Tenbergen, et Pohl 2011).

Les travaux centrés sur l'analyse de l'activité cognitive des concepteurs (Raymonde Guindon 1990) (R. Guindon et Curtis 1988)(Visser 1990)(Nguyen et Swatman 2000) (Visser 2006) proposent une explication différente sur l'absence de méthodologie constatée : le processus de conception est opportuniste. Il ne suit pas une décomposition du problème en sous-problèmes à travers les différents niveaux d'abstraction mais rebondit entre les différents niveaux d'abstraction à partir des informations disponibles, de l'approche utilisée, des analogies faites par le concepteur du fait de son expérience, de la difficulté du problème. Ces observations peuvent être mises en relation avec les confusions des industriels entre les différents niveaux d'abstraction relevés par Graaf (Graaf, Lormans, et Toetenel 2003) et Sikora (Sikora, Tenbergen, et Pohl 2011), et peuvent en partie expliquer les problèmes rapportés sur la pré-traçabilité des exigences par Gotel et Finklestein (Gotel et Finkelstein 1994). De plus, les concepteurs utilisent plusieurs méthodes et plusieurs types de représentation pour comprendre le problème, en se basant sur des analogies tirées de leur expérience (Sumner 1995)(Visser 2006). On peut noter que l'expérience est un facteur d'influence évalué comme important sur la conduite du processus par les enquêtes de grande ampleur (Chatzoglou 1997) (Alexander, Robertson, et Maiden 2005). Enfin, les concepteurs font un plan de travail, mais ils se permettent d'en dévier si des actions moins coûteuses sont possibles (Visser 1990)(Raymonde Guindon 1990). Cela entraine des problèmes pour les concepteurs : se focaliser rapidement sur une

solution (R. Guindon et Curtis 1988), et oublier de réaliser des actions ou de spécifier des informations (R. Guindon et Curtis 1988) (Visser 1990). Nous pouvons rapprocher cette observation au nombre important d'itérations rapporté par Chatzoglou (Chatzoglou 1997).

L'utilisation d'outils à vocation générale, plutôt que d'outils spécialisés, est identifiée par plusieurs études (Lubars, Potts, et Richter 1993)(Gotel et Finkelstein 1994)(Visser 1990)(Sumner 1995)(Graaf, Lormans, et Toetenel 2003)(Nair et al. 2015), en complément d'outils spécialisés. Des problèmes d'interaction utilisateur lors de la manipulation de nombreux artefacts d'ingénierie sont identifés (Ghazi et Glinz 2016).

## II.2.5. Opportunités de recherche sur les outils

Face à ces constats, plusieurs positions se dégagent dans la littérature par rapport aux outils. La première position est de favoriser la nature opportuniste du processus d'ingénierie des exigences et l'utilisation de méthodes et de représentations variées pour découvrir les exigences (Visser 1990)(Visser 2006)(R. Guindon et Curtis 1988)(Sumner 1995)(Nguyen, Carroll, et Swatman 2000). Les recommandations sur les outils sont assez précises (voir synthèse en Tableau 2) : ne pas intégrer une méthode qui contraindrait les concepteurs à un ordre strict d'activités ; supporter un accès et un basculement rapides entre plusieurs outils pour représenter et manipuler différentes sortes d'objets et leurs représentations ; supporter une navigation facile entre ces objets, sans imposer un ordre prédéterminé des activités, tout en supportant un agenda des activités du concepteur ; supporter une édition facile et une réorganisation des exigences, des problèmes de conception, des décisions ; supporter l'identification de l'origine des exigences : initialement présentes, contraintes inférées et ajoutées ; supporter la représentation d'objets intermédiaires ou partiels dans la décomposition de la solution. Cependant ces travaux sont centrés sur la découverte des exigences et ne s'intéressent pas au support des activités ultérieures de vérification et validation des exigences.

La deuxième position est de préconiser un processus structuré d'ingénierie des exigences lié à un langage (textuel ou graphique) formel pour automatiser les tâches de vérification, accompagné d'un outil flexible permettant de paramétrer le processus (Ramesh 1998)(Chatzoglou 1997)(Graaf, Lormans, et Toetenel 2003)(Sikora, Tenbergen, et Pohl 2011).

La troisième position est d'identifier l'existence d'un problème et de recommander une étude plus approfondie de l'usage des outils dans les pratiques industrielles (Gotel et Finkelstein 1994)(Karlsson et al. 2007)(Nair et al. 2015). Cette troisième voie reste à explorer : mieux comprendre l'usage des différents outils dans les pratiques industrielles d'ingénierie des exigences permettrait d'avoir une vision unifiée des problèmes, de la découverte des exigences aux activités de vérification et de validation, et de formuler des recommandations supportant l'ensemble du processus. Avant d'entreprendre cette exploration, il est intéressant de comprendre quels sont les outils proposés par les éditeurs de logiciel et par les travaux de recherche, leurs capacités par rapport aux différentes activités d'ingénierie des exigences et leur utilisabilité.

| Etude | Nbre d'entreprises et de projets | Nbre participants | Méthode de collecte | Problèmes | Représentations et outils utilisés | Recommandations outils et méthode |
|---|---|---|---|---|---|---|
| (Lubars, Potts, et Richter 1993) | 10 entreprises<br>23 projets | 87 | Interview | Gestion de la documentation volumineuse<br>Communication avec le client<br>Priorité des exigences | Exigences textuelles essentiellement<br>Diagrammes structurés (control flow, data flow)<br>Des outils à vocation générale | Un support pour gérer une documentation volumineuse et variée<br>Génération de code prototype, vue du modèle selon différents filtres et simulation du modèle |
| (Gotel et Finkelstein 1994) | 5 entreprises | 100 | Questionnaires + interviews+ observations | Accès difficile à l'information de pré-traçabilité des exigences | Des outils à vocation générale<br>Des outils spécialisés<br>Pas d'information sur les représentations utilisées | Une automatisation de la post-traçabilité des exigences par l'usage de méthodes formelles (transformations automatiques)<br>Un support à la pré-traçabilité des exigences<br>-mettre en place des rôles dédiés à la documentation<br>-favoriser l'obtention et l'enregistrement des informations en travaillant sur la conception d'environnement dédié, en s 'intéressant aux usages ;<br>-Pour faire face aux itérations, favoriser la gestion d'une information informelle et instable.<br>-accès et visualisation des traces ainsi enregistrées |
| (Chatzoglou 1997) | 74 entreprises<br>107 projets | 105 | questionnaire | Pas de méthodologie<br>Des ressources affectées différemment selon les projets | Pas vraiment précisé : prototypage et méthodologie maison<br>(critère de choix : pratique et avantageuse) | Préconise l'usage de méthodologie qui permettra de mieux gérer les ressources et une réduction du nombre d'itérations dans les projets<br>Préconise de porter attention à la qualité des outils |
| (Hall, Beecham, et Rainer 2002) | 12 entreprises | 200 | Groupe de travail | Communication entre développeurs<br>Compétences et ressources<br>Des exigences vagues au départ, un nombre croissant d'exigences, complexité de l'application, processus non défini | Aucune indication | Les problèmes sont organisationnels : pas de recommandation sur les outils, mais construction d'un modèle de maturité pour le processus d'ingénierie des exigences. |

| Etude | Nbre d'entreprises et de projets | Nbre participants | Méthode de collecte | Problèmes | Représentations et outils utilisés | Recommandations outils et méthode |
|---|---|---|---|---|---|---|
| (Raymonde Guindon 1990)<br>(R. Guindon et Curtis 1988) | 1 entreprise<br>1 projet d' ascenseur | 3 | Think aloud | Un processus opportuniste de conception, qui ne suit pas une décomposition du problème en sous-problèmes. Les concepteurs utilisent plusieurs méthodes qui les guident, mais en dévient en reconnaissant une classe connue de problèmes, ou en allant dans un niveau d'abstraction plus bas pour trouver des idées => des problèmes d'oubli, de focus sur une solution. | Système de développement de Jackson + diagrammes de contrôle de données et d'état transition + orientée objet | Ne pas intégrer une méthode qui contraindrait les concepteurs à un ordre strict d'activités, ce qui empêcherait la démarche opportuniste nécessaire à la découverte de la décomposition de la solution ;<br>Supporter un accès et un basculement rapides entre plusieurs outils pour représenter et manipuler différentes sortes d'objets et leurs représentations. Certains objets sont des exigences informelles, des informations sur le problème du domaine, des problèmes et critères sur le système et sur le processus, des décisions de conception exprimées dans une notation formelle ou semi-formelle.<br>Supporter une navigation facile entre ces objets, sans imposer un ordre prédéterminé des activités, tout en supportant un agenda des activités du concepteur ;<br>Supporter une édition facile et une réorganisation des exigences, des problèmes de conception, des décisions ;<br>Supporter l'identification de l'origine des exigences : initialement présentes, contraintes inférées et ajoutées.<br>Supporter la représentation d'objets intermédiaires ou partiels dans la décomposition de la solution. |
| (Visser 1990) | 1 entreprise<br>1 projet d'installation d'une machine outil | 1 | Think aloud<br>Pendant 3 semaines | L'activité réelle de l'ingénieur dévie du plan initial basé sur une décomposition structurée et hiérarchique du problème.<br>Le caractère opportuniste du processus est lié aux données disponibles à un moment donné : connaissances de l'ingénieur, informations à sa disposition, état de la conception et de sa représentation. | Schémas de séquence + Grafcet pour la spécification finale | L'outil doit permettre d'abandonner la solution du problème à un certain niveau, afin de traiter des éléments de solution à un autre niveau, et pouvoir reprendre l'état de la solution au niveau abandonné |

| Etude | Nbre d'entreprises et de projets | Nbre participants | Méthode de collecte | Problèmes | Représentations et outils utilisés | Recommandations outils et méthode |
|---|---|---|---|---|---|---|
| (Sumner 1995) | 1 entreprise de développement logiciel d'applications à dialogue vocal | = | Etude longitudinale | Une multitude d'outils utilisée par les concepteurs => les changements et la cohérence entre représentations sont assurées manuellement par des copier-coller. | FlowChart<br>Tableau pour chaque entité du diagramme<br>Plans de test en texte<br>Outils à vocation générale | possibilité de pouvoir paramétrer les outils (tuning pour faire évoluer les outils)<br>pouvoir créer des liens automatiques entre les outils, à partir d'abstraction pour identifier les dépendances entre représentations |
| (Nguyen et Swatman 2000)<br>(Nguyen, Carroll, et Swatman 2000) | 1 projet | 3 | Recherche active | La nature opportuniste du processus d'ingénierie des exigences<br>Une dynamique de la complexité des modèles en dents de scie | Méthode orientée objet- FOOM<br><br>IBIS/QOC pour enregistrer le rationale | des outils pour surveiller la dynamique de la complexité des modèles pour les managers. Les outils doivent assister les concepteurs dans l'enregistrement de rationale et l'édition des modèles. |
| (Graaf, Lormans, et Toetenel 2003) | 8 entreprises de logiciel embarqué » | 36 | interviews | Gestion des exigences problématique, dans l'édition des des liens<br><br>Problème d'interprétation des diagrammes sans sémantique<br><br>Faible adoption des nouvelles méthodes d'ingénierie | Exigences textuelles avec traitement de texte et feuille de calcul pour la traçabilité<br>Diagrammes UML avec outil de dessin<br>Pas d'usage de méthode formelle | Besoin de flexibilité dans les outils |
| (Karlsson et al. 2007) | 8 entreprises orientées marché | 14 | Interviews + 1 focus group | Volatilité des exigences<br>Un manque d'outils supportant un processus flexible<br>Duplicatas d'exigences<br>Des problèmes de surcharge d'exigences et de gestion des priorités<br>Fournir des retours aux clients sur leurs suggestions d'évolutions est coûteux | Exigences textuelles<br>Et diagrammes UML | Pas de recommandations formulées sur les outils, mais volonté de prolonger l'étude en posant des questions spécifiques sur les outils et la façon de gérer la traçabilité |

| Etude | Nbre d'entreprises et de projets | Nbre participants | Méthode de collecte | Problèmes | Représentations et outils utilisés | Recommandations outils et méthode |
|---|---|---|---|---|---|---|
| (Sikora, Tenbergen, et Pohl 2011) | 7 entreprises de logiciels embarqués | 17 | Interviews + questionnaire | Faible utilisation des modèles<br>Difficulté dans la gestion des exigences et de la traçabilité entre différents niveaux d'abstraction, outils existants inadaptés<br>Beaucoup de temps passé dans la vérification des critères qualité type DO178B des exigences<br>Difficulté de faire les liens entre exigences et design | Exigences textuelles principalement<br><br>Les modèles comme support à la communication | Pas de recommandations sur les outils, mais sur des guides méthodologiques pour favoriser l'utilisation des modèles qui permettrait d'augmenter le niveau d'automatisation des tâches de vérification et de traçabilité. |
| (Nair et al. 2015) | 11 domaines critiques | 52 | questionnaire | techniques de vérification manuelle sur la complétude de la preuve et l'analyse d'impact d'une modification | Exigences textuelles dans des documents structurés : plans de tests, matrices, spécification comme preuves de conformité | la provision de preuve est un challenge important => il est nécessaire de mener une étude plus approfondie pour savoir quels outils sont adaptés aux activités relatives à la sécurité et à quel moment posent-ils des problèmes |
| (Ghazi et Glinz 2016) | 29 entreprises | 29 | Interviews par vidéoconférence | Problèmes d'interface utilisateur : passage entre artefacts, travail sur une fenêtre trop petite, changement de focus, connaissance des relations entre artefacts, recherche de la bonne fenêtre, arrangement des fenêtres, mémorisation, recherche du focus courant | Aucune indication précise sur les artefacts et les outils utilisés | Améliorer l'interface utilisateur pour gérer de grands et multiples artefacts |

Tableau 2: synthèse de l'état de l'art sur les études de terrain

# II.3. Outils d'ingénierie des exigences : Utilisabilité et visualisations

Dans un premier temps, nous précisons les notions d'utilisabilité et de flexibilité d'un système. Dans un deuxième temps, nous proposons un état de l'art sur les travaux relatifs à la définition et l'évaluation de l'utilisabilité des outils et des visualisations en ingénierie des exigences. Nous concluons en dégageant les opportunités de recherche.

## II.3.1. Utilisabilité et Flexibilité

Le génie logiciel reconnait l'utilisabilité comme un attribut de qualité pour tout produit logiciel dans le standard ISO/IEC 9126 (« ISO/IEC 9126-1:2001 - Software engineering -- Product quality -- Part 1: Quality model » 2001) et dans sa révision ISO/IEC 25010:2011 pour tout système (« ISO/IEC 25010:2011 - Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models » 2011).

Parmi les huit attributs de qualité d'un système, l'utilisabilité d'un système est définie par les caractéristiques suivantes : identification appropriée (ou compréhensibilité), facilité d'apprentissage, opérabilité (contrôlabilité, tolérance aux erreurs et conformité avec les attentes de l'utilisateur), protection des erreurs, esthétique de l'interface utilisateur et accessibilité.

Nous retrouvons dans les caractéristiques de l'utilisabilité les règles proposées par la littérature en IHM. Norman (Norman 2014) propose quatre règles : la visibilité (en regardant l'utilisateur peut dire l'état du produit et les actions alternatives possibles), un bon modèle conceptuel (cohérence dans la présentation des opérations et leurs résultats), de bonnes correspondances (possibilités de déterminer les relations entre actions et résultats, les contrôles et leurs effets), des retours continus sur les résultats des actions (feedback). Constantine et Lockwood (Larry L. Constantine et Lockwood 1999) définissent cinq règles à respecter : l'accès direct sans aide ou formation, l'efficacité pour un utilisateur spécialisé, la progression dans l'usage avec l'expérience, le support au travail réel des utilisateurs, l'importance du contexte d'utilisation. On trouve la flexibilité d'opération comme critère d'utilisabilité dans les règles de progression et de support au travail de Constantine et Lockwood. En effet, la règle de la progression précise que le système doit faciliter un avancement continu dans la connaissance et les compétences, et permettre des changements progressifs dans l'usage pour un utilisateur expérimenté. De plus, la règle du support au travail réel précise que le système doit rendre le travail plus facile, plus simple et plus rapide ou rendre des nouvelles choses possibles. La flexibilité est particulièrement demandée par les experts, qui réalisent des tâches complexes et sophistiquées avec une utilisation non-standard :

> *Experts need interfaces that operate in multiple modes tailored for efficiency on a variety of tasks and that can be changed frequently and easily to fit highly particular demands of the task at hand.(Larry L. Constantine et Lockwood 1999)*

Enfin, l'interaction sans mode et les actions réversibles préconisées par Nielsen (Nielsen 1993) et Tesler (Tesler 2012) vont dans le sens de ne pas fixer de façon rigide les actions de l'utilisateur, donc de la flexibilité. Harris et Henderson (Harris et Henderson 1999) dénoncent des flux de travail « gelés » par l'informatisation qui impose aux utilisateurs des règles prédéterminées pour assurer la cohérence et la coordination des activités. Les conséquences sont des mécanismes

d'accommodation mise en place par les utilisateurs pour la réalisation des activités à l'extérieur du système et donc invisibles.

Ce sont les mêmes observations que Suchman fait dans les activités de comptabilité et de tenues de compte (Suchman 1983) : les procédures permettent aux utilisateurs d'orienter leur travail, mais ne représentent pas la façon dont ils réalisent leur travail. La séquence d'actions requise n'est pas prédéterminée, mais adaptée aux situations particulières et aux contingences : l'action est située. Les préconisations sont de construire des systèmes qui aident les utilisateurs à traiter ces particularités (Suchman 1983)(Harris et Henderson 1999) et leur permettent de faire toutes les activités, pas seulement celles qui sont requises par les procédures et les normes. Pour cela, il faut s'intéresser aux accommodations réalisées par les utilisateurs, et les autoriser dans le système. L'intégration des accommodations permet leur enregistrement et peut guider de futures évolutions du système. On incorpore ainsi le processus du changement du système dans le système, ce qui facilite le respect de la cohérence requise par les normes et la coordination des activités (Harris et Henderson 1999).

Dourish (Dourish 1995), constatant la forte imbrication entre les pratiques de travail et le système en usage, défend l'idée que tout système doit proposer un cadre dans lequel les utilisateurs peuvent changer et adapter le système basique à leurs propres usages. Cette adaptation n'est pas seulement dynamique par rapport à un contexte pour les utilisateurs, mais aussi sur le long terme pour les concepteurs et développeurs : il faut des techniques pour intégrer la conception du système pendant tout le cycle de vie du système. Le paramétrage des systèmes encourage un phénomène de co-adaptation entre les utilisateurs et le système (Mackay 1990). Cependant, le paramétrage nécessite du travail et du temps, et se perd d'une version de logiciel à l'autre (Dourish 1997). Pour répondre à ce problème, Dourish propose un modèle réflexif du système, comprenant des abstractions de niveau méta qui, dans un premier temps, permettent de paramétrer le fonctionnement du système avant usage. Dans un deuxième temps, en rendant ces mêmes abstractions disponibles via l'interface utilisateur, cela rend le système flexible à l'usage. Ainsi, il s'agit de concevoir le système interactif en réfléchissant aux représentations informatisées de ces abstractions qui sont les ressources de l'action. Il applique cette approche de méta-niveau pour construire une boite à outils pour des applications de travail collectif (collecticiels), Prospero (Dourish 1998). Il introduit des techniques de méta-niveau de divergence/synchronisation de données distribuées, et de garantie de cohérence entre les données. Il montre que l'utilisation de ces deux techniques peut permettre un support à un travail collaboratif de nature opportuniste. A partir de ces mêmes constats, Beaudouin et MacKay (Beaudouin-Lafon et Mackay 2000) proposent trois principes de génération d'interfaces visuelles : la réification, le polymorphisme et la réutilisation.

Ces travaux sont essentiellement centrés sur le travail de bureau, et donc particulièrement adaptés au travail d'ingénierie des exigences, à la fois régi par les normes décrites dans le chapitre 1, et en demande de flexibilité d'après l'état de l'art sur les études de terrain (II.2). Cependant, le courant de pensée de l'ingénierie de résilience (Leveson 2002) prolonge cet intérêt dans le domaine des systèmes critiques : l'adaptation des procédures pour faire face à des contextes particuliers et des pressions est un facteur d'accident. Elle apparait inévitable tant les contextes sont nombreux et changeants. Leveson en conclut qu'il faut changer de vision, et contrôler le comportement en donnant des opportunités de développer des mécanismes d'adaptation (coping skills). Cela passe notamment par la conception de systèmes qui supportent une adaptation sûre des performances en réponse à des contextes, et permettent d'évaluer les conséquences et la dangerosité des décisions individuelles en rendant les erreurs observables et les actions réversibles. En résumé, la flexibilité du système ne se réduit pas à du paramétrage du système avant usage : elle dépend de l'utilisabilité du système interactif définie selon des principes à respecter (Norman 2014)(Nielsen 1993)(Larry L. Constantine

et Lockwood 1999), et de la prise en compte du travail réel des utilisateurs nécessitant des adaptations (Suchman 1983)(Mackay 1990)(Dourish 1997)(Harris et Henderson 1999)(Leveson 2002). Ce sont pour ses raisons que l'utilisabilité d'un système est difficile à spécifier (Juristo, Moreno, et Sanchez-Segura 2007), les principes généraux d'utilisabilité n'intégrant pas les tâches et les actions spécifiques que les utilisateurs du système veulent réaliser.

La norme ISO 9241-11 dédiée à l'utilisabilité propose la définition suivante de l'utilisabilité :

> *Degré selon lequel un produit peut être utilisé, par des utilisateurs identifiés, pour atteindre des buts définis avec efficacité (précision et degré d'achèvement selon lesquels l'utilisateur atteint des objectifs spécifiés), efficience (rapport entre les ressources dépensées et la précision et le degré d'achèvement selon lesquels l'utilisateur atteint des objectifs spécifiés) et satisfaction (Absence d'inconfort, et attitudes positives dans l'utilisation du produit), dans un contexte d'utilisation spécifié (Utilisateurs, tâches, équipement matériel, logiciel et documents) et environnements physique et social d'utilisation d'un produit).*

Elle précise que, pour spécifier l'utilisabilité, il est nécessaire d'identifier les objectifs et de décomposer l'efficacité, l'efficience, la satisfaction et les éléments du contexte d'utilisation en sous-composants dotés d'attributs mesurables et vérifiables. D'autre part, six styles sont proposées par Lauesen (S. Lauesen 1998) pour spécifier les exigences d'utilisabilité de façon précise et s'assurer de la prise en compte de l'utilisabilité dans la spécification d'un système : performance (spécification du temps de la tâche), défaut (spécification des problèmes qui conduisent l'utilisateur à faire une erreur), subjectif (satisfaction de l'utilisateur), processus (spécification du processus à mettre en place pour garantir l'utilisabilité, tel qu'un processus itératif et basé sur du prototypage), design (spécification de l'interface utilisateur avec des copies d'écran) et recommandation générale, telle que celles présentées précédemment (Norman 2014)(Nielsen 1993)(Larry L. Constantine et Lockwood 1999). Nous proposons de combiner ces deux définitions pour obtenir une vue complète de la spécification de l'utilisabilité comme illustrée en Figure 37. Dans ce cadre, une exigence d'utilisabilité est formulée en suivant la syntaxe suivante :

Afin d'atteindre [but], le système permet à [utilisateur] de réaliser [tâche]

avec [mesure d'utilisabilité] [valeur].

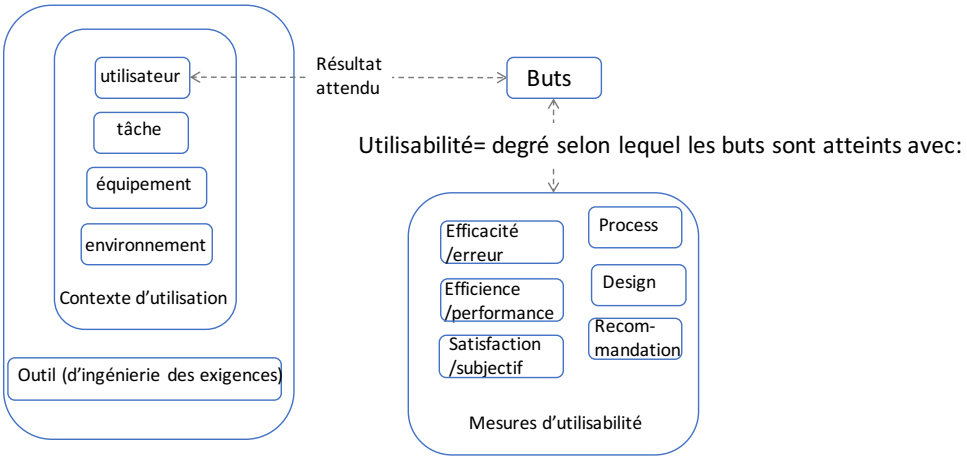



Figure 37: cadre de spécification de l'utilisabilité

Nous retenons (Harris et Henderson 1999)(Suchman 1983) la nécessité d'observer plus particulièrement les tâches d'accommodation des utilisateurs afin de les intégrer dans le système en usage sous forme d'exigences d'utilisabilité et garantir sa flexibilité (*pliancy*). Nous retenons de (Dourish 1997)(Dourish 1998) le principe de réflexion du système : l'enjeu est d'identifier et représenter des abstractions qui facilitent le paramétrage du système avant usage pour les développeurs, et permettent l'évolution des usages pour les utilisateurs.

## II.3.2. Les outils proposés par les éditeurs de logiciel

Il existe de nombreux outils d'ingénierie des exigences proposées par les éditeurs de logiciels : Alexander (Alexander s. d.) propose une liste d'outils disponibles, qui en énumère 63 (à la date de rédaction de la thèse), allant d'outils gratuits disponibles en téléchargement à des outils propriétaires avec licence, en passant par des plates-forme web collaboratives avec abonnement. Face à cette multiplication des outils, le rapport technique ISO/IEC TR 24766 (ISO/IEC 2009) propose une aide à la comparaison et à l'évaluation des outils pour les entreprises dans un processus d'achat. Dans cet objectif, il décrit les capacités attendues des outils d'ingénierie des exigences, et propose de classer ces capacités selon huit catégories : élicitation, analyse, spécification, modélisation, vérification et validation, management, traçabilité, et « autres ». Nous trouvons l'utilisabilité dans la catégorie « autres », avec les aspects relatifs au collecticiel et à l'interface graphique. La collaboration sur les exigences est envisagée à travers la gestion des accès aux données : gestion des rôles et des groupes utilisateurs, accès aux données en lecture ou écriture selon les rôles. Les aspects relatifs à l'interface utilisateur sont du niveau des recommandations classiques d'utilisabilité. En effet, le rapport préconise que les outils fournissent des vues multiples (sans préciser lesquelles), des entrées et des manipulations interactives graphiques des données (sans préciser), une interface web, les fonctions de copier-coller, et des capacités classiques des systèmes d'exploitation tels que le multifenêtrage et l'accès concurrent à plusieurs fonctions. La société savante INCOSE a utilisé le cadre proposé par le rapport technique ISO/IEC TR 24766 pour conduire l'évaluation de 34 outils. Cependant la méthode est contestable car ce sont les vendeurs d'outils qui ont évalué leur propre outil sur la base d'un questionnaire.

Carrillo et al. (Carrillo De Gea et al. 2012) utilisent également le rapport ISO/IEC TR 24766 pour conduire une évaluation de 38 outils, sur la base d'un questionnaire de 146 éléments, renseigné par les vendeurs. De plus, ils réalisent une analyse statistique des résultats, en les corrélant notamment au nombre de licences en utilisation. Les résultats montrent que les outils offrent le meilleur support à l'élicitation des exigences. L'analyse, la spécification et la traçabilité sont ensuite les activités les mieux supportées. Les activités de vérification et validation et autres capacités arrivent en troisième. Ce sont les activités de modélisation et de management qui sont le plus mal supportées. Les auteurs évaluent le score global suffisamment haut pour considérer que les outils offrent un bon support sur l'ensemble du processus d'ingénierie des exigences. En termes de modélisation, l'étude montre que les exigences textuelles sont bien supportées par les outils, alors que les artefacts SysML, les modèles orientés but, et les diagrammes de flot de données manquent de support outillé. Si nous mettons en regard ce résultat avec les pratiques industrielles observées, cela pourrait montrer que les vendeurs d'outils se sont alignés sur les pratiques des industriels. Sur les activités de gestion des exigences, les faiblesses détectées sont relatives à un modèle ouvert de données, afin de pouvoir opérer sur les données via des scripts et autoriser des applications externes à traiter les données. La capacité à faire des traces flexibles entre différents éléments (texte, graphique et table) est la capacité la moins bien notée sur la traçabilité. Sur les « autres capacités », dans laquelle sont inclues l'interface graphique et le support à la collaboration, l'intégration des données est notée comme un point faible. La corrélation significative entre le nombre de licences et les « autres capacités » suggère que les

utilisateurs ont un intérêt dans les capacités transverses aux activités telles que l'administration de l'outil, l'interface utilisateur et l'intégration des données. L'étude conclut sur une seule recommandation claire sur les outils, relative à des mécanismes permettant de combiner de l'information de plusieurs systèmes, surtout quand l'environnement de travail est distribué.

Les mêmes auteurs (Carrillo de Gea et al. 2014) proposent une révision du cadre proposé par ISO/IEC TR 24766 en mettant en avant 3 catégories de capacités : modélisation, traçabilité et collaboration. Sur la base d'un questionnaire de 168 questions, ils ont obtenu l'évaluation de 29 outils d'ingénierie des exigences par les vendeurs. Une analyse statistique des réponses leur permet de mettre en évidence 3 clusters d'outils, par rapport à la couverture des capacités. Le premier cluster correspond aux outils couvrant toutes les capacités. Le deuxième cluster est proche du premier et couvre de façon un peu moins complète les capacités de spécification et de traçabilité, et beaucoup moins les capacités de modélisation. Le troisième cluster couvre de façon beaucoup moins complète l'ensemble des capacités (voir Tableau 3).

| Cluster | Nombre | Outils |
|---|---|---|
| C1 | 18 | Aligned elements, Bright Green Projects, Cognition Cockpit, Cradle, G-Marc, inteGREAT, MKS Integrity, PACE, Polarion Requirements, Psoda, RaQuest, DOORS, ReqMan, Reqtify, Requirements Composer, RTIME, TopTeam Analyst, TraceCloud |
| C2 | 7 | Avenqo, Caliber RM, Cameo Requirements+, RequisitePro, SpiraTeam, TestTrack RM, TrackStudio |
| C3 | 4 | Leap SE, QFDcapture, RMTrak, TrackStudio |

Tableau 3 : clusters d'outils issus de l'analyse statistique de (Carrillo de Gea et al. 2014)

Nous constatons que plus de la moitié des outils (18) proposés par les éditeurs de logiciel couvre l'ensemble des capacités attendues pour l'ingénierie des exigences. Par conséquent, ces études ne nous permettent pas d'expliquer le constat des études de terrain sur la faible adoption des outils spécialisés. Cependant, les auteurs éliminent de façon volontaire les attributs de qualité de logiciel, et notamment l'utilisabilité des outils, dans les capacités évaluées.

Dans la suite de ce paragraphe, nous proposons une évaluation experte de l'utilisabilité des outils présents dans la liste de Alexander (Alexander s. d.), à partir des informations proposées sur les sites web des éditeurs. Nous avons notamment étudié le vocabulaire utilisé sur le site web, des captures d'écran de l'interface graphique, et la mise en valeur de la flexibilité et de la facilité d'utilisation. Tout d'abord, il existe des outils qui sont liés à une méthode et une expression des exigences telles que nous les avons présentées dans la partie Les différentes expressions des exigences. Par exemple, jUCMNav est un plugin Elcipse gratuit adapté à l'utilisation de la notation URN (http://jucmnav.softwareengineering.ca/foswiki/ProjetSEG), Objectiver est un outil adapté à la méthode KAOS (http://www.objectiver.com/) (voir Figure 38).

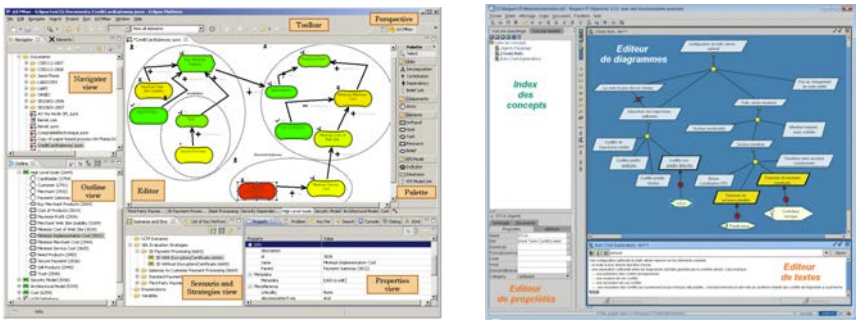



Figure 38: des outils spécifiques aux méthodes orientées but: JUCMNav à gauche, Objectiver à droite

Papyrus (http://www.eclipse.org/papyrus/index.php#applications) est un outil graphique de modélisation des exigences utilisant la notation UML (Figure 39), et propose des capacités de configuration pour s'adapter aux « besoins de l'entreprise » (« Fully customizable environment »).

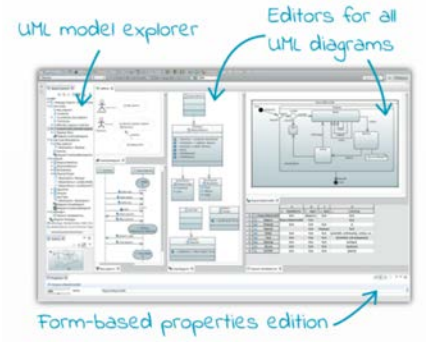



Figure 39: Papyrus, un plugin eclipse pour la modélisation UML/SysML

Au contraire, d'autres outils mettent en avant la capacité de l'outil à gérer des exigences textuelles et des représentations graphiques ainsi que la traçabilité entre artefacts de différente nature. Par exemple, Visual Trace Spec propose des vues multiples dans des onglets séparés, sous forme arborescente, tabulaire et matricielle (voir Figure 40). Vissure Requirements (www.visuresolutions.com) repose sur le même principe et promeut la flexibilité du processus que l'outil permet de mettre en place, grâce à un méta-modèle configurable (voir Figure 41).

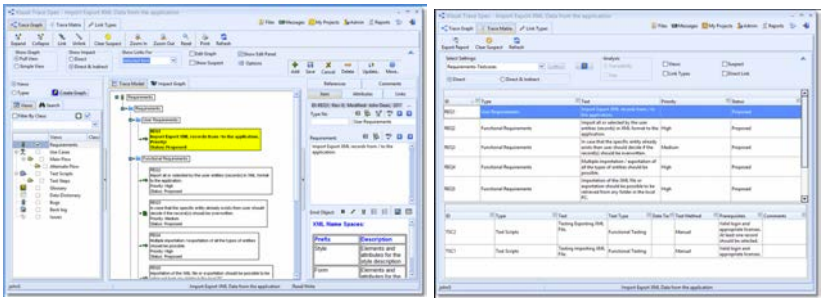


Figure 40: captures d'écran de Visual Trace Spec

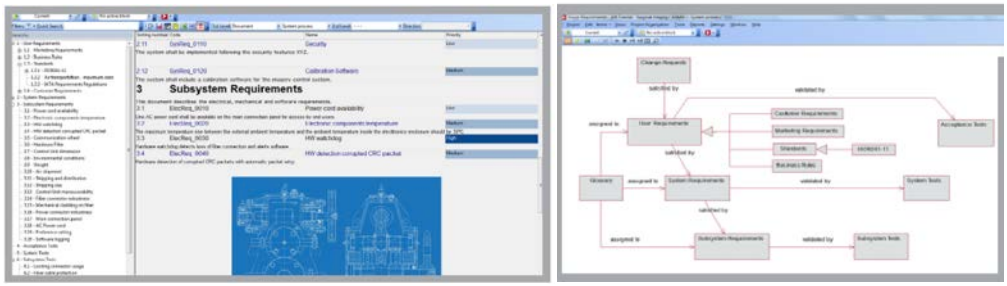


Figure 41: Vissure requirements: gestion d'exigences textuelles et graphiques

Sur ce même principe de gestion d'artefacts textuels et graphiques, une nouvelle génération d'outils promeut la visualisation des exigences et la gestion flexible des traces tels que Yakindu (https://www.itemis.com/en/yakindu/traceability/ **:** "Link your tools according to your process standard") et BluePrint **(** http://www.blueprintsys.com/blog/how-visualization-can-transform-your-it-project **: «** How Visualization Can Transform Your IT Project ») (voir Figure 42).

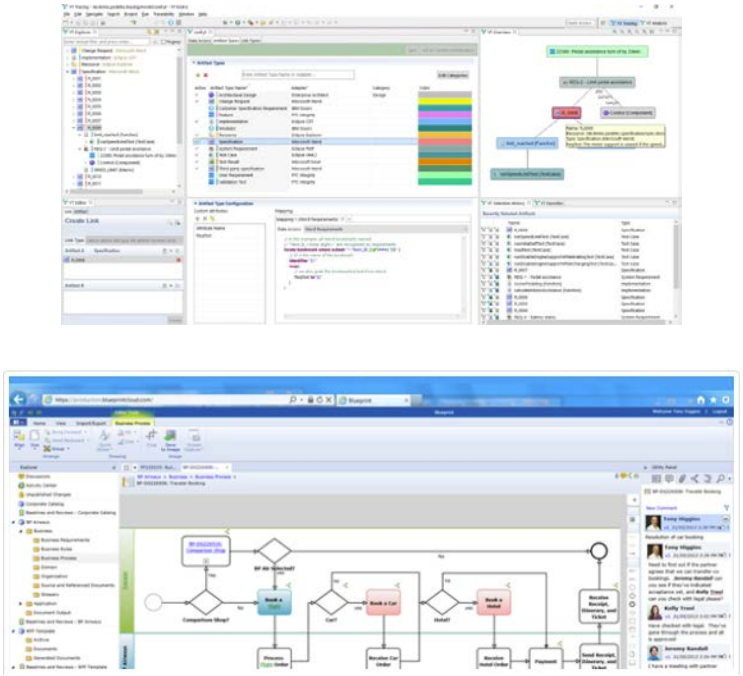


Figure 42: captures d'écran de Yakindu et BluePrint Software montrant la visualisation de plusieurs types d'artefacts

Dans le domaine des systèmes critiques, les études que nous avons rapportées dans la partie II.2 montrent que l'outil DOORS (http://www-03.ibm.com/software/products/en/ratidoor) est largement répandu pour la gestion des exigences textuelles (voir Figure 43). Dassault Systèmes propose un outil complémentaire, Reqtify (https://www.3ds.com/products-services/catia/products/reqtify), centré sur l'analyse des traces entre exigences, cas de tests et résultats, à partir de documents et données issus de sources extérieures (voir Figure 44).

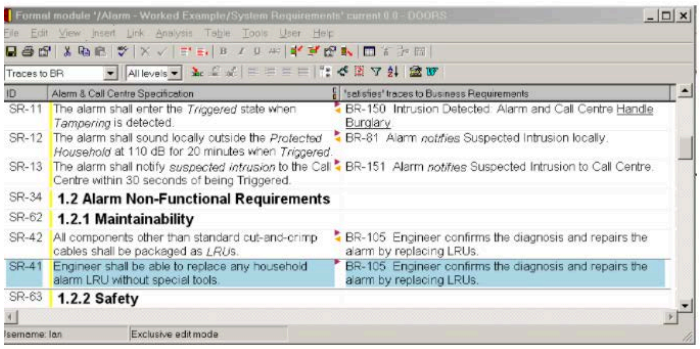



Figure 43: gestion des exigences textuelles sous DOORS

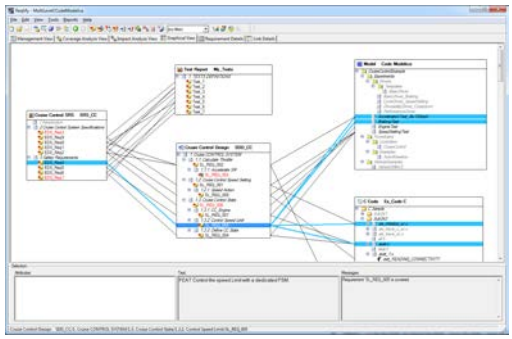


Figure 44: visualisation des traces sur Reqtify

Enfin, nous pouvons distinguer une dernière catégorie d'outils, bâtis sur des plateformes web collaboratives, tels que Irise (https://www.irise.com/), Jira (https://www.atlassian.com/software/jira), Yonix (http://www.yonix.com/) ou Jama (https://www.jamasoftware.com/solutions/). Cette catégorie s'inscrit dans le mouvement Agile, avec son vocabulaire reconnaissable de *blacklog* et *user stories* (Schwaber et Beedle 2001). Ces outils proposent des tableaux de bord type Kanban (Anderson 2010), avec de la manipulation directe pour déplacer les *user stories* d'un *blacklog* à l'autre (Figure 45), et de la collaboration autour *des user stories* (Figure 46).

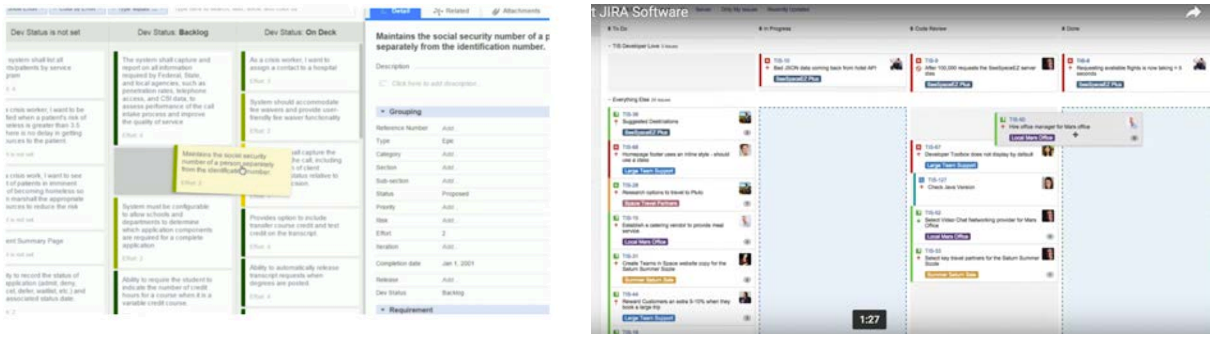



Figure 45: manipulation directe pour gérer le contenu d'un blacklog proposée par Irise (à gauche) et Jira (à droite)

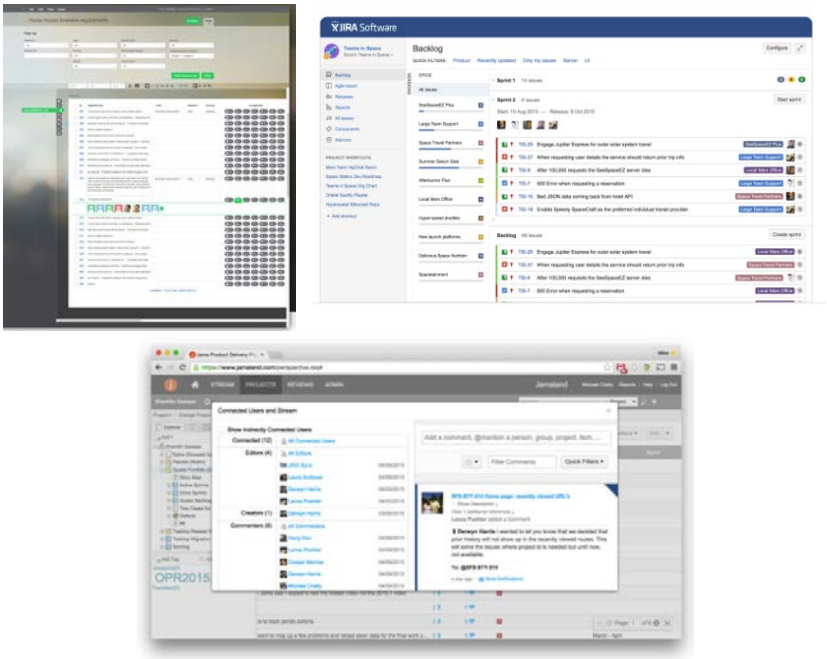



Figure 46: collaboration autour des user stories proposée par Yonix, JIRA et Jama

## II.3.3. Les outils évalués par les industriels

Les publications par des industriels relatives à l'utilisation d'outil sont rares : de fait, leur métier est de produire des systèmes, et la compétition entre industriels ne pousse pas à un partage d'expérience.

Sur la base de leur expérience industrielle dans des projets de développement de systèmes chez DaimlerChrysler, Hoffmann et al. (Hoffmann et al. 2004) affiche comme objectif de préciser les exigences pour les outils de gestion des exigences (c'est le titre de l'article). Les auteurs prennent comme cadre d'analyse le standard de qualité d'un logiciel ISO/IEC 9126 (qui a été replacé depuis par ISO/IEC 2510). Les auteurs présentent un catalogue d'exigences selon trois acteurs différents : le développeur, l'administrateur du projet, et l'administrateur de l'outil. Les auteurs insistent sur l'importance de l'interface utilisateur :

> *userfriendliness is generally an important requirement for tools in such an environment. A significant number of tools have received negative feedback from engineers because they lack a user friendly interface.*

Les auteurs n'utilisent pas le terme d'utilisabilité et annoncent que les exigences sont purement fonctionnelles. Cependant, ils émettent des exigences en termes de vues (« graphical views of the requirements should be available », « all users must be able to customize the standard views without changing the template », « these views must be freely configurable, including complex filters on objects, relations, and attributes »). De plus, les auteurs émettent une recommandation spécifique sur la traçabilité : « the tool must feature a practical, user-friendly and concise graphical representation and navigation of the traces, e.g, matrices, trees or graphs ». Ils expliquent cette recommandation par l'impopularité de la traçabilité auprès des développeurs, car coûteuse en temps et utile seulement dans les phases ultérieures.

L'utilisation de l'outil JIRA est décrite par une entreprise canadienne réalisant des systèmes de gestion d'information de passagers (Filion et al. 2017) : l'outil est utilisé pour gérer les liens entre les différents niveaux d'exigences (client, système, logiciel et matériel) et les tests associés. Cependant,

les auteurs ont besoin de réaliser des rapports de traçabilité, tels que le pourcentage d'exigences couvertes, le pourcentage d'exigences non couvertes, le pourcentage d'exigences testées et vérifiées, etc. Ils rapportent que l'outil JIRA ne permet pas de faire des audits ou de construire des vues graphiques et des matrices de traçabilité. Pour pallier ces problèmes, ils utilisent en complément de JIRA un outil de visualisation de données, easyBI (https://eazybi.com/features). L'outil permet d'importer des données de différents formats (excel, googlesheet, postegreSQL, oracle) et met en avant la manipulation directe des données par drag'n drop pour construire et explorer différents types de visualisations (jauge, tableau, barre, scatterplot, carte). Ce retour d'expérience montre l'intérêt de cet industriel pour les visualisations, dans des tâches d'analyse et d'audit des traces.

## II.3.4. Les langages visuels évalués et améliorés par les Physics of Notation

Les études de terrain que nous avons analysées dans la partie II.2 montrent une faible adoption des langages visuels proposés par la recherche en ingénierie des exigences, au profit d'exigences exprimées en langage naturel. Moody (D. Moody 2009) défend l'idée que les chercheurs ont sous-estimé l'importance de la représentation visuelle de ces langages. Il propose le cadre des « Physics of Notation » pour évaluer et concevoir des notations visuelles efficaces d'un point de vue cognitif. Il se base sur un modèle humain de la perception et de la cognition (Card, Newell, et Moran 1983) qui montre que les capacités de perception sont automatiques, très rapides et exécutées en parallèle, alors que les processus cognitifs interviennent de façon consciente, plus lentement et séquentiellement. Sur cette base, il propose d'améliorer les notations visuelles afin d'exploiter au mieux les capacités de perception de l'être humain. Il propose neuf principes interdépendants pour guider la conception et l'évaluation des notations visuelles : clarté sémiotique (une correspondance un à un entre construction sémantique et symbole graphique), discriminabilité perceptuelle (les symboles graphiques doivent être discernables), la transparence sémantique (la représentation visuelle suggère la signification), la gestion de la complexité (avec des mécanismes de modularisation et de niveaux d'abstraction), l'intégration cognitive (avec des mécanismes de contextualisation pour aider l'utilisateur à assembler de l'information et des indices perceptuels pour simplifier la navigation entre diagrammes), l'expressivité visuelle (en exploitant les capacités des variables visuelles définies par Bertin (Bertin 1983), présentées en Figure 47), le codage double en utilisant du texte pour compléter les graphiques), l'économie graphique dans le nombre de symboles graphiques utilisés, le principe d'adaptation cognitive selon les tâches et l'audience en distinguant les novices des experts.

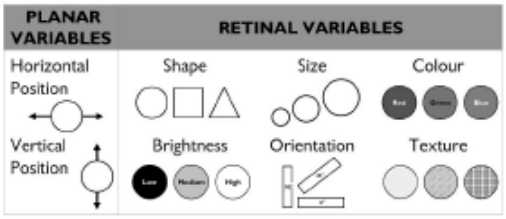



Figure 47: variables visuelles définies par Bertin (Bertin 1983)

Différents travaux ont été menés sur des langages visuels de l'ingénierie des exigences en utilisant ce cadre : le langage UML dans 13 types de diagramme (Daniel Moody et Hillegersberg 2008), le langage orienté-but i* (D. L. Moody, Heymans, et Matulevičius 2010)(Caire et al. 2013), la notation UCM (Genon, Amyot, et Heymans 2010), les statecharts (Anwer et El-Attar 2014), le langage orienté-but Tropos (Boucher s. d.). Une nouvelle notation visuelle est proposée (Laurent et al. 2010) pour modéliser des environnements distribués. Les analyses réalisées montrent que ces langages visuels

ont des sémantiques opaques. Cependant, Genon et al. (Genon, Amyot, et Heymans 2010) relèvent des difficultés dans la génération de propositions à partir des principes, d'autant plus que ces principes sont interdépendants et ne peuvent pas être tous optimisés à la fois. Ce sont les principes relatifs aux symboles graphiques (clarté sémiotique, discriminabilité, transparence sémantique, expressivité visuelle et économie graphique) qui permettent de générer des améliorations concrètes, en utilisant les variables visuelles de Bertin (Figure 47) comme espace de conception. Par exemple, des symboles graphiques révisés pour i* sont proposés par Moody et al. (D. L. Moody, Heymans, et Matulevičius 2010), comme illustré en Figure 48. L'évaluation de reconnaissance de symboles menée avec des novices (Caire et al. 2013) montrent que les symboles révisés ont une plus grande transparence sémantique que les symboles standards. Cependant, de l'aveu même des auteurs, cette évaluation reste limitée car elle ne concerne que la reconnaissance de symboles graphiques, et non la compréhension de diagrammes complets.


Figure 48: les symboles i* de départ (en haut), et les symboles issus des PoN (en bas) (D. L. Moody, Heymans, et Matulevičius 2010)

L'espace de conception proposé par les variables visuelles de Bertin a été utilisé par Conversy (Conversy 2014a) pour comprendre la perception de code exprimé en langage graphique et en langage textuel. Il propose de relier les propriétés d'expressivité des variables (sélectivité, ordre et quantité- voir Figure 49) avec les tâches de lecture et de compréhension du code.

| | shape | lum | color | pos | size | link | cont |
|---|---|---|---|---|---|---|---|
| selective | | ✓ | ✓ | ✓ | ✓ | | ✓ |
| ordered | | ✓ | | ✓ | ✓ | | ✓ |
| quantitative | | | | ✓ | ✓ | | |

Figure 49: expressivité des variables visuelles

Nous pouvons noter que la gestion de la complexité par la définition de niveaux d'abstraction est une propriété proposée par les pères fondateurs des langages visuels de spécification (Ross 1977)(DeMarco 1979)(Harel 1987)(Yourdon 2006), comme expliqué dans la partie Les différentes expressions des exigences. Enfin, les principes de gestion de la complexité, d'intégration cognitive et d'adaptation cognitive de représentation des données constituent un domaine à part entière, celui de la visualisation d'information (Card, Mackinlay, et Shneiderman 1999), et nécessitent de s'intéresser aux tâches des utilisateurs (Ben Shneiderman 1996).

## II.3.5. La visualisation d'information appliquée à l'ingénierie des exigences

Les travaux sur la visualisation en ingénierie des exigences ont émergé depuis une dizaine d'années, notamment grâce à un groupe de travail international spécifique entre 2006 et 2010 au sein de la

communauté de *Requirements Engineering* (IEEE 2006). Nous partons des revues de littérature pour se focaliser sur les propositions de visualisations interactives.

### II.3.5.1. Les revues de littérature

Winkler (S. Winkler 2008) se centre sur l'utilisabilité des visualisations des traces et réalise une revue de littérature des usages des traces afin d'identifier les différentes tâches et les différents utilisateurs. Le résultat est une liste de tâches croisée avec différents profils utilisateurs (Figure 50). La partie droite du tableau exprime des hypothèses de l'auteur sur l'accès aux données selon trois modalités (rapporter, rechercher et parcourir). Son objectif est de montrer la multitude d'usages et de rôles utilisateurs concernés, et les défis que cela pose en termes d'utilisabilité sur les visualisations.

| | user role | | | | | | | access | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Process Mgr. | Customer | Project Mgr. | Auditor | Req. Engineer | Designer | Maintainer | Report | Search | Browse |
| Prioritizing requirements | | × | × | | × | | | | | × |
| Estimating change impact | | × | × | | × | × | × | × | | × |
| Proving adequateness | | × | × | × | | | | × | | |
| Validating artifacts | | | × | × | × | × | | | × | × |
| Testing the system | | × | | × | | | | × | | |
| Supporting audits | | × | | × | | | | | × | |
| Improving changeability | | | | | × | × | × | | | × |
| Extracting metrics | × | | × | | | | | × | | |
| Monitoring progress | | | × | | | | | × | | |
| Assessing the process | × | | × | | | | | × | × | |
| Tracking rationale | | | | | × | × | × | | | × |
| Understanding the system | | | | | | × | × | × | × | × |
| Establishing accountability | | | × | | × | × | × | | | × |
| Documenting reengineering | | | | | × | × | | | | × |
| Reusing elements | × | | × | | × | × | | | | × |
| Extracting best practices | × | | | | | | | × | × | |

Figure 50: liste de tâches et des utilisateurs relatifs à la visualisation des traces (S. Winkler 2008)

Dans une revue ultérieure sur la traçabilité (Stefan Winkler et Pilgrim 2010), les auteurs montrent que la notion de trace regroupe des notions multiples de liens entre différents artefacts d'ingénierie en ingénierie des exigences, et qu'il n'y a pas de consensus sur les définitions. Ils regrettent l'absence de vue globale, les problèmes étant traités séparément par des sous-communautés de recherche, avec d'un côté le développement dirigé par les modèles, et de l'autre l'ingénierie des exigences. Ils préconisent une approche globale, centrée sur des solutions pratiques et un support outillé pour une application à des projets réels de l'industrie, dans lesquels de multiples artefacts sont utilisés. L'utilisabilité dans l'enregistrement et la visualisation des traces est pointée comme primordiale dans cette approche.

Dans leur proposition d'agenda de recherche sur la traçabilité du logiciel, Cleland-Huang et al (Cleland-Huang et al. 2014) identifient la visualisation des traces comme une direction à explorer, afin de disposer de meilleurs moyens pour la définition, la création, la maintenance et l'utilisation de l'information de traçabilité. En particulier, ils préconisent la réalisation d'études in-situ pour évaluer et comprendre les tâches et besoins spécifiques à la traçabilité, et le développement de nouvelles façons de présenter l'information aux différentes parties prenantes.

Cooper et al. (Cooper et al. 2009) élargissent le périmètre d'intérêt des visualisations et proposent un cadre d'analyse des visualisations à partir d'un processus unifié d'ingénierie des exigences. Les activités d'ingénierie des exigences sont similaires aux catégories proposées par ISO/IEC TR 24766 :

élicitation, modélisation et analyse, communication et négociation, vérification et validation. Nous notons cependant deux différences : les activités de négociation et communication sont identifiées comme des activités de premier plan, alors que les activités de management des exigences n'apparaissent plus de façon explicite. Ce cycle de vie des exigences est composé de 4 phases : contexte et mise en place, élaboration, raffinement, spécification, ces phases se succédant de façon chronologique (voir Figure 51).

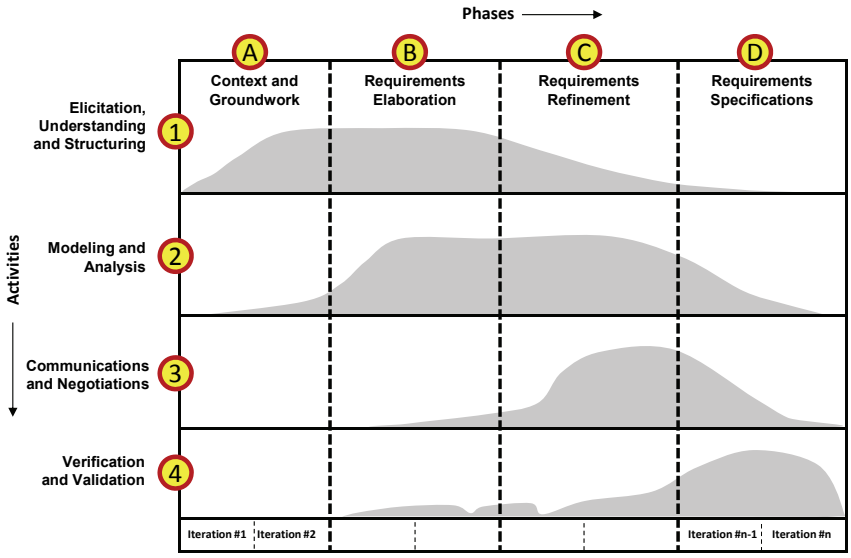



Figure 51: un processus unifié d'ingénierie des exigences (Cooper et al. 2009)

Les auteurs utilisent ce cadre pour évaluer les visualisations issues de 29 articles de recherche, et catégorisent les visualisations : tabulaire, relationnelle, séquentielle, hiérarchique, quantitative/métaphorique. Dans leur effort de catégorisation, les auteurs se rendent compte du recouvrement entre activités et phases, lié notamment à la nature itérative du travail fait sur les exigences. Ils notent une prévalence des visualisations de type relationnelle et hiérarchique, sur les phases de raffinement et de spécification. C'est pour la phase d'élaboration des exigences que la plus grande variété de visualisation est constatée (séquentielle, tabulaire, hiérarchique et relationnelle). En revanche, peu de visualisations sont proposées pour les activités de vérification et validation des exigences. Les auteurs concluent en préconisant le développement de visualisation pour les tâches de vérification et validation, et l'ouverture vers des visualisations quantitatives/métaphoriques, sans formuler de recommandations plus précises. Ils préconisent également d'aller vers une meilleure compréhension des tâches à instrumenter à travers les visualisations, conscients des limites de leur cadre d'évaluation. L'interaction sur les visualisations n'est pas du tout prise en compte dans leur évaluation.

Niu et al. (Niu, Reddivari, et Chen 2013) (Reddivari 2013) identifient l'interaction sur les visualisations d'exigences comme un moyen d'exploration, de découverte de connaissances et de raisonnement efficace. Ils proposent un cadre issu du domaine de *Visual Analytics* (Keim et al. 2008), augmenté par l'ajout de l'utilisateur comme acteur principal dans la boucle de découverte de connaissances (voir Figure 52). Les travaux récents en *Visual Analytics* passent en effet du paradigme de « l'humain dans la boucle » au paradigme « l'humain est la boucle » (Endert et al. 2014), reconnaissant les problèmes d'utilisabilité qui peuvent survenir dans la multiplication d'algorithmes déconnectés qu'il faut paramétrer.

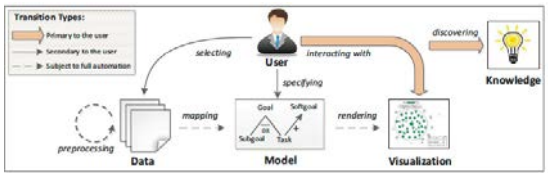



Figure 52: un cadre de *visual analytics* pour l'ingénierie des exigences (Niu, Reddivari, et Chen 2013)

Le cadre proposé ne fait pas référence aux activités d'ingénierie des exigences, et se centre sur un processus de visualisation de données par l'utilisateur: il sélectionne des données, qui correspondent à un « modèle » du système. Il spécifie le « modèle », qui est transformé en visualisation. L'utilisateur peut alors interagir sur la visualisation pour comprendre et découvrir les données. L'interaction sur la visualisation est proposée comme de première importance pour l'utilisateur (Figure 52). Nous pensons que l'utilisation du terme « modèle » est à prendre avec précaution, car il désigne un modèle de données dans le domaine de *Visual Analytics*, alors qu'il désigne les différentes représentations des exigences, telles que nous les avons décrites en partie  Les différentes expressions des exigences dans le domaine de l'ingénierie des exigences.  La flèche « specifying » entre l'utilisateur et le modèle en Figure 52, ainsi que la représentation graphique représentant un modèle orienté but, ajoute de la confusion. Une évaluation de huit articles de recherche relatifs à la visualisation des exigences est réalisée par les auteurs en utilisant ce cadre, associé à des questions opérationnelles pour chaque élément du cadre (utilisateur, données, modèle, visualisation et connaissance). L'origine des questions opérationnelles n'est pas explicitée. En particulier, pour l'utilisateur, les questions proposées (rôles multiples, performance temps-réel, intégration à un environnement de développement existant, guide d'utilisation) ne sont pas explicitement reliées à des principes d'utilisabilité connus (partie Utilisabilité et Flexibilité) comme ou à des besoins issus des industriels, tels que la flexibilité. Leur évaluation montre que les aspects de visualisation, en termes de vues multiples, navigation entre vues, recherche, filtrage, requêtes et annotation sont peu couverts par les huit articles de recherche. Les aspects liés à la découverte de connaissances, tels que la détection d'anomalies, les explications détaillées, l'aide au raisonnement, sont également peu traités. Les auteurs ne formalisent pas les résultats sous forme d'exigences à remplir par les visualisations et les interactions, et ne relient pas ces exigences à des activités d'ingénierie des exigences.

Une revue systématique de littérature sur la visualisation en ingénierie des exigences est proposée par Abad et al. (Abad, Noaeen, et Ruhe 2016), afin d'analyser et classifier les différentes techniques de visualisations selon plusieurs dimensions de l'ingénierie des exigences : les parties prenantes, le domaine et les activités. Ils ont au départ identifié 559 articles ; seuls 26 articles sont retenus pour une analyse de type « grounded theory ». Pour analyser les visualisations, les auteurs utilisent un cadre de visualisation des connaissances (Burkhard 2005), différent de la visualisation d'information (p243-244),  notamment sur la place de l'interaction, qui n'est qu'un type de visualisation parmi d'autres (voir Figure 53).

| FUNCTION | KNOWLEDGE TYPE | RECIPIENT | VISUALIZATION TYPE |
|---|---|---|---|
| Coordination | Know-what | Individual | Sketch |
| Attention | Know-how | Group | Diagram |
| Recall | Know-why | Organization | Image |
| Motivation | Know-where | Network | Map |
| Elaboration | Know-who | | Object |
| New Insight | | | Interactive Visualization |
| | | | Story |

Figure 53: un cadre de visualisation des connaissances (Burkhard 2005)

Les résultats de leur analyse montrent qu'il n'y a pas de proposition de visualisations supportant l'ensemble du cycle de vie d'ingénierie des exigences, et qu'il y a un réel besoin de développer des visualisations de type *map*, *storytelling* et interactives. Nous retenons des revues de littérature un intérêt croissant pour les visualisations interactives, sur lesquelles nous centrons plus particulièrement la suite de cet état de l'art, à partir des travaux étudiés par les revues de littérature et en les étendant.

### II.3.5.2. Les propositions de visualisations interactives

Feather et al. (Feather et al. 2006a) explorent plusieurs visualisations en support à une approche d'analyse de risques dans le cadre de projets de la NASA. Ils présentent une succession de visualisations en support à la prise de décision en termes de risques, coûts et bénéfices de solutions envisagées pour répondre à des exigences : bar chart, treemap, 2D-chart, kiviart (ou radar), matrice, courbe coût-bénéfice (voir Figure 54). Les auteurs parlent de leurs propres besoins, et ont l'avantage d'avoir des données réelles, caractérisées par de nombreuses exigences. Cela les oriente vers l'exploration de visualisations quantitatives. La démarche des auteurs est de montrer comment une juxtaposition de visualisations permet d'exploiter leur modèle de calcul et de simulation de mitigation de risques. Les visualisations ne sont pas interactives, ni interconnectées. Cooper et al. (Cooper et al. 2009) identifie une opportunité d'étendre l'utilisation de ces vues à d'autres tâches, notamment les tâches de vérification et validation.

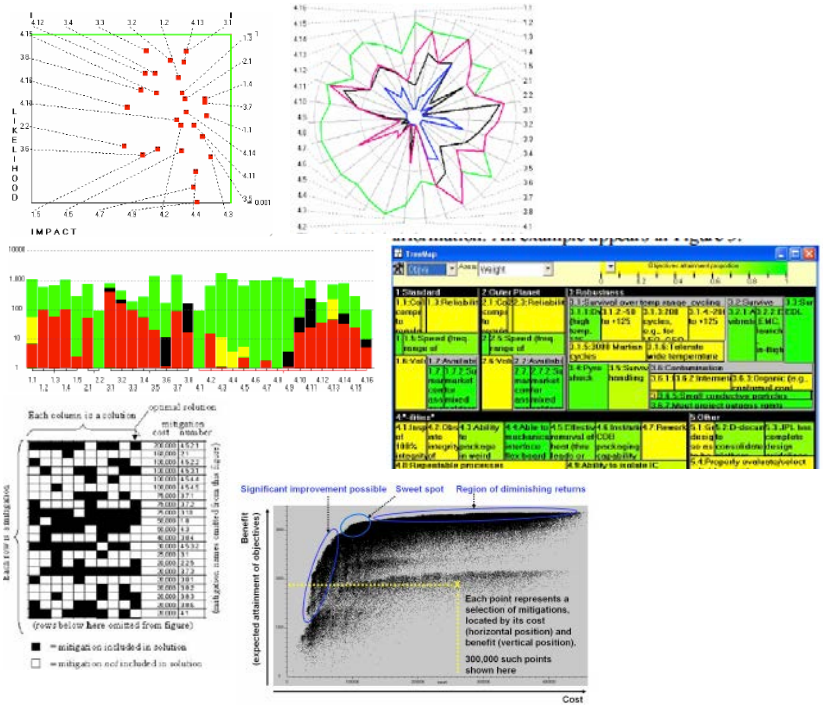



Figure 54: exploration de visualisations quantitatives pour une prise de décision basée sur un modèle de risques

Duan et Cleland-Huang (Duan et Cleland-Huang 2006) proposent d'automatiser la génération de liens entre artefacts par une approche probabiliste basée sur la fréquence des termes. Les auteurs proposent de visualiser les résultats sous forme de matrice, sous laquelle est ajoutée un histogramme représentant une métrique de conception sélectionné par l'utilisateur. Les matrices de la Figure 55 montrent les liens candidats (trouvés automatiquement) entre exigences et composants logiciels, et l'histogramme montre le degré de couplage et cohésion entre composants. Les auteurs explorent l'utilisation de cette matrice sur quatre projets. La matrice est surtout utilisée pour évaluer la validité

des liens candidats proposés par l'algorithme, dont l'utilisateur peut contrôler le seuil par un *slider*. Les auteurs relèvent un problème de passage à l'échelle sur le projet le plus conséquent (voir à droite de la Figure 55), et envisagent d'utiliser la technique de *fisheye* pour le régler dans de futurs travaux.

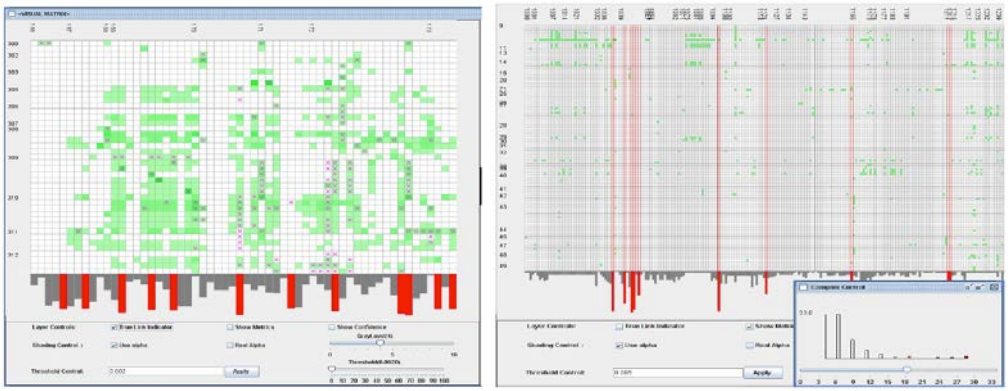


Figure 55: VisMatrix: visualisation des liens probables entre artefacts (Duan et Cleland-Huang 2006)

Kamalrudin et al. (Massila Kamalrudin, Hosking, et Grundy 2011) proposent d'automatiser la génération de cas d'utilisation essentiels à partir d'exigences exprimées en langage naturel, et intègrent ce traitement dans un outil. Quand l'utilisateur sélectionne une phrase dans le texte, le cas d'utilisation correspondant est mis en évidence. Les ingénieurs peuvent modifier les exigences textuelles ou les cas d'utilisation, et l'outil met à jour l'autre forme ou signale des incohérences (voir Figure 56). Les auteurs conduisent une évaluation de l'utilité et l'utilisabilité de l'outil avec 11 étudiants en génie logiciel de dernière année, en leur demandant de faire des exercices d'extraction des cas d'utilisation, puis de modification. Pour l'évaluation, ils utilisent un questionnaire classique sur l'utilité, la facilité d'utilisation, la facilité d'apprentissage et la satisfaction, et un questionnaire simplifié issu des dimensions cognitives des notations (Blackwell et Green 2012) évaluant la visibilité, la viscosité, la diffusion, l'effort mental, la propension à l'erreur, la proximité de correspondance, la cohérence, les dépendances cachées, l'évaluation progressive, l'engagement prématuré. Les résultats sont positifs sur l'utilité et l'utilisabilité, entre 80 et 90%. Les auteurs utilisent les dimensions cognitives pour montrer que leur outil automatique d'extraction permet une meilleure compréhension et application des cas d'utilisation essentiels, évalués comme difficiles à comprendre par des études antérieures.

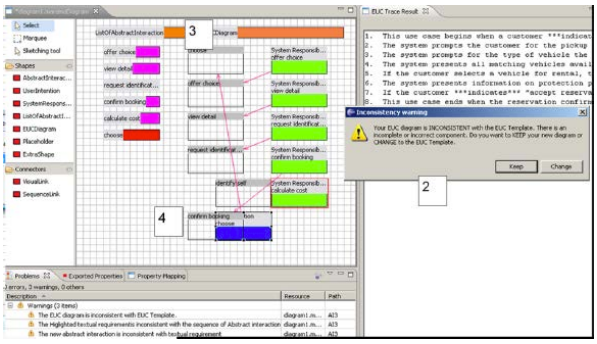


Figure 56: une vue des exigences textuelles et des cas d'utilisation générés automatiquement (M. Kamalrudin, Grundy, et Hosking 2010)

Reddivari et al. (Reddivari, Chen, et Niu 2012) proposent RecVisu, une visualisation basée sur un mécanisme de clusters calculés automatiquement à partir de thèmes présents dans les exigences textuelles (Figure 57- A), un contrôle sur la constitution des clusters (Figure 57- B) et une vue zoomée

pour voir les détails d'un cluster (Figure 57- C). RecVisu+ (Niu, Reddivari, et Chen 2013) intègre des améliorations à RecVisu telles qu'une barre d'évaluation de cohérence de chaque cluster, et des interactions directes pour réaliser des actions sur les clusters (renommer, diviser, annoter) (Figure 58). Les auteurs conduisent une étude de cas sur un projet réel, impliquant 3 ingénieurs, dans laquelle ils demandent aux participants d'utiliser RecVisu+, lors de 4 séances alternant observation et discussion avec les participants. Les résultats de leur analyse qualitative des retours collectés identifient quatre catégories de tâches en ingénierie des exigences à instrumenter par de la visualisation :

- vue d'ensemble : pour résumer un grand espace d'exigences et identifier des zones d'intérêt ;
- anomalies : pour localiser des exigences particulières et traiter les déviations ;
- hétérogénéité : pour lier les exigences à d'autres artefacts, comparer des intérêts de parties prenantes multiples ;
- causalité : réaliser une analyse sémantique, faire des comparaisons multi-variables, et supporter le raisonnement exploratoire.

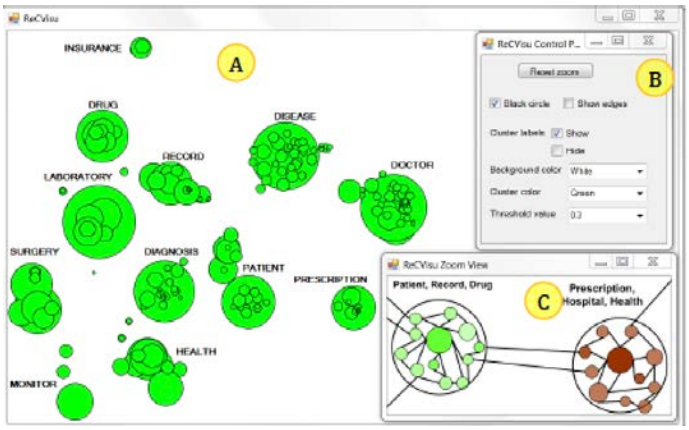



Figure 57: visualisation de clusters d'exigences (Reddivari, Chen, et Niu 2012)


Figure 58: des interactions directes sur la visualisation à base de cluster (Niu, Reddivari, et Chen 2013)

Les auteurs ne formalisent pas les résultats sous forme d'exigences en tant que telles pour les outils d'ingénierie des exigences, mais plutôt comme des améliorations à apporter à leur outil. Nous retenons de cette étude les quatre catégories de tâches à instrumenter comme candidates à une

exploration plus large de visualisations. En effet, les auteurs n'explorent qu'une seule visualisation, à base de clusters, dont la pertinence repose sur la qualité de leur algorithme de traitement du texte.

Savio et al. (Savio et al. 2012) propose ReBlock, une visualisation des exigences sur une pyramide découpée en tranches et en facettes. Le nombre de tranches montre le niveau d'abstraction, et chaque partie prenante possède sa facette (Figure 59). La rotation de la pyramide autour de ses différents axes permet de bénéficier de vues sous plusieurs angles (vues de dessus par exemple à gauche de la Figure 59). Les exigences appartenant au même niveau d'abstraction sont codées avec une même couleur. En cliquant sur une exigence, sa description textuelle, ses attributs et ses liens avec les exigences de plus haut niveau et plus bas niveau s'affichent (absence de visuel pour illustrer cet aspect dans l'article).

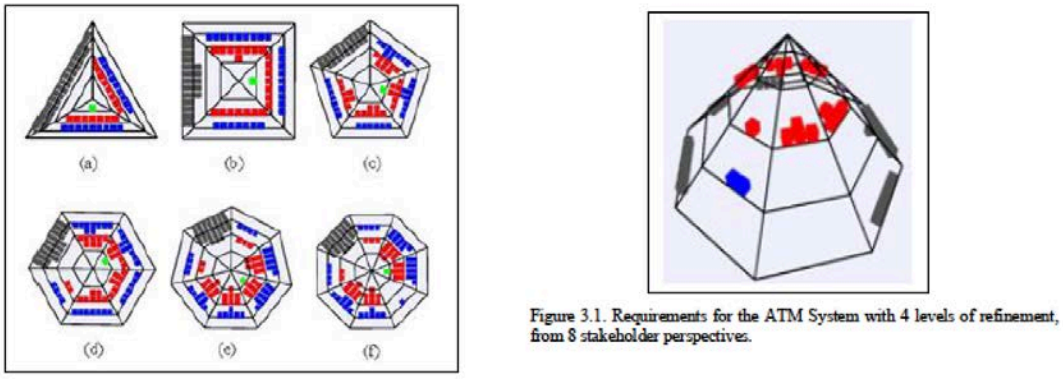



Figure 59: ReBlock : une pyramide d'exigences (Savio et al. 2012)

Les auteurs rapportent les problèmes rencontrés lors de leur utilisation de ReBlock sur une application d'ultrasons : le nombre important d'exigences, que ce soit à la base de la pyramide, ou à son sommet, implique un problème d'arrangement spatial des blocs. Une solution est de placer les blocs les uns derrière les autres pour la base. Pour le sommet, un mécanisme de cluster est mis en place, mais pose des problèmes de représentation de traces avec le niveau inférieur. Un problème non résolu par les auteurs reste le placement des exigences qui viennent de plusieurs parties prenantes, la duplication sur plusieurs facettes d'une même exigence posant des problèmes de traçabilité.

Ugai et al. (Ugai, Hayashi, et Saeki 2010) proposent de visualiser les intérêts des différentes parties prenantes sur une carte d'ancres, générée à partir d'un modèle orienté-but. Les intérêts des différentes parties prenantes sont considérés selon les exigences non fonctionnelles (sécurité, utilisabilité, maintenabilité, etc), et renseignés dans le modèle sur chaque but via une matrice (voir haut de la Figure 60). En bas de la Figure 60 sont proposées les cartes d'ancre suite au renseignement individuel de 9 parties prenantes (3 développeurs, 3 utilisateurs et 3 administrateurs) sur 34 buts. L'analyse de la visualisation proposée par les auteurs montrent qu'il faut la mener en corrélation avec le graphe de buts. Par exemple, pour expliquer pourquoi les critères à gauche du graphe n'intéressent aucune des parties prenantes, les auteurs évoquent le contenu du graphe de buts. Or les deux visualisations ne sont pas liées entre elles.

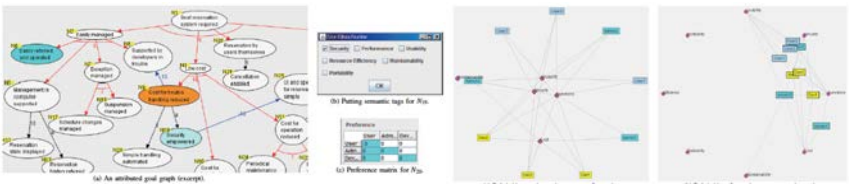


Figure 60: carte d'ancre des parties prenantes selon les exigences non fonctionnelles (Ugai, Hayashi, et Saeki 2010)

Suppakul et Chung (Supakkul et Chung 2010) rapportent le même problème dans une démarche similaire de visualisation des exigences non fonctionnelles d'un modèle orienté but (Figure 61) :

> *NFR patterns presented in this paper visualize knowledge of NFRs currently in a standalone manner with little explicit relationships with other requirements artifacts such as agents and functional requirements.*

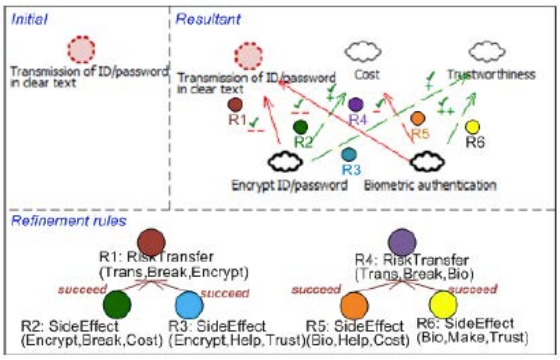



Figure 61: visualisation des exigences non fonctionnelles (Supakkul et Chung 2010)

Martinie et al. (Martinie et al. 2010), dans une démarche de justification des choix par rapport à des exigences non fonctionnelles, proposent de représenter les exigences fonctionnelles sur un graphe amélioré de notation semi-formelle QOC (question-option-critère) (MacLean et al. 1991) (voir haut de la Figure 62). Cependant la complexité du cas d'étude (spécification de boutons pour les cockpits de nouvelle génération) entraine des problèmes de lisibilité du graphe. Une représentation alternative, sous forme de matrice, est proposée, en exploitant l'expressivité des variables visuelles (Figure 49). Le nombre d'exigences de l'exemple est cependant limité à 9.

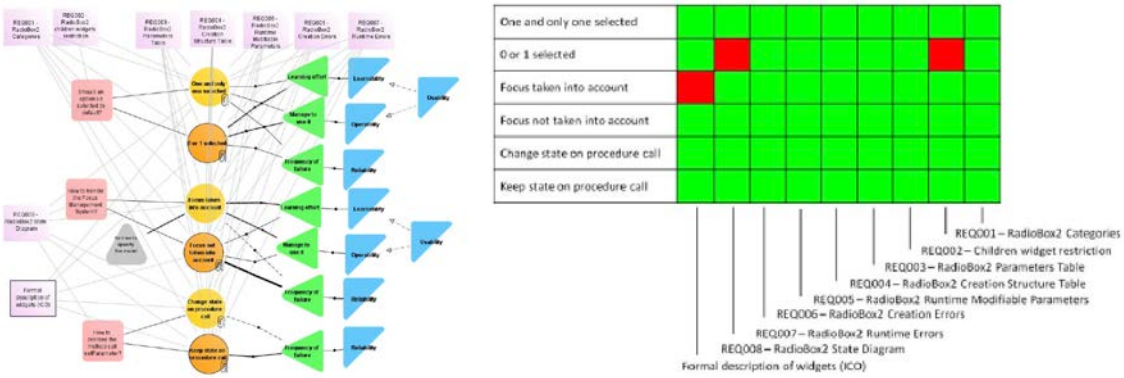



Figure 62: visualisation des exigences fonctionnelles sur le graphe, et sous forme matricielle (Martinie et al. 2010)

Merten et al. (Merten, Jüppner, et Delater 2011) explorent une visualisation de type *sunburst* (voir Figure 63) pour la vue d'ensemble de la spécification. Les différents artefacts d'ingénierie des exigences (but, cas d'utilisation, profils utilisateurs, etc) sont représentés et différenciés par la couleur, comme le montre la légende accompagnant la visualisation de la Figure 63. Ils explorent également une visualisation de type *netmap* (Figure 63) pour représenter les liens entre les cas d'utilisation, les profils utilisateurs et les buts. Le survol d'un élément permet d'afficher des informations supplémentaires. Les auteurs montrent comment l'utilisation de ces deux visualisations permet de répondre à trois questions : les cas d'utilisation sont-ils définis et présents dans la spécification ? chaque cas d'utilisation correspond-il à un profil utilisateur ? existe-t-il des profils utilisateurs identifiés mais non référencés dans la spécification ? Les visualisations ne sont pas interconnectées, mais utilisées l'une après l'autre. Les auteurs concluent en envisageant le développement de filtres afin de gérer le passage à l'échelle de leurs visualisations.

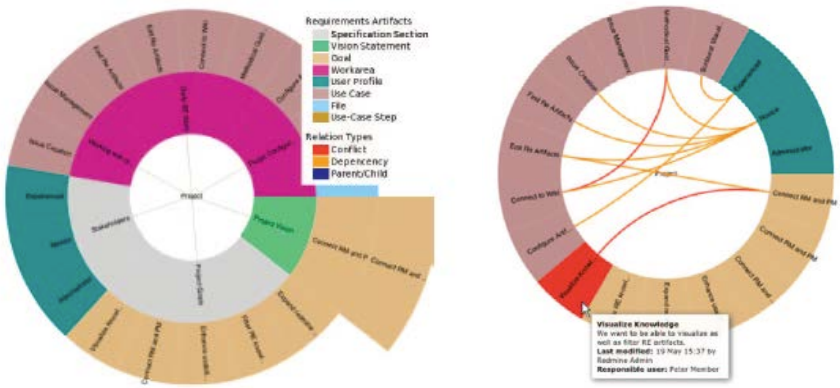



Figure 63: visualisation sunburst pour la vue d'ensemble de la spécification (à gauche) et netmap pour les liens entre cas d'utilisation, profils utilisateurs et buts (à droite) (Merten, Jüppner, et Delater 2011)

### II.3.5.3. Les plateformes collaboratives entre utilisateurs et développeurs : vers l'obèse data ?

Reconnaissant l'importance de la collaboration entre utilisateurs et développeurs et la difficulté de rassembler tout le monde au même endroit, de nombreux travaux proposent des outils web collaboratifs pour permettre aux utilisateurs d'éliciter les exigences (Whitehead 2007).

Lohmann et al. (Lohmann, Ziegler, et Heim 2008) proposent un outil web dont l'objectif est de favoriser la participation d'utilisateurs finaux distribués géographiquement dans l'élicitation des exigences pour des systèmes web : l'outil permet aux utilisateurs d'exprimer des exigences à partir de leur expérience d'interaction, dans une démarche de conception participative distribuée. L'utilisateur a accès dans la barre de navigation à une icône déclenchant l'accès à la saisie d'une exigence textuelle, avec la possibilité de désigner l'élément du système concerné par interaction directe. Un outil d'analyse des exigences exprimées ainsi par les utilisateurs est proposé pour les développeurs (voir Figure 64) : il est constitué d'une carte interactive permettant de zoomer sur une zone géographique, associée à un tableau d'exigences sur lequel on peut faire des recherches par mots-clés. La sélection d'une exigence permet au développeur de simuler l'état de l'application selon le contexte enregistré au moment de la saisie. Les auteurs ont exploré une visualisation complémentaire des exigences, sous forme de graphe (à droite sur la Figure 64), pour identifier les relations, similitudes et conflits entre exigences. L'utilisation de vues multiples coordonnées semble prometteuse. Cependant, les auteurs ne s'intéressent pas directement à la gestion des exigences par les développeurs mais à une meilleure implication des utilisateurs dans l'élicitation des exigences, à travers des mécanismes de

récompense. En particulier, à partir du moment où ce sont les utilisateurs qui expriment les exigences avec leurs propres mots, des questions se posent sur l'identification des similitudes entre exigences et une reformulation potentielle par les développeurs en conservant le lien avec les exigences initiales.


Figure 64: un outil d'analyse d'exigences exprimées par les utilisateurs (Lohmann, Ziegler, et Heim 2008)

On retrouve la même approche de plateforme collaborative entre utilisateurs et développeurs autour d'annotations intégrées dans des captures d'écran dans AnnotatePro (Rashid et al. 2006). Cependant, les efforts des auteurs se concentrent plus sur l'obtention des informations de la part des utilisateurs finaux que sur le traitement ultérieur des exigences par les développeurs (Figure 65) : les retours collectés sont présentés sous forme de liste, la sélection d'un élément permettant d'afficher les annotations de l'utilisateur.

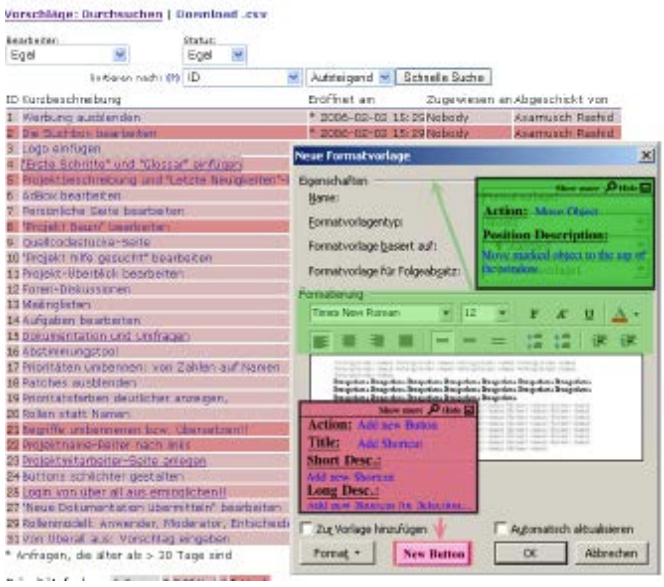



Figure 65: la vue des exigences dans AnnotatePro (Rashid et al. 2006)

Ainsi, de nombreux travaux proposent des moyens aux utilisateurs finaux pour exprimer des besoins enrichis par des informations contextuelles (capture d'écran, environnement virtuel, photo). Perez et Valderas (Pérez et Valderas 2009) proposent une interface graphique permettant aux utilisateurs finaux de définir de nouveaux services, en les guidant dans la rédaction (voir Figure 66). De même, iRequire (Seyff, Graf, et Maiden 2010) (Seyff, Ollmann, et Bortenschlager 2011) est un outil pour

téléphone mobile permettant à l'utilisateur d'enregistrer des besoins, oralement ou textuellement, documentés par des photos de son environnement (Figure 67).


Figure 66: support à l'utilisation final pour définir un nouveau service (Pérez et Valderas 2009)

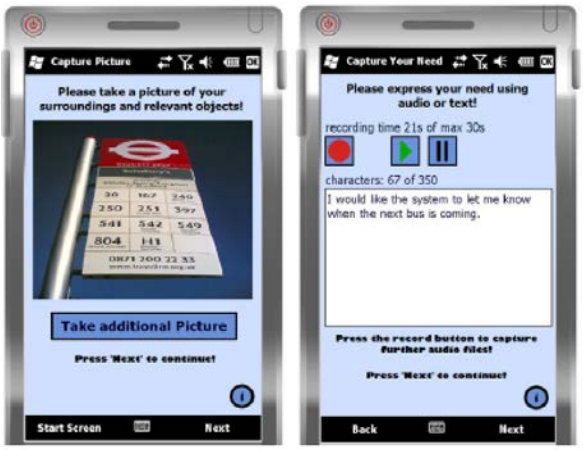



Figure 67: iRequire permet à un utilisateur de rapporter un besoin et son contexte (Seyff, Ollmann, et Bortenschlager 2011)

Enfin, les wikis sont étudiés pour faciliter le partage d'informations entre parties prenantes autour des exigences. WikiWinWin (Yang et al. 2008) propose d'utiliser TWiki (twiki.org) comme support à la négociation des exigences. ShyWiki (Solis et Ali 2010) propose un support à la conduite de brainstorming entre les différentes parties prenantes (Figure 68): les exigences peuvent être manipulées spatialement, groupées, reliées et fusionnées, les pages de wiki étant utilisées comme des tableaux blancs.

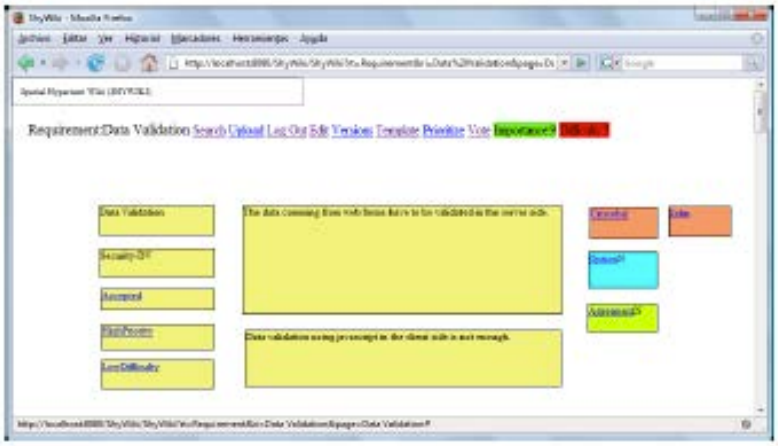



Figure 68: ShyWiki, un wiki comme support au brainstorming entre parties prenantes (Solis et Ali 2010)

Ces approches ont le mérite de stimuler la participation des utilisateurs finaux dans le processus d'ingénierie des exigences, en leur proposant des systèmes utilisables et collaboratifs. Favoriser l'élicitation des exigences par les utilisateurs finaux par des outils contextuels et collaboratifs permet une meilleure implication des utilisateurs dans la conception du système, mais pose de nouveaux problèmes aux ingénieurs : comment faire le tri, détecter des similitudes, transformer en exigences bien exprimés et assurer un retour individualisé vers les utilisateurs pour maintenir leur implication ? Le traitement des données (tri, doublon, formalisation en exigences, priorités) et un support outillé pour les ingénieurs ne sont pas abordés par ces travaux. Nous pouvons rapprocher ces éléments de l'étude des pratiques industrielles de Karlsson et al. (Karlsson et al. 2007) reportant le problème des ingénieurs dans la gestion d'une grande quantité d'information provenant des utilisateurs finaux.

## II.4. Synthèse et questions de recherche

L'état de l'art des pratiques industrielles en ingénierie des exigences (partie II.2) a mis en évidence un problème sur le support outillé en ingénierie des exigences : la faible adoption de méthodes formelles ne permet pas aux ingénieurs de bénéficier de vérification automatisée de cohérence et complétude des exigences. Ces tâches sont réalisées manuellement et prennent du temps. Les exigences sont exprimées textuellement dans les documents de spécification. De plus, une utilisation d'outils à vocation générale est rapportée, alors que de nombreux outils sont proposés par les éditeurs de logiciel, offrant des capacités fonctionnelles couvrant l'ensemble des activités d'ingénierie des exigences (Carrillo de Gea et al. 2014).

La facilité d'utilisation est pointée comme primordiale par Hoffmann et al. (Hoffmann et al. 2004), à partir de leur expérience professionnelle dans l'industrie automobile. Ils précisent que les outils doivent proposer des vues graphiques des exigences, librement configurables par des filtres. Ils ne relient pas ces exigences à des activités d'ingénierie des exigences, sauf sur la traçabilité, car elle exige un effort de la part des ingénieurs, sans bénéfices immédiats pour eux. Winkler (S. Winkler 2008)(Stefan Winkler et Pilgrim 2010) identifie l'utilisabilité des visualisations comme un point crucial à explorer, étant donnée la diversité des acteurs intéressés par l'utilisation des traces entre les différents artefacts d'ingénierie. L'utilisation d'outils à vocation générale rapportée par les études de terrain, et l'intérêt des industriels dans la « facilité d'utilisation » nous conduisent aux questions de recherche suivantes :

**RQ1 : Quelles sont les activités d'ingénierie des exigences réalisées par les ingénieurs en aéronautique ?**

**RQ2 : Comment les outils supportent-ils ces activités ?**

Dans une recherche d'amélioration des outils, l'application des principes des *Physics of Notation* (D. Moody 2009) reste pour l'instant limitée à l'amélioration de l'expressivité des symboles des langages visuels, et ne permet pas de conclure quant à une meilleure lisibilité des modèles (Caire et al. 2013)(Genon, Amyot, et Heymans 2010). L'état de l'art en visualisation d'ingénierie des exigences montre que les visualisations, en n'anticipant pas la grande quantité de données à afficher, se confrontent à des problèmes de lisibilité pour des cas réels (Duan et Cleland-Huang 2006) (Martinie et al. 2010) (Merten, Jüppner, et Delater 2011)(Savio et al. 2012).

Pour répondre à la problématique de grande quantité de données, nous pensons que le domaine de la visualisation d'information, dont l'objectif est « l'utilisation de représentations visuelles et interactives de données abstraites supportées par l'ordinateur pour amplifier la cognition » (Card, Mackinlay, et Shneiderman 1999), constitue un cadre pertinent pour l'ingénierie des exigences. En particulier, le principe de manipulation des vues interactives (Heer et Shneiderman 2012)(voir Figure 69) propose la sélection, la navigation, les vues coordonnées et l'organisation de fenêtres. Or les travaux actuels proposent soit des visualisations multiples qui sont faiblement interactives et non coordonnées (Feather et al. 2006b)(Duan et Cleland-Huang 2006)(Martinie et al. 2010) (Ugai, Hayashi, et Saeki 2010)(Supakkul et Chung 2010), soit une seule visualisation interactive (Reddivari, Chen, et Niu 2012) (Niu, Reddivari, et Chen 2013)(Savio et al. 2012). L'utilisation de visualisations multiples interconnectées représentant les différents artefacts d'ingénierie des exigences, reste à explorer.

**RQ3 : Comment améliorer l'utilisabilité des outils d'ingénierie des exigences, en exploitant les principes de la visualisation d'information ?**

**Taxonomy of interactive dynamics for visual analysis.**

| | |
|---|---|
| ***Data and View Specification*** | **Visualize** data by choosing visual encodings. |
| | **Filter** out data to focus on relevant items. |
| | **Sort** items to expose patterns. |
| | **Derive** values or models from source data. |
| ***View Manipulation*** | **Select** items to highlight, filter, or manipulate them. |
| | **Navigate** to examine high-level patterns and low-level detail. |
| | **Coordinate** views for linked, multidimensional exploration. |
| | **Organize** multiple windows and workspaces. |
| ***Process and Provenance*** | **Record** analysis histories for revisitation, review, and sharing. |
| | **Annotate** patterns to document findings. |
| | **Share** views and annotations to enable collaboration. |
| | **Guide** users through analysis tasks or stories. |

Figure 69: taxonomie des tâches interactives pour l'analyse visuelle de Heer et Schneiderman (Heer et Shneiderman 2012)

Enfin, de nombreux outils favorisent la participation des utilisateurs dans l'élicitation des exigences, ce qui a pour conséquence d'amplifier la quantité d'informations à traiter par les ingénieurs. Une proposition de vues multiples coordonnées est faite pour répondre à cet enjeu (Lohmann, Ziegler, et Heim 2008). Cependant, nous pouvons nous demander s'il ne s'agit pas d'un problème accidentel

(Brooks 1987), et s'il ne serait pas intéressant de privilégier la qualité des données rapportées par les utilisateurs, plutôt que la quantité.

**RQ4 : Comment faire participer les utilisateurs à la spécification du système futur à partir de leurs usages du système actuel ?**

# Chapitre III. Méthode

Notre objectif de recherche est d'explorer comment les outils assistent réellement les ingénieurs dans leur travail sur les exigences (RQ1 et RQ2). Notre hypothèse est que l'utilisabilité des outils utilisés est faible et peut être améliorée (RQ3). Afin de comprendre en profondeur l'ingénierie des exigences en tant qu'activité sociotechnique, nous adoptons une approche qualitative (Wohlin et Aurum 2015). Nous exposons notre méthode d'étude de cas, complétée par du design d'artefacts pour améliorer la compréhension du phénomène.

## III.1. Une méthode d'étude de cas

Nous avons choisi comme méthode de recherche l'étude de cas (case study), car elle permet d'enquêter sur un phénomène en profondeur et en contexte réel (Yin 2009). L'état de l'art sur les études de terrain révèle la multiplicité des acteurs et des outils utilisés en ingénierie des exigences, qu'il est difficile de reproduire dans des expérimentations. De plus, notre état de l'art montre que les travaux utilisant la méthode d'étude de cas obtiennent des informations plus précises sur les pratiques industrielles que les enquêtes de grande ampleur (voir synthèse en Tableau 2). L'inconvénient de l'étude de cas est la difficulté de répliquer et généraliser les résultats obtenus. Nous pourrons utiliser les résultats des enquêtes de grande ampleur de l'état de l'art pour confronter nos résultats. Notre démarche, résumée dans la Figure 70, est faite d'allers-retours entre le contexte industriel, pour collecter des données sur les pratiques des ingénieurs, et notre contexte de recherche, pour préparer la collecte et analyser les données collectées.

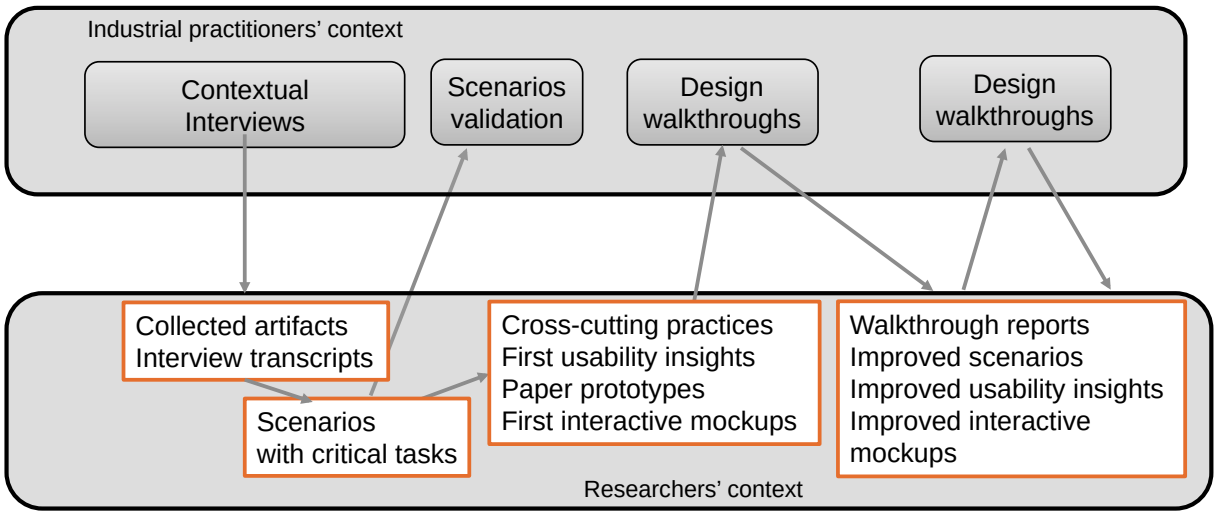



Figure 70: notre démarche d'étude de cas

Dans la suite de cette partie, nous détaillons la sélection des sites, les participants, la collecte des données et l'analyse des données.

## III.1.1. Sélection des sites et participants

Nous avons choisi d'observer le phénomène d'ingénierie des exigences dans le domaine aéronautique. L'aéronautique est un des domaines les plus exigeants vis-à-vis de l'ingénierie des exigences, du fait de sa criticité par rapport à la sécurité et la longue durée de vie de ses produits. Le standard DO-178C (RTCA et EUROCAE 2011) fournit à l'industrie aéronautique des directives pour démontrer un niveau acceptable de sécurité : les ingénieurs doivent fournir des preuves de liens entre les exigences et leur implémentation comme moyen principal de démonstration de conformité. Sur la Figure 71, les arcs entre les différents niveaux représentent les activités demandées par la DO178-C pour les systèmes embarqués (bord), basées sur de la traçabilité entre les différents niveaux. Les systèmes de contrôle aérien sont soumis à des directives de même type avec le standard ED-109/DO-278 depuis 2012 (EUROCAE et RTCA 2012), considéré comme un complément de la DO-178C pour les systèmes sol.

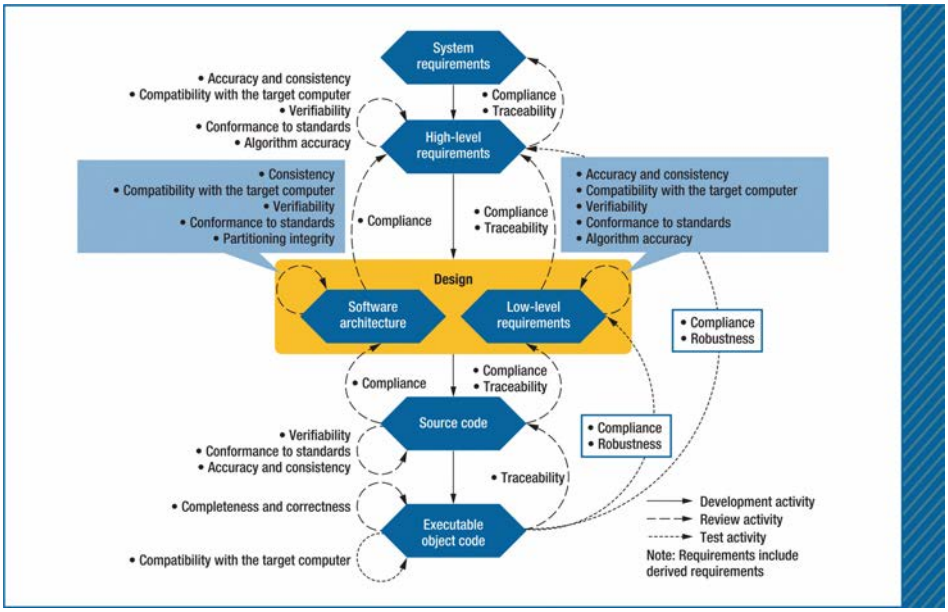



Figure 71: la traçabilité entre exigences (en haut du schéma) et code (en bas du schéma) exigée par la DO-178C

Nos travaux impliquent un total de 15 ingénieurs en exigences, avec un minimum de quatre années d'expérience, provenant de quatre entreprises différentes : un fabriquant d'avion, un fournisseur d'avionique, un sous-traitant d'un fabriquant d'avion, un fournisseur de système de contrôle aérien. Afin de garantir la confidentialité des données et l'anonymat des participants, nous ne nommons pas les entreprises et les désignons par les lettres A, B, C et D dans la suite du document. Le Tableau 4 présente les détails des participants selon l'entreprise, les années d'expérience, et la technique de collecte de données.

| Entreprise | Participants | | | |
|---|---|---|---|---|
| | années d'expérience | interview contextuelle | design walkthrough | validation scénario |
| (A) Constructeur avion 2 participants | > 10 | * | | * |
| | 4 | * | | |
| (B) Sous-traitant constructeur avion 1 participant | 4 | * | | |
| (C) Equipementier Avionique 3 participants | > 10 | * | | |
| | > 10 | * | | |
| | > 10 | * | | |
| (D) Fournisseur de système de contrôle aérien 9 participants | > 10 | * | ** | * |
| | 4 | * | ** | * |
| | 4 | * | | |
| | > 10 | * | | |
| | > 10 | | * | |
| | > 10 | | * | |
| | > 10 | | * | |
| | > 10 | | * | |
| | > 10 | | * | |
| Total | | 10 | 9 | 3 |

Tableau 4: profils et implication des participants

## III.1.2. Collecte des données

Nous avons utilisé deux techniques de collecte de données : l'interview contextuelle et le design d'artefacts associé à des design walkthroughs. Nous avons commencé avec les interviews contextuelles pour comprendre les tâches actuellement réalisées par les ingénieurs et identifier les problèmes rencontrés par les praticiens industriels. L'interview contextuelle est une entrevue individuelle conduite dans l'environnement de travail de l'utilisateur qui se centre sur les observations du travail en cours (Holtzblatt, Wendell, et Wood 2005a). Pendant l'interview, l'interviewer pose des questions sur ce qui se passe pendant le travail et réagit de façon flexible aux réponses de l'utilisateur, rebondissant sur les problèmes émergeant pendant l'interview. Une relation maitre-apprenti se construit, dans laquelle l'utilisateur est le maitre, et l'interviewer l'apprenti, qui se forme sur le travail de l'utilisateur. Notre objectif est de capturer les activités réelles d'ingénierie des exigences dans les environnements de travail, au-delà des processus officiels et des pratiques déclarées. Le focus des interviews est le support outillé dans les tâches d'ingénierie des exigences. Nous utilisons les questions suivantes sur les artefacts d'ingénierie des exigences comme point de départ : d'où vient l'information ? comment l'information est-elle créée ? comment est-elle utilisée et par qui ? (Holtzblatt, Wendell, et Wood 2005a).

Nous avons interviewé dix ingénieurs dans leur contexte de travail. Chaque interview contextuelle a duré deux heures, que nous avons enregistrées, à l'exception de trois interviews pour des raisons de confidentialité. Pendant les interviews, nous avons collecté des artefacts, que nous avons utilisés dans les transcriptions et compte-rendu pour une meilleure compréhension et pour contrebalancer les opinions des participants. Après huit interviews, nous avons détecté des problèmes significatifs sur l'intégration et la visualisation de données. Afin de mieux comprendre ces problèmes et acquérir de nouvelles perspectives, nous avons conçu plusieurs maquettes pour tirer bénéfice de

visualisations variées et conduire une recherche orientée design (design-oriented research) (Fallman 2003)(Fallman 2007) : dans ce type de recherche, la contribution est la connaissance qui vient de l'étude des propositions de design en utilisation. Les propositions de design sont considérées comme un moyen, et non comme une fin. Ainsi, notre objectif n'est pas de développer des maquettes comme un produit final, mais plutôt comme des capteurs ou des instruments pour rassembler de nouvelles perspectives sur les tâches et l'utilisabilité à travers des discussions avec les praticiens industriels.

Nous avons conçu et développé les maquettes en utilisant la libraire JavaScript d3 (Bostock, Ogievetsky, et Heer 2011) qui propose une nouvelle approche de représentation et de visualisation de documents pour les navigateurs web. La librairie d3 permet de manipuler les visualisations comme des graphes de scène, d'accéder facilement aux éléments de la scène et de bénéficier d'animations et d'interactions performantes. Par conséquent, la librairie d3 nous a semblé adapter au développement itératif de visualisations interactives d'artefacts d'ingénierie des exigences. Comme suggéré par les travaux antérieurs, nous avons exploré les visualisations hiérarchiques et quantitatives, ainsi que les interactions pour les actions de filtrage. Ensuite, étant donnée l'utilisation intensive de matrices dans les artefacts collectés pendant les interviews, utilisation corroborée par l'état de l'art des études de terrain (voir partie II.2), nous avons exploré des visualisations alternatives de matrices. Enfin, nous avons exploré comment la coordination entre visualisations (Heer et Shneiderman 2012) peut supporter la navigation et la traçabilité. Nous avons raffiné les visualisations et les interactions à travers un processus itératif, en utilisant des prototypes papier (Holtzblatt, Wendell, et Wood 2005b) et des maquettes interactives utilisant d3. Les prototypes papier n'étaient pas suffisants pour traiter le problème de grande quantité de données et tester le passage à l'échelle des visualisations, qui s'est révélé crucial dans notre état de l'art. Nous avons alimenté les maquettes d3 en données réelles que nous avons récupérées lors des interviews de l'entreprise D. La Figure 72 montre une vue d'ensemble des différentes maquettes que nous avons développées.

Nous avons conduit neuf sessions de design walkthrough (Holtzblatt, Wendell, et Wood 2005b) (Mackay 2003) impliquant sept participants (voir Tableau 4: profils et implication des participants). Un design walkthrough est un partenariat de co-design: alors que l'utilisateur interagit avec le prototype pour réaliser une tâche qu'il a besoin de faire ou qu'il a fait dans un passé récent, l'utilisateur et le chercheur découvrent de nouveaux problèmes. Ils interprètent ensemble ce qui est en train de se passer lors de l'usage et réfléchissent à des alternatives de design. Dans cet objectif, nous avons alimenté les maquettes avec des données réelles, comprenant environ 4000 exigences. Pour chaque maquette, nous avons présenté les caractéristiques et collecté des retours qualitatifs des participants. A cette occasion, nous avons également demandé aux participants de nous montrer comment ils réalisent les tâches discutées avec leurs outils actuels. Nous avons rédigé un compte-rendu de chaque walkthrough, que nous avons fait valider par les participants

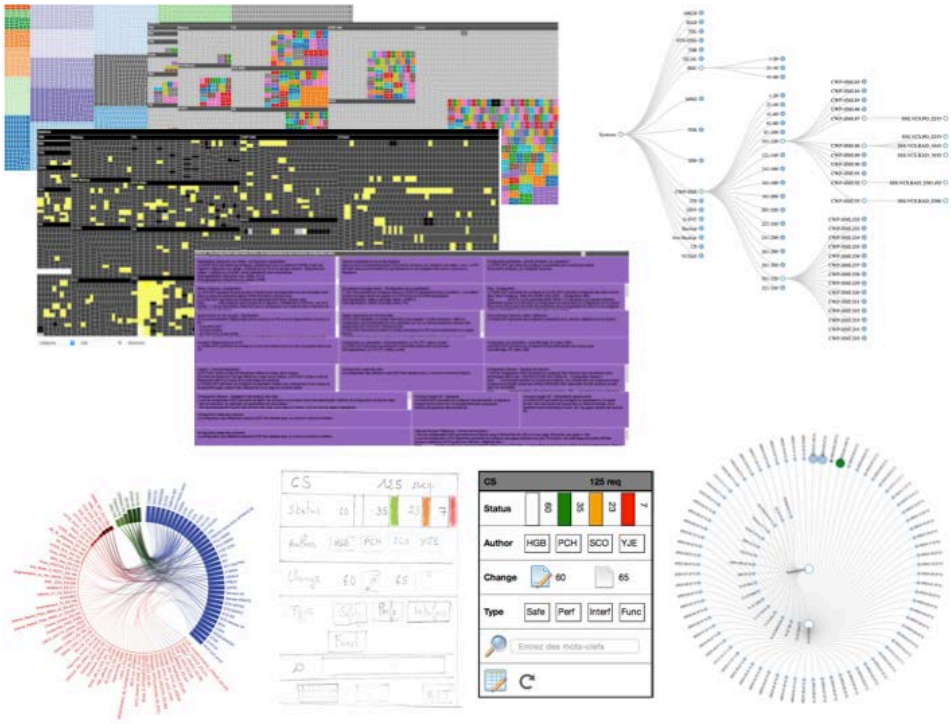



Figure 72: vue d'ensemble des différentes maquettes développées et utilisées pour la collecte de données

## III.1.3. Analyse des données

Nos questions de recherche étant centrées sur l'utilisabilité des outils supportant les tâches d'ingénierie des exigences, nous utilisons comme cadre d'analyse le cadre d'utilisabilité défini par la norme ISO 9241, complété par les styles d'utilisabilité de Lauesen (Soren Lauesen et Younessi 1998), que nous avons introduit dans la partie Utilisabilité et Flexibilité, et que nous rappelons en Figure 73. Nous avons transformé les données collectées (transcription d'interviews, compte-rendu de walkthrough et artefacts collectés) en scénarios cohérents par rapport aux utilisateurs impliqués (ingénieur en exigences, fournisseurs de composants, équipe d'ingénieurs en exigences, managers) et à leurs objectifs (en haut de la Figure 73). Un scénario synthétise une histoire sur les utilisateurs et leurs activités et constitue une technique intéressante pour rapporter de l'information concrète (Larry L. Constantine et Lockwood 1999)(Carroll 2000). Cette préparation des données permet d'obtenir des données comparables et de garder les participants engagés dans l'analyse : trois participants, des entreprises A et D, ont validé les scénarios.

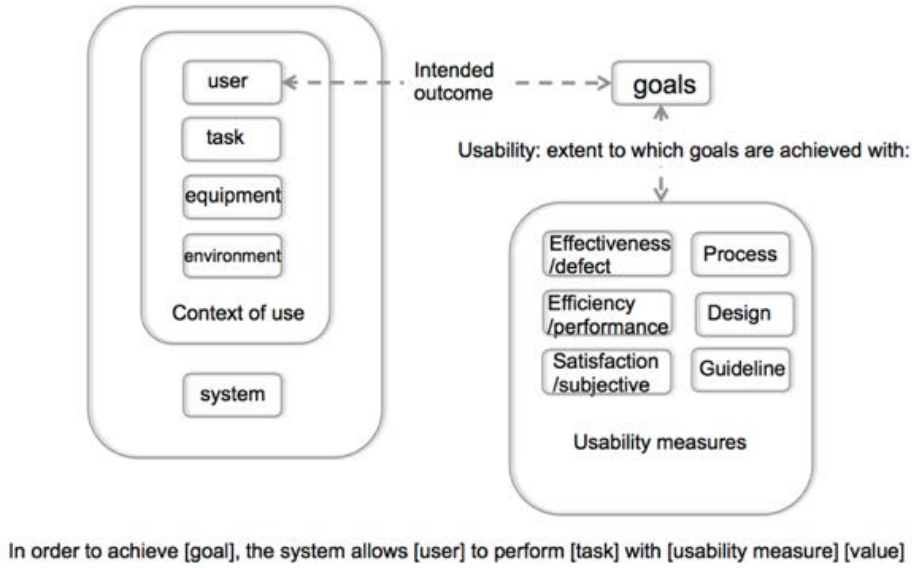



Figure 73: rappel de notre cadre d'utilisabilité

Nous avons ensuite réalisé une analyse étape par étape des scénarios, en identifiant les tâches critiques en termes d'utilisabilité : tâches propices aux erreurs (efficacité), tâches coûteuses en temps et fastidieuses (efficience), et tâches faites sous stress ou inconfort (satisfaction). Nous avons également utilisé le cadre des dimensions cognitives des notations (Green 1989) : il fournit un vocabulaire précis pour analyser l'utilisabilité des environnements (langage + outil) de programmation, et qualifier les problèmes et qualités en utilisabilité. Les dimensions cognitives, et les questions associées pour les évaluer (Blackwell et Green 2012) sont présentées dans le Tableau 5. A partir de cette analyse de l'utilisabilité, nous avons dérivé des premières exigences d'utilisabilité, en respectant la formulation du cadre défini par ISO9241 (Figure 73) : *Afin d'atteindre [objectif], le système permet à [utilisateur] de réaliser [tâche] avec [mesure d'utilisabilité] [valeur]*, où « le système » désigne les outils d'ingénierie des exigences. En adoptant cette formulation, notre objectif est d'aller au-delà des recommandations générales d'utilisabilité (Larry L. Constantine et Lockwood 1999)(Norman 2014)(Nielsen 1993), et de justifier nos exigences par rapport aux objectifs et activités des ingénieurs en exigences.

| Dimension cognitive | Questions |
|---|---|
| **Visibilité et possibilité de juxtaposition** | • Quelle est la facilité de voir ou trouver les diverses parties de la notation pendant la création ou la modification ? Pourquoi ?<br>• Quelles sont les choses plus difficiles à voir ou trouver ?<br>• Si vous avez besoin de comparer ou combiner différentes parties, pouvez-vous les voir en même temps ? Si non, pourquoi ? |
| **Viscosité** | • Quand vous avez besoin de faire des modifications à un précédent travail, quelles est la facilité de faire ce changement ? pourquoi ?<br>• Existe-t-il des modifications particulières difficiles à faire ? Lesquelles ? |
| **Concision** | • La notation vous permet-elle de dire a) ce que vous voulez de façon assez brève b) ou est-ce interminable ? pourquoi ?<br>• Quels sont les éléments qui prennent plus d'espace à décrire ? |
| **Opérations mentales difficiles** | • Quel genre d'éléments exigent le plus effort mental avec cette notation ?<br>• Existe-t-il des éléments qui semblent spécialement complexes ou difficiles à résoudre/calculer (par exemple quand il faut combiner plusieurs éléments) ? Quels sont-ils ? |

| | |
|---|---|
| **Propension à l'erreur** | • Des erreurs semblent-elles particulièrement communes ou faciles à faire ? Lesquelles ?<br>• Cela vous arrive-il de faire des petits lapsus/écarts qui vous irritent ou vous font sentir stupide ? Des exemples ? |
| **Proximité de la modélisation** | • A quel point la notation est-elle proche du résultat que vous décrivez ? Pourquoi ?<br>• Quels sont les éléments qui semblent particulièrement étranges à faire ou décrire ? |
| **Expressivité des rôles** | • A la lecture de la notation, est-facile de dire de quelle partie il s'agit dans le schéma global ? Pourquoi ? Y a-t-il de parties qui sont particulièrement difficiles à interpréter ? Lesquelles ?<br>• Existe-t-il des parties dont vous ne savez pas ce qu'elles signifient, mais que vous mettez juste parce que cela a toujours été fait ainsi ? Lesquelles ? |
| **Dépendances cachées** | • Si la structure du produit comporte des parties qui sont liées fortement l'une à l'autre, et que des modifications sur l'une affecteraient l'autre, ces dépendances sont-elles visibles ? Quelles sortes de dépendances sont cachées ?<br>• Dans quelle mesure cela empire-t-il quand vous créez une description particulièrement grande ?<br>• Ces dépendances restent-elles les mêmes, ou existe-t-il des actions qui les rendent caduques ? Si oui lesquelles ? |
| **Evaluation progressive** | • Quelle est la facilité de s'arrêter au milieu de la création d'un modèle et de vérifier le travail accompli ? Pouvez-vous le faire au moment que vous voulez ? Si non, pourquoi ?<br>• Pouvez-vous évaluer le progrès du travail que vous avez fait, ou vérifier à quelle étape du travail vous êtes ? Si non, pourquoi ?<br>• Est-ce que vous pouvez tester des versions partiellement renseignées ? Si non pourquoi ? |
| **Caractère provisoire (exploratoire)** | • Est-il possible d'ébaucher des éléments quand vous explorez des idées, ou quand vous n'êtes pas sûrs de comment faire ? Quelles caractéristiques de la notation permet de faire cela ?<br>• Quelles sortes de choses pouvez-vous faire quand vous ne voulez pas être trop précis sur le résultat exact que vous essayez d'obtenir ? |
| **Engagement prématuré** | • Quand vous travaillez avec la notation, pouvez-vous faire le travail dans n'importe quel ordre que vous souhaitez, ou est- ce que le système vous oblige à penser d'abord et à prendre des décisions ?<br>• Si oui, quelles décisions avez-vous besoin de faire à l'avance ? Quels sont les problèmes que cela cause dans votre travail ? |
| **Cohérence** | • Quand différentes parties de la notation signifient la même chose, les similarités sont-elles clairement identifiables ? Des exemples ?<br>• Existe-t-il des éléments qui devraient être similaires, mais que la notation rend différents ? Lesquels ? |
| **Notation secondaire** | • Est-il possible de faire des notes pour soi-même, ou d'exprimer de l'information qui ne fait pas partie de la notation ?<br>• Si c'était imprimé sur une feuille de papier, qu'est-ce que vous annoteriez, écririez ou dessineriez ?<br>• Est-ce que vous rajouter des signes supplémentaires (ou couleur ou choix de format) pour clarifier, insister ou répéter une information ? Si oui lesquelles et en quoi est-ce une aide ? |
| **Gestion de l'abstraction** | • Est-ce que le système vous offre des moyens de définir des nouveaux moyens ou termes dans la notation, afin que vous puissiez l'étendre pour décrire de nouvelles choses ou exprimer vos idées plus clairement/explicitement ? Lesquels ?<br>• Est-ce que le système insiste pour que vous commenciez par définir des nouveaux termes avant de faire tout autre chose ? |

Tableau 5: dimensions cognitives des notations (Green 1989) et questions permettant de les évaluer (Blackwell et Green 2012)

# III.2. Résultats attendus

Une méthode d'étude de cas classique n'intègre pas l'usage de prototypes. Cependant, nous pouvons décrire le rôle du design dans notre étude de cas en utilisant le cadre conceptuel de triangulation proposé par Mackay et Fayard (Mackay et Fayard 1997) selon trois axes: observation, design d'artefacts et théorie. Comme illustré sur la Figure 74, les analyses des interviews contextuelles (axe observation) vont nous permettre d'identifier la place de l'utilisabilité des outils dans le processus d'ingénierie des exigences (axe théorie) et d'orienter le design des visualisations interactives pour les exigences (axe design). Les design walkthroughs sur les visualisations (axe observation) vont nous permettre de réviser la place de l'utilisabilité des outils dans le processus d'ingénierie des exigences.

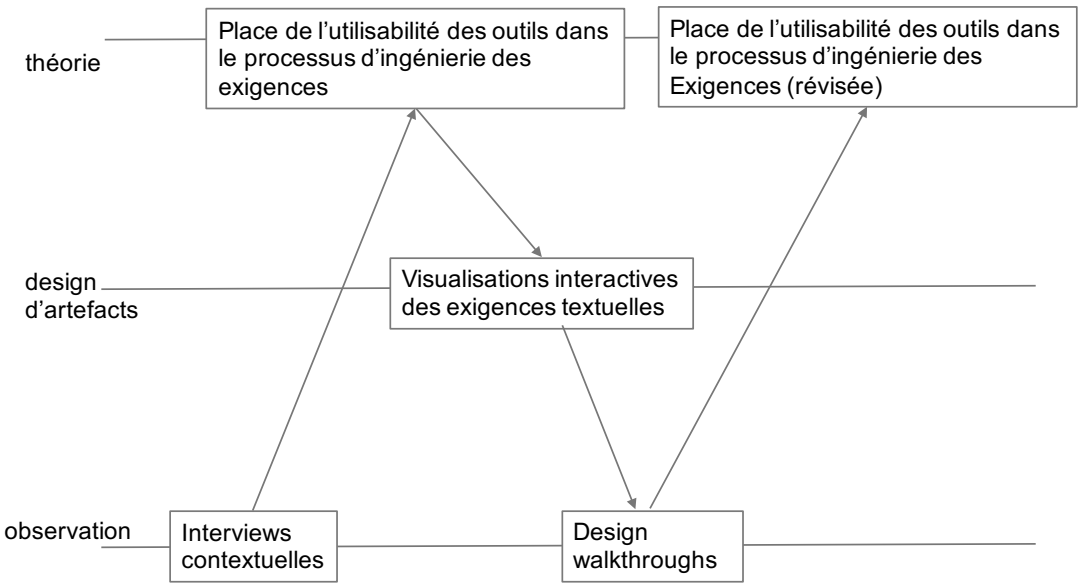



Figure 74: méthode d'étude de cas complétée par du design d'artefacts, en utilisant la triangulation entre disciplines (Mackay et Fayard 1997)

# Chapitre IV. Pratiques et outils en ingénierie des exigences dans l'aéronautique

Dans ce chapitre, nous présentons les résultats relatifs aux pratiques et outils en ingénierie des exigences dans l'aéronautique (zone verte sur la Figure 75). Après avoir présenté les caractéristiques des projets et des données d'ingénierie que nous avons observées, nous présentons les pratiques d'ingénierie des exigences transverses que nous avons découvertes parmi les différents projets et entreprises par rapport aux outils :

1. Conception exploratoire et collaborative sans outil spécifique;
2. Expérimentation d'une modélisation d'exigences avec outil spécifique;
3. Raffinement et gestion des exigences dans le référentiel.

Chaque contexte d'utilisation est structuré de façon similaire : une présentation générale des parties prenantes, de leurs objectifs et de leurs activités, suivie d'une description détaillée des activités sous forme de scénarios. Dans chaque scénario, nous fournissons des détails sur les outils utilisés et intégrons notre analyse de l'utilisabilité entre crochets. Plutôt que de formuler des recommandations, nous avons décidé d'appliquer le processus d'ingénierie des exigences sur notre propre étude. Ainsi, nous transformons chaque analyse (positive ou négative) d'utilisabilité en exigence pour les outils d'ingénierie des exigences, que nous référençons dans le scénario et présentons à la fin du scénario. De cette façon, nous maintenons une chaine de preuves (Yin 2009)-p123 entre interviews, scénarios, analyse et résultats, contribuant à la validité conceptuelle de l'étude et sa solidité. Nous proposons en synthèse les problèmes et qualités d'utilisabilité identifiés et une nouvelle vision du processus d'ingénierie des exigences.

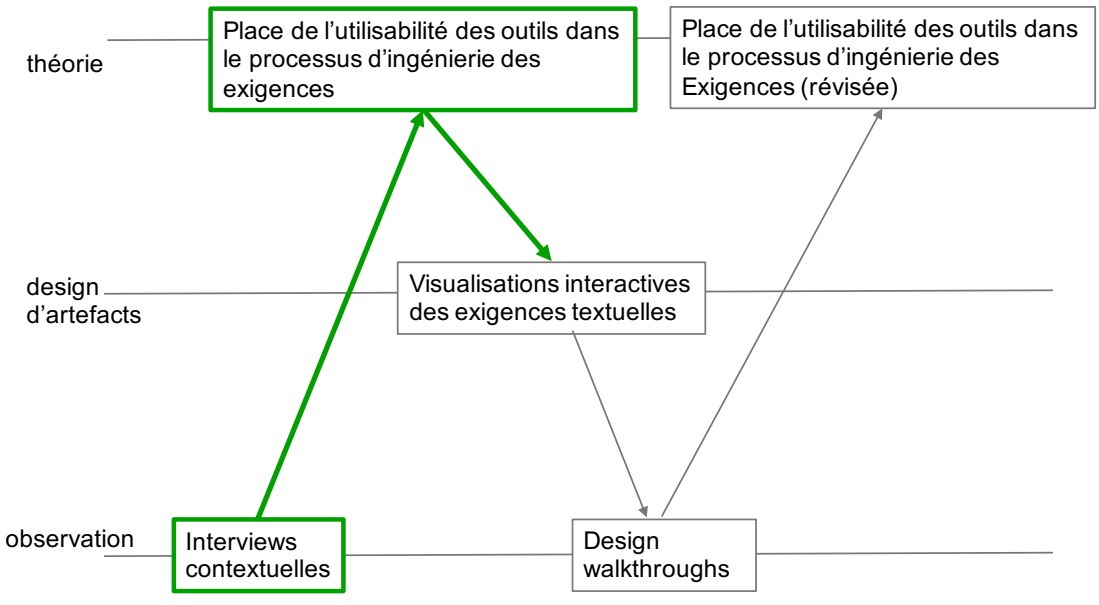



Figure 75: délimitation des résultats présentés dans ce chapitre

# IV.1. Caractéristiques des projets et des données d'ingénierie

Les industriels affichent l'adoption d'un cycle en V qui préconise une séquence chronologique des processus d'ingénierie système dans le temps pour des raisons organisationnelles et contractuelles: il détermine une répartition du travail entre les différentes parties prenantes (client, entreprise ingénierie système, fournisseurs), comme illustré dans la Figure 76 - adaptée d'une proposition faite par (Sage 2000)- et les documents produits permettent de contractualiser les relations dans le processus d'achat (Boehm 2006)(LUZEAUX, RUAULT, et WIPPLER 2011). Neuf projets différents ont été observés au sein des quatre entreprises aéronautiques A, B, C et D. Parmi ces neuf projets, nous comptons cinq projets *d'évolution* de systèmes existants. Le système est déjà opérationnel, les composants du système et les interfaces entre ces composants sont connus, il s'agit de faire évoluer le système existant avec un nouveau service. Trois projets sont relatifs à la prise en compte d'évolutions de systèmes bord (IHM cockpit d'un hélicoptère, service d'anticollision d'un avion de transport, évolution d'une suite avionique d'un turbopropulseur). Deux projets sont relatifs à la prise en compte d'évolutions de système sol (IHM et service de collaboration pour la gestion des départs, service de communication entre actuel et futur système de contrôle du trafic aérien).

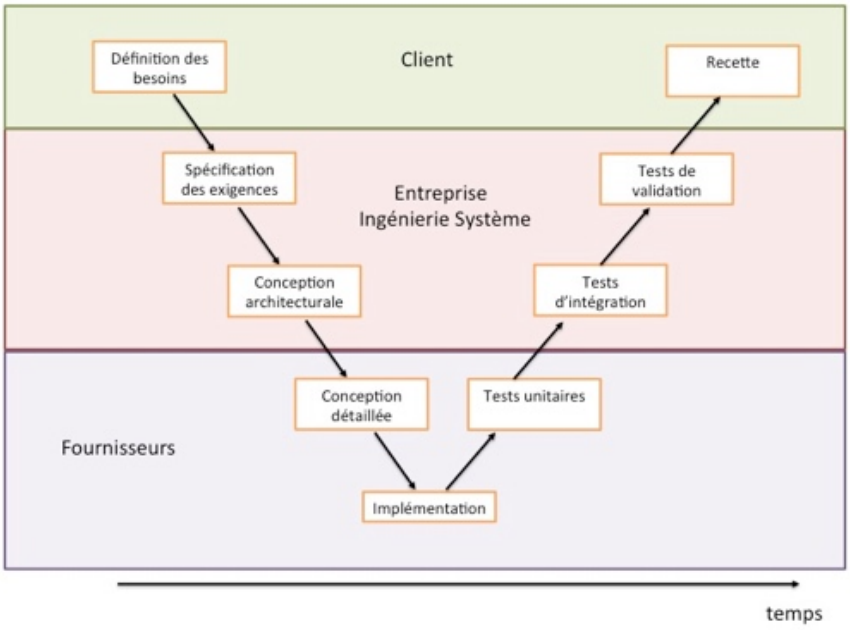
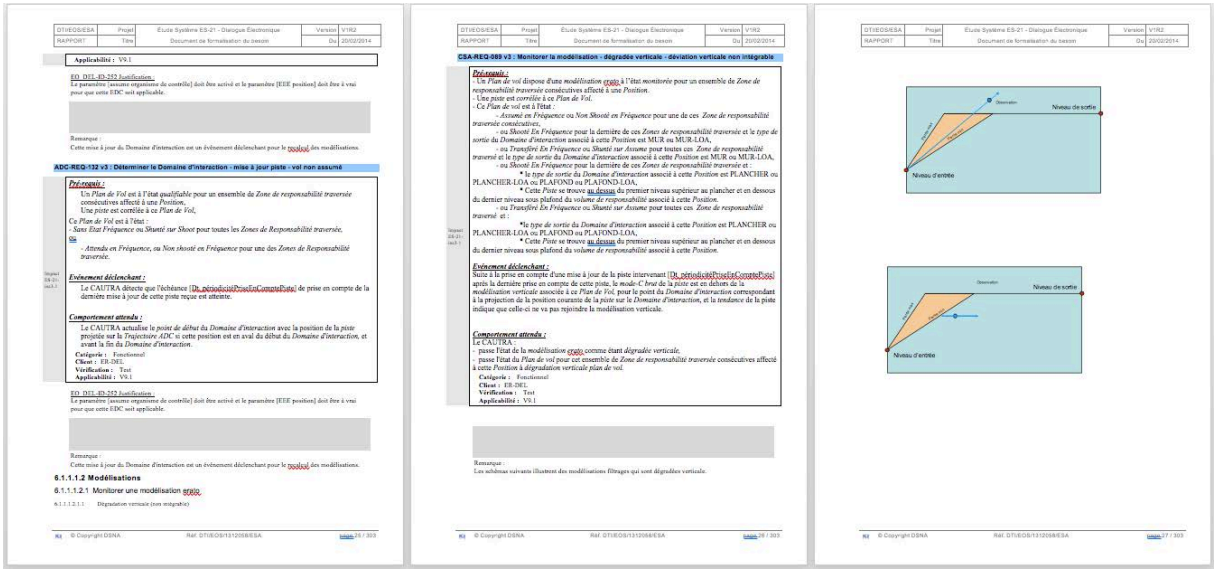



Figure 76: prescription d'un cycle en V dans les entreprises observées

Cette situation est assez courante pour l'ingénieur système, la durée de vie des systèmes aéronautiques sol et bord étant d'au moins 20 ans. Pendant cette durée de vie, la maintenance corrective et évolutive doit être assurée : la prise en compte des anomalies (ou faits techniques) suit souvent le même processus d'ingénierie système qu'une évolution, conformément à la norme ISO15288 présentée dans la partie I.3.2. En effet, la mise en place d'une nouvelle version d'un système est coûteuse, car elle exige la production de tous les documents de preuve de conformité pour la certification. Les industriels cherchent donc à optimiser le contenu d'une nouvelle version, en cherchant éventuellement à intégrer des évolutions dans une version initialement prévue pour la seule correction d'anomalies. Parmi les quatre autres projets observés, un projet est relatif à un nouveau système de radiotéléphone pour le contrôle aérien, avec appel d'offre, discussion et choix des fournisseurs. Les trois autres projets concernent des projets exploratoires, où des travaux de

modélisation système sont mis en œuvre en préparation d'affaires futures. Les documents produits sont des documents de spécification (System SubSystem Specification, Software Requirements Specification), qui font des centaines de pages, et dans lesquels le nombre d'exigences système va de 100 à 2500 pour une nouvelle fonction. Par exemple, le système FMS (Flight Management System) d'un avion est défini par 16 000 exigences. Quand on passe au niveau sous-système, une exigence système est affinée en au moins 3 exigences sous-systèmes. Quand on passe au niveau logiciel, chaque exigence sous-système est encore affinée, avec un facteur plus important s'il s'agit d'IHM. En effet, les exigences peuvent être très détaillées, avec des captures d'écran issues de prototypage à l'appui (voir exemples en Figure 77). Nous avons collecté de nombreux artefacts produits par les ingénieurs en exigences en complément des documents de spécification : des présentations PowerPoint, des fichiers Excel, des schémas, dont nous expliquons les rôles dans les scénarios reportant les pratiques observées.



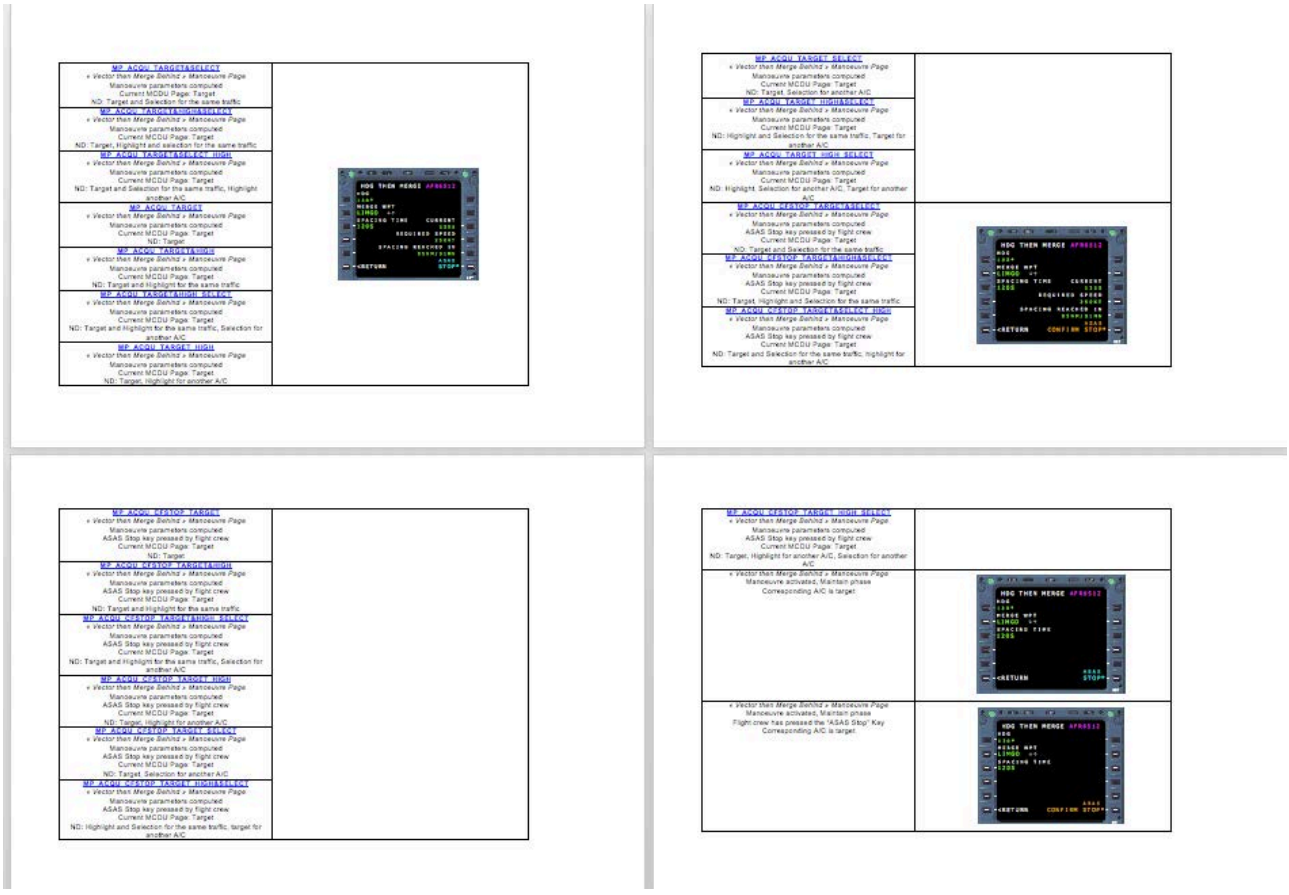


Figure 77: extraits des documents de spécification collectés : en haut, spécification textuelle d'un algorithme avec schémas à l'appui, au milieu une spécification purement textuelle, en bas une spécification sous forme de state charts textuels, avec copie d'écran associée.

## IV.2. Conception exploratoire et collaborative sans outil spécifique

Nous avons observé le contexte de conception exploratoire et collaborative sans outil spécifique dans les entreprises B, C et D du Tableau 4. Ce contexte d'utilisation survient au stade préliminaire du processus d'ingénierie des exigences, dans le cadre de la définition d'un nouveau service pour un système existant, dont les composants sont connus et stables. Le nouveau service est défini par des exigences système de haut niveau. Les parties prenantes de ce contexte sont l'ingénieur en exigences et les fournisseurs des composants système. L'objectif de l'ingénieur en exigences est d'éliciter les exigences pour les composants système et éventuellement les interfaces entre composants, à partir des exigences de haut niveau.

L'ingénieur en exigences adopte une façon de travailler différente des activités séquentielles prescrites du cycle en V (voir Figure 76). En effet, il ne va pas réfléchir seul à l'allocation des exigences système entre composants, puis rédiger les exigences pour chaque composant, pour enfin les transmettre aux fournisseurs des composants. A l'inverse, il choisit de lister les questions de conception que soulèvent les exigences système à remplir (ou les anomalies à traiter), et travaille avec les fournisseurs des composants lors d'« ateliers » sur ces questions de conception. Ce travail débouche sur un panel de solutions techniquement réalisables en termes d'allocation d'exigences, car proposées et discutées avec les fournisseurs. Cela permet ensuite à l'ingénieur de faire un choix

entre ces solutions et de rédiger les exigences d'interfaces entre composants. Les exigences affinées par composants sont initiées par l'ingénieur système et complétées par les fournisseurs. Cette méthode est adoptée pour des raisons d'efficacité. Un des ingénieurs système interviewés, se positionnant au sein de son entreprise comme un garant des processus, tient un discours contradictoire : il regrette que "l'IS (ingénierie système) n'ait pas le vent en poupe" et que "les gens ne suivent pas les processus". Mais il constate aussi l'efficacité d'un travail préalable à la rédaction des exigences en collaboration avec les fournisseurs des différents composants du système. Un autre ingénieur interviewé, travaillant au niveau composant, se réjouit de ce travail avec les ingénieurs système, fustigeant « la forme contre le fond », et indiquant l'importance d'« avoir les bureaux les uns à côté des autres ».

Enfin, nous observons qu'à ce stade initial du processus d'ingénierie des exigences, les ingénieurs collaborent plus autour de questions de conception qu'autour des exigences. Le but des ingénieurs est de créer une paire problème-solution satisfaisante. Ces observations sont cohérentes avec le modèle de coévolution du problème et de la solution de Dorst et Cross (Dorst et Cross 2001), que nous pouvons reprendre et enrichir avec la notion d'exigence pour décrire les activités des ingénieurs à ce stade initial du processus d'ingénierie des exigences (Figure 78). Le nombre d'allers-retours entre les deux dimensions n'est pas forcément prévisible, rendant parfois l'avancement du travail difficile à suivre et prévoir.
Nous détaillons dans les paragraphes suivants les activités réalisées par l'ingénieur en exigences et les ingénieurs composants, dans deux scénarios :

- Lister les questions de conception en dessinant l'architecture ;
- Partager et discuter les questions de conception et l'architecture.

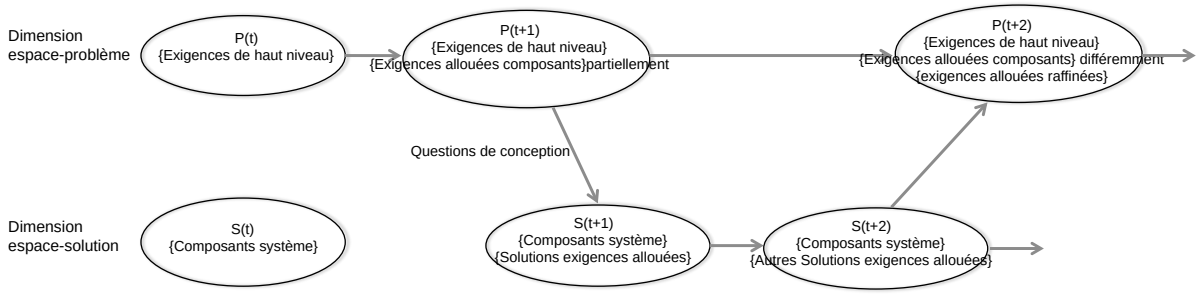



Figure 78: une coévolution des exigences allouées aux composants du système, issue d'une collaboration entre ingénieurs en exigences et fournisseurs de composants

## IV.2.1. Lister les questions de conception en dessinant l'architecture

L'ingénieur système utilise un outil de dessin tel que Visio pour construire un schéma de l'architecture. Il se sert de ce schéma pour identifier les questions de conception, et cela de façon exhaustive. La construction du schéma lui permet d'avoir une vision d'ensemble, et de « être sûr de ne rien louper » [proximité de la modélisation - élevée] [propension à l'erreur- faible] [**R1**]. Il préfère utiliser des logiciels de dessin plutôt que des outils de modélisation système car les dessins vont évoluer dans le temps et doivent être facilement modifiables [caractère exploratoire- élevé] [viscosité- faible] [**R2**] : « Ce sont des hypothèses d'étude, et du fait que c'est un dessin, cela se change de manière très simple ». C'est au moment où il construit le schéma qu'il va discerner/découvrir les questions à résoudre : « avec ce genre de schéma, les sujets apparaissent d'eux-mêmes ». En parallèle à la construction du schéma, il va rédiger les questions dans un fichier Excel (exemple en Figure 80), en les structurant par thème/questions/référence vers le document de spécification système [notation secondaire - élevée].

Les liens entre les questions de conception listées et le schéma ne sont pas explicités, ils sont dans la tête de l'ingénieur [dépendances cachées – élevées] [**R3**,**R4**]. Quand les liens sont explicités, comme dans la Figure 79, ils sont faits par l'ingénieur dans un fichier Word, dans lequel le schéma est inséré, et sur lequel il rajoute une flèche et du texte se référant au numéro du problème [efficience – perte de temps lors de l'édition] [efficience – perte de temps lors de la lecture liée à l'indirection] [**R4**]. Certaines exigences ne soulèvent pas de questions pour l'ingénieur système, qui envisage immédiatement comment elles vont se décliner sur les composants [**R5**].

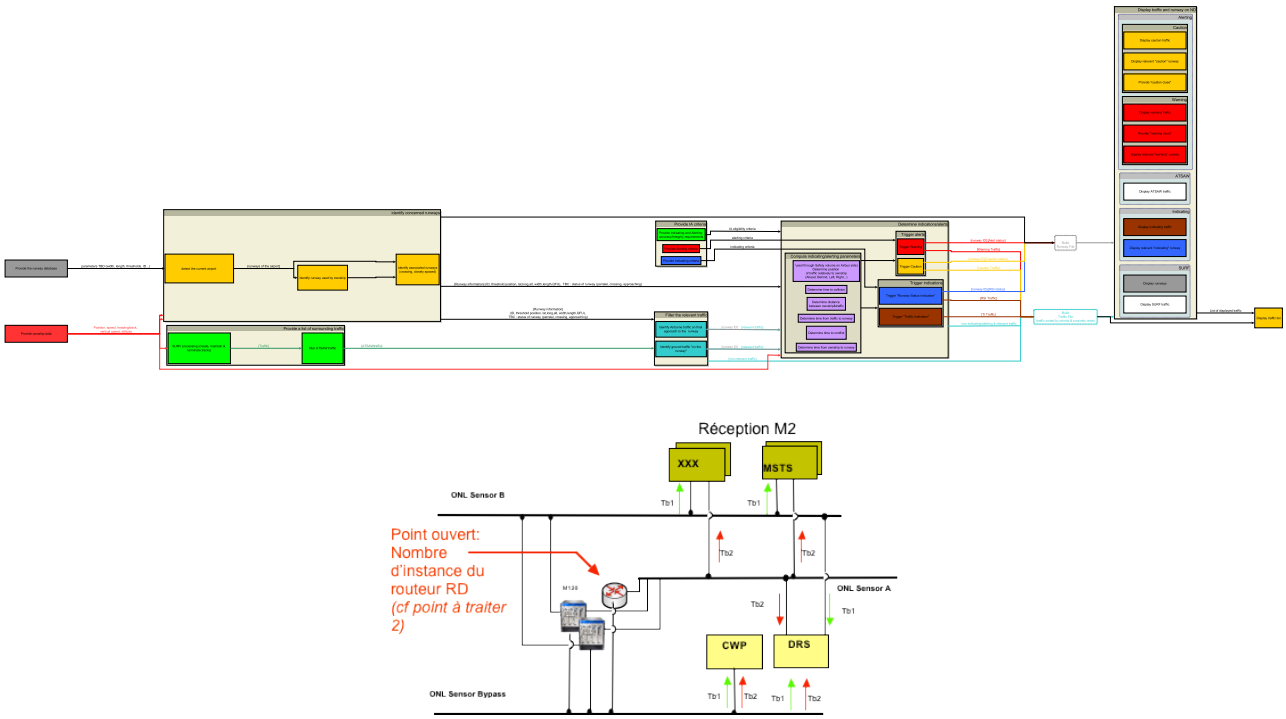



Figure 79: exemples de schéma d'architecture

| Theme | sujet | Commentaire ESA | Chapitre EDS | Chapitre EDSP | atelier | date atelier | sortie attendues |
|---|---|---|---|---|---|---|---|
| Association / Correspondance | Correspondances (SFPL/segment) – (RI/PLN) | | 6.5.2.3.1 | | 1 | | chapitre 6.5.2.3.1 renseigné |
| | Association OLDI 4F – (RI/PLN) | | 6.5.2.3.2 | | | | |
| | Commentaire MOECER n°40 sur EDSP : | La solution préconisée est mise en œuvre mais s'appuie sur une 'correspondance' entre DDS:segment et réentrances multiples RI sans faille. | | 4.3.2 | | | |
| Conception | Architecture fonctionnelle | Allocation des fonctions aux composants WRAPPER / STPV | 6,2 | | | | |
| | Identification des notifications DDS nécessaires pour les EIC | le but de ces ateliers consistent à traiter les points durs éventuels identifiés par MOECER | Transverse | | | | |
| | Identification des inputs RMI nécessaires pour les EIC | le but de ces ateliers consistent à traiter les points durs éventuels identifiés par MOECER | Transverse | | | | |

Figure 80: exemple de tableau rassemblant les questions par thème

En résumé, afin d'éliciter les questions de conception posées par des exigences système initiales, les outils d'ingénierie des exigences doivent permettre à l'ingénieur en exigences de :

**R1** : dessiner des vues comportementales du système en utilisant des éléments graphiques tels que des rectangles et des flèches (efficacité : pas besoin de se souvenir ou de respecter la sémantique d'une notation pour favoriser la proximité de la modélisation) ;

**R2** : changer les vues comportementales du système avec un nombre minimum d'actions (efficience dans les actions d'édition pour supporter l'exploration) ;

**R3** : créer une question de design à partir d'un élément graphique (flèche ou rectangle) de la vue comportementale avec un nombre minimum d'actions (efficience de la création d'une question pour favoriser la traçabilité) ;

**R4** : Visualiser et éditer une liste tabulaire de questions de design en lien avec les exigences système initiales (design : la liste tabulaire des questions de design est une vue complémentaire de la vue graphique) (efficience dans la navigation : les deux vues sont coordonnées) ;

**R5** : Editer une exigence, un dessin ou une question de design dans n'importe quel ordre et dans n'importe quelle vue (efficacité dans le nombre de questions de design exprimées) (efficience : pas d'ordre prédéfini pour supporter l'exploration et éviter l'engagement prématuré).

L'utilisation de dessins par le concepteur, comme mémoire externalisée mais aussi comme moyen de réinterpréter ce qu'il a dessiné et découvrir une nouvelle direction de conception, a déjà été décrite par Schon et Wiggins (Schon et Wiggins 1992). Notre analyse montre que les outils à vocation générale, telles que Word, Excel, PowerPoint ou des logiciels de dessin, sont préférés car ils supportent ces pratiques. Les exigences que nous proposons vont dans le sens d'un support de ces pratiques par les outils d'ingénierie des exigences.

## IV.2.2. Partager et discuter les questions de conception et l'architecture

Dans un deuxième temps, l'ingénieur envoie par messagerie électronique le schéma d'architecture et la liste des questions de conception aux fournisseurs des composants du système et propose une réunion de travail. L'objectif de la réunion pour l'ingénieur est de présenter le schéma et les questions de conception identifiées aux fournisseurs, et voir avec eux s'« ils ont déjà la solution de leur côté pour réaliser la fonction ». La réunion peut avoir lieu en vidéoconférence, quand un des fournisseurs n'est pas sur place. Nous n'avons pas assisté à ce genre de réunion, mais selon les interviewés, la réunion aborde une par une les questions de conception identifiées dans le document, permet d'avoir une compréhension partagée des problèmes pour envisager et confronter des pistes de solution réalisables [visibilité – faible, chacun a sa vue] [**R6**]. Les pistes de solution ne sont pas notées en séance dans les documents, mais précisées a posteriori par messagerie électronique [opérations mentales difficiles]. En effet, lorsqu'on demande les résultats de ces réunions, l'ingénieur utilise sa messagerie en filtrant sur le mot *atelier* ou sur la date de la réunion [efficience faible – perte de temps dans la recherche d'information] [**R7**]. De fait, il n'existe pas vraiment de compte-rendu formel de ces ateliers : l'ingénieur compte alors essentiellement sur sa mémoire, sa connaissance du système, son expertise « je n'ai pas besoin de compte-rendu pour répondre aux questions techniques ». En revanche, en cas de transfert de l'activité vers un collègue, « ce serait plus difficile ». A l'issue des échanges, lors des ateliers et par messagerie, l'ingénieur peut être conduit à modifier une exigence système initiale (par exemple la performance spécifiée n'est pas atteignable). Il n'utilise pas le mode « Suivi des modifications » de Word, parce qu'il veut souligner le changement du contenu seulement, pas le changement de formatage, et d'une manière persistante. Pour atteindre cet objectif, il écrit « WAS » (*était* en anglais) devant l'énoncé qu'il veut modifier, barre l'énoncé, puis écrit « IS » (*est* en anglais) suivi du nouvel énoncé [efficacité élevée dans la mise en valeur persistante d'un changement d'exigence] [notation secondaire possible dans Word]. Par contre, les discussions motivant cette modification sont dans les boîtes électroniques des différentes parties prenantes, et ne sont donc pas reliées à la modification d'exigence [dépendances cachées] [**R8**].

En résumé, afin de collaborer autour des questions de conception levées par un ensemble d'exigences système initiales, les outils d'ingénierie des exigences doivent permettre à l'ingénieur en exigences et aux fournisseurs de composants de :

**R6** : Partager la liste tabulaire des questions de conception et la vue comportementale dans une réunion co-localisée ou distribuée ;

**R7** : Discuter une question de conception dans n'importe quelle vue (graphique ou tabulaire) (efficience dans l'édition et l'accès aux discussions pour favoriser la traçabilité) ;

**R8** : Changer l'énoncé d'une exigence d'une façon visible et persistante (exemple de design : ETAIT ~~+ ancien énoncé~~ et EST+ nouvel énoncé) en lien avec une discussion (efficacité dans la traçabilité des changements).

Dans un souci d'efficience dans le travail de l'ingénieur, nous n'intégrons pas dans nos exigences l'utilisation d'une notation de design rationale, telles que IBIS (Kunz et Rittel 1970) ou QOC (MacLean et al. 1991), qui permettent de justifier des choix, mais sont coûteuses en termes d'utilisation. En effet, ces notations exigent des ingénieurs des actions supplémentaires en prescrivant une structuration spécifique des données (Conklin et Yakemovic 1991). Notre approche est de privilégier le support à la collaboration entre ingénieurs en intégrant dans les artefacts manipulés par les utilisateurs des moyens de collaboration. Dans cette approche orientée processus (Conklin et Yakemovic 1991), une plus-value secondaire est l'enregistrement des discussions et le disponibilité d'un historique des discussions pour consultation ultérieure.

## IV.3. Expérimentation d'une modélisation d'exigences avec outil spécifique

Nous avons observé le contexte d'utilisation d'expérimentation d'une modélisation d'exigences avec outil spécifique dans les entreprises A, C et D du Tableau 4. Les entreprises s'orientent vers des outils de modélisation SysML afin de contrer la croissance incontrôlée des exigences textuelles, évoquant la maxime « un schéma vaut mieux qu'une tonne de texte ». Les entreprises observées testent la modélisation de système sur des projets de recherche ou des futures affaires, mais n'envisagent pas de transposer les référentiels de systèmes existants : la transposition serait trop coûteuse. La force des outils testés est de pouvoir être connectés à l'aide de frameworks à d'autres outils, par exemple OpenModelica pour décrire des comportements, Reqtify pour la gestion des liens. L'intérêt officiel de ces outils est de pouvoir éditer l'architecture fonctionnelle, préciser des performances et des contraintes sur les fonctions dans un deuxième temps, pour ensuite être en mesure de simuler le fonctionnement du système modélisé. La modélisation et la simulation de solutions alternatives permettent alors de comparer les solutions et de faire des choix argumentés.

Nous avons pu constater lors des interviews que les outils de modélisation ne sont pas utilisés de cette façon. En effet, seule la solution retenue est modélisée, le travail de comparaison des solutions et de décision se faisant par jugement d'expert lors des ateliers entre ingénieurs que nous avons précédemment décrits. La conséquence est que les solutions alternatives ne sont pas toujours enregistrées, ni les discussions et compromis qui ont conduit à cette solution. Cette situation peut en partie s'expliquer par les problèmes d'utilisabilité que nous décrivons dans les deux scénarios suivants :

- l'édition d'un modèle ;
- la communication sur un modèle.

Dans nos scénarios, Alex est une ingénieure au sein d'un projet de R&D visant à appliquer la modélisation système sur la gestion de crise.

## IV.3.1. Edition d'un modèle

Alex travaille sur le modèle de la Figure 81, dans lequel les rectangles modélisent les quartiers d'une ville, le quartier général, les casernes, et les pompiers. Elle rajoute une sortie sur l'acteur « pompier », ce qui impacte tous les rectangles à droite du modèle, avec une nouvelle sortie. Mais elle souhaite que la nouvelle sortie soit positionnée en bas à droite du rectangle, alors qu'elle a été positionnée par ordre alphabétique au milieu du rectangle : il faut qu'elle réalise cette opération sur chacun des rectangles : « c'est boite par boite, il manque des macros » [satisfaction faible] [viscosité élevée]. Elle déplace ensuite les rectangles un par un, car ils sont trop près du bord de la fenêtre [satisfaction faible] [viscosité élevée] [**R9**]. Cette résistance aux changements peut expliquer la préférence des utilisateurs pour les éditeurs de dessin, que nous avons décrite dans le paragraphe précédent. Ensuite, Alex veut préciser une exigence sur le quartier général : elle double-clique sur le rectangle correspondant, ce qui déclenche l'ouverture d'une fenêtre par-dessus le modèle, qui contient 12 onglets (voir Figure 81), qu'elle parcourt un par un pour trouver l'attribut qu'elle veut spécifier : « on a l'impression que les propriétés sont dans l'ordre inverse de notre besoin » [satisfaction – faible] [efficience faible – perte de temps] [**R10** ]. Le travail sur le modèle n'est pas linéaire : Alex n'est pas capable de remplir toutes les propriétés en une seule fois, elle ne dispose pas de toutes les informations nécessaires. Or elle a travaillé sur un autre projet la semaine dernière, et ne se souvient plus des propriétés qu'elle a déjà remplies et celles qu'elle doit renseigner. Elle passe en revue chaque élément modélisé pour afficher la boite de propriétés dans laquelle elle consulte un à un les onglets [efficience – perte de temps] [évaluation progressive – faible] [**R11**]. La perte de temps est d'autant plus grande si deux ingénieurs travaillent de façon concurrente sur le même modèle, un utilisateur ne pouvant pas se souvenir de ce que l'autre a fait.

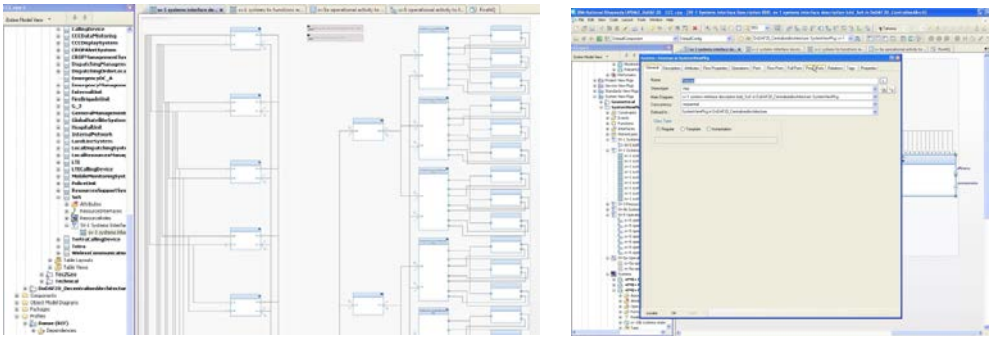


Figure 81: exemples de modèle de système (à gauche) et fenêtre des propriétés (à droite)

En résumé, afin de raffiner le modèle d'un système, les outils doivent permettre à l'ingénieur en exigences de :

**R9** : Modifier les propriétés graphiques d'un ensemble d'objets avec un nombre minimum d'actions (efficience dans les actions d'édition pour favoriser la modélisation d'alternatives) (satisfaction de se concentrer sur les tâches de modélisation) ;

**R10** : Chercher et éditer les propriétés par mot clé à partir d'un élément graphique (efficience dans les actions de recherche pour préciser la modélisation) ;

**R11** : Retrouver l'état d'avancement des propriétés à partir d'un élément graphique (efficience dans le filtrage d'information et l'évaluation progressive).

### IV.3.2. Présentation du modèle

Alex veut présenter le modèle aux parties prenantes du projet. Elle considère que le modèle final est difficile à lire et à comprendre en tant que produit indépendant. Elle produit un support de présentation de type vidéo qui intègre une description illustrée des différents acteurs (voir Figure 82), afin de présenter le contexte pas à pas et de façon dynamique avant de montrer le modèle. Pour réaliser la vidéo, elle produit des diapositives PowerPoint et utilise le logiciel CamStudio pour capturer une démonstration interactive du modèle dans l'outil de modélisation. L'utilisation de l'outil spécifique n'est en effet pas possible sans accès à des répertoires partagés sur le réseau de l'entreprise.


Figure 82: exemple de présentation du contexte du modèle

Par conséquent, l'ingénieure a passé du temps à mettre en forme la modélisation SysML, mais passe encore du temps dans la mise en forme d'un support de présentation pour communiquer avec les parties prenantes. On peut noter que le support préparé augmente la proximité de la modélisation (carte présentant les districts, la répartition des stations et le lieu de la crise), qui favorise la communication. Notre scénario renforce le besoin d'améliorer l'expressivité des notations proposée par Moody (D. Moody 2009), mais montre la nécessité de travailler sur des modèles plus grands pour tester la lisibilité.

En l'état actuel, modéliser un système complexe en entier avec la notation SysML n'est pas envisageable pour les entreprises participant à l'étude : les outils ne favorisent pas l'exploration et la réflexion, du fait des problèmes d'utilisabilité que nous venons de relever dans les scénarios, et les modèles produits ne facilitent pas la communication, la modélisation étant trop éloigné de ce qu'elle décrit.

## IV.4. Raffinement et gestion des exigences dans le référentiel

Nous avons observé le contexte d'utilisation "Raffinement et gestion des exigences dans le référentiel" dans les entreprises C et D du Tableau 4. Ce contexte survient dans les stades ultérieurs du processus d'ingénierie des exigences. Les utilisateurs sont les ingénieurs en exigences qui sont responsables de la gestion des exigences d'un système en opérations. Plusieurs évolutions (et corrections d'anomalies) sont à l'étude, et plusieurs versions sont en cours de développement. L'objectif des ingénieurs est de construire et maintenir les référentiels d'exigences des différentes versions, afin de fournir les documents de traçabilité nécessaires au processus de certification (RTCA et EUROCAE 2011). Ils utilisent principalement un outil spécifique, DOORS, pour gérer les référentiels et générer la documentation, et les feuilles de calcul comme solution de contournement. Les ingénieurs expriment beaucoup d'insatisfaction à ces stades : les activités sont évaluées comme

routinières et « administratives », caractérisées par moins de créativité et plus de rigueur. Les principales activités que nous avons observées sont dirigées par la recherche de la complétude et la traçabilité des exigences :

- rédaction d'exigences cohérentes ;
- conduite d'analyse de couverture ;
- gestion du contenu des versions de système.

Nous présentons en détail les activités sous forme de scénarios, et analysons l'utilisabilité des outils en usage. L'équipe est composée de dix ingénieurs. Claire, Fred et Phil sont des membres de l'équipe. Ils utilisent le logiciel DOORS pour éditer et gérer les exigences. Chacun a deux écrans, pour pouvoir afficher DOORS en plein écran.

## IV.4.1. Rédaction d'exigences cohérentes

Claire est en train de rédiger une exigence : elle commute constamment entre le modèle de données affiché sur le deuxième écran, et l'outil DOORS, afin d'utiliser les termes exacts dans l'énoncé de l'exigence [efficience- perte de temps] [R12]. En effet, les concepts du domaine qui doivent être utilisés sont définis dans un diagramme entité-relations dans un autre outil dédié. Une règle interne à l'entreprise spécifie que les concepts du domaine doivent apparaître en italiques dans une exigence textuelle bien formulée, afin d'en faciliter ultérieurement la lecture. Pour satisfaire cette règle, Claire change manuellement le formatage du texte de toutes les occurrences du domaine [efficience – perte de temps] [R12 R13]. Claire veut ensuite insérer un lien entre l'exigence qu'elle vient de rédiger et une exigence de plus haut niveau : elle cherche l'exigence en utilisant l'arborescence de navigation à gauche, mais elle ne la trouve pas. C'est Phil qui vient de la créer mais Claire n'y a pas accès en lecture, c'est pour cela qu'elle ne la trouve pas [efficience – perte de temps] [visibilité faible selon droit d'accès]. Elle demande à Phil de lui donner accès (ils sont dans le même bureau), ce qui lui permet d'afficher l'exigence. Elle règle l'affichage de façon à visualiser les deux exigences afin de pouvoir établir le lien entre les deux exigences par glisser-déposer [efficience – perte de temps]. Fred est un nouvel arrivant dans le projet, il vient de créer l'exigence « interagir avec la séquence de vols », et travaille sur sa formulation. Or il existe déjà une exigence « manipuler la séquence de vols » : c'est le responsable de projet qui le lui signale à la fin de la semaine, il le sait car il travaille sur le système depuis plusieurs années maintenant [efficience – perte de temps sur la semaine] [efficacité – redondance entre exigences potentiellement non détectée] [R14]. La détection de ce doublon entre deux exigences est tardive, fortuite et dépendante de l'expertise du responsable de projet. Cela ne peut pas constituer une solution pérenne, d'autant que le roulement du personnel est inévitable sur des systèmes ayant de longs cycles de vie.

En résumé, afin de rédiger des exigences cohérentes, les outils doivent permettre à l'ingénieur en exigences de :

**R12** : sélectionner et ajouter des termes du modèle de données pendant l'édition de l'énoncé de l'exigence (efficience dans les actions d'édition de l'exigence et du modèle de données) (efficacité : les mots sont cohérents avec le modèle de données) ;

**R13** : faciliter la mise en valeur des mots du modèle de données dans l'énoncé de l'exigence (efficience dans les actions d'édition) ;

**R14** : détecter les similitudes d'exigences pendant l'édition de l'énoncé d'une exigence (efficacité : pas de redondance).

Les exigences que nous proposons vont dans le sens d'une vérification interactive de la cohérence pendant la rédaction des exigences, basée sur des algorithmes syntaxiques permettant de détecter des similitudes et de les signaler à l'utilisateur. Cette logique est cohérente avec la proposition de Berry et al. (D. Berry et al. 2012) de réfléchir à la répartition humain-machine, étant données les limites des algorithmes de traitement de texte. A noter qu'en l'état actuel de nos connaissances, certains problèmes d'utilisabilité, liés à l'édition et partage de liens sont difficiles à transformer directement en exigences d'utilisabilité.

## IV.4.2. Conduite d'analyse de couverture

Maintenant, Claire veut lire les exigences raffinées qui ont été écrites par Phil, dans le but de vérifier leur conformité par rapport aux règles de rédaction des exigences, et leur couverture par rapport aux exigences système de plus haut niveau. Elle part d'une exigence système, et sélectionne la flèche jaune attachée à l'exigence. Quel que soit le nombre d'exigences raffinées, la représentation est la même flèche, ce qui communique une impression de structure plate, décrite comme peu claire par les participants [satisfaction faible] [visibilité faible]. La sélection de la flèche déclenche l'ouverture d'une liste déroulante présentant des chemins de répertoires, par exemple /Project/Baseline/Evolutions/R347 (voir Figure 83), ce qui ne donne aucune information sur le contenu de l'exigence [satisfaction faible] [visibilité faible]. Claire sélectionne les exigences raffinées une par une dans la liste [possibilité de juxtaposition faible] pour faire une vérification individuelle de leur conformité, et évaluer que l'ensemble des exigences raffinées permet de couvrir l'exigence système initiale. De plus, Claire doit se rappeler quelles exigences elle a déjà lues dans la liste et celles qui restent à lire [évaluation progressive faible]. Souvent, pour faire le travail, l'équipe décide de faire un export Excel (exemple en Figure 84) [visibilité et possibilité de juxtaposition élevées] et rajoute une partie (en bleu à droite) pour le traitement des écarts identifiés [notation secondaire –élevée]. La colonne « écart oui/non » leur permet de filtrer facilement sur les écarts identifiés afin de les traiter avec les fournisseurs des composants [efficience élevée]. Cependant, dans le cas où il faut modifier une exigence, Claire doit retourner dans DOORS et chercher l'exigence à partir de son identifiant, en se plaçant dans le bon module [efficience – perte de temps].

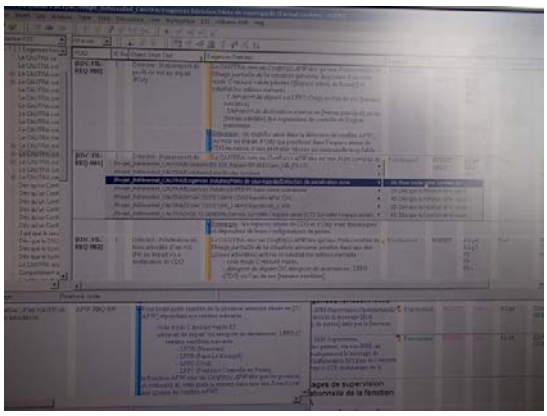
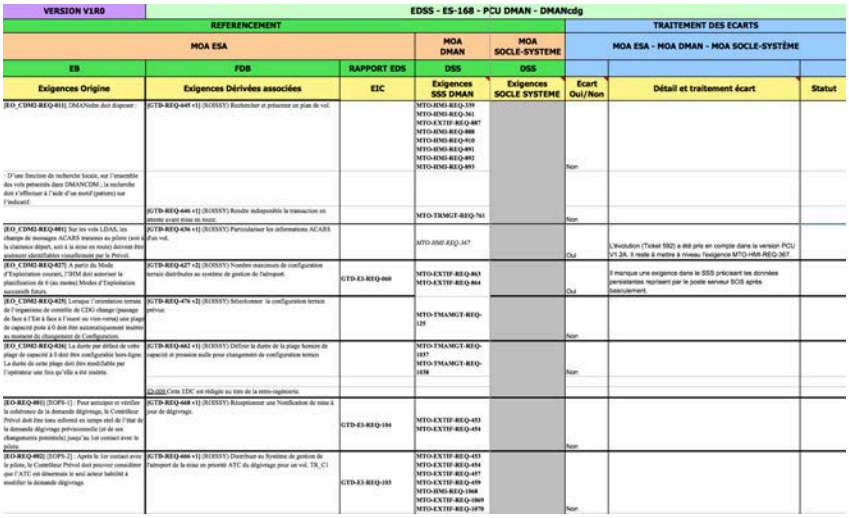


Figure 83: accès à partir d'une exigence système aux exigences raffinées dans DOORS



| VERSION V1R0 | EDSS - ES-168 - PCU DMAN - DMANcdg | | | | | | |
|---|---|---|---|---|---|---|---|
| REFERENCEMENT | | | | | TRAITEMENT DES ECARTS | | |
| MOA ESA | | | MOA DMAN | MOA SOCLE-SYSTEME | MOA ESA - MOA DMAN - MOA SOCLE-SYSTÈME | | |
| EB | FDB | RAPPORT EDS | DSS | DSS | | | |
| Exigences Origine | Exigences Dérivées associées | EIC | Exigences SSS DMAN | Exigences SOCLE SYSTEME | Ecart Oui/Non | Détail et traitement écart | Statut |
| [EO_CDM2-REQ-011] DMANedm doit disposer | [GTB-REQ-645 v1] (ROSSSY) Rechercher et présenter un plan de vol. | | MTO-HMI-REQ-339<br>MTO-HMI-REQ-361<br>MTO-EXTIF-REQ-487<br>MTO-HMI-REQ-888<br>MTO-HMI-REQ-910<br>MTO-HMI-REQ-891<br>MTO-HMI-REQ-892<br>MTO-HMI-REQ-893 | | Non | | |
| D'une fonction de recherche locale, sur l'ensemble des vols présentés dans DMANCDM ; la recherche doit s'effectuer à l'aide d'un motif (pattern) sur l'indicatif. | | | | | | | |
| | [GTB-REQ-646 v1] (ROSSSY) Rendre indisponible la transaction en attente avant mise en route. | | MTO-TRMGT-REQ-761 | | Non | | |
| [EO_CDM2-REQ-001] Sur les vols LDAS, les champs de messages ACARS transmis au pilote (soit à l'un vol, la clairance départ, soit à la mise en route) doivent être aisément identifiables visuellement par le Prévol. | [GTB-REQ-636 v1] (ROSSSY) Particulariser les informations ACARS | | *MTO-HMI-REQ-367* | | Oui | L'évolution (Ticket 592) a été pris en compte dans la version PCU V1.2A. Il reste à mettre à niveau l'exigence MTO-HMI-REQ-367. | |
| [EO_CDM2-REQ-027] A partir du Mode d'Exploitation courant, l'IHM doit autoriser la planification de 6 (au moins) Modes d'Exploitation successifs futurs. | [GTB-REQ-627 v2] (ROSSSY) Nombre maximum de configuration terrain distribuées au système de gestion de l'aéroport. | GTD-E3-REQ-060 | MTO-EXTIF-REQ-863<br>MTO-EXTIF-REQ-864 | | Oui | Il manque une exigence dans le SSS précisant les données persistantes reprises par le poste serveur SOS après basculement. | |
| [EO_CDM2-REQ-028] Lorsque l'orientation terrain de l'imposition de contrôle du CDG change (passage de flux à l'Est à flux à l'ouest ou vice-versa) une plage de capacité point à 0 doit être automatiquement insérée au moment du changement de Configuration. | [GTB-REQ-476 v2] (ROSSSY) Sélectionner la configuration terrain prévue. | | MTO-TMAMGT-REQ-125 | | Non | | |
| [EO_CDM2-REQ-026] La durée par défaut de cette plage de capacité à 0 doit être configurable hors-ligne. La durée de cette plage doit être modifiable par l'opérateur une fois qu'elle a été insérée. | [GTB-REQ-662 v1] (ROSSSY) Définir la durée de la plage horaire de capacité et pression nulle pour changement de configuration terrain | | MTO-TMAMGT-REQ-1057<br>MTO-TMAMGT-REQ-1058 | | Non | | |
| | D-000/Cette EDC est rédigée au titre de la rétro-ingénierie. | | | | | | |
| [EO-REQ-001] [EOPS-1] : Pour anticiper et vérifier la cohérence de la demande déprivage, le Contrôleur Prévol doit être tenu informé en temps réel de l'état de la demande déprivage prévisionnelle (et de ses changements potentiels) jusqu'au 1er contact avec le pilote. | [GTB-REQ-668 v1] (ROSSSY) Réceptionner une Notification de mise à jour de déprivage. | GTD-E3-REQ-104 | MTO-EXTIF-REQ-453<br>MTO-EXTIF-REQ-454 | | Non | | |
| [EO-REQ-002] [EOPS-2] : Après le 1er contact avec le pilote, le Contrôleur Prévol doit pouvoir consulter que l'ATC est désormais le seul acteur habilité à modifier la demande déprivage. | [GTB-REQ-666 v1] (ROSSSY) Distribuer au Système de gestion de l'aéroport de la mise en priorité ATC du déprivage pour un vol. TS_C1 | GTD-E3-REQ-103 | MTO-EXTIF-REQ-453<br>MTO-EXTIF-REQ-454<br>MTO-EXTIF-REQ-457<br>MTO-EXTIF-REQ-459<br>MTO-HMI-REQ-1068<br>MTO-EXTIF-REQ-1069<br>MTO-EXTIF-REQ-1070 | | Non | | |

Figure 84: exemple d'export Excel pour vérifier la couverture

## IV.4.3. Gestion de contenus de version

Il y a simultanément quatre versions en cours de définition pour le système, chronologiquement V9, V9.1, V9.2 et V9.3. C'est une situation courante, qui permet au comité de pilotage de mettre des priorités sur les nombreuses évolutions issues des utilisateurs du système et du contexte règlementaire changeant. Au sein de l'équipe d'ingénierie des exigences, Phil établit et maintient la matrice de la Figure 85 : il a listé les évolutions horizontalement en haut (une par colonne), et les composants système verticalement à gauche (une par ligne). Si l'évolution impacte un composant, Phil reporte 1 dans la cellule correspondante. Phil rapporte que le comité de pilotage se plaint de cette visualisation, mais il ne comprend pas pourquoi : « tu peux trouver toutes les informations que tu veux dans cette matrice » [satisfaction – faible].

L'équipe d'ingénierie des exigences évalue en parallèle l'impact de 57 évolutions sur le système, avec des intitulés reflétant leur contenu. Une évolution, que nous nommerons E-A pour des raisons de confidentialité, est planifiée pour être intégrée dans la version V9, alors qu'une autre évolution, que nous nommerons E-D, est planifiée dans la version consécutive V9.1. Claire modifie l'exigence R1, en lien avec l'évolution E-A et applicable en V9, et Fred modifie en parallèle la même exigence R1, en lien avec l'évolution E-D et applicable en V9.1. La version V9.1 étant consécutive à la version V9, les modifications de R1 liées à E-A sont insérées dans l'exigence R1 modifiée pour l'évolution E-D. Cela veut dire que l'équipe gère trois versions de l'exigence R1 en parallèle :

- R1-V0, avant E-A and E-D ;
- R1-V1 pour la version V9, dérivée de R1-V0 et tracée vers l'évolution E-A ;
- R1-V2 pour la version V9.1, dérivée de R1-V1 et tracée vers l'évolution E-D.

Les services de gestion de configuration de l'outil DOORS permettent aux deux ingénieurs d'en être conscients et de modifier de façon concurrente la même exigence R1 [efficacité – gestion de configuration des exigences]. Pendant ce temps, le comité de pilotage décide de changer les priorités des évolutions : l'évolution E-A devient moins prioritaire et par conséquent est intégrée dans la version ultérieure V9.2. La conséquence est que la chronologie entre les évolutions E-A et E-D est modifiée : « nous devons détricoter tout le travail fait », exprime un ingénieur interviewé.

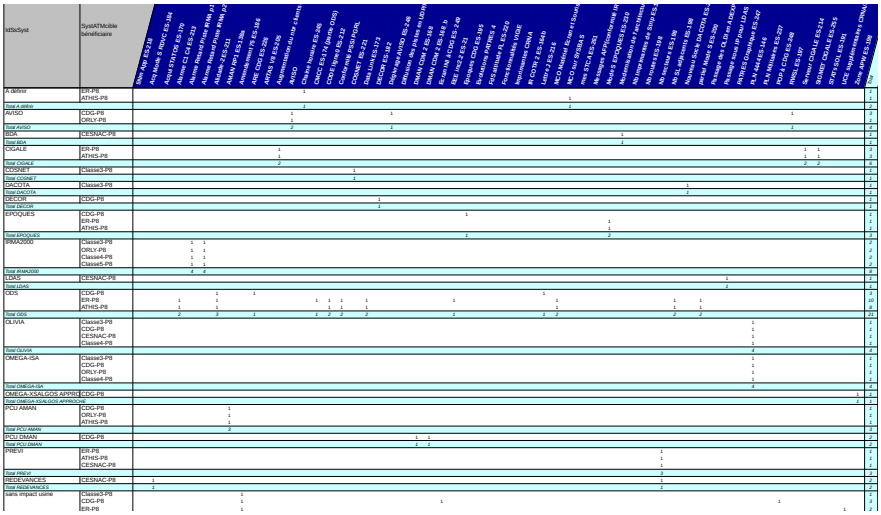


Figure 85: une matrice évolutions-composants construite par un ingénieur

En effet, afin d'obtenir des référentiels cohérents pour les versions V9, V9.1 et V9.2, l'équipe doit :

- modifier R1-V1 applicable en V9 (enlever les modifications liées à E-A et la trace correspondante) ;
- modifier R1-V2 applicable en V9.1 (enlever les modifications liées à E-A et la trace correspondante, vérifier les changements liées à E-D),
- et finalement modifier R1 applicable en V9.2 (intégrer les changements liés à E-A et la trace correspondante).

L'équipe doit procéder ainsi pour toutes les exigences impactées par l'évolution E-A : « trois semaines de travail sont nécessaires pour traiter les 182 exigences ». C'est typiquement une tâche de gestion d'exigences qui est considérée comme « décourageante » par les ingénieurs [satisfaction – faible]. Cependant cela reste une tâche obligatoire pour conserver la cohérence des référentiels exigée par la certification.

# IV.5. Synthèse

## IV.5.1. Le cycle en V comme Vernis logico-déductif

Les pratiques industrielles observées sont différentes de l'activité prescrite du cycle en V. En effet, les ingénieurs responsables de la spécification des exigences engagent un travail collaboratif de réflexion avec les fournisseurs des composants. Les ingénieurs utilisent des schémas libres, faits sur tableau blanc ou éditeur de dessin, comme moyen de raisonnement pour découvrir les questions de conception. Ce travail leur permet d'explorer et confronter des solutions techniquement réalisables et de construire une compréhension partagée des nouvelles exigences à remplir. A l'issue de ces ateliers de travail, l'ingénieur système va initier la rédaction des exigences par composants, avec des allers-retours avec les fournisseurs sur des questions laissées en suspens.

Les ingénieurs interviewés expriment une grande satisfaction sur cette façon de travailler : ils considèrent que la conception est le cœur de leur travail. Cependant, ils ressentent une contradiction entre l'efficacité de leurs pratiques et la non-application du processus prescrit du cycle en V. La littérature montre que cette contradiction est légitime. L'activité cognitive du concepteur n'est pas une classique activité de résolveur de problème, qui définit son problème en termes clairs, et cherche

ensuite une solution dans un espace bien défini. Le raisonnement du concepteur est désordonné (Rittel 1987)(Dorst et Cross 2001)(Lawson 2006), du fait de la nature des problèmes de conception. Le problème change pendant qu'il est traité, et ce sont des allers-retours entre problème et solution qui permettent aux concepteurs de façonner le problème, d'atteindre une adéquation satisfaisante entre la structure du système et le but à atteindre par le système (Mattessich 1978b). Le raisonnement du concepteur apparaît finalement comme un processus d'argumentation et de conversation (Rittel 1987)(Schon 1983) : le concepteur débat avec lui-même ou les autres, des problèmes surgissent, des propositions concurrentes sont développées en réponse et une recherche des pour et contre de chaque proposition est faite. Au final, le concepteur prend une décision sur une proposition, après modification des propositions débattues. Les problèmes et les propositions sont interconnectés et imbriqués. Ce raisonnement est difficile à décrire et illustrer en termes de tâches discernables et chronologiques à réaliser dans un projet, tel que le fait le cycle en V. Des alternatives au cycle en V ont été proposées, sous la forme de processus concurrents, itératifs, agiles (Boehm 2006)(Schwaber et Beedle 2001). On peut émettre  plusieurs propositions de processus (INCOSE 2015) s'efforçant de représenter un aspect itératif et incrémental dans la conception de système (voir Figure 86). Mais l'identification des itérations (ou incréments) impliquent un découpage du problème à résoudre en sous-problèmes, constituant le point d'entrée d'une itération. Or cela ne correspond pas non plus au raisonnement du concepteur que nous avons décrit précédemment. En effet, le découpage en itérations implique l'abandon des « méta-problèmes » (Rittel 1987) : suis-je en train de traiter le bon problème, ou ce problème est-il le symptôme d'un autre problème, de plus haut niveau, auquel je devrais plutôt m'attaquer ?

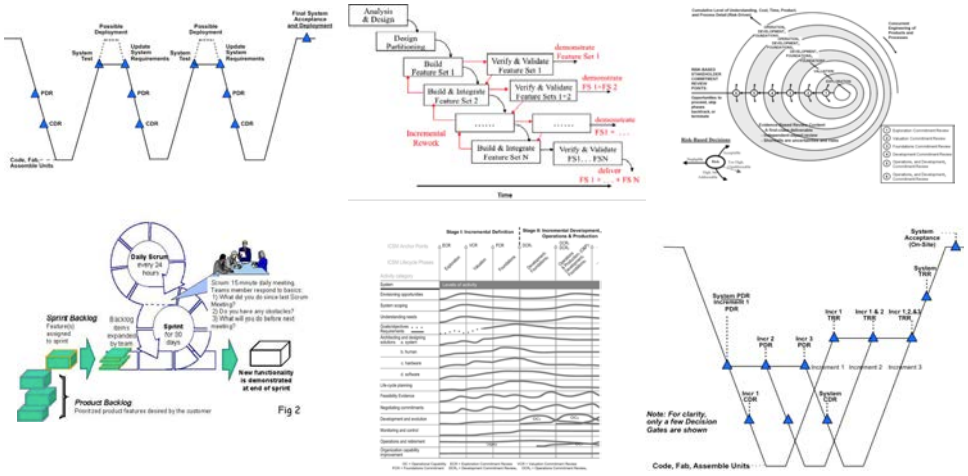


Figure 86: d'autres cycles que le cycle en V

Les différents cycles s'évertuent à capturer une organisation temporelle des tâches, un cheminement et un mode de conduite à suivre par les concepteurs qui n'est pas le leur. Les concepteurs sont perplexes (Boehm et Turner 2003) : il faudrait choisir entre des processus séquentiels et réputés rigoureux, tel le cycle en V, et des processus itératifs et agiles qui les empêchent d'avoir une appréhension globale du problème. Par conséquent, nous pensons qu'il est vain de continuer à réfléchir à des nouvelles formes de cycle, vouées à ne pas capturer le raisonnement désordonné du concepteur. Ce sont des plans, pour mobiliser des ressources, mais ils ne décrivent pas de façon précise le déroulement des actions dans le temps (Suchman 1987) :

> *Plans are resources for situated action, but do not in any strong sense determine its course. While plans presuppose the embodied practices and changing circumstances of situated action,*

> *the efficiency of plans as representations comes precisely from the fact that they do not represent those practices and circumstances in all of their concrete detail.*

Nous proposons d'assumer le cycle en V comme un vernis logico-déductif pour l'ingénierie des systèmes complexes, permettant de reconstruire une rationalité exigée pour la certification des systèmes, mais inopérante pour la conception. Cela doit permettre aux ingénieurs de suivre le processus d'argumentation désordonné nécessaire à la conception, sans se sentir à la marge des processus prescrits. Le fait de ne pas suivre strictement de méthode lors de la conception de solutions rejoint les positions de Feyerabend (Feyerabend et Hacking 2010) sur la production de connaissances scientifiques et la créativité. En prenant notamment comme exemples des découvertes scientifiques provoquées par la décision de penseurs de briser volontairement les règles, il préconise l'absence de méthode pour la croissance des connaissances : *Science is never a completed process*.

## IV.5.2. Une vision située de l'ingénierie des exigences

Nous cherchons à représenter l'ingénierie des exigences de façon cohérente à ce que nous avons observé. Nous venons de voir que les notions de cycle et séquence temporelle sont utiles en tant que plans, mais qu'elles ne traduisent pas les pratiques. Par conséquent, nous proposons de déconstruire la représentation séquentielle du cycle de vie du système de la norme IEEE1220, en n'ordonnant plus les phases dans le temps. En effet, à chaque instant, nous avons observé que des *situations* coexistent dans deux contextes, le contexte d'ingénierie du système et le contexte opérationnel d'utilisation du système (voir Figure 87), avec des parties prenantes différentes : client, utilisateurs, ingénieurs en exigences, fournisseurs des composants, réglementation.

La reconnaissance de la coexistence des deux contextes pour le système, à la fois en cours d'utilisation dans le contexte opérationnel et en cours de construction (définition et réalisation) dans le contexte d'ingénierie système nous permet de mieux retranscrire la richesse des situations observées selon les parties prenantes en présence et mettre en évidence l'aspect sociotechnique de l'ingénierie des exigences (Goguen 1994). Notre représentation permet de montrer le système de façon centrale comme un *champ commun de travail* (Schmidt et Simonee 1996) à toutes les parties prenantes :

> *Cooperative work is constituted by the interdependence of multiple actors who, in their individual activities, in changing the state of their individual field of work, also change the state of the field of work of others and who thus interact through changing the state of a* ***common field of work****.*

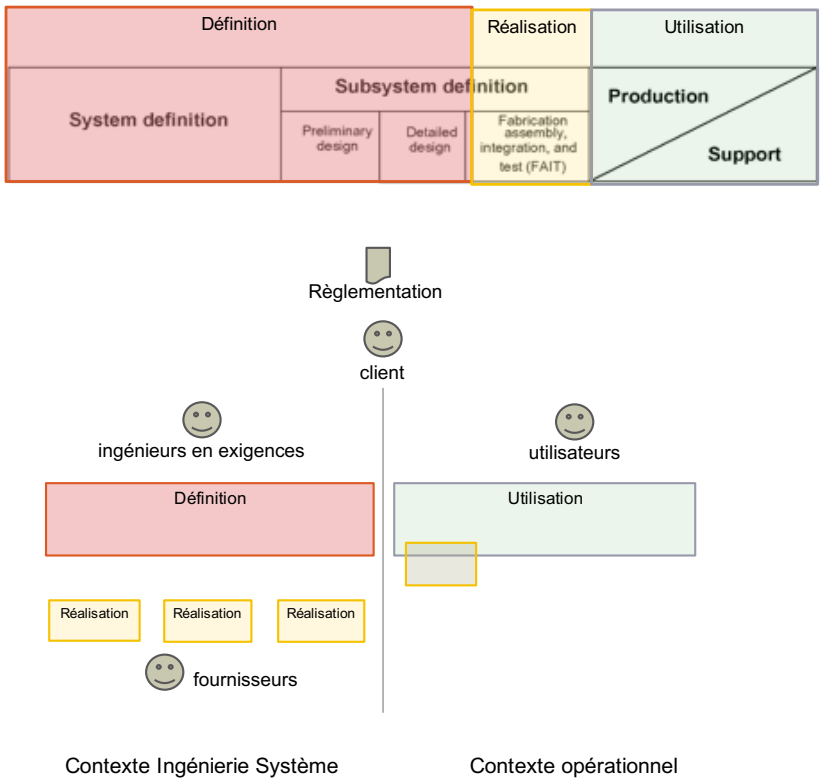



Figure 87: le cycle de vie d'un système selon IEEE1220, et notre proposition avec la représentation de la coexistence simultanée des contextes opérationnel et d'ingénierie du système et des parties prenantes

Les activités de parties prenantes sur le système sont distribuées dans le temps et dans l'espace, mais aussi semi-autonomes dans leurs stratégies et leurs objectifs : les exigences sont le point d'articulation (Schmidt et Simonee 1996) nécessaire à la coordination de ce travail désordonné de définition du système. Ainsi, comme représenté en Figure 88, à chaque instant :

- en contexte opérationnel :
  - une version de système est utilisée par les utilisateurs ;
  - les utilisateurs émettent de façon continue des demandes et des anomalies à partir de leur utilisation du système (en haut à droite de la Figure 88 ).
- en contexte d'ingénierie système (à gauche en Figure 88) :
  - les ingénieurs en exigences réalisent des études de définition système sur les demandes d'évolution et les anomalies pour aboutir à la spécification des exigences, selon les pratiques que nous venons de décrire dans ce chapitre ;
  - les fournisseurs de composants développent des versions de composants à partir des exigences spécifiées.

Le client, entre les deux contextes, a un rôle d'arbitre : il transmet les demandes des utilisateurs, en faisant des choix, du contexte opérationnel vers le contexte d'ingénierie système (en haut de la Figure 88). Dans le sens inverse, il décide des priorités sur les contenus de version. La réglementation émet des exigences, applicables sur tous les éléments et dans les deux contextes : le système en utilisation, le système en cours de définition, la façon de travailler des utilisateurs sur le système en

utilisation, la façon de travailler des ingénieurs en exigences et des fournisseurs de composants sur l'ingénierie du système.

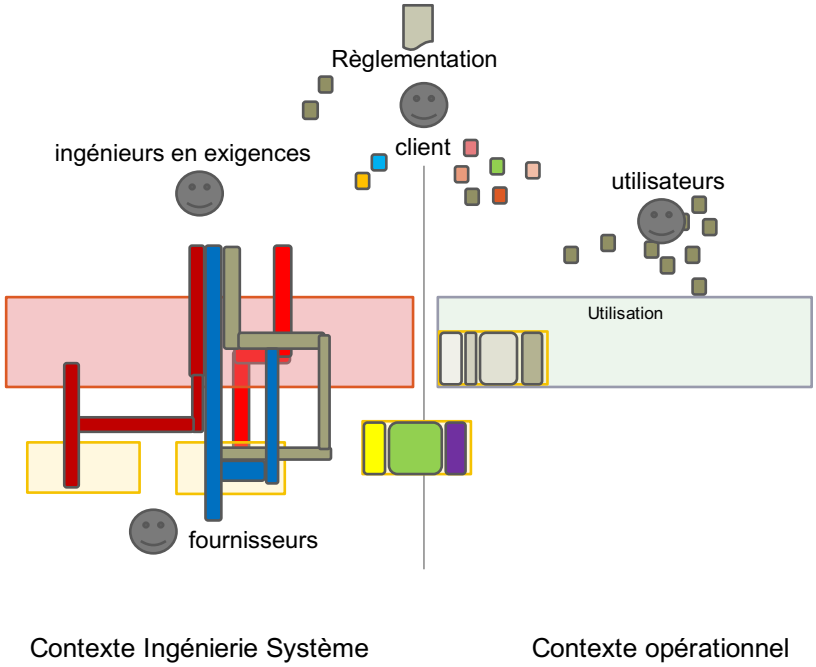



Figure 88: notre vision située du processus d'ingénierie des exigences

## IV.5.3. Les outils en usage dans le processus d'ingénierie des exigences

L'analyse détaillée des pratiques transverses aux entreprises nous permet d'identifier cinq aspects du processus d'ingénierie des exigences qui changent significativement dans le temps: la traçabilité, la maturité des exigences (la dimension spécification de Pohl (Pohl 1994)), la collaboration, la représentation du comportement du système et la satisfaction utilisateur. L'évaluation de ces cinq aspects révèle deux états opposés dans le temps que nous couplons avec les outils en usage (voir Figure 89). Dans le stade préliminaire du processus, représenté par la ligne grise en Figure 89, correspondant au contexte d'utilisation « Conception exploratoire et collaborative sans outil spécifique », les ingénieurs en exigences font rapidement face à la conception, c'est-à-dire « une adéquation satisfaisante d'une structure de système à un objectif spécifique à réaliser par le système » (Mattessich 1978a). Leur travail est collaboratif et centré sur les questions de conception avec les fournisseurs des composants. Ils dessinent des représentations du comportement du système comme moyen de raisonnement pour découvrir les questions de conception et décider l'allocation des exigences. Les outils à vocation générale sont utilisés pour supporter ce travail : des éditeurs de dessin et des tableaux blancs pour exprimer des vues dynamiques et éliciter les questions, Excel ou PowerPoint pour enregistrer les questions, messagerie électronique et vidéoconférence pour les partager. Les ingénieurs sont très satisfaits par ce travail : ils considèrent la conception comme leur activité principale et essentielle. Ils ne se plaignent pas des outils utilisés, qui sont souples, *pliant* selon Harris et Henderson (Harris et Henderson 1999). Par exemple, Excel est le support idéal pour l'accommodation : les utilisateurs ont juste à ajouter une colonne s'ils veulent traiter un nouvel attribut de l'exigence. Cependant, les différents fichiers produits sont stockés au mieux dans un répertoire partagé, mais plus fréquemment sur les ordinateurs personnels et les boîtes de messagerie. Les ingénieurs ne prennent pas le temps de mettre à jour les outils officiels avec leurs

enregistrements séparés. Les conséquences sont une dispersion et une perte des informations en termes de pré-traçabilité des exigences.

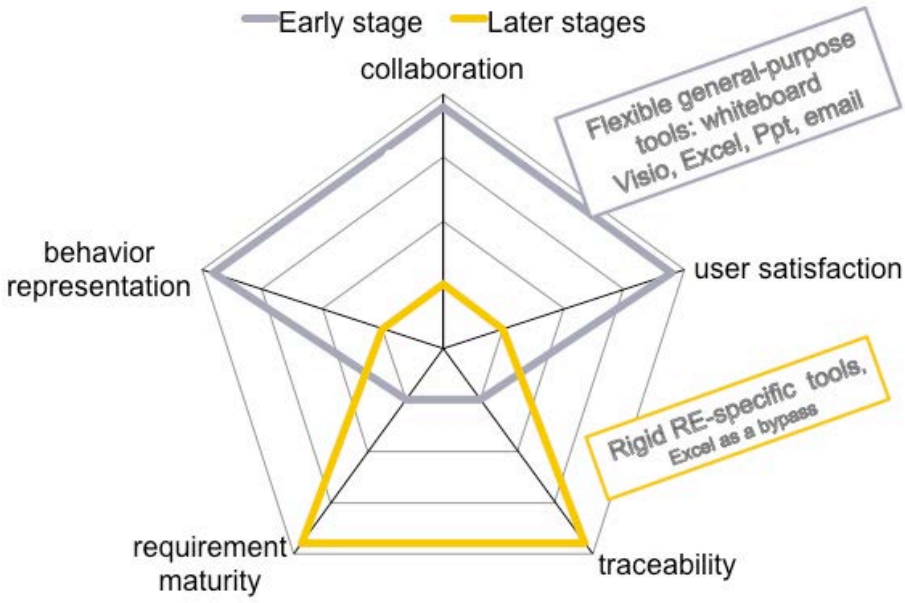



Figure 89: deux états opposés dans le processus d'ingénierie des exigences

Dans les stades ultérieurs, représentés par la ligne jaune sur la Figure 89, le travail des ingénieurs est centré sur la raffinement des exigences et la vérification de cohérence et complétude. Les ingénieurs expriment beaucoup d'insatisfaction dans ces stades : les tâches sont évaluées comme « routinières » et « administratives », caractérisées par moins de créativité et plus de rigueur. Ils utilisent des outils dédiés et Excel comme moyen de contournement pour réaliser ce travail. Comme le montrent nos scénarios, les outils spécifiques contraignent les ingénieurs à un flux de travail rigide et inadapté (Harris et Henderson 1999)(Suchman 1983), menant à des pertes de temps significatives. De plus, la collaboration avec les fournisseurs des composants n'est pas efficace : la liste des exigences raffinées transmet une vision fragmentaire du système, et rend les documents illisibles sauf par son rédacteur. Les conséquences sont une perte de motivation et une perte de temps dans l'établissement et la maintenance de la traçabilité d'un grand nombre d'exigences, s'additionnant à un intérêt intrinsèquement moins grand des ingénieurs pour ces activités.

L'existence de cette coupure nette au niveau des outils est contre-productive : elle entraine une perte d'information et une exacerbation de l'opposition entre créativité et formalisation, alors qu'il y a nécessité d'une continuité sur les exigences.

## IV.5.4. Quelles conséquences pour les outils ?

En reconnaissant le chaos qui entoure l'acte de conception, et en reconstruisant a posteriori un raisonnement logico-déductif nécessaire à la certification, l'ingénierie peut redevenir source de plaisir. Le défi est d'instrumenter conjointement l'absence de chemin prédéfini et la recherche de cohérence. Afin de préserver une absence de chemin prédéfini tout en favorisant l'enregistrement des informations dans les outils officiels, ces derniers doivent évoluer afin de supporter les utilisateurs dans la réalisation des activités du contexte d'utilisation Conception exploratoire et collaborative sans

outil spécifique, actuellement supportés par les outils à vocation générale : dessiner des représentations du système, partager et discuter des questions de conception, créer et mettre à jour les exigences à partir des questions de conception. Dans cet objectif, nous avons formulé les exigences d'utilisabilité R1 à R8. Ces exigences sont complétées par les exigences R9 à R11 issues de l'analyse des problèmes d'utilisabilité des outils de modélisation (voir la synthèse en Tableau 6). Notre constat est similaire à ceux de Visser (Visser 1990) et Guindon (R. Guindon et Curtis 1988), et les exigences sur les outils se rejoignent, avec un nouvel aspect sur la collaboration entre ingénieurs en exigences et fournisseurs. Afin de garantir la cohérence exigée par les standards pour démontrer l'atteinte d'un niveau de sécurité (RTCA et EUROCAE 2011), les outils officiels sont utilisés mais ne sont pas satisfaisants pour les utilisateurs, comme le montre l'analyse des scénarios du contexte d'utilisation « Raffinement et gestion des exigences dans le référentiel ». L'analyse de nos scénarios d'un point de vue de l'utilisabilité nous a permis de formuler les exigences d'utilisabilité R12 à R14 (voir synthèse en Tableau 6). Cependant, l'analyse seule n'est pas suffisante pour pouvoir formuler des exigences sur les outils, tant l'insatisfaction exprimée est grande : nous proposons un recours au design d'artefacts pour les identifier.

| Exigences d'utilisabilité | Origine |
|---|---|
| Afin d'éliciter les questions de conception posées par des exigences système initiales, les outils d'ingénierie des exigences doivent permettre à l'ingénieur en exigences de :<br><br>**R1** : Dessiner des vues comportementales du système en utilisant des éléments graphiques telles que des rectangles et des flèches (efficacité : pas besoin de se souvenir ou de respecter une signification sémantique d'une notation pour favoriser la proximité de la modélisation)<br><br>**R2** : Changer les vues comportementales du système avec un nombre minimum d'actions (efficience dans les actions d'édition pour supporter l'exploration)<br><br>**R3** : Créer une question de design à partir d'un élément graphique (flèche ou rectangle) de la vue comportementale avec un nombre minimum d'actions (efficience de la création d'une question pour favoriser la traçabilité)<br><br>**R4** : Visualiser et éditer une liste tabulaire de questions de design en lien avec les exigences système initiales (design : la liste tabulaire des questions de design est une vue complémentaire de la vue graphique) (efficience dans la navigation : les deux vues sont coordonnées)<br><br>**R5** : Editer une exigence, un dessin ou une question de design dans n'importe quel ordre et dans n'importe quelle vue (efficacité dans le nombre de questions de design exprimées) (efficience : pas d'ordre prédéfini pour supporter l'exploration et éviter l'engagement prématuré) | Conception exploratoire et collaborative sans outil spécifique |
| Afin de collaborer autour des questions de conception levées par un ensemble d'exigences système initiales, les outils d'ingénierie des exigences doivent permettre à l'ingénieur en exigences et aux fournisseurs de composants de :<br><br>**R6** : Partager la liste tabulaire des questions de conception et la vue comportementale dans une réunion co-localisée ou distribuée ;<br><br>**R7** : Discuter une question de conception dans n'importe quelle vue (graphique ou tabulaire) (efficience dans l'édition et l'accès aux discussions pour favoriser la traçabilité) ;<br><br>**R8** : Changer l'énoncé d'une exigence d'une façon visible et persistante (design : ETAIT +~~ancien énoncé~~ et EST+ nouvel énoncé) en lien avec une discussion (efficacité dans la traçabilité des changements) | Conception exploratoire et collaborative sans outil spécifique |
| Afin de raffiner le modèle d'un système, les outils doivent permettre à l'ingénieur en exigences de :<br><br>**R9** : Modifier les propriétés graphiques d'un ensemble d'objets avec un nombre minimum d'actions (efficience dans les actions d'édition pour favoriser la modélisation d'alternatives) (satisfaction de se concentrer sur les tâches de modélisation)<br><br>**R10** : Chercher et éditer les propriétés par mot clé à partir d'un élément graphique (efficience dans les actions de recherche pour préciser la modélisation) | Expérimentation d'une modélisation d'exigences avec outil spécifique |

| | |
|---|---|
| **R11** : Retrouver l'état d'avancement des propriétés à partir d'un élément graphique (efficience dans le filtrage d'information et l'évaluation progressive) | |
| Afin de rédiger des exigences cohérentes, les outils doivent permettre à l'ingénieur en exigences de :<br>**R12** : sélectionner et ajouter des termes du modèle de données pendant l'édition de l'énoncé de l'exigence (efficience dans les actions d'édition de l'exigence et du modèle de données) (efficacité : les mots sont cohérents avec le modèle de données)<br>**R13** : faciliter la mise en valeur des mots du modèle de données dans l'énoncé de l'exigence (efficience dans les actions d'édition)<br>**R14** : détecter les similitudes d'exigences *pendant* l'édition de l'énoncé d'une exigence (efficacité : pas de redondance dans les exigences) | Raffinement et gestion des exigences dans le référentiel |

Tableau 6:Exigences d'utilisabilité pour les outils d'ingénierie des exigences issues de l'analyse des scénarios

# Chapitre V. Visualisations interactives des exigences

Afin de comprendre les enjeux d'utilisabilité dans les activités de vérification des exigences, nous avons conçu et développé des prototypes de visualisations interactives des exigences, dans une démarche de triangulation entre théorie, design d'artefacts et observations (voir Figure 90 d'après Mackay et Fayard 1997). Dans ce chapitre, nous exposons d'abord les principes de conception des visualisations. Ensuite, nous présentons les résultats des design walkthroughs menés au sein de l'entreprise D et les exigences d'utilisabilité que nous en déduisons pour un support au processus d'ingénierie des exigences. Enfin, nous présentons des retours d'expérience des visualisations hors de l'entreprise D.

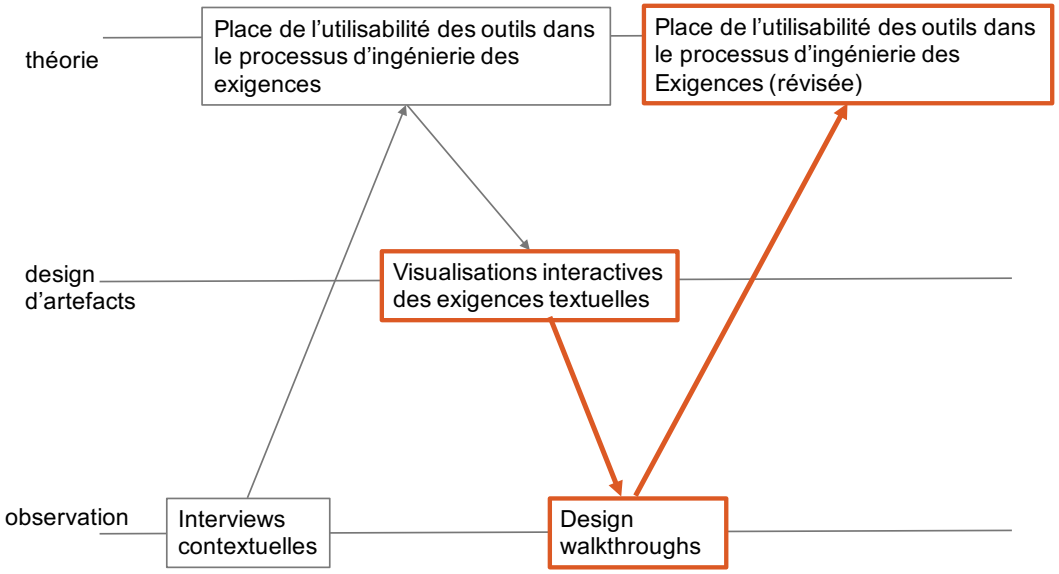



Figure 90: délimitation des résultats présentés dans ce chapitre

# V.1. Principes de conception des visualisations interactives

## V.1.1. Utilisation des structures et groupements construits par les ingénieurs

Lors de nos observations, nous avons collecté des données produites par les participants, afin de compléter notre analyse (nous les avons utilisées pour illustrer nos scénarios) mais également afin d'alimenter nos prototypes. Notre étude des données montre que les ingénieurs structurent les exigences textuelles avec des hiérarchies et des groupes qui ont un sens pour les différents acteurs impliqués dans le processus d'ingénierie des exigences : décomposition fonctionnelle pour les architectes, structuration par cas d'utilisation pour les utilisateurs finaux, décomposition par composants du système pour les fournisseurs de composants. L'ingénierie des exigences ne se réduit pas à de l'analyse visuelle de données : ce sont les ingénieurs qui construisent ces données, et les structurent selon l'avancement de la spécification. Les représentations structurées reflètent la spécification du système à venir.

Les travaux sur la visualisation en ingénierie des exigences se concentrent sur des algorithmes de reconstruction automatique de l'information (Duan et Cleland-Huang 2006) (Cleland-Huang et Habrat 2007)(Reddivari, Chen, et Niu 2012), et proposent la visualisation des résultats de ces algorithmes. Notre analyse des pratiques industrielles nous conduit à adopter une approche différente, qualifiée de *content-based structuring* par la littérature en visualisation d'information (Herman, Melançon, et Marshall 2000). Nous partons du principe d'exploiter les hiérarchies et groupement, pertinents par construction pour les utilisateurs, pour élaborer les visualisations. Les visualisations doivent d'abord refléter les structures construites par les utilisateurs, conformément au principe d'interprétabilité proposé pour la visualisation d'information de hiérarchies (Elmqvist et Fekete 2010). De façon similaire aux méthodes d'analyse structurée (partie Les différentes expressions des exigences), les ingénieurs construisent des structures via des numérotations, des appellations, et de l'organisation de l'information en tableau en utilisant des feuilles de calcul. Par exemple, quand un ingénieur numérote les exigences textuelles R1, puis R1.1, R1.2 et R1.3, cela signifie que l'exigence textuelle R1 est raffinée par les exigences textuelles R1.1, R1.2 et R1.3. Il a ensuite la possibilité d'affiner R1.2 en R1.2.1 et R1.2.2. Les avantages sont multiples : identifier un même niveau d'abstraction par le nombre de chiffres, ajouter une exigence dans un même niveau d'abstraction en incrémentant le dernier chiffre, affiner une exigence en ajoutant un chiffre. Ainsi, l'ajout d'exigences dans un même niveau d'abstraction et l'ajout d'un niveau d'abstraction supplémentaire sont des tâches faciles à réaliser : la structure des exigences est évolutive. De même, quand l'ingénieur indexe une exigence par UC_, cela signifie que cette exigence est un cas d'utilisation. S'il le nomme UC_pilot_take_off, cela veut dire que c'est un cas d'utilisation, concernant le pilote, pendant le décollage. Enfin, en ajoutant une colonne « Justification » dans une feuille de calcul contenant des exigences en lignes, le texte rédigé dans la cellule à l'intersection est une justification qui porte sur cette exigence : la présentation sous forme de tableau établit de facto le lien.

Les outils tels que Doors, en proposant une numérotation automatique, empêchent la construction d'une structure par la numérotation ou l'appellation, et rendent obligatoires des actions supplémentaires pour expliciter la structure via des liens. L'extrait présenté en Figure 91 est issu du forum de discussion de DOORS (https://goo.gl/SSFH13): il illustre la volonté d'un ingénieur d'ordonner et de structurer des exigences par des identifiants, que l'outil ne permet pas de satisfaire. L'animateur du forum qualifie cette volonté de *big mistake*, et demande à l'utilisateur de ne pas donner de signification à la numérotation.

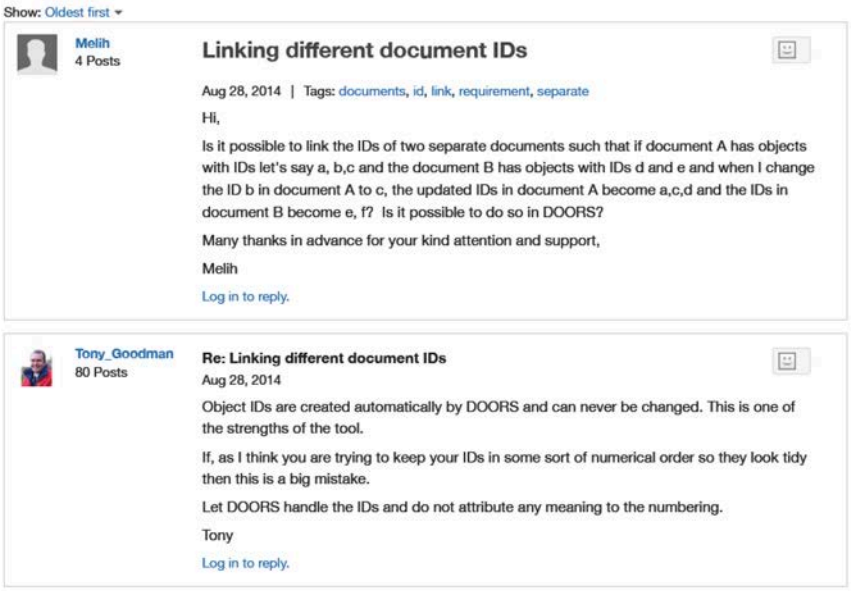



Figure 91: extrait du forum de discussion de DOORS

De même, une justification étant considérée comme un objet de la base de données, à saisir dans une boîte de dialogue, il devient nécessaire de renseigner explicitement un lien entre la justification et l'exigence à laquelle elle se rapporte. En cela l'édition des liens « de traçabilité » est en partie un problème accidentel (Brooks 1987), conséquence d'une vision d'informaticien orientée base de données obligeant les ingénieurs à saisir des liens entre différentes informations- objets d'une base de données, alors que ces liens sont contenus dans la numérotation, l'appellation et la présentation de l'information en tableau (même ligne, fusion de cellules, onglets différents). Notre conclusion est que tout est question de structuration : les ingénieurs en exigences font d'abord du *modelage* avant de faire de la *modélisation*, utilisant le texte, la numérotation et la présentation en tableau comme matières malléables permettant de *façonner* la structure du système de façon efficace. En ce sens, les feuilles de calcul et les éditeurs de texte représentent un excellent outil de travail : la numérotation peut être partiellement automatisée, et l'arrangement en tableau encode implicitement des relations entre éléments, disponibles pour du traitement automatique. Par conséquent, nous proposons de renverser la situation actuelle, en permettant aux ingénieurs d'utiliser la numérotation et l'appellation pour signifier un ordre et une structure dans les exigences : ils fixent les règles de nommage des exigences, et utilisent des outils à vocation générale pour organiser les exigences et les informations sur ces exigences.

## V.1.2. Exploration des techniques de visualisation 2D de graphes et de hiérarchies

Cependant, il reste le problème de la communication de la structure modelée : une feuille de calcul avec plusieurs onglets et des milliers de ligne n'est pas utilisable pour avoir une vue d'ensemble et présenter l'avancement du travail aux parties prenantes. En reprenant la métaphore du modelage, le potier a un avantage sur l'ingénieur : il produit une forme tangible et visible, qu'il peut présenter à son client et retoucher selon ses retours. Pour remédier à ce problème, nous proposons d'intégrer les règles de nommage dans un parser qui traite les données pour en déduire des structures des exigences, à partir desquelles nous proposons des visualisations (voir Figure 92). Les bases de données peuvent également être alimentées de cette façon, transparente pour les ingénieurs.

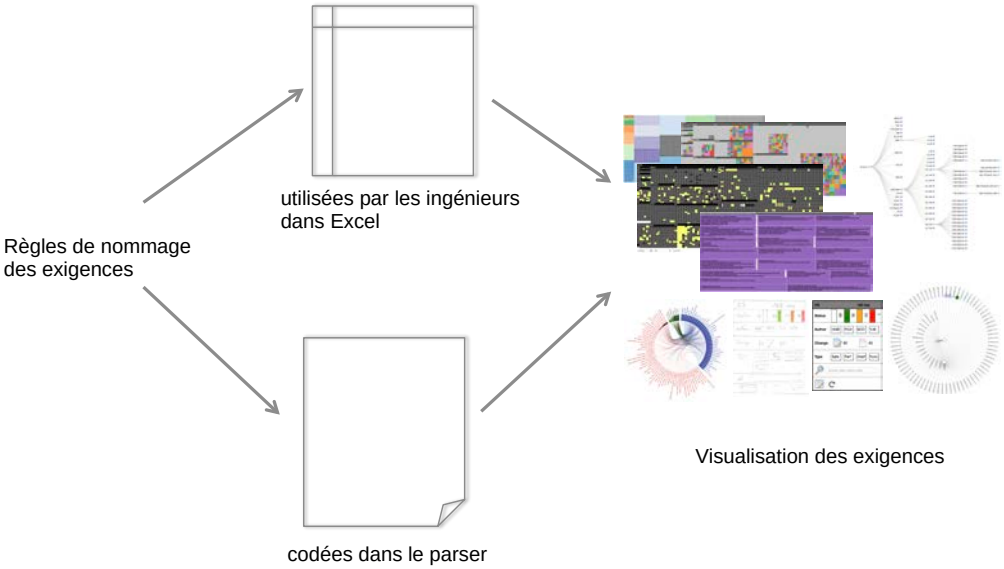



Figure 92: Excel comme langage pivot pour la visualisation des exigences

Les structures de données hiérarchisées peuvent être représentées par des arbres, qui sont un type spécial de graphes : l'enjeu de la visualisation de graphes est la taille du graphe, posant des problèmes d'encombrement visuel et d'utilisabilité de l'interaction. Sur la base de revues de Herman et al. (Herman, Melançon, et Marshall 2000) et de Elmqvist et Fekete (Elmqvist et Fekete 2010), nous explorons les différentes techniques de visualisation de graphes et d'agrégation de hiérarchies (voir Figure 93), associées aux techniques d'interaction de navigation et de manipulation de hiérarchies, en revisitant les problèmes issus des scénarios du contexte d'utilisation « Raffinement et Gestion des exigences dans le référentiel ». De plus, nous utilisons la taxonomie des tâches interactives pour la visualisation d'information proposée par Heer et Shneirdeman (Heer et Shneiderman 2012) (voir Figure 69).

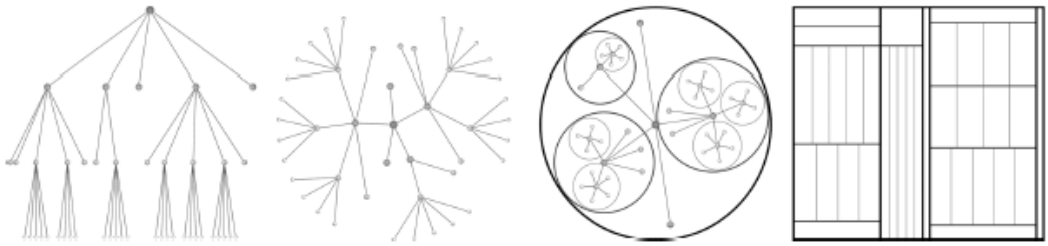


Figure 93: les techniques de visualisation de données hiérarchiques : arbre, arbre radiale, bulle, treemap

Nous n'avons délibérément pas exploré les visualisations en trois dimensions, alors qu'il existe des techniques de visualisation d'arbres en 3D, telles que ConeTree (G. G. Robertson, Mackinlay, et Card 1991) ou Information Cube (Rekimoto et Green 1993) (voir Figure 94). En effet, les techniques de visualisation 3D impliquent des problèmes d'occlusion de l'information, comme rapportés par Savio et al. (Savio et al. 2012) avec la pyramide ReBlock (Figure 59). Ces problèmes restent partiellement résolus par la semi-transparence de l'information et la rotation interactive multiaxe de la vue 3D, et sont accompagnés de nouveaux problèmes d'utilisabilité de l'interaction (Herman, Melançon, et Marshall 2000) (B. Shneiderman 2003). Les nouveaux dispositifs d'affichage à tête haute pourraient offrir de nouvelles façons d'interagir avec les données (Cordeil et al. 2017), mais ne sont pas explorés dans cette thèse.

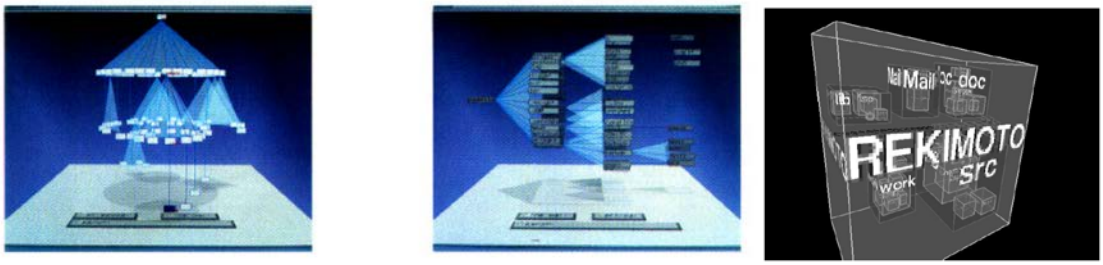



Figure 94: visualisation 3D de hiérarchies : Cone Tree à gauche (G. G. Robertson, Mackinlay, et Card 1991) et Information Cube à droite (Rekimoto et Green 1993)

Enfin, l'encombrement visuel d'un diagramme de réseau lié à la congestion des arcs peut être résolu en utilisant une matrice d'adjacences pour représenter le réseau (Bertin 1983), comme illustré en Figure 95. Ghoniem et al. (Ghoniem, Fekete, et Castagliola 2005) ont montré que les matrices d'adjacence sont particulièrement efficaces pour les graphes denses, sauf pour la recherche de chemin. Cependant, les données d'exigences, du fait de leur structure hiérarchique, forment plutôt des graphes clairsemés, de faible densité, pour lesquels les matrices d'adjacence ne sont pas adaptées. Ainsi, la matrice VisMatrix (Duan et Cleland-Huang 2006) est constituée de centaines de colonnes et de lignes représentant les artefacts d'ingénierie, mais ne restitue pas la hiérarchie des artefacts et occupe beaucoup d'espace pour peu de liens.

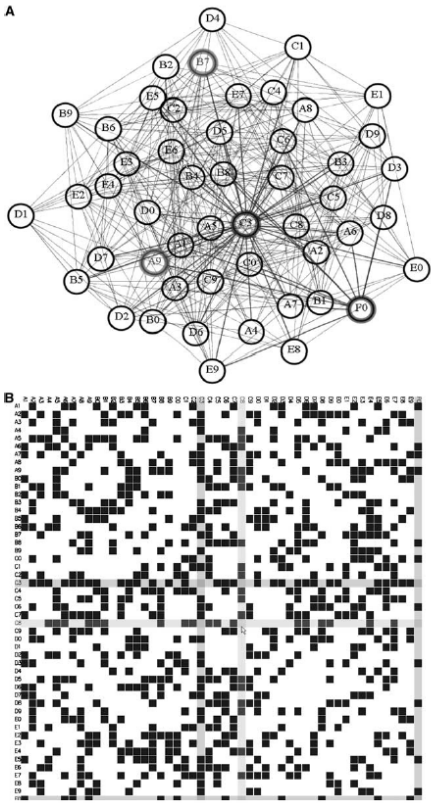



Figure 95: deux visualisations du même graphe (50 noeuds, 400 arcs): diagramme de réseau et matrice (Ghoniem, Fekete, et Castagliola 2005)

Pour répondre à ce problème, Henry et Fekete ont fait plusieurs propositions de combinaison de visualisations de matrice et de diagramme avec MatrixExplorer (N. Henry et Fekete 2006) MatLink (Nathalie Henry et Fekete 2007) et NodeTrix (N. Henry, Fekete, et McGuffin 2007) (voir Figure 96). Cependant, ces propositions sont conçues et évaluées pour des tâches d'analyse de réseaux sociaux qui sont différentes des tâches des ingénieurs en exigences que nous avons observées. Nous retenons de MatrixExplorer l'intérêt de visualisations synchronisées et du filtrage interactif pour les utilisateurs.

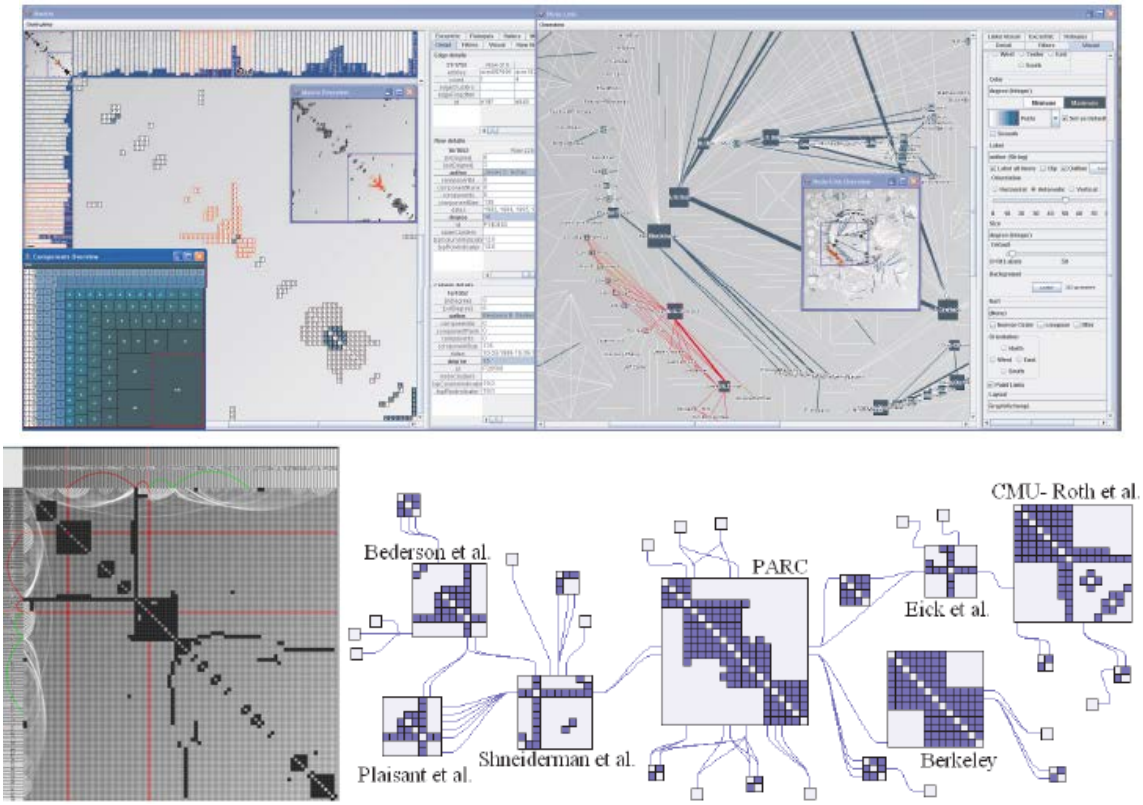



Figure 96: MatrixExplorer, MatLink et NodeTrix combinent matrice d'adjacence et diagramme pour l'analyse de réseaux sociaux

# V.2. Visualisations interactives et design walkthrough

Nous présentons les différentes visualisations interactives des exigences et les discussions menées avec les participants de l'entreprise D (voir Tableau 4) à partir de ces visualisations. Nous proposons une analyse des discussions, en cherchant à comprendre les raisons (le pourquoi) des propositions et opinions exprimées par les participants. A partir de cette analyse, nous tirons des conclusions, que nous numérotons pour en assurer le suivi, et que nous traitons à travers de nouvelles propositions de design, jusqu'à l'obtention de nouvelles exigences d'utilisabilité.

## V.2.1. Arbre dépliable des exigences

Le premier prototype développé présente les exigences sous la forme d'un arbre dépliable (voir Figure 97), inspiré de SpaceTree (Plaisant, Grosjean, et Bederson 2002) : de la gauche vers la droite, le premier niveau est le système, le deuxième niveau est la catégorie fonctionnelle de l'exigence (par exemple ARCH pour Architecture, MNG pour Management). La sélection d'une catégorie affiche les exigences de cette catégorie avec une animation. Cependant, le grand nombre d'exigences dans une

même catégorie, jusqu'à 1360, pose un problème d'affichage, que nous avons contourné en groupant les exigences vingt par vingt. L'utilisation de cette visualisation a été évaluée comme intéressante par les participants, afin de visualiser la structure des exigences mais aussi pour ajouter une nouvelle catégorie dans la structure. Un des participants utilise l'outil XMind (http://www.xmind.net/) qui est un outil de *mind mapping* et offre des visualisations similaires : il a produit une carte montrant la structure des exigences et l'utilise dans des présentations faites auprès des utilisateurs finaux, pour avoir une vue d'ensemble.

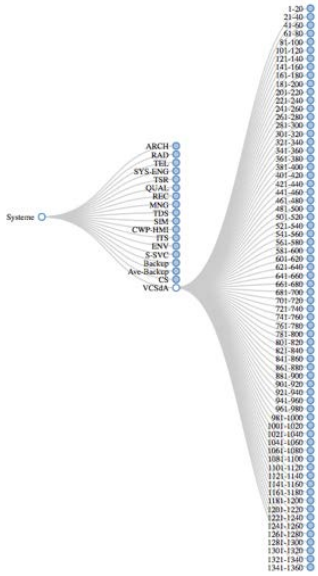



Figure 97: visualisation de la structure des exigences avec un arbre dépliable (collapsible tree)

Inversement, le groupement par nombre de vingt est évalué comme non pertinent, l'identifiant numérique de l'exigence n'ayant pas ici de signification. Dans un contexte de vérification des exigences par analyse de couverture, les participants ont évalué comme fastidieux les interactions successives pour afficher le texte de l'exigence raffinée (voir Figure 98). Nous concluons que (C1) l'arbre dépliable est approprié pour communiquer sur la structure des exigences (C2) il est nécessaire d'explorer d'autres visualisations pour afficher une grande quantité d'exigences (C3) il est nécessaire de travailler sur les interactions pour mieux supporter le scénario d'analyse de couverture (C4) il existe un intérêt de modifier la structure via la visualisation.

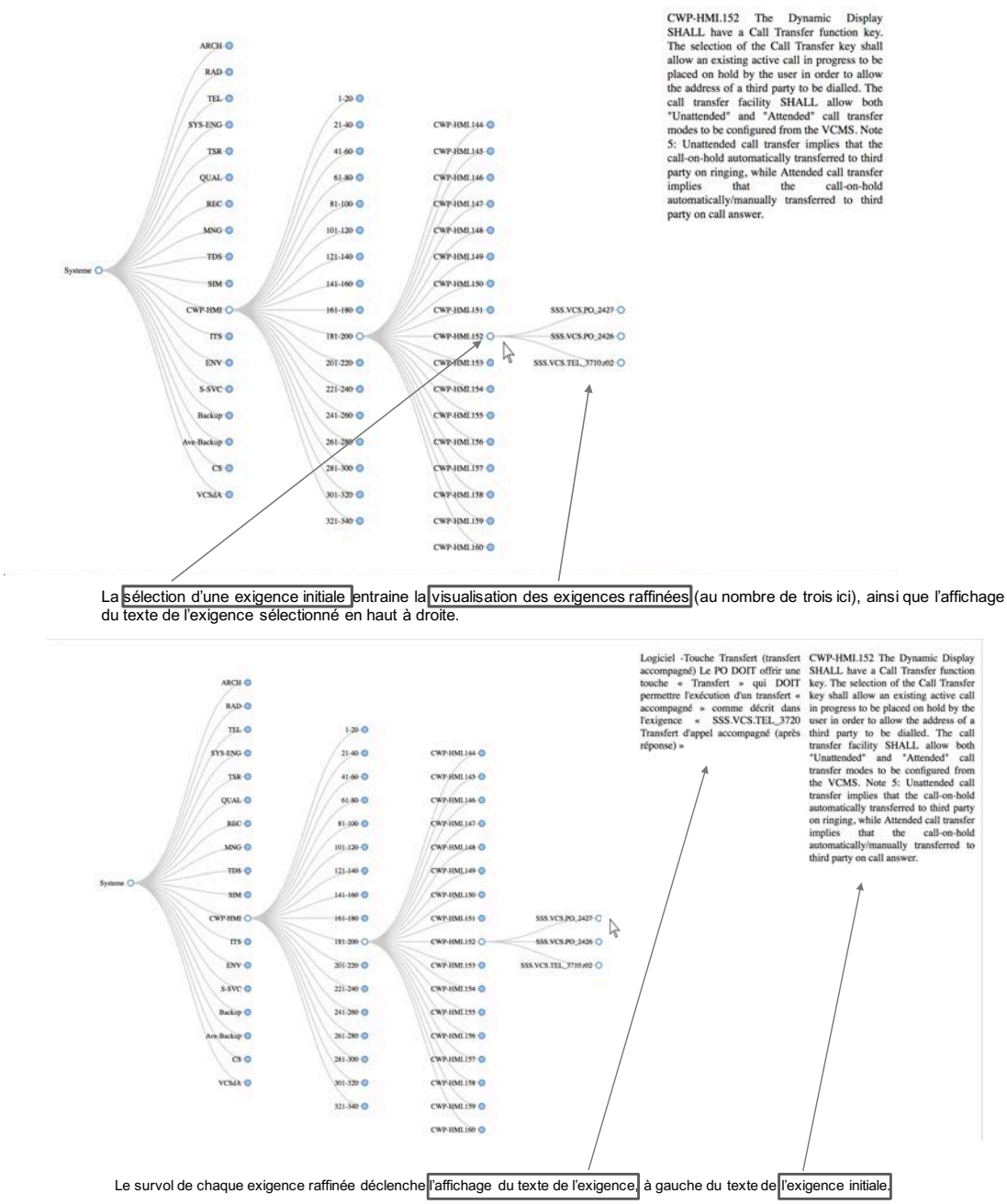



Figure 98: affichage des exigences raffinées par survol successif d'un arbre dépliable

## V.2.2. Vue radiale des exigences

Le deuxième prototype que nous avons testé pour afficher un plus grand nombre d'exigences (C2) est une vue radiale en grappe (*rotating cluster layout*) présentée en Figure 99 : le système est au centre de la visualisation et les catégories d'exigences sont organisées radialement (à gauche de la Figure 99). La sélection d'une catégorie affiche les exigences de cette catégorie, arrangées radialement de la même façon (à droite de la Figure 99). Un plus grand nombre d'exigences peut être affiché en utilisant cette technique de visualisation. Cependant, des problèmes d'interaction ont été relevés lors des design walkthroughs, mettant en exergue de nouvelles directions de design.

En effet, quand l'utilisateur veut fermer la catégorie sélectionnée et en afficher une autre, le grand nombre d'exigences rend la catégorie difficile à pointer, du fait de la très petite taille du cercle, réduit pour laisser la place aux exigences. Les participants ont trouvé cela « dérangeant », tout en notant que cela pouvait être évité : ils n'ont pas besoin de voir les autres catégories quand ils affichent les exigences d'une catégorie spécifique. Cependant, à notre question sur l'utilité de développer deux catégories en même temps, ils ont évalué la possibilité intéressante afin de lire deux exigences réparties dans deux catégories différentes mais convoquées séquentiellement dans le fonctionnement opérationnel du système. Cela arrive dans une tâche de vérification des exigences avec une approche orientée scénarios. Les participants ont rapporté des difficultés dans la lecture du texte, dues à son positionnement incliné et au scanning visuel coûteux pour trouver une catégorie. Une boite de recherche de ce type  a été proposée par les participants pour se passer du scanning visuel et trouver l'information plus efficacement, pas seulement sur la catégorie de l'exigence, mais aussi son énoncé ou son identifiant. Cette discussion est cohérente avec les conclusions de l'étude sur le *SpaceTree* (Plaisant, Grosjean, et Bederson 2002), qui préconisent de mettre en valeur les résultats d'une recherche sans déclencher de déploiement automatique de l'arbre pour éviter des animation constantes et non contrôlées. Finalement, nous avons proposé d'enrichir les données initiales avec un état d'avancement codé par la taille du cercle (voir à gauche de la Figure 99) : les participants ont confirmé l'intérêt d'avoir un retour d'information sur la progression du travail, mais souhaitaient pouvoir mettre à jour l'état de l'exigence de façon interactive depuis la visualisation.

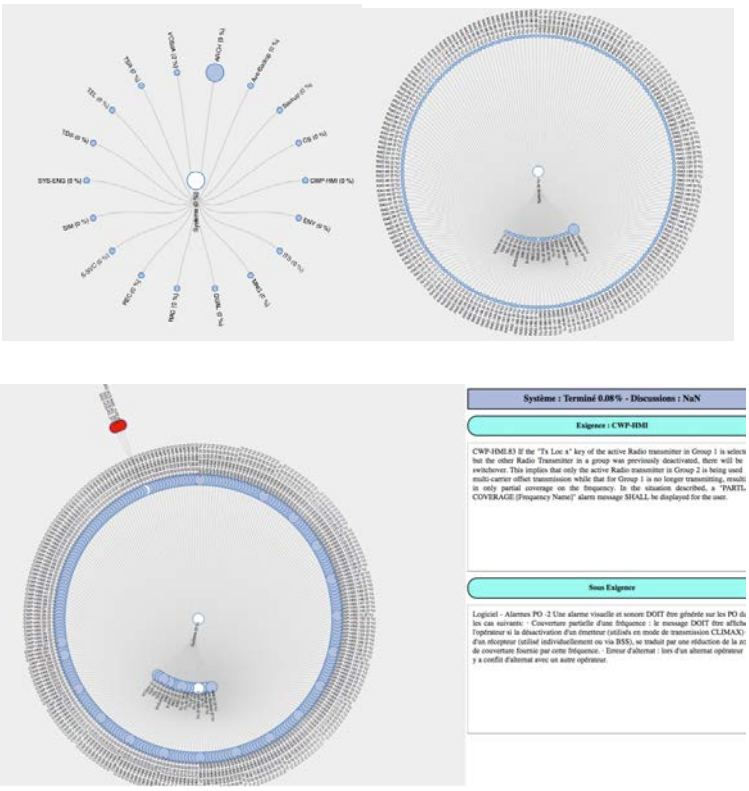



Figure 99: visualisation des exigences en utilisant une vue radiale

Nous concluons que (C5) l'affichage rotatif en grappe n'est pas utilisable dans un contexte d'ingénierie des exigences : d'autres visualisations doivent être explorées pour afficher un grand nombre d'exigences (C6) il existe un besoin de visualiser des liens de type scénario entre exigences de différentes catégories (C7) il existe un besoin sur la recherche d'information par mots clé (C8) il

existe un besoin de mette à jour l'état d'avancement d'une exigence et de visualiser l'état d'avancement global.

## V.2.3. Treemap des exigences

Suite aux retours obtenus (conclusions C2 et C5) sur les deux visualisations que nous venons de présenter, nous avons développé plusieurs prototypes autour de visualisation de type treemap (Ben Shneiderman 1992), et plus particulièrement le treemap zoomable (Blanch et Lecolinet 2007) pour travailler sur la navigation et les interactions (conclusion C3). La technique du treemap est conçue pour afficher des hiérarchies en utilisant au mieux l'espace de l'écran. Dans la Figure 100, le système en cours de spécification représente la surface globale, subdivisée en rectangles représentant les catégories des exigences. L'espace alloué à chaque catégorie d'exigences dépend du nombre d'exigences. La catégorie des exigences est codée avec de la couleur et les exigences qui n'ont pas été raffinées sont codées en noir, montrant « le trou » dans la spécification. Les exigences raffinées d'une même exigence système forment un groupe sur le treemap et l'identifiant de l'exigence est affiché dans un tooltip au survol du groupe (au centre de la Figure 100).

Figure 100: une visualisation des exigences en treemap

Les utilisateurs peuvent naviguer dans la hiérarchie en sélectionnant la barre de titre pour descendre d'un niveau, ou en sélectionnant directement le groupe d'exigences pour descendre directement de deux niveaux et atteindre le texte des exigences (voir Figure 101). L'implémentation du treemap avec la librairie d3 (Bostock, Ogievetsky, et Heer 2011) intègre une animation au déclenchement de la navigation, permettant à l'utilisateur de conserver son orientation spatiale (Blanch et Lecolinet 2007). Cependant, la grande quantité de données à afficher entraine des problèmes de performance dans les transitions animées.

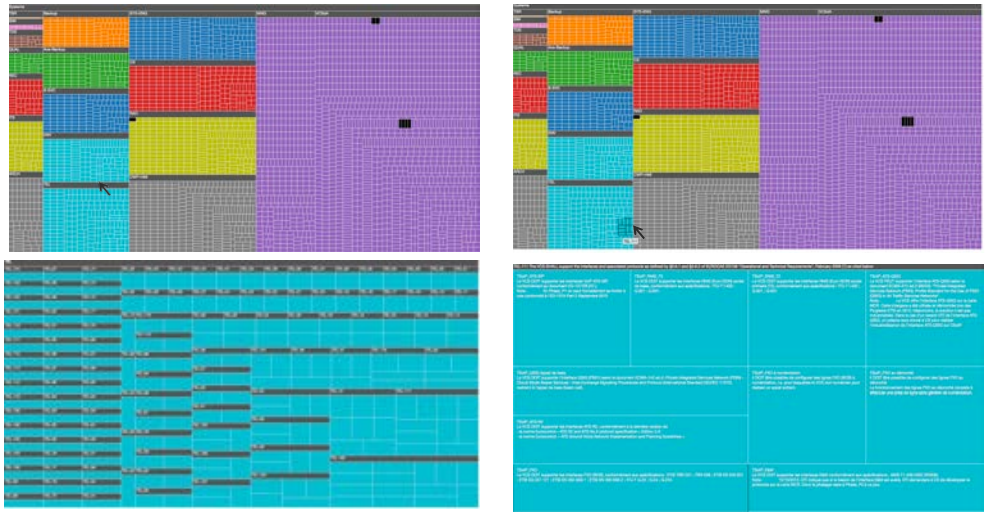


Figure 101: navigation dans la hiérarchie par sélection d'une catégorie (à gauche) ou d'un groupe (à droite)

Enfin, nous proposons aux utilisateurs des actions (valider, rejeter, modifier, commentaire) par clic droit sur une exigence afin d'offrir un support à la tâche de vérification des exigences (voir Figure 102) (conclusion C8).


Figure 102: actions disponibles sur une exigence du treemap

Nous avons présenté le treemap auprès des participants comme une visualisation partagée entre les différents ingénieurs, afin de favoriser la collaboration. Les participants sont enthousiastes vis-à-vis de cette visualisation et de la navigation, et nous avons analysé avec eux les raisons : elle leur permet d'avoir une vue d'ensemble des exigences [R15], de se concentrer sur un niveau d'abstraction [R16], puis de lire une exigence système et les exigences raffinées *en même temps* pour une vérification de conformité [R17]. Dans notre implémentation, les exigences ont toutes la même surface (égale à 1). Cependant, il est possible de donner une valeur différente à la surface. Les différents attributs d'une exigence (voir partie Les différentes expressions des exigences) sont potentiellement éligibles :

- la priorité de l'exigence pour les utilisateurs finaux => plus l'exigence est importante aux yeux des utilisateurs, plus elle prend de la place visuellement;
- le coût de l'exigence => plus l'exigence est chère, plus elle prend de la place, et les services qui coûtent le plus chers prennent plus de place.
- la criticité en termes de sécurité => plus l'exigence est critique, plus elle prend de la place.
- le côté innovant ou création de valeur => plus l'exigence est nouvelle, plus elle prend de la place.

En suivant les principes de flexibilité (partie Utilisabilité et Flexibilité), il serait intéressant de laisser l'utilisateur choisir parmi les attributs de l'exigence celui qu'il affecte à la surface d'une exigence. Les actions disponibles par clic droit sur les exigences ont été évaluées comme pratiques pour la vérification et le raffinement des exigences : "c'est vraiment un environnement de travail" [R18]. Les discussions sur les actions disponibles par clic droit ont confirmé le besoin d'avoir accès au modèle

de données pour vérifier ou corriger un terme [exigence déjà exprimée par R12**].** Cependant, les participants préfèreraient utiliser le code couleur « vert-orange-rouge » pour discriminer les exigences, respectivement « vérifiée-modifiée/commentée-rejetée » sur la vue globale, afin de pouvoir suivre l'avancement du processus de vérification des exigences [R19]. Les suggestions des utilisateurs nous permettent plus d'identifier les informations intéressantes à coder que la façon de les coder. En effet, la couleur est une variable visuelle parmi les six possibles (taille, forme, texture, couleur, luminosité, orientation) que nous pouvons utiliser pour coder de l'information (Bertin 1983). Une étude portant sur la perception graphique des treemaps (Kong, Heer, et Agrawala 2010) montre que l'épaisseur des bords (qui agit sur la luminosité) permet de coder efficacement la hiérarchie de l'arbre. Cette alternative à la couleur permet d'utiliser cette dernière pour coder d'autres informations qui restent à explorer. Par exemple, dans une version du prototype, nous utilisons la couleur pour mettre en relief les résultats d'une recherche sur mot clé : les exigences contenant le mot « radio », renseigné par l'utilisateur dans la boite de recherche en bas de l'écran, sont coloriées en jaune sur le treemap (voir Figure 103).

L'élément du « trou noir » a été apprécié des participants pour pouvoir détecter une anomalie dans la spécification en un coup d'œil sur la vue globale, en comparaison avec la lecture fastidieuse de tableaux [R20]. Cependant, la prochaine étape pour les participants est de traiter ces anomalies, ce qui est cohérent avec le mantra de la visualisation d'information « *overview first, zoom and filter, then details on-demand* » (B. Shneiderman 1996).

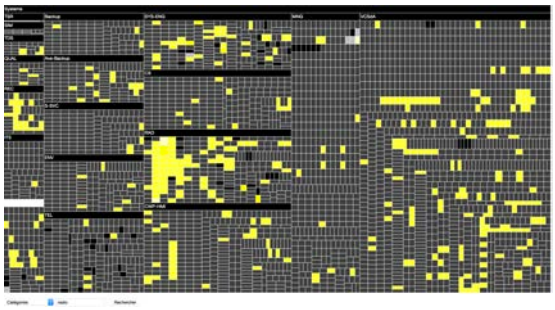


Figure 103: utilisation de la couleur pour mettre en relief les résultats d'une recherche sur le treemap

Ils ont émis le même souhait sur les exigences modifiées, commentées ou rejetées, les considérant aussi comme des problèmes à traiter. Ces retours montrent l'intérêt d'un filtrage dynamique avec retour d'information immédiat sur la visualisation des exigences [R21]. Les participants nous rapportent que les filtres proposés dans DOORS sont prédéfinis et accessibles à partir d'une liste déroulante : le côté statique ne permet pas aux utilisateurs de trouver des propriétés globales ou de répondre à des questions spécifiques (B. Shneiderman 1996). C'est pour cela qu'il peut être plus rapide pour un utilisateur d'exporter les données dans un format .csv et utiliser les filtres dans une feuille de calcul.

En résumé, afin de conduire l'analyse de couverture, l'outil doit permettre à l'ingénieur en exigences de :

**R15** : Visualiser une vue d'ensemble des exigences du système sans action (efficience dans la compréhension de la vue) (treemap comme exemple de design) ;

**R16** : Accéder aux exigences d'un niveau d'abstraction donné avec un minimum d'actions à partir de la vue d'ensemble des exigences (efficience dans la navigation) (treemap comme exemple de design) ;

**R17** : Lire l'exigence initiale et les exigences raffinées simultanément (efficience dans la lecture) (treemap comme exemple de design) ;

**R18** : Accéder à des actions sur l'exigence à partir de la visualisation : valider, rejeter, commenter, modifier (efficience dans la vérification) (treemap avec les actions en clic droit comme exemple de design d'interaction) ;

**R19** : Voir l'état d'avancement des exigences (validée/modifiée-commentée/rejetée) sur la vue globale des exigences (évaluation progressive forte) ;

**R20** : Identifier les exigences non raffinées d'un coup d'œil à partir de la vue d'ensemble des exigences (efficience dans la détection d'anomalies) (rectangle noir sur le treemap comme exemple de design) ;

**R21** : Filtrer les exigences de façon dynamique à partir de la vue globale des exigences avec retour d'information immédiat sur la vue globale (efficience dans la détection d'anomalies) (utilisation de la couleur sur le treemap comme exemple de design).

## V.2.4. Filtrage dynamique interactif et traitement des résultats

Afin de conduire des discussions plus approfondies sur le filtrage dynamique et le traitement des résultats, nous avons conçu des prototypes papier et logiciel d'un inspecteur d'exigences (voir Figure 104), en s'inspirant de l'inspecteur interactif d'objets multiples ManySpector (Hoarau et Conversy 2012).

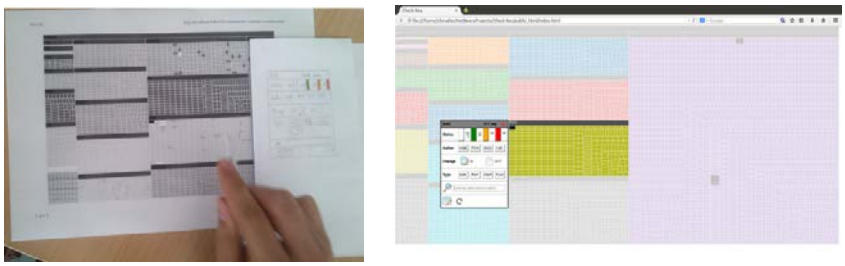


Figure 104: prototypes papier et logiciel du filtrage interactif

L'inspecteur peut être déclenché à n'importe quel niveau d'abstraction (système, catégorie, exigence système) et montre toutes les valeurs des attributs des exigences du niveau sélectionné : état, auteur, changement, type. La sélection d'une valeur permet de filtrer les exigences ayant cette valeur. Par exemple, dans la Figure 105, la sélection de HGB comme auteur permet de filtrer sur les exigences rédigées par HGB et de les mettre en valeur sur le treemap. Une liste interactive des exigences est obtenue en cliquant sur l'icône  en bas à gauche de l'inspecteur (à gauche de la Figure 106). La sélection d'une exigence de la liste entraine l'ouverture du treemap sur cette exigence (à droite de la Figure 106).

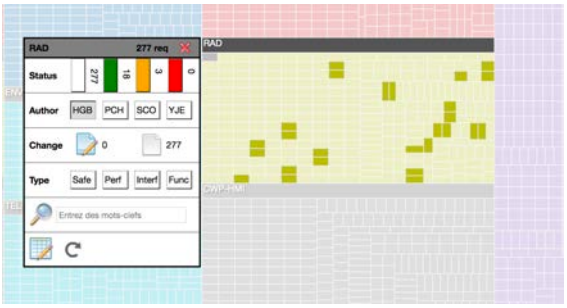



Figure 105: inspecteur sur la catégorie RAD


Figure 106: la liste des exigences filtrées (à gauche) permet de naviguer d'exigence en exigence dans le treemap (à droite)

Nous avons intégré la boite de recherche dans l'inspecteur, en tant qu'outil de filtrage par mot-clé, dans l'idée de supporter la vérification de cohérence. Par exemple, si l'utilisateur entre « timeline », toutes les exigences contenant ce mot sont mises en valeur sur le treemap et peuvent être examinées par l'utilisateur pour réaliser la comparaison et détecter les similitudes. Les filtres fonctionnent avec des OU pour un même attribut (sinon ce serait systématiquement un ensemble vide) et avec des ET pour des attributs différents : sur la Figure 107, les exigences contenant le mot « radio » sont mises en évidence sur le treemap. La sélection de HGB permet de mettre en évidence, parmi celles-ci, les exigences rédigées par HGB. Ce filtrage permet de répondre à la situation : « je me souviens que j'ai rédigées la semaine dernière des exigences sur la radio et je veux les retrouver pour continuer à travailler dessus, suite à des informations que j'ai récupérées ».

Initialement, les filtres proposés sont :

- l'auteur de l'exigence afin de répondre au scénario Rédaction d'exigences cohérentes pour supporter la vérification croisée d'exigences ;
- l'état de l'exigence (vérifiée, commentée/modifiée/rejetée) pour confirmer l'exigence R19 relative à l'évaluation de l'état d'avancement du travail de vérification.


Figure 107: mise en évidence des exigences contenant le mot "radio' (à gauche), puis application du filtre HGB (à droite)

A travers les discussions avec les participants, il apparaît que la recherche par mot-clé est intéressante pour vérifier la cohérence des exigences [**R21_1** ], mais que le traitement des exigences filtrées serait plus intéressante à faire dans une visualisation tabulaire directement, plutôt que de naviguer d'exigence en exigence dans le treemap. La navigation dans le treemap à partir de la liste entraîne une perte de contexte et ne facilite pas la comparaison entre exigences [**R22**]. De plus, les participants soulignent que le filtrage par mot clé est également intéressant pour conduire une analyse d'impact suite à une demande d'évolution : en faisant des recherches successives sur des mots pertinents par rapport à la demande, il est possible d'identifier les exigences devant être modifiés pour la réalisation de l'évolution [**R21_2** ]. L'intérêt de filtrer par auteur est confirmé par les participants afin de supporter la vérification croisée des exigences [**R21_3** ]. D'autres filtres ont été identifiés à travers les discussions : le type d'exigence (safety, performance, interface, fonction) est pertinent pour pouvoir mettre des priorités dans le processus de vérification des exigences. Par exemple, les ingénieurs vont vérifier les exigences critiques (safety) en premier, et être certains qu'il n'y a plus de commentaires dessus [**R21_4** ]. Cependant, nous ne prétendons pas fournir une liste exhaustive des filtres. La liste présentée ici couvre les filtres que nous avons discutés avec les participants autour de l'analyse de couverture et l'analyse d'impact.

En résumé, nous consolidons l'exigence R19 sur la visualisation de l'état d'avancement du travail sur les exigences : cette exigence est cruciale afin de supporter le raisonnement désordonné et non linéaire des ingénieurs tout en évitant les oublis (R. Guindon et Curtis 1988) et en étant capables de produire un état d'avancement vers le management (Nguyen, Carroll, et Swatman 2000).

**R19 :** Afin d'évaluer l'avancement du travail, l'outil doit permettre à l'ingénieur de filtrer dynamiquement les exigences par état d'avancement à partir de n'importe quel niveau d'abstraction avec retour d'information immédiat sur la vue d'ensemble (efficience dans le filtrage) (inspecteur et treemap comme exemples de design).

De plus, nous raffinons l'exigence R21 sur le filtrage dynamique avec les exigences suivantes :

**R21_1** : Afin de réaliser une vérification de cohérence, l'outil doit permettre à l'ingénieur de filtrer dynamiquement les exigences par mots clés à partir de n'importe quel niveau d'abstraction avec retour d'information immédiat sur la vue d'ensemble (efficience dans le filtrage) (inspecteur et treemap comme exemples de design) ;

**R21_2** : Afin de réaliser une analyse d'impact, l'outil doit permettre à l'ingénieur de filtrer dynamiquement les exigences par mots clés à partir de n'importe quel niveau d'abstraction avec retour d'information immédiat sur la vue d'ensemble (efficience dans le filtrage) (inspecteur et treemap comme exemples de design) ;

**R21_3** : Afin de réaliser une vérification croisée des exigences, l'outil doit permettre à l'ingénieur de filtrer dynamiquement les exigences par auteur à partir de n'importe quel niveau d'abstraction avec retour d'information immédiat sur la vue d'ensemble (efficience dans le filtrage) (inspecteur et treemap comme exemples de design) ;

**R21_4** : Afin de prioriser le processus de vérification de cohérence, l'outil doit permettre à l'ingénieur de filtrer dynamiquement les exigences par type (safety, performance, fonctionnelle, interface) à partir de n'importe quel niveau d'abstraction avec retour d'information immédiat sur la vue d'ensemble (efficience dans le filtrage) (inspecteur et treemap comme exemples de design).

Enfin, nous formulons l'exigence R22 sur la visualisation du résultat du filtrage, dans laquelle nous privilégions l'absence de mode (Tesler 2012) avec un traitement direct des exigences filtrées.

**R22** : Afin de traiter les exigences filtrées, l'outil doit permettre à l'ingénieur en exigences d'afficher les exigences issues d'un filtrage dynamique dans une visualisation tabulaire éditable (efficience dans le traitement des anomalies).

### V.2.5. Visualisation de liens entre artefacts d'ingénierie avec le *chord diagram*

Nous avons exploré la visualisation du *chord diagram* (D. Holten 2006)(DHR Holten 2009) comme visualisation alternative des matrices : les éléments sont arrangés radialement autour d'un cercle, et les relations entre les éléments sont représentées sous forme d'arcs connectant les données. La technique de *bundling* permet de diminuer l'encombrement visuel par une agrégation des arcs au centre de la visualisation. La sélection d'un élément met en évidence les arcs et les autres éléments auxquels il est connecté. Ainsi, en reprenant les données de la matrice de la Figure 85, les évolutions système sont représentées en rouge, les composants du système en bleu et les sites opérationnels en vert (voir Figure 108).

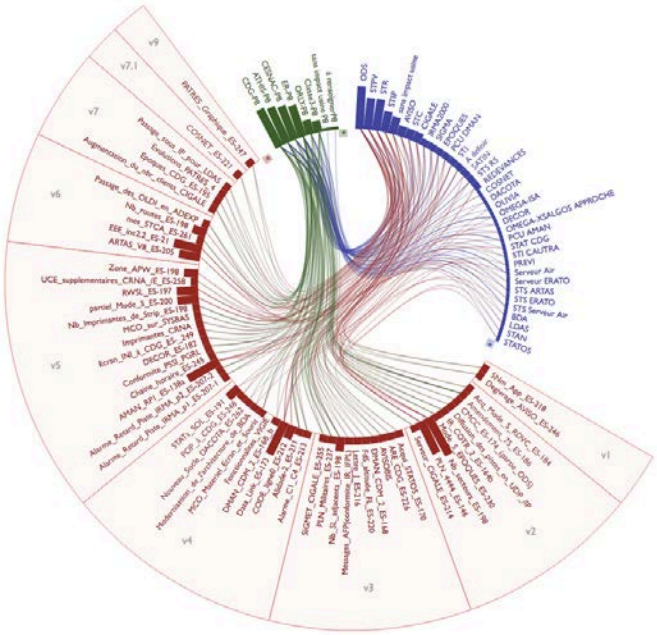


Figure 108: la visualisation de chord diagram pour voir les composants système (en bleu) implémentant les évolutions (en rouge) selon les sites opérationnels (en vert)

La sélection d'une évolution met en valeur les liens menant vers les composants du système qui l'implémentent et les sites opérationnels impactés (Figure 109 - a). De façon similaire, la sélection d'un site met en valeur les liens menant vers les évolutions que le site va connaître ainsi que les composants impactés (Figure 109 - b). Enfin, la sélection d'un composant permet d'identifier les évolutions impactant le composant et les sites concernés (Figure 109 - c).

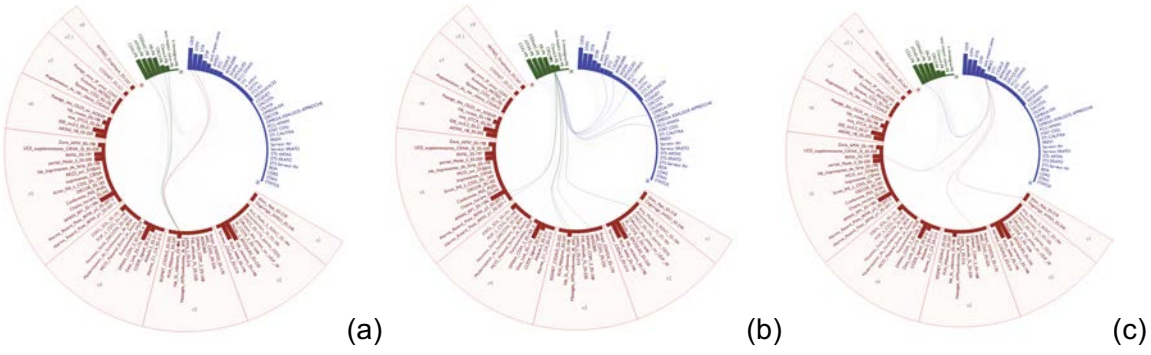



Figure 109: la sélection d'un élément d'une catégorie permet de mettre en valeur les liens menant vers les éléments impactés des autres catégories

Les participants ont évalué la visualisation intéressante dans le contexte de communication avec un comité de pilotage du système [R23] : ils ont proposé d'ajouter de l'information à la sélection d'une évolution, pour un meilleur support d'aide à la décision, telle que des statistiques sur les exigences constituant une évolution (nombre de nouvelles exigences, nombre d'exigences modifiées, nombre d'exigences inchangées). Ils ont suggéré que cela pouvait améliorer l'analyse d'impact d'une nouvelle version du système, en naviguant d'une évolution à l'autre [R24]. Les participants ont également envisagé d'utiliser la visualisation comme un outil de communication au sein de l'équipe d'ingénierie des exigences, afin d'avoir une meilleure conscience de la situation sur les évolutions en cours d'étude. En effet, le nombre d'évolutions à l'étude peut être élevé, et les ingénieurs n'en ont pas

forcément une vision d'ensemble. Cette visualisation a d'ailleurs été livrée aux participants sur leur demande à des fins d'intégration dans leur intranet [R25].

En résumé, afin de gérer les versions système de façon flexible, l'outil doit permettre à l'ingénieur en exigences et au comité de pilotage de :

**R23** : visualiser les évolutions à l'étude et les composants impactés correspondant à chaque évolution (efficience dans la sélection et la lecture) (*chord diagram* comme exemple de design) ;

**R24** : naviguer d'une évolution à l'autre avec des statistiques sur les exigences (nouvelles, modifiées, inchangées) (efficience dans l'aide à la décision) (*chord diagram* comme exemple de design d'interaction) ;

**R25** : Afin d'avoir une conscience de la situation des évolutions à l'étude, l'outil doit permettre à l'équipe d'ingénieurs en exigences de partager une visualisation des évolutions et des composants impactés (efficacité dans la cognition de l'équipe) (*chord diagram* comme exemple de design).

Plus largement, les participants étaient prêts à utiliser la visualisation comme un outil de filtrage, leur permettant par exemple d'extraire les exigences relatives à une évolution et classées par composants du système, « par drag'n drop de la sélection dans une fenêtre à part ». Cette remarque est à nouveau cohérente avec le mantra *overview first, zoom and filter, then details on-demand* (B. Shneiderman 1996) et le principe de vues multiples coordonnées (Heer et Shneiderman 2012). Enfin, un des participants a imaginé l'utilisation de la visualisation comme un disque tournant, permettant de « dérouler les tests ». Dans ce cas, la visualisation permettrait de visualiser les cas de tests d'un côté, et les exigences de l'autre. L'ensemble de ces discussions révèle l'intérêt de la visualisation de *chord diagram* pour montrer une vue globale des liens entre différents artefacts d'ingénierie, positionnés radialement, dans un contexte de communication avec une partie prenante. Les matrices sont en effet difficiles à comprendre par des utilisateurs novices (Abello, Ham, et Krishnan 2006) (Ghoniem, Fekete, et Castagliola 2005). Cela nous permet de généraliser la proposition de Merten (Merten, Jüppner, et Delater 2011) qui utilise une visualisation similaire pour voir les liens entre cas d'utilisation, profils utilisateurs et buts [R26].

**R26** : Afin de communiquer avec les différentes parties prenantes, l'outil doit permettre à l'ingénieur en exigences de visualiser une vue d'ensemble interactive des liens entre différents artefacts d'ingénierie comprenant :

- les composants du système pour communiquer avec les fournisseurs ;
- les évolutions ou les cas d'utilisation pour communiquer avec les utilisateurs finaux ;
- les sites pour communiquer avec les personnes responsables du déploiement.
- (Efficience dans le filtrage) (*chord diagram* comme exemple de design).

## V.2.6. Vues multiples coordonnées pour la visualisation de liens entre artefacts

La conclusion (C6) sur le besoin de visualiser des liens de type scénario entre exigences de différentes catégories concerne la visualisation du comportement du système : il s'agit de visualiser l'ordre d'appel des exigences pour un cas d'utilisation donné. Ce point s'est révélé être non satisfaisant dans notre analyse des pratiques industrielles (voir chapitre 4) et reste à explorer. En termes de visualisation d'information, il s'agit de visualiser des relations (dans notre cas les scénarios ou cas d'utilisation du système) dans des données hiérarchiques (les exigences du système).

### V.2.6.1. Coordination entre treemap et *chord diagram*

Fekete et al (Fekete et al. 2003) proposent de superposer la visualisation de liens sur le treemap pour visualiser des relations dans des données hiérarchiques. Cependant, cela entraine un encombrement visuel (à gauche de la Figure 110), qui peut être résolu par la technique de *bundling* (D. Holten 2006) (à droite de la Figure 110). Cette technique permet d'identifier des patterns mais ne permet pas de suivre un chemin. Or nous cherchons à visualiser l'ordre d'appel des exigences, qui se traduit par un suivi de chemin.

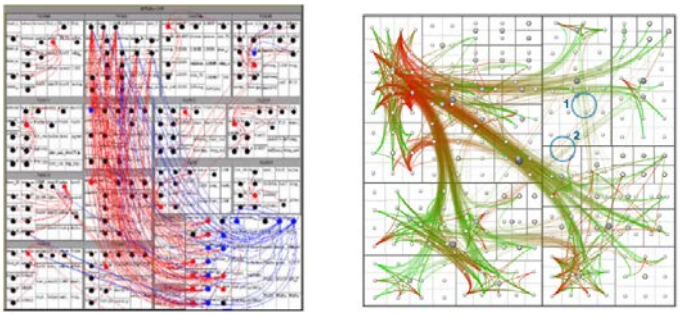


Figure 110: visualisation de relations sur un treemap (Fekete et al. 2003)

Chen et al. proposent de combiner une visualisation treemap avec un arbre dépliable pour visualiser les liens entre du code source et la documentation : la sélection d'un nœud dans le treemap représentant le code (en cyan sur la Figure 111) déclenche la mise en évidence de la documentation correspondante dans le treemap de la documentation (en magenta dans la Figure 111), ainsi que la construction d'un arbre montrant les détails de dépendances (en bas de la Figure 111). L'évaluation d'utilisabilité auprès de 15 participants montre que cette approche améliore la compréhension du logiciel, avec cependant des problèmes de visualisation des hiérarchies liés à l'utilisation de la couleur.

Ce résultat nous conforte dans notre choix d'explorer l'utilisation de vues multiples coordonnées pour visualiser les cas d'utilisation du système et le treemap des exigences, conformément aux principes de visualisation d'information proposés par Heer et Shneiderman (Heer et Shneiderman 2012).


Figure 111: combinaison d'un treemap avec un arbre dépliable (Chen, Hosking, et Grundy 2012)

Nous disposons de deux visualisations : le treemap des exigences et le *chord diagram* des évolutions système/composants du système/sites opérationnels. Le treemap des exigences comprend déjà de nombreuses interactions pour la navigation, le support à la vérification des exigences et le filtrage interactif. Par conséquent, nous privilégions la coordination des vues à partir d'interactions sur le *chord diagram.* Ainsi, la sélection d'une évolution sur le *chord diagram* met en évidence sur le treemap

les exigences impactées par l'évolution sélectionnée, avec la couleur spécifiée par l'utilisateur sur le *chord diagram*, comme illustré sur la Figure 112.

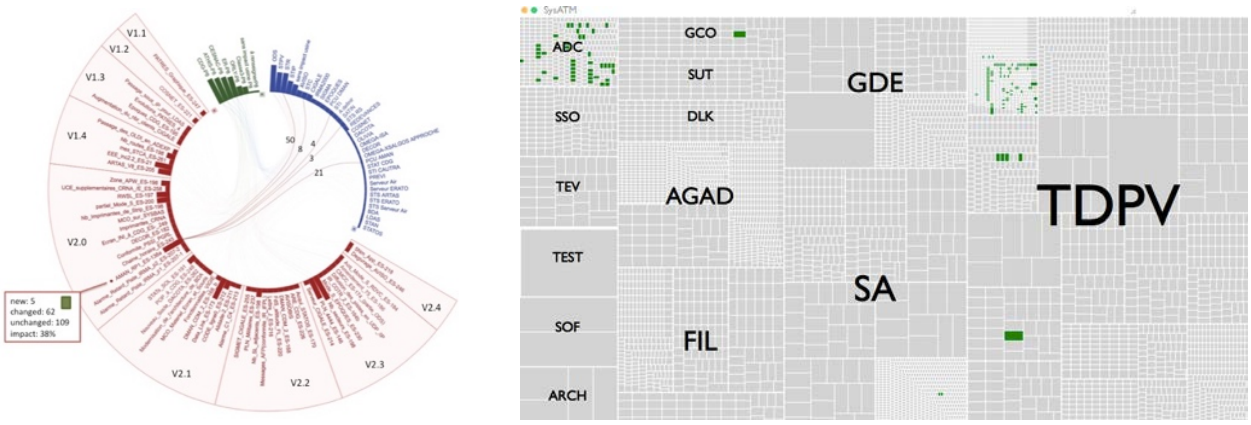



Figure 112: coordination entre visualisations : la sélection d'une évolution sur le chord diagram montre les exigences correspondantes sur le treemap

La sélection d'une deuxième évolution, avec le choix d'une couleur différente peut permettre d'identifier les exigences impactées par les deux évolutions sur le treemap, par superposition de couleur. Ainsi, en reprenant le scénario Gestion de contenus de version, le comité de pilotage peut évaluer la charge de travail consécutive à un changement de chronologie dans les évolutions, en identifiant les exigences potentiellement communes aux deux évolutions. Cette solution permet de visualiser sur le treemap les exigences requises par une évolution ou un cas d'utilisation, mais ne permet pas de visualiser l'ordre d'appel des exigences. Les discussions sur les résultats du filtrage interactif ayant montré la préférence des utilisateurs pour une visualisation tabulaire des résultats, nous avons éliminé le parcours des exigences à travers le treemap pour privilégier une visualisation tabulaire des exigences, en utilisant l'axe des y pour représenter l'ordre des exigences.

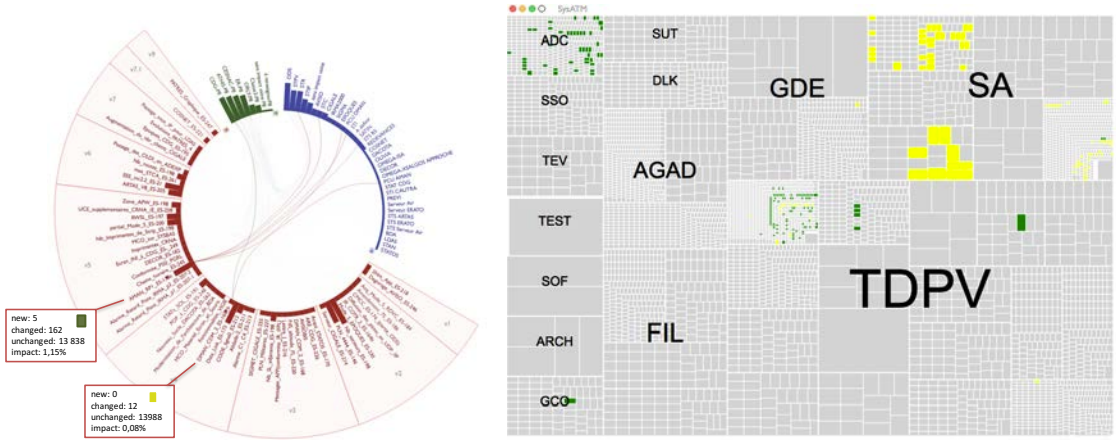



Figure 113: la sélection de deux évolutions sur le chord diagram, associées à des couleurs différentes, peut permettre d'identifier sur le treemap l'existence d'exigences impactées par les deux évolutions

### V.2.6.2. Vue détaillée des exigences ordonnées

A partir du treemap, nous proposons de visualiser les exigences textuelles ordonnées par glisser-déposer dans une zone, comme illustré en (a) sur la Figure 114. Dans la vue détaillée des exigences

ordonnées, les exigences deviennent des objets manipulables de l'interface, dont l'arrangement spatial sur l'axe vertical de la fenêtre reflète un ordre d'appel. Chaque exigence est un objet graphique que l'utilisateur peut supprimer (en sélectionnant le bouton rouge) s'il considère que l'exigence n'est pas pertinente pour ce cas d'utilisation, ou déplacer par glisser-déposer s'il souhaite modifier la séquence d'actions ((b) sur la Figure 114).

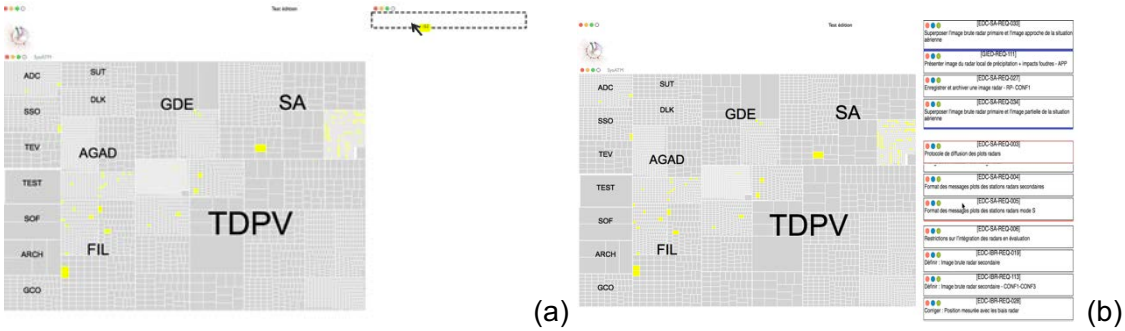



Figure 114: (a) glisser-déposer des exigences mises en évidence dans le treemap vers une zone dédiée à une lecture ordonnée des exigences (b) glisser-déposer dans la vue des exigences pour modifier leur ordre

Dans la même philosophie que DigiStrips (Mertz, Chatty, et Vinot 2000a) et Mammi (Conversy et al. 2011), qui proposent une manipulation interactive des strips en support au travail des contrôleurs aériens, nous proposons une manipulation interactive des exigences en support au travail des ingénieurs. L'avantage de cette représentation est de combiner l'expression textuelle des exigences avec des possibilités de manipulation directe réservée généralement aux graphiques pour spécifier des relations entre exigences. Ainsi, en approfondissant cette métaphore, on peut imaginer différentes interactions pour construire une spécification plus précise des cas d'utilisation. Par exemple, en tirant un trait entre deux ou plusieurs exigences, on peut spécifier une séquence obligatoire, ce qui aurait pour effet de grouper les exigences. De même en faisant une croix entre deux exigences, on peut spécifier une séquence interdite (voir Figure 115). Ces propositions d'interactions sont au stade de prototypes papier et n'ont été ni développés ni soumis aux participants de notre étude.


Figure 115: exemples d'interactions sur les exigences pour spécifier des séquences

L'utilisateur peut sauvegarder les différents groupes d'exigences ordonnées construits de cette manière, à travers une barre de titre interactive. La synchronisation des visualisations est assurée en temps réel : l'entrée dans la barre de titre déclenche la création d'un nouvel item dans le *chord diagram* (voir Figure 116).

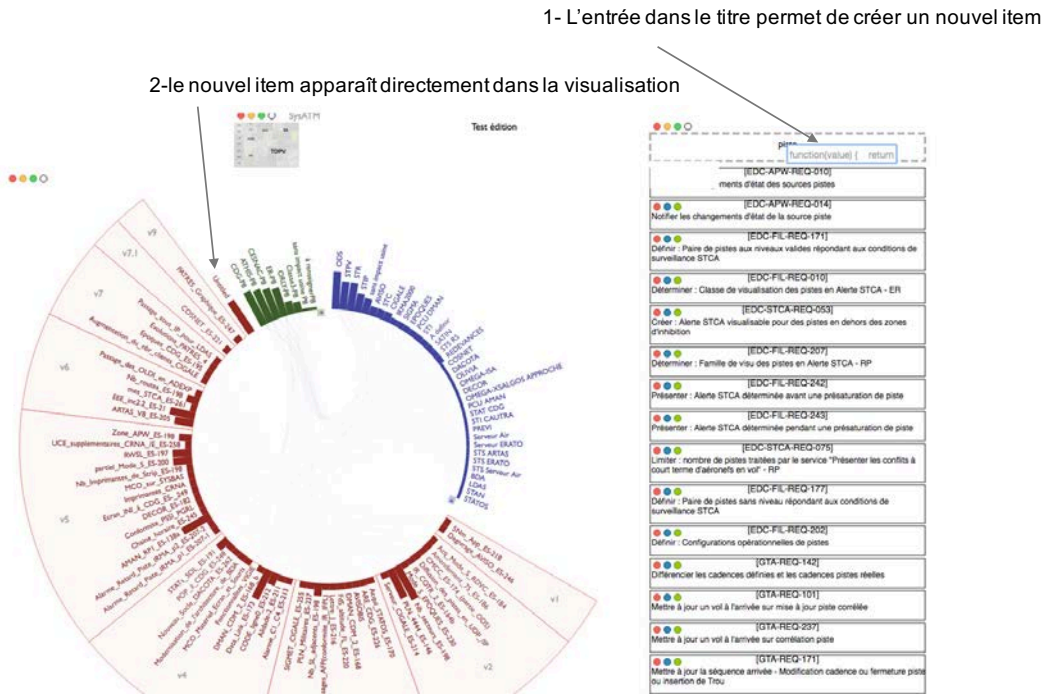



Figure 116: synchronisation des visualisations à la création d'un item

L'intérêt de ces interactions est de créer des liens entre artefacts d'ingénierie à partir de leurs visualisations, sans les manipuler explicitement : il n'existe pas un menu avec une action « créer un lien ». La création d'une évolution peut être déclenchée directement depuis le *chord diagram,* à partir d'un bouton « + », disposé radialement. Le nom de l'évolution peut être renseigné et des liens vers les composants peuvent être réalisés en désignant les composants par simple clic (voir Figure 117). Ces liens sont visualisés en pointillés pour montrer qu'il n'y a pas d'exigences sous-jacentes : l'idée est de proposer à l'utilisateur d'enregistrer l'évolution et les hypothèses sur les composants pouvant réaliser cette évolution, sans avoir formalisé les exigences. Notre objectif est ici de favoriser l'exploration et supporter la représentation d'objets intermédiaires ou partiels dans la décomposition de la solution, conformément aux recommandations émises par Guindon (R. Guindon et Curtis 1988)(Raymonde Guindon 1990) et à nos observations des pratiques industrielles (chapitre 4). On peut considérer cette action comme équivalente à l'ajout d'une ligne dans une feuille calcul et des précisions de type « à définir » dans les colonnes.

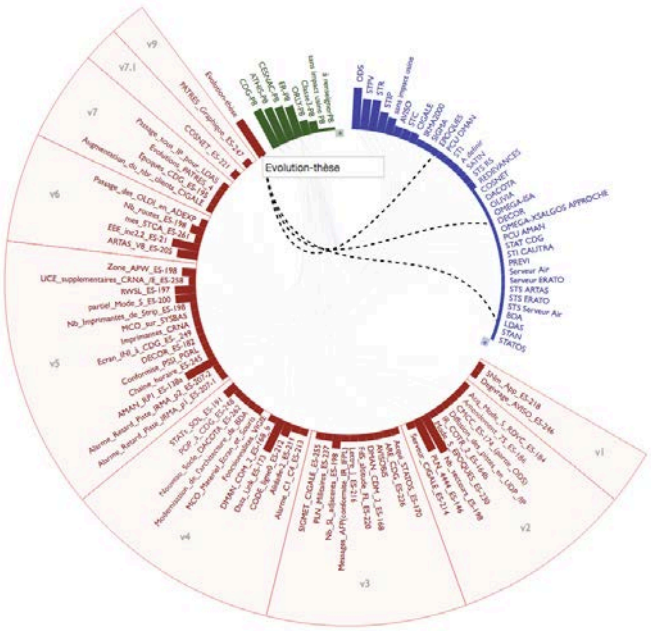



Figure 117: création d'une évolution et de liens à partir de la visualisation du *chord diagram*

### V.2.6.3. Organisation des visualisations à l'écran

Nous disposons désormais de trois visualisations différentes. Or l'utilisation de vues multiples coordonnées pose le problème de l'organisation des vues à l'écran (Heer et Shneiderman 2012). Une disposition en mosaïque permet aux utilisateurs disposant d'écrans assez larges de voir toutes les visualisations en même temps. De plus, nous proposons pour chaque visualisation, de façon analogue aux fenêtres sous MacOS, quatre boutons associés aux actions suivantes :

- Réduction maximale sans lisibilité de la visualisation (voir Figure 114) ;
- Réduction moyenne avec lisibilité de la visualisation ;
- Affichage normal de la visualisation ;
- Poignée de déplacement de la visualisation dans la fenêtre.

Ces boutons permettent à l'utilisateur de configurer les fenêtres selon son intérêt dans la communication avec les parties prenantes. Par exemple, pour une validation des cas d'utilisation par les utilisateurs finaux du système en cours de spécification, la séquence peut être celle illustrée en Figure 118.

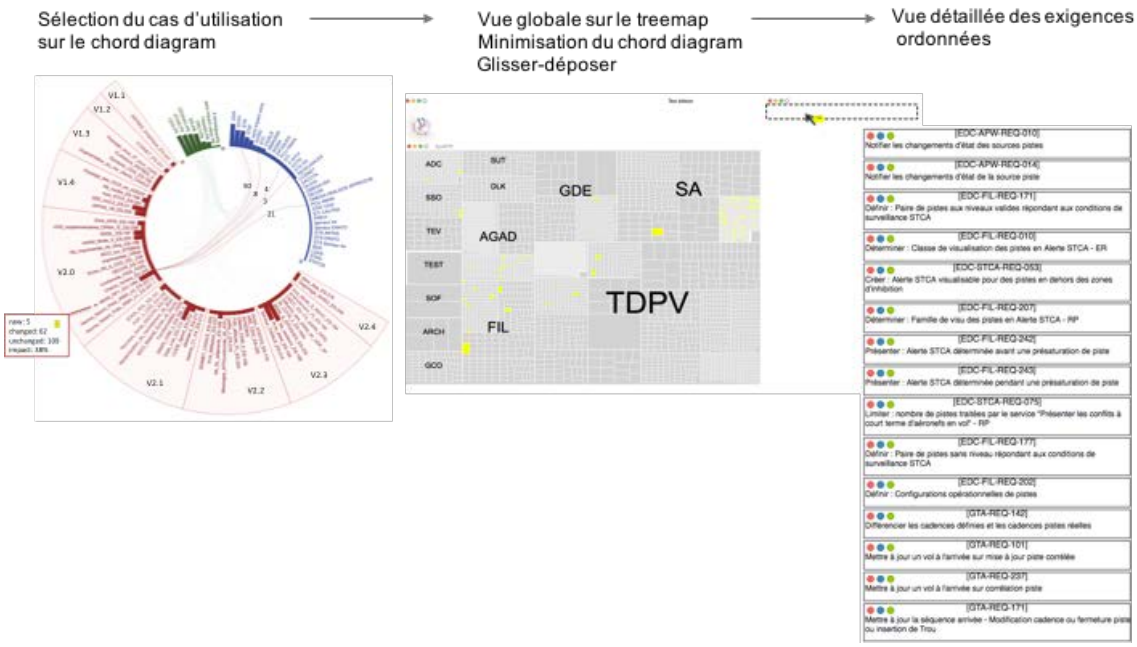



Figure 118: exemple de séquence d'utilisation des visualisations interactives

# V.3. Retours d'expérience sur l'utilisation des prototypes de visualisations interactives

L'objectif de nos prototypes est de servir de support à une recherche orientée design et non de constituer un produit fini (Fallman 2003)(Mackay et Fayard 1997). Afin d'éviter le piège de l'exemple jouet qui élude le passage à l'échelle et compromet le transfert des résultats vers l'industrie (Ivarsson et Gorschek 2009), nos prototypes sont alimentés par des données réelles de l'entreprise D, comprenant entre 4000 et 14 000 exigences. Les designs walkthroughs que nous avons conduits avec les ingénieurs en exigences à l'aide des prototypes nous ont permis d'approfondir nos connaissances sur leurs pratiques et sur la place de l'utilisabilité des outils dans le processus d'ingénierie des exigences.

Nous n'avons pas mené d'expérimentations contrôlées sur les prototypes de visualisations interactives des exigences du fait de la nature des activités d'ingénierie des exigences : nous avons déjà décrit, dans l'état de l'art (chapitre 2) et dans l'analyse des pratiques industrielles (chapitre 4), le caractère opportuniste des activités, sans chemin prédéfini dans leur réalisation, ainsi que la multiplicité des parties prenantes. Ces caractéristiques sont difficiles à reproduire en laboratoire. Cependant, dans une démarche de généralisation de nos résultats, nous avons cherché à évaluer l'utilisation des prototypes dans d'autres contextes que l'entreprise D. Nous avons exploité deux contextes d'utilisation au sein de l'ENAC : le projet d'avion tout électrique E-Fan 2.0 et les projets de synthèse des étudiants du niveau Master 2 IHM. Dans cette partie, nous reportons les retours d'expérience dans ces deux contextes d'utilisation.

## V.3.1. Spécification du cockpit de l'avion électrique E-Fan

Le projet E-Fan avait pour but de concevoir, construire et industrialiser un avion biplace côte à côte tout électrique pour la formation initiale des pilotes (les 20 premières heures de double commande et 5 premières heures de solo en local). La propulsion électrique est alimentée par un pack de batteries,

sous certification LSA Light Sport Aircaft, poids de l'avion < 600 kg). Le projet était mené par Airbus Group Innovation (AGI). Un consortium s'est articulé au fil de la construction du projet, regroupant une diversité d'acteurs (académiques, PME, grands groupes) : Safran, Zodiac Aerospace, Evtronic, Aéro Composites Saintonge, SERMA Technologies, le CEA, l'ENSAM, ainsi que DAHER-SOCATA et plusieurs PME régionales sous-traitantes, mais également deux grandes écoles aéronautiques : l'ENAC et l'ISAE.

Ainsi, l'ENAC a été impliquée pendant deux ans, d'octobre 2014 à septembre 2016, dans le premier cercle des partenaires du projet, avec un triple rôle : celui d'expert (avionique et tableau de bord), de prescripteur (formation des élèves-pilotes) et d'utilisateur (exploitant d'avion-école).

Notre intervention se situe dans le cadre de l'expertise sur la conception des systèmes bord. Nous avons mené le recueil des besoins des pilotes-instructeurs de l'ENAC et la formalisation du besoin sous forme d'exigences pour la spécification du cockpit, en mettant en place une démarche participative que nous détaillerons dans le chapitre suivant. Le résultat est l'identification de 13 cas d'utilisation et de 168 exigences, illustrés par des prototypes interactifs de tableau de bord intégrant un système de gestion de l'énergie, qui ont été livrés à la Banque Public d'Investissement, financeur du projet, en juillet 2016.

L'utilisation des prototypes de visualisations des exigences dans ce contexte, avec des données différentes, nous a conduit à réfléchir au fichier en entrée des visualisations. Jusqu'à présent, la structure des données en entrée n'était pas modifiable, car elle nous était fournie par nos utilisateurs. Dans le contexte de l'E-Fan 2.0, en tant que responsable de la spécification du cockpit, nous pouvions réfléchir à la façon de structurer les données.


Figure 119: onglets du fichier

Nous avons conçu une correspondance entre onglets du fichier et visualisations (voir Figure 119) :

- un onglet (*functions* sur Figure 119) décrit la hiérarchie du treemap (voir Figure 120);
- chaque onglet qui suit ( *useCases, regulations, components* sur la Figure 119) correspond à un artefact d'ingénierie, positionné comme entité sur le *chord diagram* (voir Figure 121) ;
- un dernier onglet (reqs sur la Figure 119) contient les exigences, et les liens avec les autres artefacts d'ingénierie (Figure 122).

Un parser permet de générer les visualisations à partir de ce fichier. Le travail de spécification peut être ainsi fait dans le fichier Excel, puis être présenté à l'aide des visualisations interactives.

| id | title | mnemo |
|---|---|---|
| F0 | système eFan | SysEfan |
| F1 | Interfaces | ITF |
| F1.1 | Efan-Tablet Interface | EFAN TAB |
| F1.2 | Efan-Fleet Data concentra | EFAN Data |
| | | |
| F2 | primary flight data | PRIM |
| F2.1 | Efan flying & communicati | FLY |
| F2.2 | Power&Energy indication | NRJ |
| | | |
| F3 | System monitoring | SYS |
| F3.1 | Failure management | FAIL |
| F3.2 | EPU monitoring | MON |
| | | |
| F4 | Mission monitoring | Mission |
| F4.1 | Interactive map | Map |
| F4.2 | Gauges | Gauges |
| | | |
| F5 | Numeric checklist | CHCK |
| | | |
| F6 | Com Nav Surv | CNS |
| F6.1 | Communication | COM |
| F6.2 | Navigation | NAV |
| F6.3 | Surveillance | SURV |
| | | |
| F7 | Training | TRN |
| F7.1 | Event memorization | MEM |
| F7.2 | Event playback | PBCK |
| F7.3 | Student management | STUD |
| | | |
| F8 | to be defined | TBD |
| | | |
| F9 | Fleet Management | Fleet |

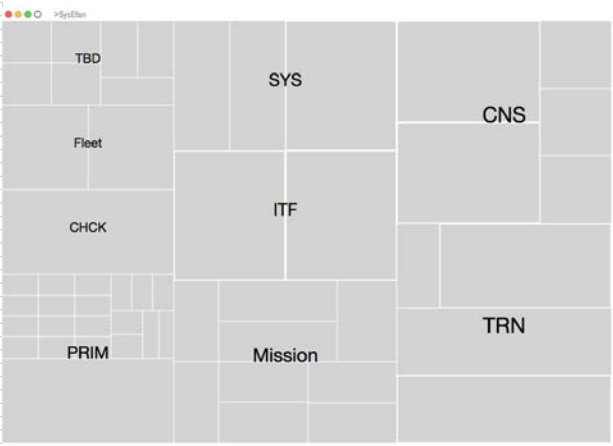


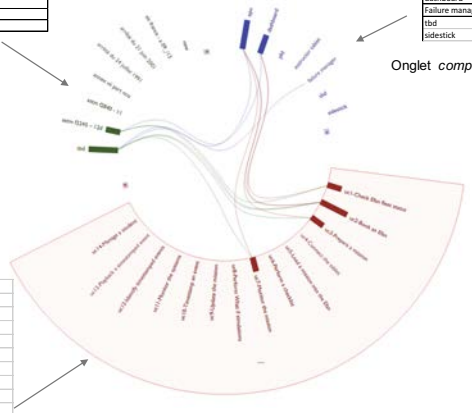

Figure 120: description de la hiérarchie des fonctions dans un onglet, et treemap associé

| title |
|---|
| ASTM F2245 – 12d |
| ASTM F2840 - 11 |
| Annex VII Part NCO |
| Arrêté du 24 juillet 1991 |
| arrêté du 21 juin 2001 |
| AIC France - A 09/13 |

Onglet *regulations*

| name |
|---|
| PFD |
| EPU |
| Instructor tablet |
| dashboard |
| Failure manager |
| tbd |
| sidestick |

Onglet *components*



| title |
|---|
| uc1-Check Efan fleet status |
| uc2-Book an Efan |
| uc3-Prepare a mission |
| uc4-Connect the tablet |
| uc5-Load a mission into the Efan |
| uc6-Perform a checklist |
| uc7-Monitor the mission |
| uc8-Perform What if simulations |
| uc9-Update the mission |
| uc10-Timestamp an event |
| uc11-Monitor the systems |
| uc12-Identify timestamped events |
| uc13-Playback a timestamped event |
| uc14-Manage a student |

Onglet *useCases*

Figure 121: construction du *chord diagram* à partir d'onglets du fichier Excel

| UseCase | function | req id | req version | requirement text | parent req | regul | component | additional info | other info |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| UC1<br>UC2 | NRJ | REQID_001 | | **The eFan shall include at least the equipment required by the CS-LSA certification documents:**<br>**- ASTM F2245 -12d Design and Performance of a Light Sport Airplane**<br>**- ASTM F2840 – 11 Design and Manufacture of Electric Propulsion Units for** | | ASTM F2245 – 12d | dashboard | | |
| UC3 | NRJ | REQID_002 | | **In order to be able to fly in VFR day conditions, the eFan shall comply with the following documents:**<br>**- "Arrêté du 24 juillet 1991" ref EQUA9101162A**<br>**- "Arrêté du 21 juin 2001" ref EQUA0100683A**<br>**- EASA Annex VII - Part-NCO, Non-commercial air operations with other-than-complex motor-powered aircraft** | | | | | |
| UC1<br>UC2 | | REQID_003 | | **The eFan shall include an airspeed indicator** | | ASTM F2245 – 12d §8<br>Arrêté du 24 juillet 1991 §2.6.2.1 a<br>Annex VII Part NCO.IDE.A.120 | | | |
| | | REQID_004 | | **The eFan shall include an altimeter** | | ASTM F2245 – 12d §8<br>Annex VII Part NCO | | | |
| | **TBD** | REQID_005 | | **The eFan shall include a tachometer informing of the motor RPM and maximum allowable RPM.** | | ASTM F2245 – 12d §8<br>ASTM F2840 – 11 §6.13.3 | | | |
| | **TBD** | REQID_006 | | **The eFan shall include an engine kill switch** | | ASTM F2245 – 12d §8 | | | |
| | **TBD** | REQID_007 | | **The eFan shall include the engine instruments as required by the engine** | | ASTM F2245 – 12d §8 | | | |
| | **TBD** | REQID_008 | | **The eFan shall include a master switch.** | | ASTM F2245 – 12d §8 | | | |
| | **TBD** | | | **The eFan shall include overload protection devices.** | | ASTM F2245 – 12d §8 | | | |
| | **TBD** | REQID_009 | | **The eFan shall incorporate a "fuel gauge" including minimum energy reserves and numerical readout showing energy remaining.** | | ASTM F2840 – 11 §6.13.1<br>ASTM F2245 – 12d §8 | | | |
| | **TBD** | REQID_010 | | **The eFan EPU should incorporate a warning light to inform the pilot that there is a minimum energy left.** | | ASTM F2840 – 11 §6.13.1 | | | |
| | **TBD** | REQID_011 | | **The eFan shall incorporate a temperature gauge for the motor temperature status and maximum allowable temperature.** | | ASTM F2840 – 11 §6.13.4 | | | |
| | **COM** | REQID_012 | | **The eFan shall inform the pilot when the EPU is set in RUN mode** | | ASTM F2840 – 11 §6.13.5 | | | |

Figure 122: extrait de l'onglet reqs contenant les exigences et les liens avec cas d'utilisation, textes réglementaires, fonctions et composants

Pour le projet E-Fan, nous avons identifié les textes réglementaires de certification (ASTM International 2016) comme artefacts d'ingénierie à positionner autour du *chord diagram*. Ces textes règlementaires décrivent les instruments de vol obligatoires pour que l'avion soit autorisé à voler dans

l'espace européen par l'agence européenne de sécurité (EASA). La représentation permet ainsi de répondre aux questions suivantes, relatives aux exigences de certification de l'avion : « quels sont les cas d'utilisation qui permettent d'atteindre l'objectif de certification de l'avion électrique ? » « Quels sont les composants du système qui portent les exigences de certification ? ».

Ce travail sur E-Fan 2.0 a été mené alors que la rédaction de la spécification était pratiquement terminée : nous ne pouvons donc pas conclure sur une amélioration que pourraient apporter les visualisations interactives des exigences sur la conduite du processus d'ingénierie des exigences. Cependant, nous avons apprécié la souplesse de pouvoir positionner un artefact particulier (les textes réglementaires) sur la visualisation afin de répondre aux questions récurrentes de la part des autres partenaires du projet. Enfin, ce travail nous a permis de nous abstraire des données spécifiques de l'entreprise D, et de consolider un workflow, avec un fichier et un parser, potentiellement utilisables dans d'autres contextes. C'est ce que nous avons voulu tester dans le contexte des projets des étudiants du Master 2 IHM.

## V.3.2. Spécification de projets étudiants en Master 2 IHM

Le Master2 IHM (www.masterihm.fr) est une formation co-habilitée par l'Université Paul Sabatier et l'ENAC, dont l'objectif est de former des spécialistes de la conception et du développement d'applications interactives. Les étudiants sont amenés à mettre en pratique les enseignements dans un projet tout le long du semestre de scolarité. Ce projet, appelé « chef d'œuvre », est proposé par une entreprise ou un laboratoire et réalisé par une équipe de quatre étudiants. Dans le cadre de notre enseignement relatif au management de projets centrés utilisateur et l'ingénierie des exigences d'utilisabilité, nous demandons aux étudiants de produire une spécification du système interactif issu du chef d'œuvre. Pour la promotion 2016-2017, nous avons fourni le modèle de fichier décrit précédemment, accompagnés des visualisations, pour une utilisation sur les sept chefs d'œuvre de la promotion. La consigne donnée aux étudiants est la production des cas d'utilisation, l'identification de contraintes réglementaires éventuelles dans l'onglet *regulations*, et la rédaction détaillée des exigences dans l'onglet *reqs*, en respectant les règles de syntaxe de la norme ISO 29148 (présentées dans la partie Les différentes expressions des exigences).

Nous leur avons présenté la correspondance entre onglets et entités du *chord diagram*, et laissé la possibilité d'utiliser les onglets pour d'autres artefacts d'ingénierie. Notre objectif était d'évaluer (1) l'applicabilité du fichier Excel et des visualisations sur des projets divers, (2) la possibilité laissée aux étudiants d'adapter le contenu au contexte de leur projet. Les étudiants de master 2 Informatique n'ont pas le niveau de connaissances et de compétences de praticiens industriels. Cependant, c'est une population facile d'accès et classiquement utilisée en génie logiciel dans les évaluations, comme nous avons pu le constater dans l'état de l'art (Ivarsson et Gorschek 2009).

| Chef d'œuvre | Client | Nbre de cas d'utilisation | Nbre d'exigences | Nbre de composants |
|---|---|---|---|---|
| VOLTA-R: réalité augmentée pour hélicoptère électrique | Aquinéa | 17 | 72 | 5 +tbd |
| Programmation dynamique de Drones | ENAC | 6 | 47 | 7 + Soft (pour tbd) |
| EyeSat: centre de contrôle commande | CNES-ISAE | 8 | 39 | 10 |
| Saisie de l'activité paramédicale | MIPIH | 12 | 42 | 2 |
| République numérique et mobilité | Berger-Levrault | 12 | 7 | 4 |
| Refonte cartographique pour la gestion de flotte | Orange-Ocean | 28 | 45 | 3 |
| Focus numérique pour contexte physique | IRIT | 9 | 24 | 4 |

Tableau 7 : synthèse des  données d'ingénierie sur les chefs d'oeuvre

Nous fournissons dans le Tableau 7 la synthèse des données d'ingénierie sur les sept chefs d'œuvre : le nombre de cas d'utilisation, le nombre d'exigences et le nombre de composants identifiés. Les cas

d'utilisation sont bien identifiés par tous les groupes : la démarche participative ou centrée utilisateur que les étudiants mettent en œuvre centre la réflexion sur les usages et facilite une identification des cas d'utilisation. Les exigences sont bien formulées, en suivant la syntaxe préconisée, sauf pour un groupe (République numérique et mobilité- voir Tableau 7) : leur sujet visait à explorer des techniques d'interaction innovantes sur une application mobile existante, sans ajout de service. Il ne s'agissait pas d'un système complexe, mais d'une application mobile sans évolution à spécifier, ce qui peut expliquer les difficultés rencontrées par le groupe projet. Deux groupes projets (Programmation dynamique de drones et EyeSat) ont choisi de mettre en composants du système les éléments de l'IHM (exemples : timeline, carte, vue synoptique) (voir Figure 123). Une stratégie opposée a été choisie par le groupe « Refonte cartographique pour la Gestion de Flotte » : les composants définis sont « écran, clavier, souris », et la colonne « Additional Info » dans l'onglet des exigences a été utilisée pour préciser les composants IHM. Dans la première stratégie, les composants IHM sont mis en valeur, alors que dans la deuxième stratégie ils deviennent des informations de second plan. Nous n'avons pas conduit de discussions avec les étudiants sur le choix de cette stratégie. Cependant, le résultat intéressant est que, dans les deux stratégies, les informations sur les composants IHM sont présentes. Le choix de la stratégie pourrait être déterminé par la maturité de la spécification, son niveau de prescription (obligatoire ou optionnel), ou l'intérêt d'une communication spécifique.

Le groupe projet EyeSat a choisi de positionner les profils utilisateur (superviseur et expert) et l'origine du besoin (réunions client, visite centre de contrôle) sur le *chord diagram* et de tracer les cas d'utilisation et les composants IHM vers ces éléments (voir en haut de la Figure 123, en vert sur le *chord diagram*). Ce choix leur permet d'utiliser l'interaction sur la visualisation pour justifier un cas d'utilisation par rapport à une réunion ou un profil. Ces résultats illustrent l'intérêt du *chord diagram* pour montrer des informations de liens entre artefacts d'ingénierie (exigence d'utilisabilité R26), et la flexibilité offerte par la correspondance entre onglets du fichier Excel et éléments de la visualisation : les groupes projet ont pu adopter des stratégies différentes, sans préconisation de notre part, avec le même niveau d'information enregistré. Deux groupes (Voltar et Programmation dynamique de drones) ont intégré dans la liste de leurs composants un « tbd » (*to be done*) ou « soft » (*software*, sans préciser le composant IHM), leur permettant d'indiquer les cas d'utilisation et exigences n'ayant pas encore été spécifiés. Cette observation confirme l'intérêt de pouvoir enregistrer de l'information, tel qu'un cas d'utilisation, sans qu'elle soit consolidée. Ces mêmes deux groupes ont réussi à maintenir le fichier Excel compatible avec les visualisations jusqu'au bout : les autres groupes ont réalisé de la mise en forme sur le fichier Excel, qui rend inopérant le parsing du fichier. Notre parser est actuellement trop sensible à la mise en forme et devrait être amélioré pour rendre cette sensibilité moins forte.


Figure 123: deux exemples d'utilisation des visualisations avec les composants IHM en bleu sur le *chord diagram*

Enfin, d'un point de vue pédagogique, l'intérêt du fichier Excel et des visualisations est de proposer un cadre pour la spécification, qui a encouragé les étudiants à plus de rigueur par rapport aux années précédentes, à travers une réflexion sur les liens entre cas d'utilisation, exigences et composants IHM. Auparavant, nous demandions aux étudiants de réaliser ces liens via un seul tableau et sans les visualisations. Les informations obtenues, sur des projets de qualité, restaient parcellaires et non liées entre elles, avec par exemple les cas d'utilisation d'un côté sous forme graphique, et les exigences dans un tableau de l'autre, de simples « ok » ou virgule verte pour démontrer la prise en compte de l'exigence. Il reste cependant délicat de comparer des promotions entre elles, d'autant que la formation du Master 2 IHM a connu des évolutions dans l'année 2016-17.

# V.4. Synthèse

## V.4.1. Des visualisations interactives et coordonnées de texte structuré

La maxime *One picture is worth a thousand words* ou en français *Un bon croquis vaut mieux qu'un long discours* n'est pas valide pour l'expression des exigences. Brooks (Brooks 1995), parlant de la description de logiciel, évoque *the flow chart curse*, la malédiction du flow chart. Et un des pères fondateurs de l'analyse structurée, DeMarco (DeMarco 2002) écrit :

> *Narrative specs are not the problem; a suitably partitioned spec with narrative text used at the bottom level makes a fine statement of work.*

Le texte, utilisé pour l'expression des exigences, des scénarios et cas d'utilisation, possède de nombreux avantages (voir II.1.2.1.2 et II.1.2.4) :

- sa lisibilité par toutes les parties prenantes, notamment les utilisateurs finaux ;
- la possibilité d'être structuré et numéroté, indiquant une hiérarchie et un ordre de lecture ;
- sa capacité à être facilement modifié : cette caractéristique est cruciale du fait de la nature changeante et instable des exigences.

Ces nombreux avantages peuvent expliquer l'utilisation du texte comme langage de spécification, rapportée par les études de terrain dans l'état de l'art et observée dans notre étude. Cependant, nous avons relevé dans nos observations des problèmes d'utilisabilité des outils spécifiques utilisés pour la gestion des exigences textuelles (chapitre 4) : faible visibilité avec une absence de vue d'ensemble et une difficulté de navigation, difficulté de filtrage, évaluation progressive du travail difficile, engagement prématuré, et une insatisfaction exprimée par les participants.

Nous avons cherché à conserver l'utilisation du texte tout en répondant aux problèmes rencontrés par les ingénieurs en exigences. Nos propositions sont :

- des visualisations interactives et coordonnées de texte structuré: le treemap zoomable des exigences pour la vue d'ensemble et la navigation, l'inspecteur des exigences pour l'évaluation progressive du travail et la recherche d'anomalies en support à la vérification des exigences, le *chord diagram* pour la visualisation de liens entre artefacts d'ingénierie et une communication améliorée avec les différentes parties prenantes, la vue des exigences ordonnée pour une manipulation directe des exigences. Ces visualisations interactives sont issues d'un travail itératif de design avec les participants de l'entreprise D. L'alimentation des prototypes par des données réelles, jusqu'à 14 000 exigences, a permis une meilleure implication des participants et le traitement direct du passage à l'échelle.
- des exigences d'utilisabilité formulées à partir de notre connaissance des activités et des design walkthroughs (voir synthèse en Tableau 8).

| Exigences d'utilisabilité | Origine |
|---|---|
| Afin de conduire l'analyse de couverture, l'outil doit permettre à l'ingénieur en exigences de :<br><br>**R15** : Visualiser une vue d'ensemble des exigences du système sans action (efficience dans la compréhension de la vue) (treemap comme exemple de design)<br><br>**R16** : Accéder aux exigences d'un niveau d'abstraction donné avec un minimum d'actions à partir de la vue d'ensemble des exigences (efficience dans la navigation) (treemap comme exemple de design)<br><br>**R17** : Lire l'exigence initiale et les exigences raffinées simultanément (efficience dans la lecture) (treemap comme exemple de design)<br><br>**R18** : Accéder à des actions sur l'exigence à partir de la visualisation : valider, rejeter, commenter, modifier (efficience dans la vérification) (treemap avec les actions en clic droit comme exemple de design d'interaction)<br><br>**R19** : voir l'état d'avancement des exigences (validée/modifiée-commentée/rejetée) sur la vue globale des exigences (évaluation progressive forte) (inspecteur et treemap comme exemples de design).<br><br>**R20** : Identifier les exigences non raffinées d'un coup d'œil à partir de la vue d'ensemble des exigences (efficience dans la détection d'anomalies) (rectangle noir sur le treemap comme exemple de design)<br><br>**R21** : Filtrer les exigences de façon dynamique à partir de la vue globale des exigences avec retour d'information immédiat sur la vue globale (efficience dans la détection d'anomalies) (utilisation de la couleur sur le treemap comme exemple de design)<br><br>**R21_1** : par mots clés à partir de n'importe quel niveau d'abstraction (efficience dans le filtrage) (inspecteur et treemap comme exemples de design)<br><br>**R21_3** : par auteur à partir de n'importe quel niveau d'abstraction (efficience dans le filtrage) (inspecteur et treemap comme exemples de design).<br><br>**R21_4** : par type (safety, performance, fonctionnelle, interface) à partir de n'importe quel niveau d'abstraction (efficience dans le filtrage) (inspecteur et treemap comme exemples de design). | Raffinement et gestion des exigences dans le référentiel<br><br>Design walkthroughs |
| **R21_2** : Afin de réaliser une analyse d'impact, l'outil doit permettre à l'ingénieur de filtrer dynamiquement les exigences par mots clés à partir de n'importe quel niveau d'abstraction avec retour d'information immédiat sur la vue d'ensemble (efficience dans le filtrage) (inspecteur et treemap comme exemples de design).<br><br>**R22** : Afin de traiter les exigences filtrées, l'outil doit permettre à l'ingénieur en exigences d'afficher les exigences issues d'un filtrage dynamique dans une visualisation tabulaire éditable (efficience dans le traitement des anomalies) | Raffinement et gestion des exigences dans le référentiel<br><br>Design walkthroughs |

| | |
|---|---|
| Afin de gérer les versions système de façon flexible, l'outil doit permettre à l'ingénieur en exigences et au comité de pilotage de :<br>**R23** : visualiser les évolutions à l'étude et les composants impactés correspondant à chaque évolution (efficience dans la sélection et la lecture) (*chord diagram* comme exemple de design) ;<br>**R24** : naviguer d'une évolution à l'autre avec des statistiques sur les exigences (nouvelles, modifiées, inchangées) (efficience dans l'aide à la décision) (*chord diagram* comme exemple de design d'interaction)<br>**R25** : Afin d'avoir une conscience de la situation des évolutions à l'étude, l'outil doit permettre à l'équipe d'ingénieurs en exigences de partager une visualisation des évolutions et des composants impactés (efficacité dans la cognition de l'équipe) (*chord diagram* comme exemple de design) | Raffinement et gestion des exigences dans le référentiel<br>Design walkthroughs |
| **R26** : Afin de communiquer avec les différentes parties prenantes, l'outil doit permettre à l'ingénieur en exigences de visualiser une vue d'ensemble interactive des liens entre différents artefacts d'ingénierie comprenant :<br>les composants du système pour communiquer avec les fournisseurs ;<br>les évolutions ou les cas d'utilisation pour communiquer avec les utilisateurs finaux ;<br>les sites pour communiquer avec les personnes responsables du déploiement.<br>(Efficience dans le filtrage) (*chord diagram* comme exemple de design) | Raffinement et gestion des exigences dans le référentiel<br>Design walkthroughs |

Tableau 8: exigences d'utilisabilité pour les outils d'ingénierie des exigences sur les visualisations interactives de texte structuré

## V.4.2. La rigueur en sortie, sans la rigidité pendant le processus d'ingénierie des exigences

Nos premiers résultats montrent que les visualisations interactives et coordonnées de texte structuré peuvent apporter une vision de la spécification du système permettant d'améliorer la communication avec les parties prenantes tout en bénéficiant des avantages du texte (lisibilité et facilité de modification). Les activités de vérification des exigences textuelles sont instrumentées par des interactions sur les visualisations. L'ingénieur peut chercher du texte, voir l'état d'avancement des exigences, détecter et compléter les informations manquantes par une navigation et un filtrage interactifs sur les visualisations.

Les visualisations interactives proposées permettent de décorréler rigueur et rigidité dans le processus d'ingénierie des exigences, en rendant possible une souplesse pendant le processus tout en éliminant progressivement toute approximation en sortie du processus. La souplesse provient de l'utilisation d'outils à vocation générale pour le modelage de la structure et la rédaction des exigences, que l'on associe à des visualisations interactives améliorant la communication sur la structure de la spécification et instrumentant une vérification progressive des exigences. Dans notre vision située de l'ingénierie des exigences, nous représentons les visualisations interactives de texte structuré comme outil utilisé par les ingénieurs en exigences pendant le processus, et les matrices de traçabilité comme preuve pour la certification en sortie du processus (voir Figure 124).

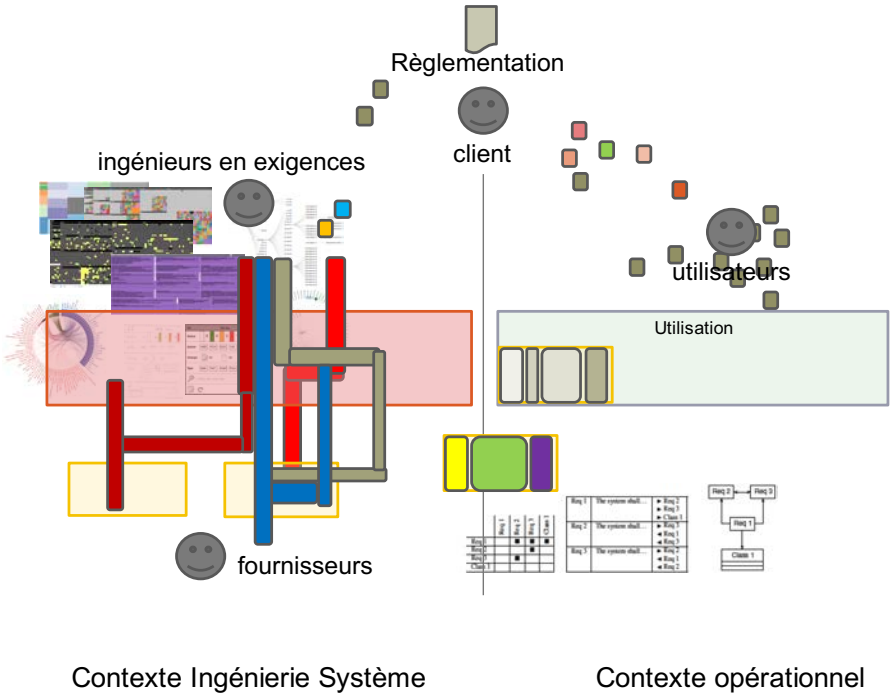



Figure 124: visualisations interactives des exigences : moins de rigidité pendant le processus et la rigueur en sortie du processus

## V.4.3. Mieux comprendre le système en utilisation pour mieux anticiper dans le système en définition ?

La finalité de nos travaux est d'étendre la capacité intellectuelle des ingénieurs à gérer la complexité d'un système, en spécifiant et comprenant au mieux son comportement. Nous avons centré nos travaux sur les outils des ingénieurs en exigences et formulé des exigences d'utilisabilité sur ces outils (Tableau 6 et Tableau 8), pour favoriser la souplesse nécessaire à la conception sans mettre en danger la rigueur requise en sortie du processus par la certification.

Cependant, l'exploitation d'un système dans un environnement changeant entraine une multiplication des conditions d'utilisation à envisager et à spécifier : la complexité réside dans une « incertitude au sein de systèmes richement organisés » (Morin 2005). Leveson (Leveson 2002) parle de complexité *interactive:* un système est interactivement complexe quand le niveau d'interactions entre les composants du système et son environnement atteint un point tel que ces interactions ne peuvent pas être totalement planifiées, comprises, anticipées et contrées. Dans une mise en perspective historique de la gestion de la complexité, Hollnagel (Erik Hollnagel 2012) fait le constat que les systèmes sont devenus tellement complexes que *les situations de travail associées* sont *chroniquement sous-spécifiés, d'où en partie imprédictibles.* Les moyens d'action utilisés jusqu'à présent pour faire face à la complexité, à savoir l'automatisation, les principes de conception, et la formation, ne sont plus suffisants. Il conclut:

> *Things therefore go right because people learn to overcome or compensate for design shortcomings (incomplete design). We need to understand how things go right—and how we can support this essential capability.*

Goguen (Goguen 1994) pointe le contexte social et situé du système comme cause d'une spécification incomplète. Il propose une spécification « rétrospective » : si les ingénieurs ne peuvent pas établir une spécification complète d'un système complexe a priori, pendant la phase de conception, ils peuvent améliorer la spécification d'un système complexe a posteriori, dans la phase d'utilisation du système. Cette proposition paraît difficilement acceptable pour des systèmes critiques, avec des enjeux de sécurité. Notre vision située de l'ingénierie des exigences, dans laquelle cohabite contexte opérationnel et contexte d'ingénierie du système, suggère le processus d'ingénierie des exigences comme moyen d'action pour faire face à la complexité interactive : quelles données issues du contexte opérationnel peuvent améliorer la compréhension des interactions entre composants du système et son environnement ? Comment mieux informer la définition du système futur à partir des données issues du système actuel ? Quels cadres d'analyse pour décrire et générer l'interaction ?

# Chapitre VI. Une ingénierie participative des exigences pour les systèmes interactifs complexes

*'Observation is a process in which we play an intensely active part. An observation is a perception, but one which is planned and prepared. We do not have an observation…but we 'make' an observation'.*
*(Popper 1979)*

*'Tout paysage se présente d'abord comme un immense désordre qui laisse libre de choisir le sens qu'on préfère lui donner'.*
*(Lévi-Strauss 1957)*

*'La liberté n'est pas la possibilité de réaliser tous ses caprices ; elle est la possibilité de participer à la définition des contraintes qui s'imposeront à tous'.*
*(Jacquard et Planès 1999)*

Ce chapitre est centré sur nos contributions relatives à l'amélioration de la spécification d'un système interactif complexe, cherchant à informer la définition du système futur à partir de données issues du système actuel en contexte opérationnel. Nous proposons de considérer l'utilisateur comme une frontière entre le système et son environnement. A ce titre, il est un poste d'observation privilégié pour comprendre et définir le comportement du système et les interactions avec l'environnement. Nous élaborons une approche d'ingénierie participative des exigences, basée sur des techniques issues de la conception participative et les cadres d'analyse de l'interaction que nous cherchons à intégrer dans l'ingénierie des exigences des systèmes complexes.

Nous exposons tout d'abord les fondations de notre approche et ses principes. Ensuite, nous illustrons l'application de notre approche sur différents projets : collaboration pour le contrôle aérien, cockpit d'un avion-école électrique, analyse d'un rapport d'accident, et nouvel instrument de vol.

# VI.1. Les utilisateurs dans l'ingénierie des exigences des systèmes interactifs complexes

## VI.1.1. Les utilisateurs dans les systèmes complexes : une frontière entre système et environnement

Nous avons relevé dans notre introduction consacrée à la définition de *système* la difficulté des normes d'ingénierie des systèmes à positionner les êtres humains dans le système : sont-ils des éléments du système ou à l'extérieur ?

Dans la théorie du système général, Le Moigne précise que la détermination des frontières entre le système et l'environnement dépend des intentions du modélisateur. Il esquisse des caractéristiques pour repérer les frontières (p242) : une digitalisation de l'information à la traversée d'une frontière, une différence de densité dans les nœuds du réseau de l'interaction (forte à l'intérieur du système, faible à l'extérieur), le couplage à un processeur au moins n'appartenant pas au système mais à son environnement, des points au voisinage desquels la loi d'évolution du système n'est plus maitrisée.

Or nous constatons les caractéristiques suivantes chez les utilisateurs : ils sont en interface avec l'environnement du système par les cinq sens, et en interface avec le système par l'IHM. Ils sont récepteurs et émetteurs d'événements qui concernent ou affectent le comportement du système. Enfin, les utilisateurs ont un rôle de codeur/décodeur et de digitalisation de l'information vers le système.

Ce constat nous conduit à identifier les utilisateurs comme des *frontières* entre l'environnement et le système. En cela, notre approche diffère radicalement du *Cognitive Systems Engineering,* qui propose de penser le couple humain-machine comme un système cognitif joint, et met par conséquent de côté le phénomène de l'interaction entre l'humain et le système (E. Hollnagel et Woods 1999)(Erik Hollnagel et Woods 2005). Une frontière étant reconnue comme une source d'information sur la mise en œuvre des séquences de comportement du système dans son environnement (Le Moigne 2006), nous cherchons à positionner les utilisateurs comme informateurs dans la définition du système.

## VI.1.2. Les utilisateurs dans l'ingénierie des systèmes interactifs complexes

### VI.1.2.1. Les utilisateurs dans la conception des systèmes interactifs

La valeur-clé de la conception participative est de promouvoir la participation des utilisateurs dans la conception des systèmes. Il existe des idées fondatrices et sous-tendues par la conception participative, développées dans les années 70-80 (Greenbaum et Kyng 1991) dans un contexte politique de démocratie au travail:

- le travail est une activité sociale impliquant une interaction entre de nombreux groupes de personnes ;
- les systèmes doivent être mieux adaptés aux compétences et aux méthodes de travail des personnes qui vont effectivement utiliser ces systèmes ;
- les barrières entre les experts techniques et les utilisateurs doivent être dépassées afin de construire une communication efficace pendant le processus de conception.

La conception participative recommande donc que les travailleurs en tant qu'utilisateurs du système final prennent part aux décisions relatives à la conception du système et à la façon dont il sera utilisé. La technologie n'étant pas développée de façon isolée, la participation aux choix technologiques implique aussi des décisions sur le contenu du travail et la conception des métiers (job design). Les usages sont inventés en même temps que le système, sur la base d'une réflexion commune entre travailleurs et ingénieurs.

Il ne suffit pas d'avoir des utilisateurs dans le processus de conception pour les faire participer efficacement à la construction du système: la mise en place et la conduite de processus et de techniques spécifiques sont primordiales pour atteindre cet objectif (Susanne Bødker et Iversen 2002). Ces techniques et processus ont été largement développés dans les années 90-2000 par le domaine de recherche de l'Interaction Homme-Machine (Nielsen 1993)(Larry L. Constantine et Lockwood 1999) (Dix et al. 2003)(Beaudouin-Lafon et Mackay 2002)(Holtzblatt, Wendell, et Wood 2004) (Norman 2014). L'utilisateur est en effet un acteur incontournable dans les systèmes interactifs, les chercheurs du domaine ont dû réfléchir à la façon de concevoir des solutions qui lui soient adaptées. Les processus proposés sont qualifiés de centré-utilisateur (Holtzblatt, Wendell, et Wood 2004), centré-usage et agile (Larry L. Constantine et Lockwood 2003) ou centré-activité (Norman 2014). Ces processus sont accompagnés de principes d'utilisabilité guidant la conception de solutions, que nous avons présentés dans notre état de l'art (partie Utilisabilité et Flexibilité). En général, nous retrouvons dans les processus proposés quatre activités principales :

- la collecte de données du contexte de l'utilisateur;
- l'analyse de ces données et la modélisation du travail des utilisateurs ;
- l'idéation et la synthèse de solutions ;
- l'évaluation par les utilisateurs des solutions proposées.

Ces activités sont menées de façon *itérative* **:** chaque activité permet d'enrichir la découverte des besoins et des exigences utilisateurs, souvent complexes et non explicitées en un seul bloc. Ainsi, la construction de solutions permet souvent de révéler de nouveaux besoins non décelés lors des activités de collecte de données et d'analyse (voir des exemples de processus en Figure 125).

Les techniques spécifiques proposées par la recherche en IHM comprennent l'observation des situations de travail, les enquêtes contextuelles, les focus groups, le prototypage papier et vidéo, les scénarios, les personas, la génération d'idées avec les utilisateurs, les design walkthrough et les tests d'utilisabilité.

Dans les années 2000-2010, les travaux de recherche en IHM se sont largement répandus, pour devenir des approches adoptées par des industriels des systèmes d'information et de télécommunication. On parle désormais de *Design thinking* chez IBM (https://www.ibm.com/design/work.shtml) ou SAP (https://designthinkingwithsap.com/en/), de *User Experience* chez Apple (https://developer.apple.com/design/tips/) et de nombreux autres industriels. Microsoft propose une inscription en ligne pour participer à la conception des produits (https://www.microsoft.com/en-us/usability). Enfin, la norme ISO 9241 partie 210 (ISO 2010) est dédiée au processus de conception centrée sur l'opérateur humain pour les systèmes interactifs(à droite de la Figure 125). Elle identifie les principes généraux caractérisant une approche centrée sur l'opérateur humain :

- *la conception est fondée sur une compréhension explicite des utilisateurs, des tâches et des environnements ;*
- *les utilisateurs sont impliqués dans la conception et le développement ;*
- *la conception est dirigée et précisée par l'évaluation centrée sur l'utilisateur ;*

- *le processus est itératif ;*
- *la conception couvre l'expérience de l'utilisateur dans son intégralité ;*

*l'équipe de conception inclut des compétences et des points de vue pluridisciplinaires.*

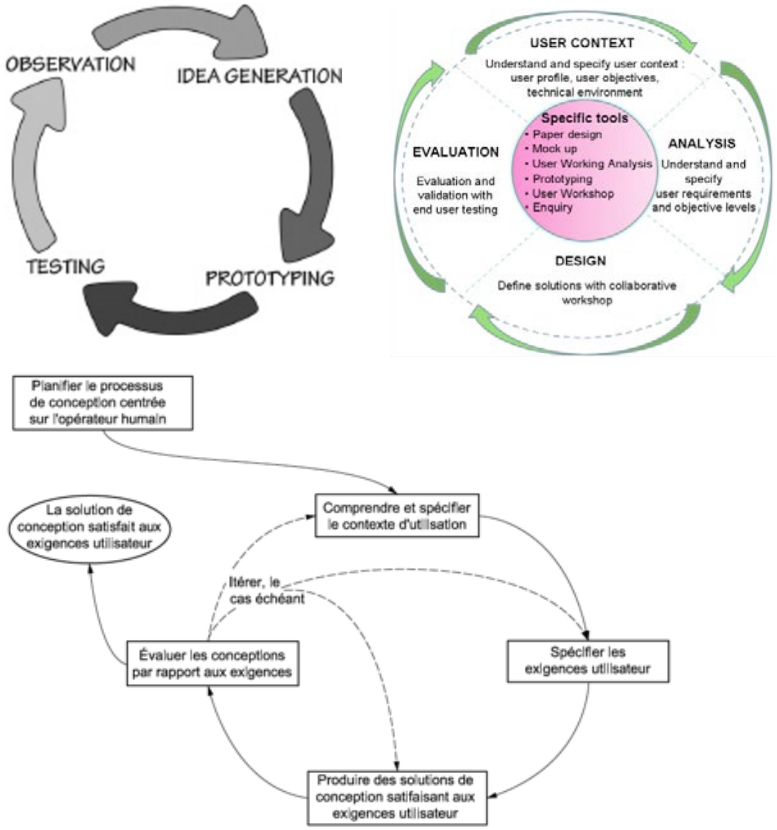



Figure 125 : des processus de conception centrée-utilisateur

### VI.1.2.2. Les utilisateurs dans l'ingénierie des systèmes complexes

Alors que les principes d'une conception participative et itérative ont été adoptés par des industriels des systèmes d'information et de communication, force est de constater qu'elle reste pour l'instant minoritaire dans le développement des systèmes complexes. Dans ce type de développement, la contractualisation est le mode de travail le plus répandu et reste incontournable tant la quantité et l'hétérogénéité des composants sont grandes (Boehm 2006). Ce sont l'acheteur et le management de l'organisation, en tant que *client*, qui dirigent la conception du système, à travers les exigences. Notre état de l'art sur les pratiques industrielles en ingénierie des exigences montre en effet que c'est le client qui est la partie prenante la plus souvent évoquée dans les problèmes de communication (partie II.2). Dans le développement de logiciels, le mouvement Agile s'est construit pour répondre à ces problèmes de communication, en favorisant la production de « logiciel de valeur pour le client *»,* plutôt que la production de documents (Schwaber et Beedle 2001)(Cockburn 2002). Le manifeste Agile (http://agilemanifesto.org/principles.html) pose 12 principes dont le premier est la « plus haute priorité donnée à la satisfaction du client à travers des livraisons rapides et continues d'un logiciel de valeur *»*. Les méthodes agiles ont été largement adoptées ces dix dernières années dans le développement logiciel (Bustard 2012).

Les méthodes Agile et Centrée-Utilisateur ont des pratiques communes, dont la plus évidente est l'itération, prônée par les deux méthodes. Dans la carte des différentes pratiques agiles

(https://www.agilealliance.org/agile101/the-agile-manifesto/), on retrouve des techniques affichant une relative prise en compte de l'utilisateur telles que les user stories, personas et usability testing (voir Figure 126).

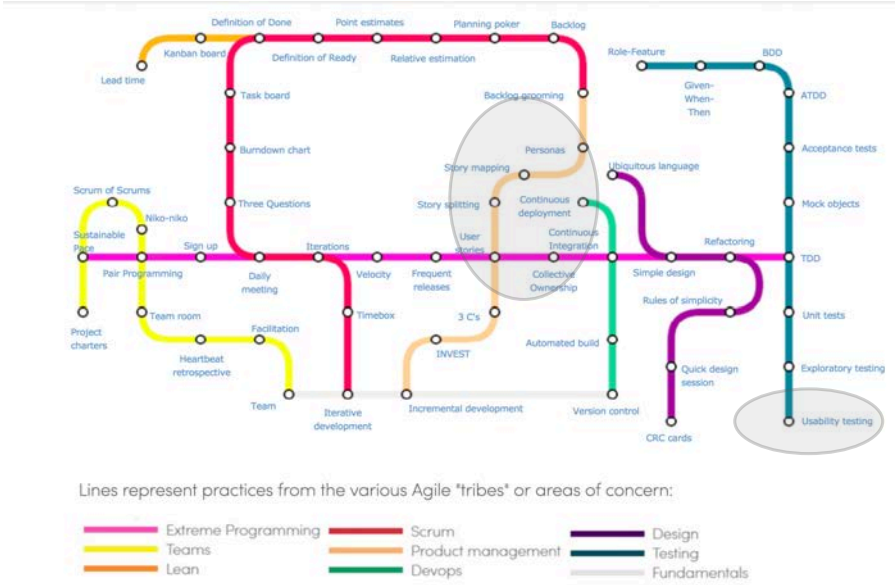



Figure 126: techniques relatives à la prise en compte des utilisateurs sur la carte des pratiques agiles

Mais ces pratiques communes restent minoritaires et l'intégration des deux processus reste difficile (Sohaib et Khan 2010)(Jurca, Hellmann, et Maurer 2014)(Salah, Paige, et Cairns 2014) : le client peut avoir une vision du produit différente de celle des utilisateurs, et la synchronisation des activités et la collaboration entre les concepteurs UX et les développeurs Agile posent problème, exigeant de gros efforts de communication et de planification. Par conséquent, nous estimons que l'intégration du mouvement agile dans l'ingénierie des systèmes complexes (Srinivasan, Dobrin, et Lundqvist 2009)(VanderLeest et Buter 2009)(Hanssen, Wedzinga, et Stuip 2017) ne permet pas de répondre à nos considérations sur les utilisateurs et l'observation du comportement du système actuel dans son environnement pour mieux informer la définition du système futur.

Dans son effort d'intégration de l'utilisabilité dans le développement industriel de logiciel, Constantine oppose le centré-utilisateur, qui serait *enraciné dans le facteur humain et l'IHM,* au centré-usage dont il promeut l'orientation d'ingénierie (L. L. Constantine et Lockwood 2003)(Larry L. Constantine 2009) (L. Constantine 2011). C'est dans cette philosophie qu'il propose une approche structurée à base de modèles.

Palanque et al. promeuvent également une approche à base de modèle de tâche et de spécification formelle pour intégrer l'utilisabilité dans l'ingénierie des systèmes complexes (Palanque, Bastide, et Paternò 1997).

Enfin, Amyot propose la notation URN (*User Requirement Notation*) : cette notation se centre sur l'expression des exigences utilisateur (objectifs ou fonctions que les utilisateurs attendent que le système réalise) mais aussi sur la description de leur raffinement en exigences système. Pour cela, la notation combine une modélisation orientée buts en langage GRL avec une modélisation de scénarios utilisant la notation *Use Case Map* (D. Amyot et al. 1999)(Daniel Amyot et Mussbacher 2000)(Daniel Amyot 2003) (voir un exemple en Figure 127).

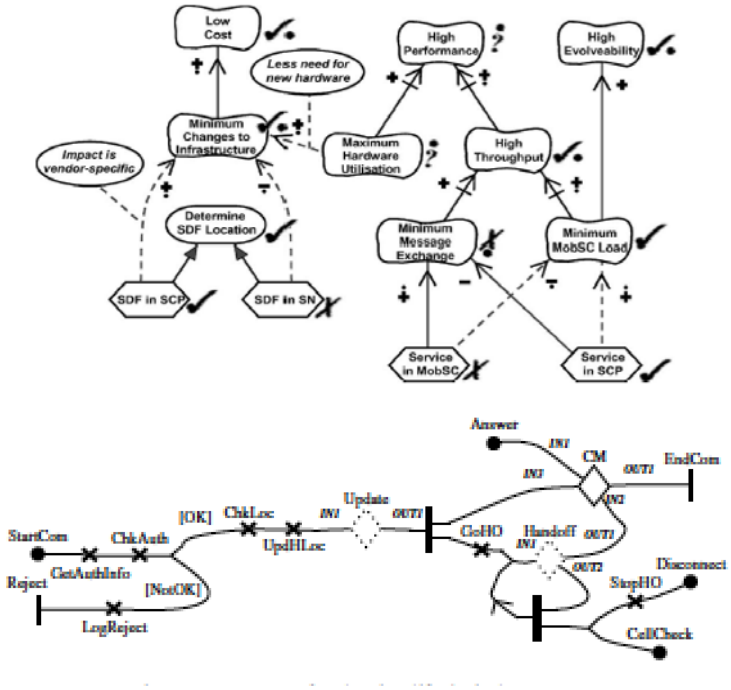



Figure 127: URN: une combinaison d'une modélisation orientée buts (en haut) avec une modélisation des scénarios en Use Case Map (en bas) (Daniel Amyot 2003)

Nous partageons leur orientation d'ingénierie et leur volonté de se centrer sur les usages plus que sur les utilisateurs. Cependant, notre état de l'art sur les pratiques industrielles (partie II.2) et les conclusions de notre étude (Chapitre IV) montrent une faible adoption des approches orientées modèles et des processus structurés en ingénierie des exigences. Par conséquent, nous recherchons une approche mixte, en conservant le focus sur les usages (plus que sur les utilisateurs) et l'orientation ingénierie, mais en évitant les formalismes de modélisation précoce, incompatibles avec la nature changeante des exigences.

### VI.1.2.3. Une prise en compte des usages dans l'ingénierie des systèmes complexes

Il existe en aéronautique des projets qui démontrent l'intérêt des approches ethnographiques pour mieux comprendre l'usage du système actuel et concevoir le système futur. Klausen et Hutchins (Klausen et Hutchins 1996) conduisent une étude ethnographique d'enregistrements audio et vidéo de cockpits dans le cadre théorique de la cognition distribuée (E. Hutchins 1995), en prenant comme unité d'analyse le cockpit, et non les pilotes. Ils mettent en évidence le rôle des représentations de l'information et de leurs mouvements dans la coordination des pilotes. Ils identifient différents chemins possibles pour l'information, certains anticipés pendant la conception, mais d'autres non prévus. Sur la base de différentes études ethnographiques, Hutchins a conçu un nouvel indicateur numérique de vitesse, avec un objectif de conserver les propriétés cognitives de l'indicateur analogique (E. Hutchins 1996). De ces différentes expériences, il conclut que la collecte et l'exploitation des « histoires d'usage » (*histories of usage*) ne constituent pas seulement une base pour assister les utilisateurs, mais des *opportunités* pour construire les systèmes et assister les futurs développements (Hollan, Hutchins, et Kirsh 2000). Ce sont les mêmes auteurs qui ont justifié les principes d'interface à manipulation directe par le cycle de l'action (E. L. Hutchins, Hollan, et Norman 1985).

Dans son étude ethnographique sur le contrôle aérien, Mackay (Mackay 1999) met en évidence la dimension collaborative du travail des contrôleurs aériens, et le rôle du strip papier dans une construction continue et partagée d'une conscience de la situation entre contrôleurs aériens. Partant

de ces résultats, plusieurs travaux de recherche ont été menés sur les interactions et la représentation graphique de strips électroniques directement manipulables sur écran tactile (Mertz, Chatty, et Vinot 2000a)(Mertz, Chatty, et Vinot 2000b) (Pavet 2001)(Garron, Journet, et Pavet 2006), ou sur l'augmentation de strips papier par des représentations numériques (Mackay et al. 1998) (Hurter et al. 2012) (Letondal et al. 2013). De ces différentes expériences, une approche *d'interaction située* est développée (Beaudouin-Lafon 2004), à partir du concept d'action située de Suchman (Suchman 1987) pour concevoir des systèmes interactifs mieux adaptés et adaptables à leur contexte d'utilisation, en dépassant la notion de tâche. Dans ce cadre, le principe d'interaction instrumentale est proposé pour étendre la manipulation directe (Beaudouin-Lafon 2000), accompagné des principes de polymorphisme et de réutilisation des commandes (Beaudouin-Lafon et Mackay 2000).

Des propositions d'utilisation de l'ethnographie et du prototypage pour informer la spécification d'un système complexe ont été faites dans le cadre d'un projet de contrôle aérien (I. Sommerville et al. 1993). Ces travaux mettent en évidence l'intérêt de méthodes ethnographiques d'observation pour capturer les besoins. Cependant, ils pointent la difficulté d'intégrer les compte-rendu d'observation des sociologues dans le processus d'ingénierie des exigences, et d'établir une correspondance entre observations et exigences sur le système.

Toujours dans le domaine du contrôle aérien, informer la spécification de systèmes complexes sociotechniques avec des modèles d'activités et des techniques de créativité a été expérimenté par Maiden et formalisé dans le processus RESCUE pour *Requirements Engineering with Scenarios for a User-centred Environment* (Maiden et Robertson 2005) (Sara Jones et Maiden 2005)(S. Jones et al. 2008) (Burnay, Horkoff, et Maiden 2016) (voir Figure 128). Ce processus a l'avantage d'intégrer les différentes activités, et d'identifier des points de synchronisation entre les activités. Cependant la modélisation de l'activité utilisée est basée sur l'approche de *Cognitive Work Analysis* de Vicente (Kim J. Vicente 1999b) (Kim J. Vicente 2000) et Diaper (Diaper et Addison 1992), ne capturant pas le travail réel des utilisateurs, mais le décomposant en séquences hiérarchisées de tâches.

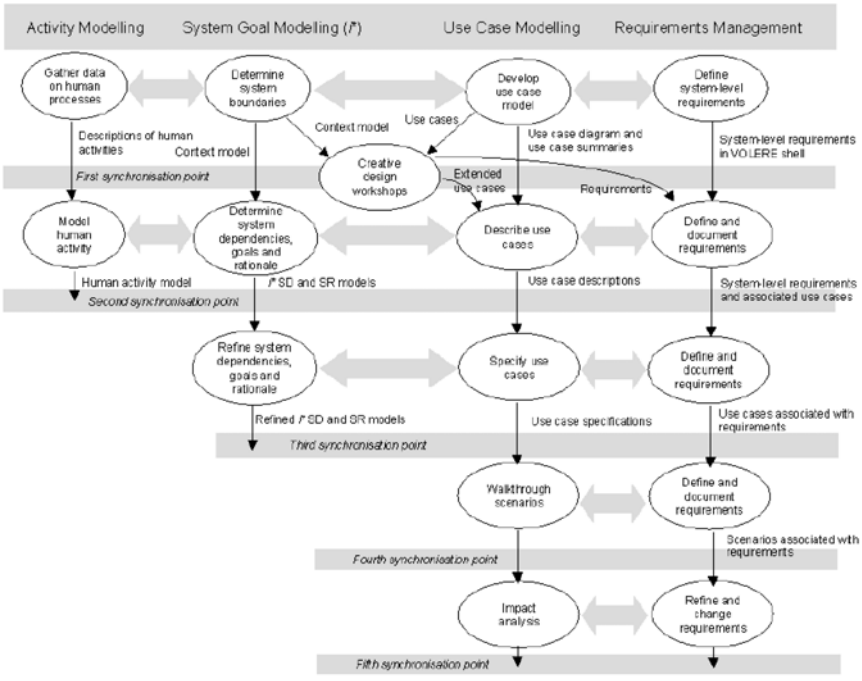



Figure 128: le processus RESCUE proposé par Maiden (Sara Jones et Maiden 2005)

Prenant comme références les deux cadres théoriques de l'action située et de la cognition distribuée, Doherty et al. (Doherty, Campos, et Harrison 2008) proposent de travailler sur la spécification des ressources pour l'action, plutôt que des tâches, pour automatiser l'analyse des comportements possibles de l'utilisateur. Cependant, ils pointent la capacité déterminante de l'analyste à identifier les situations poussant l'utilisateur à un comportement opportuniste.

Nous retenons que les méthodes ethnographiques d'observation favorisent la collecte de données sur les usages non anticipés du système actuel, au-delà des tâches prescrites. Ces données d'observation peuvent alimenter la conception du système futur, à condition qu'elles soient restituées de façon compréhensible vers les concepteurs et intégrées dans les processus d'ingénierie.

## VI.2. Une ingénierie participative des exigences

Nous proposons une nouvelle approche que nous nommons *ingénierie participative des exigences*. Elle est basée sur une articulation de techniques utilisées en conception participative pour impliquer les utilisateurs, avec un effort continu d'abstraction et de formalisation des exigences pour informer la définition du système. La finalité est une production d'exigences matures spécifiant dans le système futur la prise en compte de situations non prévues dans le système actuel, mais gérées par les utilisateurs en contexte opérationnel. L'ingénierie participative des exigences mobilise :

- **des observations et des interviews contextuelles des utilisateurs sur le système en utilisation** :

ces observations et interviews doivent avoir comme unité d'analyse le système et les ressources pour l'action mobilisées par les utilisateurs pour s'adapter à un environnement changeant. Il s'agit de capturer des instances critiques du système en utilisation (*critical instances of the typical*) (Carroll, Kellogg, et Rosson 1991)(Susanne Bødker et Iversen 2002). Nous proposons que ces observations et interviews soient réalisées par les ingénieurs en exigences, plutôt que par des sociologues, afin de réaliser les observations dans un objectif d'ingénierie et non d'étude ethnographique.

- **L'expression des exigences de haut niveau à partir des observations et interviews contextuelles** :

ces exigences de haut niveau constituent une étape intermédiaire de *coordination*, qui garantit un premier résultat tangible et utilisable dans le processus d'ingénierie des exigences. Nous avons observé qu'elles peuvent être un point de départ adéquat pour un travail collaboratif entre ingénieurs en exigences et fournisseurs pour réfléchir sur le comportement du système en définition ;

- **Pour chaque exigence de haut niveau, la rédaction de scénarios et une analyse**:

Les scénarios et leur analyse décrivent de façon détaillée les actions, les ressources pour l'action et les problèmes observés par les ingénieurs en exigences ou exprimés par les utilisateurs. Comme nous l'avons vu dans notre état de l'art, le scénario a un grand pouvoir d'expression et reste la technique la plus efficace pour impliquer les utilisateurs dans l'ingénierie des exigences. De plus, les scénarios restituent les comportements observés du système en usage aux autres ingénieurs en exigences et aux fournisseurs. L'analyse des scénarios peut permettre d'identifier des questions de conception.

- **l'idéation et le prototypage pour concevoir avec les utilisateurs des solutions de comportement du système, à partir des scénarios et des questions de conception** :

les techniques d'idéation, tel que le brainstorming, sont déjà utilisées par les ingénieurs et les utilisateurs pour l'identification des événements redoutés. Le prototypage commence à être reconnu

dans les systèmes complexes comme un moyen de travailler sur les « fonctionnalités critiques » (Kriesi et al. 2016) et les « scénarios non anticipés » (Elverum et Welo 2015). Les principes d'utilisabilité et de flexibilité doivent être mobilisés par les ingénieurs en exigences dans l'idéation et le prototypage. L'évaluation progressive des prototypes par les utilisateurs permet aux ingénieurs en exigences de gagner en confiance sur la maturité des exigences ;

- **la transformation des prototypes en exigences détaillées, pour une prise en compte des résultats dans le processus d'ingénierie des exigences** :

cette transformation est accompagnée d'un maintien de la correspondance entre scénario, exigences de haut niveau et exigences détaillées, afin de justifier les exigences détaillées et montrer leur contribution, au-delà de l'utilisabilité.

Dans la suite de ce chapitre, nous illustrons l'application de notre approche sur différents cas de projets de systèmes interactifs en aéronautique.

# VI.3. Cas illustrés de l'ingénierie participative des exigences

Dans un objectif d'illustration, nous présentons les résultats obtenus par l'application de l'ingénierie participative des exigences sur quatre cas :

- Contrôle aérien et collaboration ;
- Avion électrique et gestion de l'énergie ;
- Analyses d'incidents et d'accident : cas du rapport d'accident du vol 447 Rio-Paris
- Nouvel instrument de vol.

Pour chaque cas, nous présentons son cadre et sa significativité, suivis d'une description du déroulement du processus d'ingénierie des exigences. Nous décrivons notamment la contribution des différentes parties prenantes dans les activités suivantes :

- l'observation de situations non prévues dans le système actuel mais gérées par les utilisateurs actuellement ;
- l'expression des exigences de haut niveau sur le système pour la gestion des situations non prévues ;
- la génération et évaluation de solutions répondant aux situations non prévues ;
- l'expression des exigences détaillées pour la gestion des situations non prévues.

Suite à la description du processus, nous réalisons une synthèse des résultats et identifions les bénéfices et les obstacles à la validité de l'ingénierie participative des exigences.

## VI.3.1. Contrôle aérien et collaboration

### VI.3.1.1. Cadre et significativité du cas

L'objectif du contrôle du trafic aérien (ATC) est de maximiser à la fois la sécurité des vols et la capacité : il s'agit de faire passer le plus de vols possibles sans compromettre leur sécurité et sans créer de retard. Le trafic aérien étant en forte croissance, les autorités en Europe et aux Etats-Unis ont entrepris le financement de grands programmes, SESAR et NextGen (Erzberger 2004), pour la définition et la mise en œuvre avec les industriels de nouveaux systèmes de contrôle aérien. Les objectifs sont un gain en capacité obtenu par une automatisation des tâches de surveillance et de séparation du trafic, réduisant la charge de travail des contrôleurs aériens et maintenant le niveau de sécurité. La précision et l'efficacité de la séparation automatique dépend d'une connaissance par le

système des trajectoires prévues et modifiées. Or, le système actuel, basé sur des strips papier et des communications vocales entre pilotes et contrôleurs (voir photo en Figure 129) n'est pas systématiquement mis à jour avec les modifications et instructions données par les contrôleurs aux pilotes à la fréquence radio. Par conséquent, le strip papier représente un frein à l'automatisation, que de nombreux projets proposent de lever en le remplaçant par des outils électroniques.

Figure 129: une position de contrôle aérien en France, basée sur une image radar et un tableau de strips papier

Dans ce contexte, le projet MAMMI, financé par Eurocontrol, se base sur l'opportunité offerte par de nouveaux grands écrans multi-touch et multi-utilisateurs. Le projet implique l'ENAC pour l'expertise métier et IHM, Intuilab pour la conception et le développement d'IHM, Intactile Design pour le design graphique. Nous décrivons le cas étudié dans notre vision située de l'ingénierie des exigences en Figure 130.

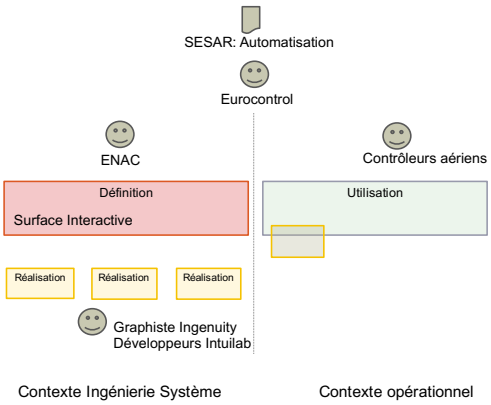



Figure 130: le cadre du cas MAMMI représenté dans notre vision située de l'ingénierie des exigences

Nous avons vu que le contrôle aérien était un domaine d'application très utilisé dans les approches ethnographiques d'observations du système actuel pour informer la construction du système futur (Hughes, Randall, et Shapiro 1992)(I. Sommerville et al. 1993)(Mackay 1999)(Sara Jones et Maiden 2005). Travaillant pour la Direction des Services de la Navigation Aérienne puis pour l'ENAC, école de formation des contrôleurs aériens français, ce domaine nous est familier et facilement accessible. De plus nous avons travaillé sur la collaboration entre contrôleurs aérien durant un précédent travail de recherche (Uninski 1998)(Sire et al. 1999). Par conséquent, ce projet présente des conditions favorables à l'ingénierie participative des exigences, et constitue une opportunité pour consolider sa mise en œuvre.

### VI.3.1.2. Description du processus

*Observation de situations non prévues dans le système actuel mais gérées par les utilisateurs*

Des études ethnographiques (Hughes, Randall, et Shapiro 1992)(Mackay 1999) (Rognin, Salembier, et Zouinar 2000) ont mis en évidence la dimension collaborative du travail des contrôleurs et le rôle du strip papier dans une conscience partagée de la situation. Dans ses observations, MacKay pointe notamment l'intervention de contrôleurs supplémentaires sur une position de contrôle en cas de pics de trafic ou de situations non prévues, à partir d'une surveillance continue de la situation réalisée par les contrôleurs présents dans la salle :

*only the necessary number of controllers need work on a position at a time, with extra hands instantly available as needed*.

Les exemples de situations non prévues sont les orages et la fermeture d'une zone par les militaires, entrainant un reroutage de tous les vols impactés. Dans le cadre de MAMMI, nous avons plus particulièrement étudié ces situations de collaboration impliquant plus de deux utilisateurs, en complétant les analyses ethnographiques existantes, par des interviews hors contexte de 6 contrôleurs aériens et 5 experts du contrôle aérien (nous n'avions en effet pas accès à une salle de contrôle). Ainsi, l'ingénieur en exigences, peut, sans avoir accès à des observations et des interviews contextuelles du système actuel en contexte opérationnel, récupérer des connaissances à partir des études ethnographiques réalisées, et les injecter dans le processus d'ingénierie des exigences. Il ressort des interviews que les situations pendant lesquelles les contrôleurs aériens sont plus de deux sur une position de contrôle représentent pratiquement la moitié du temps, entre les situations d'instruction et les situations de forte charge telles que les orages. Ces situations ne sont pas prévues de façon explicite dans le système actuel, mais elles sont gérées par les contrôleurs aériens.

*Expression des exigences de haut niveau sur le système pour la gestion des situations non prévues*

Dans le système actuel, l'accès partagé au tableau de strips et à l'image radar permet une allocation dynamique des tâches entre les différents contrôleurs, ainsi qu'une vérification continue des actions des autres. Nous avons extrait des études ethnographiques antérieures des exigences, avec une reformulation utilisant les règles syntaxiques de la norme ISO 29148 pour l'expression des exigences (ISO/IEC/IEEE 2011). Les exigences suivantes sont formulées :

- le système doit être renseigné avec les ordres de contrôleurs ;
- le système permet à plus de deux utilisateurs d'interagir simultanément ;
- le système favorise la conscience mutuelle ;
- le système favorise la communication et la coordination ;
- le système favorise l'allocation dynamique de tâches.

*Génération et évaluation de solutions répondant aux situations non prévues*

Pour répondre à ces exigences, nous avons réfléchi au dispositif physique, basé sur une une surface interactive multi-touch et multi-utilisateurs, et aux objets graphiques interactifs sur cette surface comme *ressources pour l'action* (voir Figure 131). La position est composée de deux écrans radar verticaux, présentant une vue de référence du trafic aérien. Chaque image radar peut être configurée différemment (zoom, centrage, marqueurs nautiques) selon les besoins du contrôleur. Une surface horizontale est placée sous les écrans radar ; elle centralise les moyens d'interaction sur la position de contrôle.

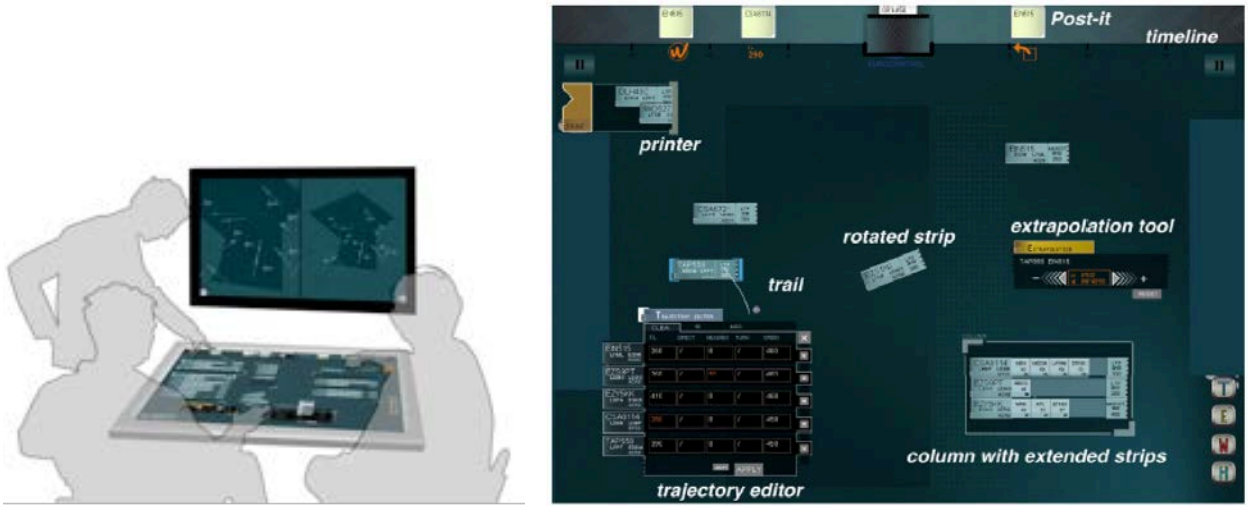



Figure 131: dispositif physique et objets graphiques interactifs comme ressources pour l'action pour plus de 2 contrôleurs

Au-delà des principes généraux d'utilisabilité (voir partie II.3.1), nous avons exploité les principes de collaboration directe (Sire et al. 1999) (Sire 2000) pendant des ateliers de conception impliquant des contrôleurs aériens et des concepteurs IHM :

- une surface partagée interactive ;
- la réification des actions dans des instruments ;
- l'accomplissement partiel des actions : une action peut être séparément préparée, vérifiée et accomplie, potentiellement par différents utilisateurs.
- un retour d'information continu pour une visibilité des actions.

Notre hypothèse est que les principes de collaboration directe contribuent à générer des solutions pour atteindre les exigences selon la matrice présentée en Figure 132.

| Design Principles → / Requirements ↓ | P1: multitouch surface | P2: reify actions into instruments | P3: partial accomplishment of actions | P4: provide feedback |
|---|---|---|---|---|
| R1: update the system | x | x | x | |
| R2: more than two users | x | x | | x |
| R3: foster mutual awareness | x | x | x | x |
| R4: foster communication and coordination | | x | x | x |
| R5: foster dynamic task allocation | | x | x | |

| Design Principles → / Interactions ↓ | P1: multitouch surface | P2: reify actions into instruments | P3: partial accomplishment of actions | P4: provide feedback |
|---|---|---|---|---|
| I1: desktop, free layout | x | x | x | |
| I2: orientable strips | x | | | |
| I3: trajectory editor | | x | x | x |
| I4: trajectory extrapolation viz | | x | | x |
| I5: post-its, and timeline | | x | x | x |
| I6: audio annotations | | x | x | |
| I7: feedback and feedthrough | | | | x |

Figure 132: hypothèse de contribution des principes de conception aux exigences (à gauche) et mise en œuvre des principes sur les interactions (à droite)

Nous avons utilisé des prototypes papier pour imaginer des interactions multi-touch sur les objets, en support à des séances d'idéation avec des contrôleurs aériens (voir exemples en Figure 133). Des designers graphiques ont ensuite travaillé sur les représentations.

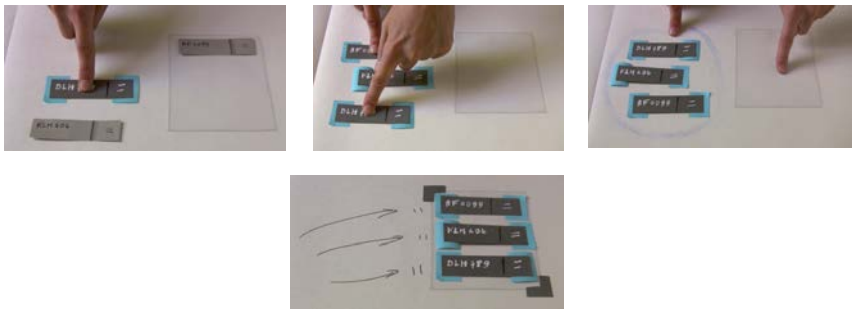


Figure 133: exploration de trois façons d'interagir avec les objets (en haut) pour créer un groupe dans une colonne (en bas)

Ce travail a abouti à des représentations et des interactions avancées (voir Figure 131 et Figure 132) telles que la métaphore du bureau et la disposition libre des outils, les strips orientables, l'éditeur de trajectoire, la visualisation de la trajectoire extrapolée, les annotations audio, et les rétroactions et trans-actions (*feedthrough*). Ces éléments ont été implémentés par des développeurs spécialisés en interaction-homme machine, sur la base des prototypes et de leur participation aux séances de conception : le passage au développement logiciel ne s'est accompagné d'aucune formalisation sous forme d'exigences. Nous avons conduit une évaluation préliminaire avec 8 participants, 3 contrôleurs aériens et 5 experts du contrôle aérien. Le protocole et les résultats sont détaillées dans notre article (Conversy et al. 2011). Nous pouvons résumer les résultats d'évaluation de la façon suivante : les principes de surface partagée et de réification des actions dans les instruments contribuent à un renseignement du système (R1), avec plus de deux utilisateurs (R2), à la communication et coordination (R4), et l'allocation dynamique de tâches (R5). Le principe de l'accomplissement partiel des actions contribue à une allocation dynamique des tâches et à un renseignement du système : un contrôleur peut préparer une modification de trajectoire dans l'éditeur de trajectoire suite à une coordination téléphonique, pour que l'autre contrôleur valide la modification au moment de la communication avec le pilote.

*Expression des exigences détaillées pour la gestion des situations non prévues*

En termes de spécification, nous en sommes restés aux exigences de haut niveau et aux deux matrices présentées en Figure 132. Nous n'avons pas fait l'effort d'affiner les exigences textuellement à partir des propositions de représentations et d'interaction : le projet MAMMI était un projet de recherche, sans objectif d'industrialisation, une spécification plus précise n'était attendue par aucune partie prenante. C'est un élément moteur du processus d'ingénierie des exigences : on ne rédige les exigences que s'il y a une partie prenante (les fournisseurs) qui les utilise.

### VI.3.1.3. Synthèse des résultats et obstacles à la validité

Nous pouvons résumer les résultats du processus d'ingénierie des exigences du cas MAMMI dans le Tableau 9.

| **Activités** | **Résultats** | **Moyens de l'ingénierie participative des exigences mis en œuvre** |
|---|---|---|
| Observation de situations non prévues dans le système actuel mais gérées par les utilisateurs | Scénarios de situations d'instruction et d'orage | Utilisation d'études ethnographiques antérieures, consolidées par interviews hors contexte de contrôleurs aériens. Centrage sur les actions et les ressources des actions (strips,, image radar, tableau de strips, actions sur les strips) |
| Expression des exigences de haut niveau sur le système pour la gestion des situations non prévues | Exigences R1 à R5 | Transformation des recommandations des études ethnographiques en exigences selon la norme ISO 29148 |
| Génération et évaluation de solutions répondant aux situations non prévues | Conversy, Stéphane, Hélène Gaspard-Boulinc, Stéphane Chatty, Stéphane Valès, Carole Dupré, et Claire Ollagnon. « Supporting Air Traffic Control Collaboration with a TableTop System ». In Proceedings of the ACM 2011 Conference on Computer Supported Cooperative Work, 425-34. CSCW '11. New York, NY, USA: ACM, 2011. | principes de collaboration directe, idéation et prototypage papier avec des contrôleurs aériens et des développeurs IHM<br>Design walkthroughs<br><br>Evaluation semi-contrôlée sur la base d'une plate-forme légère de simulation |
| Expression des exigences détaillées pour la gestion des situations non prévues | Matrices entre exigences, principes de conception et interactions<br><br>Absence d'exigences détaillées | Pas d'objectif d'industrialisation<br>Présence des développeurs aux séances d'idéation et de prototypage |

Tableau 9 : synthèse des résultats du cas MAMMI

En termes d'effort, nous pouvons dire que les observations des situations non prévues et l'expression des exigences de haut niveau ont été relativement aisées : nous sommes partis d'études ethnographiques fournies, et nous avions une bonne connaissance de la problématique de la collaboration et du contrôle aérien.

Ce sont la génération et le développement de solutions qui ont été les plus coûteuses, mobilisant une graphiste, des concepteurs en IHM et une équipe de développeurs en IHM, et nécessitant l'intégration d'un simulateur de trafic pour la conduite de l'évaluation. Le coût de conception et de développement peuvent s'expliquer par le soin apporté aux graphismes et l'usage d'une technologie innovante pour la surface interactive multi-touch (en 2008 date de début du projet). Les parties prenantes du projet sont homogènes : elles partagent les mêmes connaissances et considérations en IHM, ne nécessitant pas un effort de coordination via des exigences. Ce partage offre un environnement propice à une conception participative intégrée plus qu'à une ingénierie participative des exigences.

## VI.3.2. Avion électrique et gestion de l'énergie

### VI.3.2.1. Cadre et significativité du cas

Entre 2014 et 2016, nous avons conduit le processus d'ingénierie des exigences du cockpit de l'avion-école tout électrique E-Fan 2.0, comme évoqué préalablement en partie V.3.1. La demande initiale définit la mission de l'avion-école électrique de la façon suivante.

| La mission type que doit pouvoir réaliser l'avion est un vol « formation initiale » (50 mn en vol, 60 mn « bloc-bloc » avec 70 mn d'autonomie) qui serait composé comme suit : |
|---|
| - 5 mn de roulage au sol, sur piste ou taxiway en herbe,<br>- 5 mn de montée initiale + 5 mn de transition en croisière vers un secteur de travail hors circuit d'aérodrome compris entre 1500 et 2500 pieds d'altitude par rapport à l'aérodrome,<br>- 20 mn de travail en secteur à des vitesses comprises entre la vitesse d'attente et la vitesse de croisière,<br>- 8 mn de retour du secteur vers le circuit d'aérodrome à vitesse de croisière en descente vers 1000 ft/aérodrome,<br>- 2 tours de piste (un final + un en secours en cas de remise de gaz) d'environ 6 mn chacun,<br>- Atterrissage avec 15 mn de réserve finale minimum,<br>- 5 mn de roulage retour au parking (consommation pouvant être prélevée sur les 15mn de réserve finale). |

Nous résumons le cadre du cas E-Fan 2.0 dans la Figure 134.

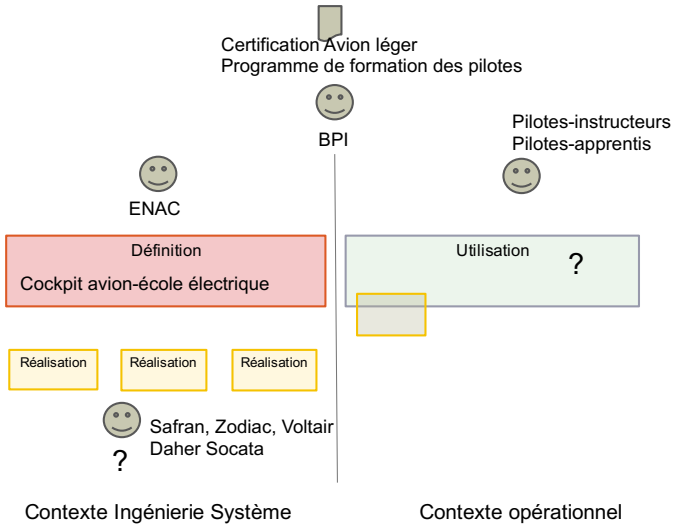



Figure 134: le cadre du cas E-Fan 2.0 dans notre vision située de l'ingénierie des exigences

Le projet E-Fan 2.0 constitue un cas significatif pour évaluer l'ingénierie participative des exigences à double titre. Tout d'abord, c'est un projet avec un objectif d'industrialisation impliquant des entreprises de pointe (Airbus, Safran, Zodiac, Daher Socata) : l'ingénierie participative des exigences produit-elle des résultats acceptables et utilisables par des partenaires industriels rompus à l'ingénierie des systèmes ? De plus, le problème à résoudre n'est pas lié à l'automatisation d'un système existant, mais à la construction d'un nouveau système intégrant une innovation technologique : quelles observations dans un contexte opérationnel peut-on conduire pour mieux informer la définition du nouveau système ? les utilisateurs peuvent-ils vraiment participer sans être un prétendu « frein à l'innovation » ?

### VI.3.2.2. Description du processus

*Observation de situations non prévues dans le système actuel mais gérées par les utilisateurs*

Les utilisateurs principaux du cockpit E-Fan 2.0 sont d'abord les pilotes instructeurs puis les pilotes apprentis. Nous avons plus particulièrement travaillé avec les pilotes instructeurs du centre de formation ENAC de Carcassonne, deux anciens pilotes apprentis, et un pilote instructeur expert. Nous avons mené une séance d'observation en vol d'instruction, des interviews contextuelles au sol, au centre de Carcassonne, et des séances d'observation de briefing de pilotes apprentis par un pilote instructeur (voir photos illustratives en Figure 135).

Figure 135: photos d'observations et d'interviews contextuelles au centre ENAC de Carcassonne

Nos observations et interviews se sont centrées sur le déroulement d'une séance d'instruction et les ressources utilisées, avec une attention particulière sur la gestion du carburant. Actuellement, les pilotes décollent avec presque deux à trois fois la quantité minimale requise de carburant, ce qui conduit à avoir une marge de carburant d'une à deux heures pour une séance d'instruction. Une réglementation ENAC précise que la réserve finale d'un avion thermique (type TB10) en fin de séance doit être de 45 minutes. Cette réserve est une garantie de sécurité en cas d'impossibilité d'atterrir sur l'aérodrome de départ, permettant de joindre un aéroport de dégagement ou de repérer et atterrir sur une aire dégagée en campagne. A l'annonce d'une autonomie de vol entre 1h et 1h15 avec l'avion électrique E-Fan 2.0, les réactions des pilotes étaient : « je ne monte pas dans un avion qui n'a qu'une heure d'autonomie avec un élève », « avec une réserve de 45 minutes, cela laisse pas beaucoup de temps pour voler », « si on se retrouve dans le tour de piste avec un Ryanair, comment on va faire ?». Plusieurs scénarios ont été rédigés, décrivant le déroulement d'un tour de piste, la mise en attente pour un trafic commercial, la gestion du carburant et du temps, le briefing avant le vol (avec le calcul du carburant à emporter selon le poids des personnes).

Les conclusions étaient qu'il fallait (1) lever le point bloquant de la faible autonomie (2) réviser la réglementation sur la réserve finale pour un avion électrique. Ainsi, l'ingénieur en exigences ne fait pas que subir la réglementation, il peut également participer à son évolution pour accompagner une nouvelle technologie.

*Génération et évaluation de solutions répondant aux situations non prévues*

Dans une première phase, nous nous sommes concentrés sur les représentations à fournir au pilote pendant le vol, sans formaliser d'exigences. Au cours de séances d'idéation avec des pilotes instructeurs et jeunes pilotes, le principe de réification des actions dans les instruments a contribué à générer l'idée d'une jauge énergie, dans laquelle les blocs de consommation correspondent aux activités opérationnelles d'un vol d'instruction : montée, vol vers une zone d'exercice, exercices, approche et descente vers le terrain (voir schéma de principe en Figure 136).

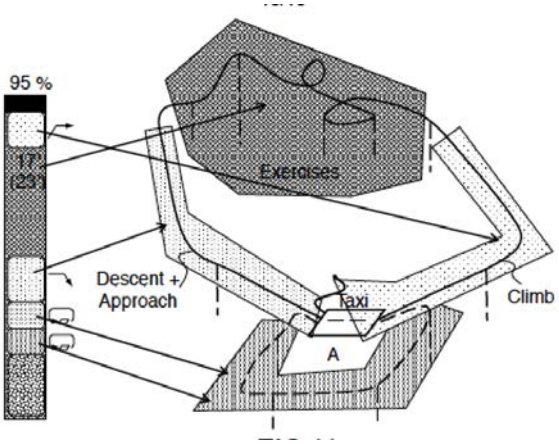



Figure 136: schéma de principe de la jauge énergie, extrait du brevet déposé

Le prototypage rapide de cette jauge associée à une carte, dans un simulateur de vol, a permis de montrer la dynamique aux pilotes et de valider l'intérêt du concept.

> *Expression des exigences de haut niveau sur le système pour la gestion des situations non prévues*

Nous avons alors pu formaliser des exigences sur la gestion de l'énergie, justifiées par nos interviews, et sur la base du concept de la jauge et de sa dynamique (voir Figure 137).

| | |
|---|---|
| **10. The system must enable the management of energy** | |
| *10.1.* | *During flight preparation* |
| Notes | *Users Interview & UC 1 - Must reach 8% of total flight time ENDURANCE precision.* |
| *10.1.1.* | *Taking into account planned flight profile* |
| Notes | *Trajectory and altitude planned.* |
| *10.1.2.* | *Taking into account planned energy profile* |
| Notes | *Selected economical mode.* |
| *10.1.3.* | *Taking into account the environment* |
| Notes | *Wind, temperature, load...* |
| *10.2.* | *In flight* |
| Notes | *Users Interview & UC 3 - Must reach 8% of remaining flight time ENDURANCE precision* |
| *10.2.1.* | *Enable to change planned flight profile* |
| Notes | *Trajectory and altitude planned* |
| *10.2.2.* | |
| Notes | *Selected economical mode.* |
| *10.2.3.* | *Taking into account the environment* |
| Notes | *Wind, temperature, load...* |
| *10.2.4.* | *Express endurance in terms of operational capabilities* |
| Notes | *Indicate remaining flight time endurance at selected economical mode and at max range presetting.* |
| *10.3.* | *In case of diversion* |
| Notes | *Users Interview & UC 2 - Must reach 8% of remaining RANGE precision at selected economical mode and at max range presetting.* |
| *10.3.1.* | *Taking into account the environment* |
| Notes | *Wind, temperature, load...* |

Figure 137: exigences de haut niveau sur la gestion de l'énergie

*Expression des exigences détaillées pour la gestion des situations non prévues*

Le concept de jauge et d'activités opérationnelles a fait l'objet d'un brevet avec un élargissement du domaine d'application à tout véhicule électrique (Pujos et al. 2016). La rédaction du brevet a poussé à l'expression d'exigences détaillées, qui n'étaient pas forcément nécessaires à ce stade du projet. Dans notre démarche d'illustration de l'ingénierie participative des exigences, ce fait contribue à montrer son potentiel innovant : les utilisateurs, en situation de conception, ne constituent pas un frein mais une source d'innovation, à condition que les ingénieurs en exigences mobilisent les techniques appropriées : prototypage, techniques de créativité et principes de conception. La nouveauté de l'activité à instrumenter a nécessité une inventivité plus grande pour appliquer le principe de réification des instruments (ici les activités opérationnelles dans une jauge). L'analogie avec des activités existantes a été introduite par les ingénieurs en exigences comme technique de créativité, afin de surmonter cette difficulté (gestion de micro-drones et conduite d'une voiture électrique). L'expression des exigences détaillées dans le brevet s'est largement appuyée sur le prototype.

*Génération et évaluation de solutions répondant aux situations non prévues*

Une fois le concept découvert et validé, nous avons voulu affiner les exigences, en étendant le concept aux scénarios de préparation de la séance d'instruction (briefing), et d'évaluation de la séance d'instruction (débriefing). Pour cela, nous avons utilisé le brevet comme base de spécification : nous l'avons fourni à une équipe de quatre étudiants du Master IHM, et leur avons demandé de concevoir un prototype pour une utilisation sur iPad. En effet, lors de nos observations, nous avons relevé que l'iPad est largement utilisé par les pilotes instructeurs pendant le briefing des pilotes apprentis (voir photo en haut de la Figure 135) et pendant la séance d'instruction en vol, avec un dispositif permettant de fixer l'iPad autour de la cuisse. Pour la construction de ce prototype (voir Figure 138), nous avons travaillé avec les utilisateurs en utilisant les techniques de prototypage et d'idéation. Nous avons mobilisé les principes de collaboration directe de la façon suivante :

- une surface interactive partagée entre instructeur et apprenti : la tablette peut servir de support en briefing pour expliquer le déroulement du vol ;
- un accomplissement partiel des actions : l'instructeur peut préparer une modification de la séance et évaluer l'impact sur la consommation énergétique sur la tablette, puis la valider pour une transmission vers le système bord ;
- un retour continu pour une visibilité des actions : un calcul continu de l'énergie est présenté dans la jauge à partir d'un déplacement d'un point sur la carte.

Figure 138: du prototype papier au design walkthrough d'un prototype logiciel pour une utilisation sur tablette

Nous avons conduit des design walkthroughs de ce prototype logiciel avec un des pilotes-concepteurs de la première phase et quatre instructeurs pilotes. Les retours du pilote-concepteur ont été positifs, et ont conduit à une interaction d'ajout d'exercice sur un point de la carte. Les retours des quatre instructeurs-pilotes étaient plus hétérogènes : nous n'avions pas intégré des principes d'utilisabilités

basiques, tel que le retour d'information sur une sélection, et ces points étaient relevés par certains. C'est un nouveau point de vigilance sur la façon de susciter des retours des utilisateurs en ingénierie participative des exigences. Cependant, en replaçant les utilisateurs dans un contexte d'utilisation du prototype pour préparer un vol, ils ont soulevé une question cruciale : comment évaluer l'impact du vent sur la consommation énergétique, qui est d'autant plus dimensionnant que les avions sont légers ? De même que les ingénieurs en exigences et les fournisseurs utilisent des dessins de comportement du système pour lever des questions de conception puis formuler des exigences sur les composants (voir partie Conception exploratoire et collaborative sans outil spécifique), les prototypes interactifs sont un moyen de découvrir des questions pour les ingénieurs en exigences et les utilisateurs. Nous avons mené une séance d'idéation sur cette problématique, en mobilisant les principes de collaboration directe : le résultat est la réification du vent dans un instrument, disponible depuis la carte (voir Figure 139).

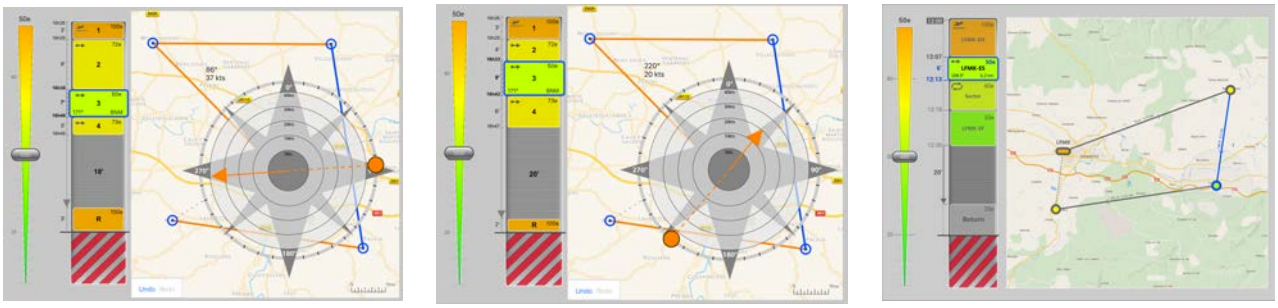


Figure 139: la réification du vent dans un instrument : la force et la direction du vent sont réglés par l'utilisateur en manipulant le vecteur, l'impact sur la consommation est présenté de façon continue dans la jauge

> *Expression des exigences détaillées pour la gestion des situations non prévues*

Ce dispositif fait l'objet d'un deuxième brevet (Hélène Gaspard-Boulinc, Conversy, et al. 2016), dont les domaines d'application sont étendus, par analogie, à la gestion d'énergie d'appareils électriques mobiles et au management de projet. Le dépôt du deuxième brevet montre l'efficacité des principes de collaboration directe combinées aux techniques de prototypage et d'idéation pour générer des dispositifs innovants, à partir d'une question issue d'évaluation avec les utilisateurs. L'intégration de l'instrument du vent s'est accompagnée d'une conception détaillée de l'interface sur les représentations, que nous avons menée en collaboration avec un designer graphiste, en utilisant le cadre des variables visuelles (Bertin 1983) pour le codage des informations (à droite de la Figure 139). Enfin, nous avons exploité la définition de contexte de notre cadre d'utilisabilité (voir Figure 37) pour structurer la spécification par différenciation des contextes (voir Figure 140) :

- au sol, avant le vol, à l'extérieur de l'avion ;
- au sol, avant le vol, à l'intérieur de l'avion ;
- en vol, à l'intérieur de l'avion ;
- au sol, après le vol, à l'intérieur de l'avion.

La combinaison des conditions de contexte pourrait être générative pour la définition du système : on peut imaginer des services du système pour un instructeur qui serait au sol alors que son élève réalise un vol « en solo ».

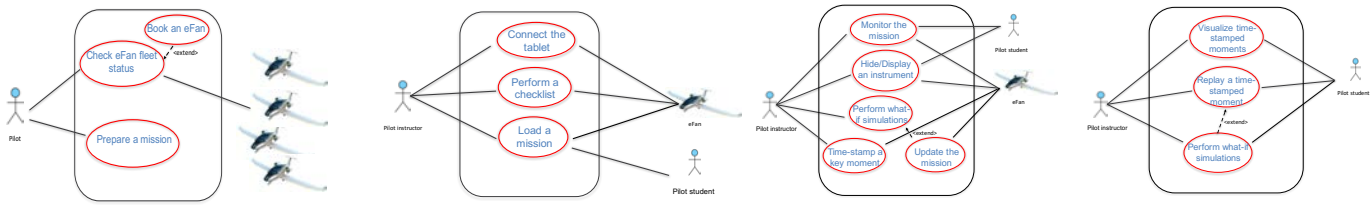



Figure 140: les cas d'utilisation structurés par contexte : avant le vol, pendant le vol et après le vol

Nous avons rédigé la spécification de chaque cas d'utilisation en traduisant les interactions conçues et évaluées avec les utilisateurs par des interactions entre acteur et système. Nous avons utilisé principalement le texte pour exprimer les cas d'utilisation sous forme d'exigences. Pour les cas d'utilisation ayant un impact sur plusieurs composants de l'avion, nous avons utilisé les diagrammes de séquence de messages, formalisme utilisé par les autres partenaires du projet : notre choix de l'expression des exigences est dirigé par l'utilisabilité de la spécification. Notre spécification a entrainé des discussions constructives avec l'équipe d'ingénieurs en charge de *l'Electric Power Unit* (EPU), sur les protocoles d'échanges de données et l'architecture à mettre en place. Le système de gestion de l'énergie sur iPad a également suscité des réactions de la part de l'ingénieur responsable de la certification de l'aéronef : l'introduction d'un nouvel équipement, non approuvé par des certifications antérieures, allonge la durée du projet et recule la date de mise sur le marché de l'avion. Cela nous a conduit à un compromis : proposer un cockpit avec des équipements basiques pour répondre aux exigences de certification, complétés par les jauges et une tablette instructeur (voir Figure 141). La spécification a été présentée aux partenaires du projet lors d'un comité plénier en mars 2016, et livrée à la Banque Public d'Investissement en juin 2016.

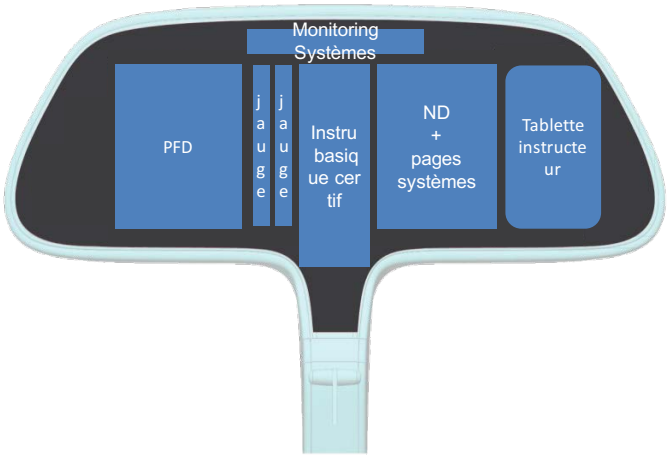



Figure 141: proposition de cockpit, avec les instruments basiques pour la certification, les jauges et la tablette instructeur

### VI.3.2.3. Synthèse des résultats et obstacles à la validité

Le Tableau 10 résume les résultats du processus d'ingénierie des exigences du projet E-Fan 2.0.

| Activités | Résultats | Moyens de l'ingénierie participative des exigences mis en œuvre |
|---|---|---|
| Observation de situations non prévues dans le système actuel mais gérées par les utilisateurs | Situations de gestion du carburant, de préparation de la séance d'instruction, de tour de piste avec un trafic commercial | Observation et interview de pilotes instructeurs en contexte centrées sur les actions et ressources des actions (carte, plan de vol, météo, plan de formation,montre) |
| Expression des exigences de haut niveau sur le système pour la gestion des situations non prévues | Exigences présentées en Figure 137 | Restitution des situations sous forme de scénarios<br>Idéation et prototypage papier et logiciel avec des pilotes instructeurs et jeunes pilotes<br>Techniques de créativité (analogie)<br>Principes de collaboration directe<br>Transformation des prototypes en exigences de haut niveau. |
| Génération et évaluation de solutions répondant aux situations non prévues | Pujos, Matthieu, Raïlane Benhacene, Hélène Gaspard-Boulinc, Denis Louviot, et Jean-Luc Vinot. Energy management system for vehicles. US 15/178,762, issued juin 2016. http://www.google.com/patents/US20160363456<br><br>Gaspard-Boulinc, Hélène, Stéphane Conversy, Mickaël Loubriat, Alexandre Duchevet, Clement Dupont, M. Riedinger, Matthieu Pujos, Raïlane Benhacene, et Denis Louviot. Activity Based Resource Management System, issued 15 décembre 2016. https://hal-enac.archives-ouvertes.fr/hal-01469780. | principes de collaboration directe<br>idéation et prototypage papier et logiciel avec des pilotes instructeurs<br><br>Variables visuelles<br>Principes de la visualisation d'information |
| Expression des exigences détaillées pour la gestion des situations non prévues | Discussions avec les équipes EPU et certification sur la faisabilité technique des exigences détaillées<br><br>Cas d'utilisation et diagrammes de séquence de messages intégrés dans la spécification livrée à la BPI. | Transformation des prototypes en cas d'utilisation et exigences détaillées<br>Structuration par contexte d'utilisation<br>Choix d' expression des exigences selon son utilisabilité |

Tableau 10: synthèse des résultats du processus d'ingénierie des exigences du cas Efan 2.0

L'observation de situations non prévues dans le système actuel mais gérées par les utilisateurs a mis en perspective les réticences exprimées par les utilisateurs face à un changement pour les transformer en opportunités d'innovation. Cela est passé par une mobilisation accrue de techniques d'idéation et de prototypage par les ingénieurs en exigences, avec des temps de développement très courts. Il s'agit également de toujours replacer les utilisateurs en contexte pendant les design walkthrough, afin que les problèmes basiques d'utilisabilité d'un prototype ne brouillent la nature des discussions et ne bloquent la génération d'idées. Les résultats du processus d'ingénierie des exigences montrent une capacité à générer une spécification structurée et utilisable par les partenaires du projet travaillant sur des composants du système.

## VI.3.3. Analyses d'incidents et d'accident : cas du rapport d'accident du vol 447 Rio-Paris

### VI.3.3.1. Cadre et significativité du cas

Une des faiblesses de l'ingénierie participative des exigences, telle que définie actuellement, provient de la difficulté de l'ingénieur en exigences d'observer les situations opérationnelles en contexte. La

difficulté ne vient pas du coût, car il n'est pas significatif par rapport aux sommes engagées sur des grands projets de définition de systèmes complexes, mais de l'accès au contexte et du côté non prévisible des observations. En ce qui concerne l'accès au contexte, nous avons vu dans le projet MAMMI qu'il était parfois impossible : nous avons utilisé les études ethnographiques réalisées par des tiers, que nous avons consolidées par des interviews d'utilisateurs hors contexte. La restitution des observations et interviews contextuelles via des scénarios peut également être capitalisée par les ingénieurs en exigences, se constituant une bibliothèque de scénarios réutilisable par l'équipe d'ingénieurs en exigences. Cependant, la mise en place d'une nouvelle version de système doit déclencher une réactualisation des scénarios via des observations et interviews contextuelles. En ce qui concerne le côté non prévisible des observations, rien ne peut garantir en effet que l'ingénieur en exigences puisse observer des instances critiques du système le jour où il réalise les observations. L'incomplétude et le côté anecdotique sont des critiques récurrentes des scénarios issus d'observations (Diaper 2002).

Pour répondre à ce point, nous évaluons l'ingénierie participative des exigences sur les rapports d'analyse d'accident et d'incident rédigés par des tiers, en les considérant comme des scénarios. En effet, les systèmes critiques possèdent des systèmes de management de la sécurité assurant un traitement systématique des incidents et des accidents à différents niveaux pour une amélioration continue de la sécurité. En aéronautique, il existe des cellules responsables de la sécurité des vols dans les compagnies aériennes, chez les constructeurs d'avion, au niveau d'un Etat, du continent et enfin au niveau mondial avec l'OACI. En France, le Bureau d'Enquêtes et d'Analyses pour la sécurité de l'aviation civile (BEA) conduit ses enquêtes avec comme « unique objectif l'amélioration de la sécurité aérienne » sans viser «la détermination des fautes ou responsabilités ». Les rapports d'analyse d'accident, disponibles en ligne sur le site du BEA (https://www.bea.aero/), reconstituent la chronologie des événements ayant conduit à l'accident. Une analyse se base sur le rejeu des données enregistrées par les « boites noires » (échanges verbaux, actions de pilotage et paramètres de vol), des simulations effectuées pour vérifier le comportement des différentes sorties visuelles, auditives ou haptiques des sous-systèmes matériels, et des jugements d'experts. Les parties prenantes sont nombreuses, et nous avons eu initialement des difficultés à instancier notre vision située de l'ingénierie des exigences pour caractériser le cas « analyse d'accident et d'incident ». Quel est le système qui représente le *champ de travail commun* de ces parties prenantes ? Le système peut être celui du transport aérien, sur lequel nous arrivons à positionner toutes les parties prenantes que nous venons de citer selon la Figure 142.

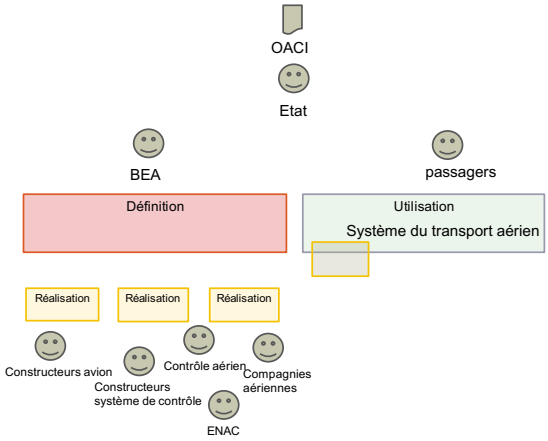
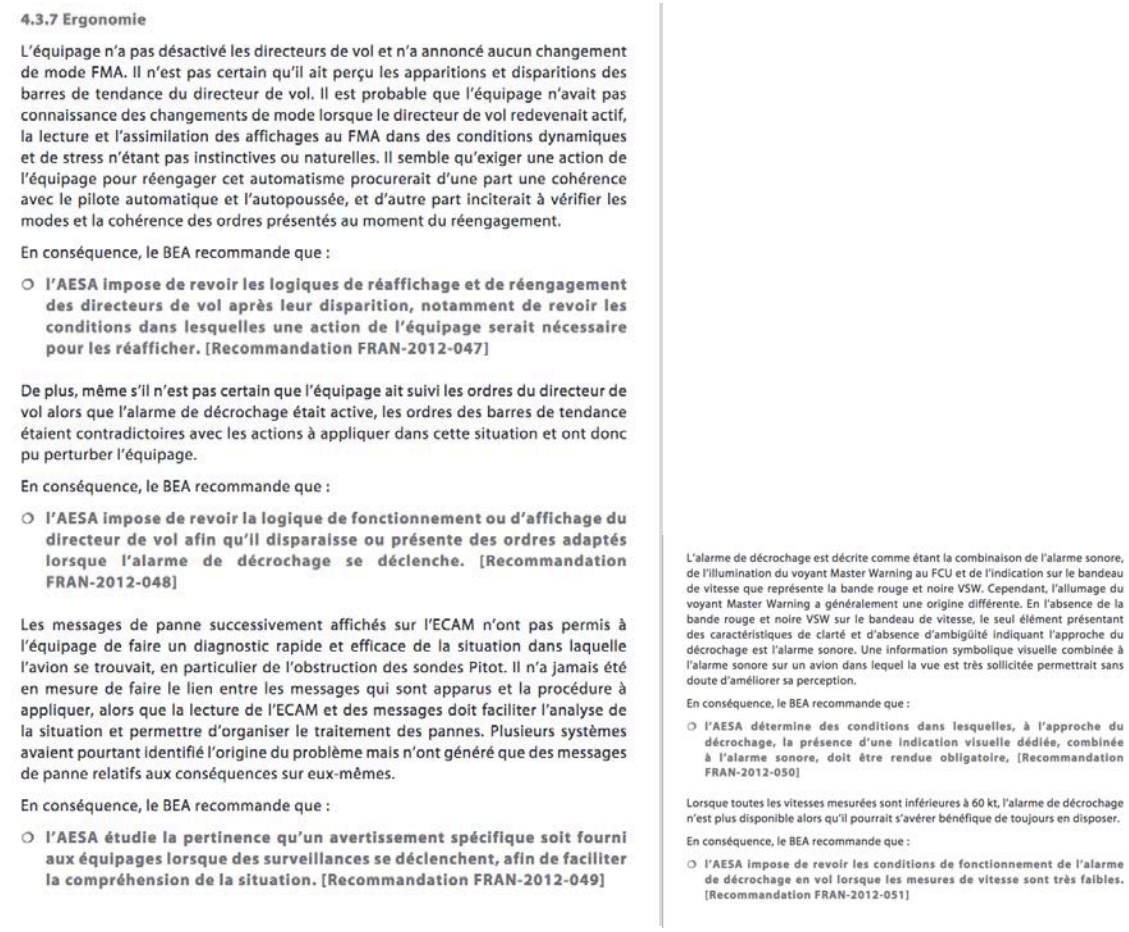



Figure 142: le système du transport aérien représenté dans notre vision située de l'ingénierie des exigences

Nous positionnons le BEA comme ingénieur en exigences : un rapport contient le scénario détaillé de l'accident, et les recommandations émises, structurées selon les différentes parties prenantes de notre schéma, sont très similaires à des exigences (voir exemples en Figure 143).



**4.3.7 Ergonomie**

L'équipage n'a pas désactivé les directeurs de vol et n'a annoncé aucun changement de mode FMA. Il n'est pas certain qu'il ait perçu les apparitions et disparitions des barres de tendance du directeur de vol. Il est probable que l'équipage n'avait pas connaissance des changements de mode lorsque le directeur de vol redevenait actif, la lecture et l'assimilation des affichages au FMA dans des conditions dynamiques et de stress n'étant pas instinctives ou naturelles. Il semble qu'exiger une action de l'équipage pour réengager cet automatisme procurerait d'une part une cohérence avec le pilote automatique et l'autopoussée, et d'autre part inciterait à vérifier les modes et la cohérence des ordres présentés au moment du réengagement.

En conséquence, le BEA recommande que :

- **l'AESA impose de revoir les logiques de réaffichage et de réengagement des directeurs de vol après leur disparition, notamment de revoir les conditions dans lesquelles une action de l'équipage serait nécessaire pour les réafficher. [Recommandation FRAN-2012-047]**

De plus, même s'il n'est pas certain que l'équipage ait suivi les ordres du directeur de vol alors que l'alarme de décrochage était active, les ordres des barres de tendance étaient contradictoires avec les actions à appliquer dans cette situation et ont donc pu perturber l'équipage.

En conséquence, le BEA recommande que :

- **l'AESA impose de revoir la logique de fonctionnement ou d'affichage du directeur de vol afin qu'il disparaisse ou présente des ordres adaptés lorsque l'alarme de décrochage se déclenche. [Recommandation FRAN-2012-048]**

Les messages de panne successivement affichés sur l'ECAM n'ont pas permis à l'équipage de faire un diagnostic rapide et efficace de la situation dans laquelle l'avion se trouvait, en particulier de l'obstruction des sondes Pitot. Il n'a jamais été en mesure de faire le lien entre les messages qui sont apparus et la procédure à appliquer, alors que la lecture de l'ECAM et des messages doit faciliter l'analyse de la situation et permettre d'organiser le traitement des pannes. Plusieurs systèmes avaient pourtant identifié l'origine du problème mais n'ont généré que des messages de panne relatifs aux conséquences sur eux-mêmes.

En conséquence, le BEA recommande que :

- **l'AESA étudie la pertinence qu'un avertissement spécifique soit fourni aux équipages lorsque des surveillances se déclenchent, afin de faciliter la compréhension de la situation. [Recommandation FRAN-2012-049]**

L'alarme de décrochage est décrite comme étant la combinaison de l'alarme sonore, de l'illumination du voyant Master Warning au FCU et de l'indication sur le bandeau de vitesse que représente la bande rouge et noire VSW. Cependant, l'allumage du voyant Master Warning a généralement une origine différente. En l'absence de la bande rouge et noire VSW sur le bandeau de vitesse, le seul élément présentant des caractéristiques de clarté et d'absence d'ambigüité indiquant l'approche du décrochage est l'alarme sonore. Une information symbolique visuelle combinée à l'alarme sonore sur un avion dans lequel la vue est très sollicitée permettrait sans doute d'améliorer sa perception.

En conséquence, le BEA recommande que :

- **l'AESA détermine des conditions dans lesquelles, à l'approche du décrochage, la présence d'une indication visuelle dédiée, combinée à l'alarme sonore, doit être rendue obligatoire, [Recommandation FRAN-2012-050]**

Lorsque toutes les vitesses mesurées sont inférieures à 60 kt, l'alarme de décrochage n'est plus disponible alors qu'il pourrait s'avérer bénéfique de toujours en disposer.

En conséquence, le BEA recommande que :

- **l'AESA impose de revoir les conditions de fonctionnement de l'alarme de décrochage en vol lorsque les mesures de vitesse sont très faibles. [Recommandation FRAN-2012-051]**

Figure 143: recommandations en ergonomie formulées par le BEA à l'issue de l'analyse de l'accident du vol 447

L'ENAC, en tant qu'école de formation des ingénieurs, des contrôleurs aériens et des pilotes, est positionnée comme fournisseur du système du transport aérien (en bas de la Figure 142). Plus particulièrement, de même que les autres fournisseurs possèdent leur cellule d'analyse et de prévention des incidents et accidents et tirent leurs propres leçons des incidents et accidents, l'équipe d'Informatique Interactive de l'ENAC possède une compétence en Interaction Humain-Machine qu'elle peut mettre au service du système du transport aérien par des analyses complémentaires, à partir des analyses réalisées par le BEA. Ces analyses complémentaires peuvent à leur tour alimenter le contenu des formations ENAC et les axes de recherche.

Dans cet objectif, nous avons étudié le rapport d'enquête de l'accident du vol 447 Rio-Paris (https://www.bea.aero/docspa/2009/f-cp090601/pdf/f-cp090601.pdf) : le rapport semble exclure toute panne des systèmes embarqués, à l'exception des capteurs appelés « sondes Pitot ». La défaillance de ces sondes est un cas prévu : les systèmes de bord sont conçus pour y réagir, et ont eu le comportement attendu par leurs concepteurs. C'est une combinaison des comportements de ces systèmes, des conditions de vol et des réactions des pilotes, qui a donné une issue fatale à cette défaillance. Par conséquent, ce cas pose les questions de la capacité d'un fournisseur (l'ENAC) à produire des exigences détaillées à partir de scénarios produits par un ingénieur en exigence (le

BEA), et l'extension de l'ingénierie participative des exigences à des situations **mal** gérées par les utilisateurs.

### VI.3.3.2. Description du processus

> *Observation de situations non prévues dans le système actuel mais gérées par les utilisateurs*

Suivant notre approche d'ingénierie participative des exigences, nous avons plus particulièrement étudié les descriptions des *actions* des pilotes, et des *ressources pour les actions.* Nous avons donc cherché dans le rapport les différents extraits relatifs à la chronologie des actions des pilotes. Nous avons alors relevé que les descriptions des actions des pilotes étaient fréquemment associées à des mots tels que « symptômes », « diagnostic », « anomalie », « signes crédibles » (voir par exemple un extrait du rapport en Figure 144 dans lequel nous mettons les termes en évidence).

Il est alors prévu que l'équipage détectera l'anomalie, qu'il « fera sens » éventuellement de cette détection, qu'il modifiera ses priorités d'action en cours, et qu'il engagera l'action correspondante, (actions de pilotage et/ou actions de traitement de la panne, associées à des procédures ou check-lists), le tout dans le délai attendu (dont l'ordre de grandeur est indiqué dans le raisonnement de certification s'il est critique).
Sur A330, l'ECAM propose des actions à effectuer dans la majorité des cas de panne ou d'urgence. A partir des informations disponibles à l'ECAM, l'équipage doit analyser et confirmer la nature de la défaillance avant d'entreprendre toute action de traitement de la panne. Dans d'autres cas, la « réaction adéquate » attendue de l'équipage suppose des actions immédiates de mémoire visant à stabiliser la situation, puis le recours à des instructions d'action disponibles sur l'ECAM, et/ou le recours à des procédures explicitées dans le QRH et classées par catégorie d'anomalie diagnostiquée.
……
Selon la fréquence d'exposition de l'opérateur à l'anomalie durant sa formation ou en opérations réelles, sa réponse peut être automatique, applicative de règles, ou construite sur la base de connaissances profondes. Les réponses automatiques supposent la reconnaissance de stimuli bien spécifiques, à laquelle la réaction est associée sans véritable interprétation. L'application de règles suppose non seulement leur connaissance, mais aussi la reconnaissance de leurs conditions d'applicabilité, et donc une bonne identification plus une certaine interprétation de l'anomalie. La construction d'une réponse par appel aux connaissances suppose une incorporation de l'anomalie dans la représentation mentale de la situation, qui peut passer par une destruction/reconstruction de celle-ci, très coûteuse en ressources et chronophage. Ainsi, la perception correcte de la situation par un équipage, qui permet d'améliorer la fiabilité et la rapidité de diagnostic et de décision, est liée non seulement à la manière dont la situation est présentée à cet équipage (interfaces, paramètres) mais aussi à sa formation et à son expérience.

Figure 144: extrait du rapport d'analyse du BEA de l'accident du vol AF447

Ce vocabulaire est proche de celui de la médecine : un médecin construit son diagnostic par observation des symptômes du patient, ce qui le conduit, à partir de ses connaissances et de son expérience, à faire des hypothèses sur des maladies possibles (Rapezzi, Ferrari, et Branzi 2005). Le médecin va faire l'hypothèse que c'est une maladie donnée, et rechercher des faits spécifiques à cette maladie pour tester son hypothèse : une question vers le patient « est-ce que vous avez mal quand vous avalez ? », des palpations, une prise de sang, un examen radiologique complémentaire. En cas d'urgence, il va d'abord stabiliser la situation du patient (position latérale de sécurité par exemple) avant d'entreprendre un diagnostic. En résumé, le médecin, face à un dysfonctionnement du patient à diagnostiquer, entreprend des *actions* pour gagner en confiance et consolider son diagnostic en *interagissant* avec le *patient*. Cette démarche est un processus *d'abduction,* défini comme un « processus permettant d'expliquer un phénomène ou une observation à partir de certains faits, événements ou lois » (Bourgeois et Lecourt 2006). Peirce (Peirce 1955) a théorisé l'abduction comme méthode scientifique de construction de connaissances, l'articulant avec les méthodes classiques de déduction et d'induction. Il la résume de la façon suivante :

*The surprising fact, C, is observed*

*But if A were true, C would be a matter of course,*

*Hence, there is reason to suspect that A is true.*

Ainsi, le pilote (médecin), face à une anomalie du système avion (patient) à diagnostiquer, entreprend des *actions* pour gagner en confiance et consolider son *diagnostic* en *interagissant* avec *l'avion* (patient).

Nous avons utilisé le *cycle de l'action* (E. L. Hutchins, Hollan, et Norman 1985) comme outil d'analyse : il est une simplification de la réalité mais constitue cependant un cadre utile et reconnu pour comprendre les actions humaines. Quand des personnes utilisent un système, elles doivent faire face à deux « gouffres » (*gulfs*) : le gouffre de l'exécution, où elles essaient de comprendre comment opérer, et le gouffre de l'évaluation, où elles essaient de comprendre ce qui est arrivé après avoir opéré.

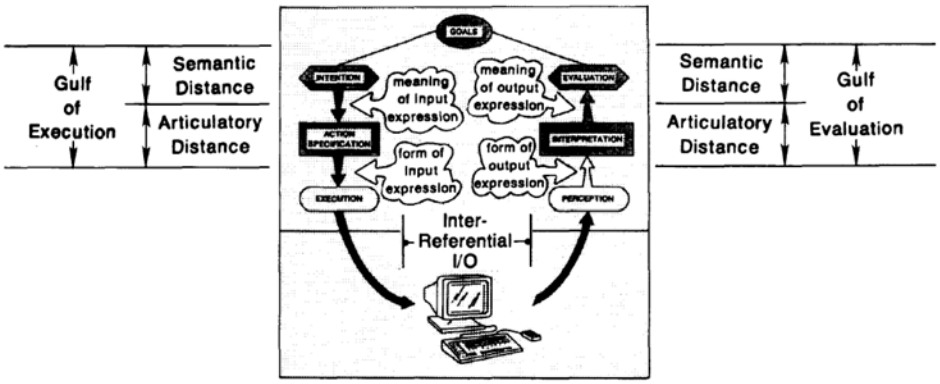



Figure 145: le cycle de l'action, avec les gouffres de l'exécution et de l'évaluation (E. L. Hutchins, Hollan, et Norman 1985)

En particulier, en cas de fonctionnement inattendu (gouffre de l'évaluation, à droite), les utilisateurs remettent en cause les éléments afférents aux concepts cités par Norman (Norman 2014): signifiants (*meaning of output expression* sur la figure Figure 145) rétro-actions et notifications (form of output expression sur la figure Figure 145). Ainsi, nous avons modélisé la chronologie des actions décrites dans le rapport d'analyse par une succession de cycles de l'action, dans lesquels nous avons représenté les moyens d'entrée (le manche), les rétro-actions sur l'instrument des informations primaires de vol (PFD pour *Primary Flight Display)*, et les notifications. La conclusion du rapport est la suivante :

> Les ordres du directeur de vol, le doute sur la pertinence de l'alarme sonore de décrochage et l'identification d'une possible situation de survitesse n'ont pas permis au PF de poser le bon diagnostic. Il a alors combiné des actions antagonistes pour répondre à la fois à une situation de survitesse (réduction de poussée, actions à cabrer) et à une situation de décrochage (application de la poussée maximale). p188

Dans le cadre du processus d'abduction et du cycle de l'action, nous pouvons reformuler de la façon suivante : les pilotes disposaient de deux *signifiants* leur suggérant des *actions* opposées :

- une action d'assiette à cabrer est indiquée par le directeur de vol ;
- une action d'assiette à piquer est suggérée par l'alarme de décrochage (STALL).

Ils ont essayé de façon alternative les deux types d'actions, pour tester le comportement de l'avion et faire un choix entre une situation de survitesse et une situation de perte de vitesse et de décrochage. Mais les tests n'ont pas permis de lever le doute, renvoyant des signifiants invisibles ou contradictoires par rapport à ce qu'ils attendaient : il y a notamment une assiette à piquer pour prendre de la vitesse qui déclenche l'alarme de décrochage de façon contradictoire avec le modèle conceptuel du pilote. Le pilote a peut-être pensé que l'alarme passe de l'état «arrêt» à «marche» quand elle retentit, alors qu'elle est passée de l'état «marche-déconnecté» à «marche-connecté». Les deux états « arrêt-connecté » et « marche-déconnecté » ont des signifiants similaires (silence), ce qui les rend indiscernables (voir Figure 146).

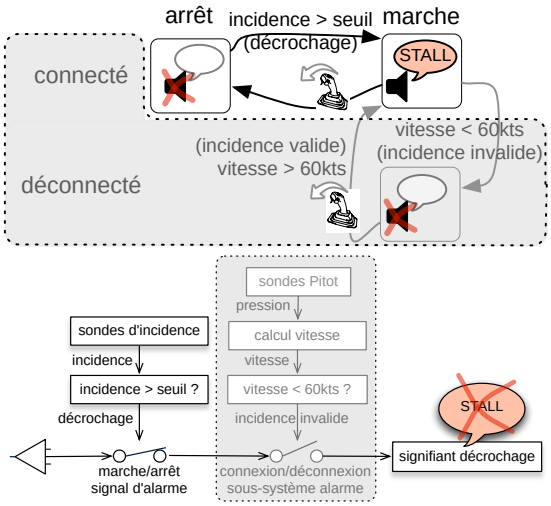



Figure 146: modèle conceptuel du pilote sur le fonctionnement de l'alarme de décrochage

Pour une analyse détaillée, nous renvoyons le lecteur vers un article plus complet (Conversy et al. 2014).

> *Expression des exigences de haut niveau sur le système pour la gestion des situations non prévues*

Le processus d'abduction est une démarche intéressante pour poser un diagnostic sur un fait surprenant. Cependant, le problème dans le cas des pilotes est que les tests qu'ils utilisent sont des actions de pilotage qui remettent en question la stabilité de la situation et que ces actions ne sont pas réversibles. Pour reprendre notre analogie avec le médecin, cela reviendrait, après avoir placé le patient en position latérale de sécurité (stabilisation de la situation), à manipuler sa tête pour tester l'hypothèse d'une fracture des cervicales, et mettre par ce geste sa vie en danger. Le médecin a un outil spécifique pour faire ce test, la radiologie, lui permettant d'analyser le patient en minimisant l'impact. De même, nous suggérons qu'il faut fournir aux pilotes des outils leur permettant de conduire de façon sûre le raisonnement par abduction. Ainsi, nous formulons une première exigence de haut niveau de la façon suivante :

| |
|---|
| Afin de réaliser un diagnostic de la situation, le système doit permettre au pilote de conduire un raisonnement d'abduction qui maintient un état stable du système. |

En reprenant l'approche méthodologique d'un raisonnement d'abduction (Svennevig 2001) nous pouvons affiner cette exigence de la façon suivante :

Afin de réaliser un diagnostic de la situation, le système doit permettre au pilote de conduire un raisonnement d'abduction qui maintient un état stable du système :

- formuler des hypothèses
- choisir l' hypothèse la plus plausible
- tester l'hypothèse choisie en interagissant avec le système sans en modifier son état.

*Génération et évaluation de solutions répondant aux situations non prévues*

A partir de là, une multitude de questions de conception peuvent se poser pour concevoir ce genre de dispositif, et il faudrait travailler l'exigence avec des spécialistes de l'intelligence artificielle, de l'IHM, des systèmes bord, et des pilotes pour concevoir des solutions. Les idées ci-dessous sont issues de notre seule réflexion au moment de la rédaction de la thèse, alimentée par nos connaissances en IHM et l'utilisation de la technique de l'analogie avec la médecine.

Pour formuler les hypothèses, on peut imaginer une première suggestion automatique d'hypothèses de type *Recommender Systems (la dernière fois qu'un pilote était dans cette situation, il a choisi telle hypothèse).* Pour choisir et tester l'hypothèse la plus plausible, il faut imaginer des représentations et des interactions sur les hypothèses. De nombreux travaux en Facteurs Humains proposent un signifiant de type icône pour montrer aux pilotes (ou conducteurs de véhicule) un niveau d'incertitude sur l'automatisme. Les résultats montrent que cela modifie le comportement du pilote/conducteur (McGuirl et Sarter 2006)(Flemisch et al. 2012) (Beller, Heesen, et Vollrath 2013) (Jung et al. 2015). Mais est-ce discernable dans un environnement déjà encombré visuellement ? Des études montrent que la performance de recherche visuelle se dégrade dans des environnements chargés visuellement et peu organisés (Doyon-Poulin, Robert, et Ouellette 2012) (N. Moacdieh et Sarter 2015)(N. M. Moacdieh et Sarter 2017).

Rasmussen et Vicente ont proposé des recommandations centrées sur la seule aide au diagnostic, avec une représentation hiérarchisée des informations, en insistant sur la perception au détriment de l'action (Rasmussen 1985) (Kim J. Vicente et Rasmussen 1990)(K. J. Vicente et Rasmussen 1992). Nous avons vu que les ingénieurs en exigences et les fournisseurs de composants utilisaient des dessins comme support au raisonnement et à la construction du comportement du système. Ces représentations seraient-elles efficaces pour les utilisateurs du système en utilisation et l'aide au diagnostic ?

La question cruciale est de trouver comment interagir avec le système sans en modifier son état. Les actions étant irréversibles, le « undo » classique en utilisabilité n'est pas possible, et peut être dangereux (Mancini, Dix, et Levialdi 1997). De même qu'il y a les rétro-actions (*feedback*) et les trans-actions (*feedthrough*), peut-on généraliser la pré-action (*feedfoward*) ? Ou utiliser un dispositif mobile (comme l'appareil de radiologie) pour faire un zoom sémantique sur un élément du système ? Le support au raisonnement par abduction dans un contexte critique est à la croisée de la visualisation d'information, de la collaboration et de l'interaction humain-machine et reste à explorer.

*Expression des exigences détaillées pour la gestion des situations non prévues*

En l'absence de prototypes, nous ne sommes pas en mesure d'aller plus loin sur l'expression des exigences. De plus, nous pourrions faire l'exercice de transformer certaines idées citées ci-dessus en exigences, mais elles seraient soumises à un changement très probable en l'absence d'évaluation avec les utilisateurs.

### VI.3.3.3. Synthèse et obstacles à la validité

Nous présentons les résultats du processus d'ingénierie des exigences dans le Tableau 11.

| **Activités** | **Résultats** | **Moyens de l'ingénierie participative des exigences mis en œuvre** |
|---|---|---|
| Observation de situations non prévues dans le système actuel mais gérées par les utilisateurs | Le raisonnement par abduction<br><br>Conversy, Stéphane, Stéphane Chatty, Hélène Gaspard-Boulinc, et Jean-Luc Vinot. « The Accident of Flight AF447 Rio-Paris: A Case Study for HCI Research ». In Proceedings of the 26th Conference on L'Interaction Homme-Machine, 60-69. IHM '14. New York, NY, USA: ACM, 2014. doi:10.1145/2670444.2670459. | Analyse du rapport d'analyse de l'accident AF447, basée sur la recherche des actions des pilotes<br>Technique d'idéation (association d'idées)<br>Cycle de l'action pour décrire les actions et restituer le déroulement des actions<br>Absence d'utilisateurs |
| Expression des exigences de haut niveau sur le système pour la gestion des situations non prévues | Exigences sur le support à l'abduction | Technique d'idéation (analogie avec la médecine)<br>Approche méthodologique de l'abduction transformée en sous-exigences<br>Absence d'utilisateurs |
| Génération et évaluation de solutions répondant aux situations non prévues | Des idées émises partiellement en IHM | Connaissances en IHM et Facteur Humain<br>Technique d'idéation (analogie)<br>Absence de prototypage et d'idéation<br>Absence d'utilisateurs |
| Expression des exigences détaillées pour la gestion des situations non prévues | | |

Tableau 11: synthèse des résultats du processus d'ingénierie des exigences sur le cas 'analyse d'accident du vol Rio-Paris 447'

Les résultats sont contrastés : en quoi pouvons-nous qualifier le processus de participatif étant donnée la non-participation des utilisateurs ? En position de fournisseur, nous avons utilisé un scénario d'accident rédigé par un ingénieur en exigences, et mobilisé une technique de décomposition des actions selon le cycle de l'action, complétée par une analogie. Nous avons produit des exigences de haut niveau, mais elles restent à consolider et à affiner. Nous aurions pu interviewer des pilotes pour consolider nos exigences sur le processus d'abduction, en utilisant les cycles de l'action comme support à l'interview. Nous pouvons qualifier le processus d'ingénierie des exigences mis en œuvre de centré-utilisateur plus que de participatif.

Etendre l'ingénierie participative des exigences à des situations **mal** gérées par les utilisateurs, tels que les incidents et les accidents, est difficile. En cas d'incident, les utilisateurs impliqués se considèrent en situation d'échec et la conduite d'interview nécessite un support fidèle en termes de rejeu (Helene Gaspard-Boulinc, Jestin, et Fleury 2002). De plus, la survenue d'un incident constitue un cadre d'intervention difficile, dans lequel les utilisateurs sont sensibles à la gestion de la règle et aux sanctions (Fleury, De Wit, et Gaspard-Boulinc 2004). Dans le cas d'un accident, les utilisateurs non impliqués restent le plus souvent prudents, en l'absence d'une connaissance approfondie de la situation et des intentions des utilisateurs impliqués.

Cependant, nous pouvons comparer les exigences que nous avons réussi à formuler, ainsi que les idées émises en mobilisant nos connaissances en Interaction Humain-Machine, aux recommandations en ergonomie émises par le BEA. Nos exigences de haut niveau apportent un éclairage différent mais non contradictoire, et ouvrent sur des axes de recherche élargissant les

propositions de recommandations du BEA (voir pages 217-218 en Figure 143). Enfin, la construction des modèles conceptuels de la Figure 146 est un exercice intégré dans l'enseignement en IHM pour les élèves-ingénieurs ENAC (Conversy et al. 2014).

Une ingénierie des exigences centrée-utilisateur pourrait être une voie intéressante (1) pour les fournisseurs n'ayant pas accès aux utilisateurs, (2) pour l'analyste d'incident et d'accident ayant des difficultés à impliquer les utilisateurs. Elle n'est pas suffisante : une consolidation des exigences avec l'ingénieur en exigences et les utilisateurs reste nécessaire à réaliser pour poursuivre le processus sur une base saine.

## VI.3.4. Nouvel instrument de vol

### VI.3.4.1. Cadre et significativité du cas

L'étude d'un nouvel instrument de vol, que nous utilisons comme dernier cas d'illustration de l'ingénierie participative des exigences, est à l'initiative d'un pilote instructeur expert, concepteur de la formation initiale des pilotes à l'ENAC, et responsable du projet E-Fan 2.0 à l'ENAC.

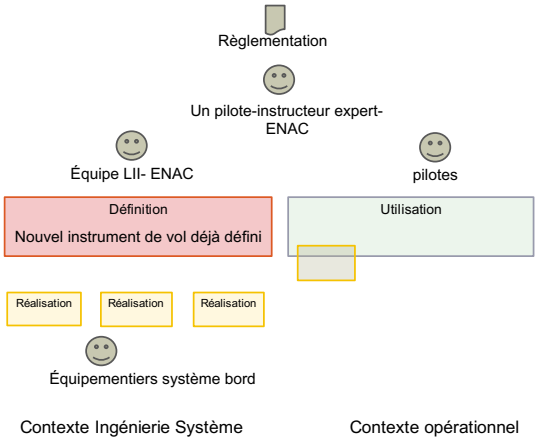



Figure 147: cas 'nouvel instrument de vol' dans notre vision située de l'ingénierie des exigences

Il est venu nous consulter avec une idée de « smart PFD », dans lequel « la bonne information est présente au bon endroit au bon moment », combinant « deux philosophies d'indicateur d'attitude, occidentale et russe ». Il avait construit une première solution sur une impression papier de l'instrument actuel, en barrant les éléments et les replaçant, avec des explications manuscrites (voir Figure 148). Il est intéressant de noter que cette situation est analogue à celles que nous avons observées et rapportées en partie IV.2 entre ingénieurs en exigences et fournisseurs de composants : l'utilisation d'artefacts partagés mêlant proposition de solution et questions pour démarrer le travail, chacun raisonnant avec son outil de travail (instrument de vol ou architecture fonctionnelle).

L'étude constitue une évaluation significative car elle démarre par une solution, assez précise, proposée par un utilisateur expert : l'ingénieur en exigences apporte-t-il une plus-value ? Plus généralement, les utilisateurs experts peuvent avoir des idées très précises sur l'amélioration du système : pourquoi ne pas les laisser faire plutôt que de mettre en place une ingénierie participative des exigences ?

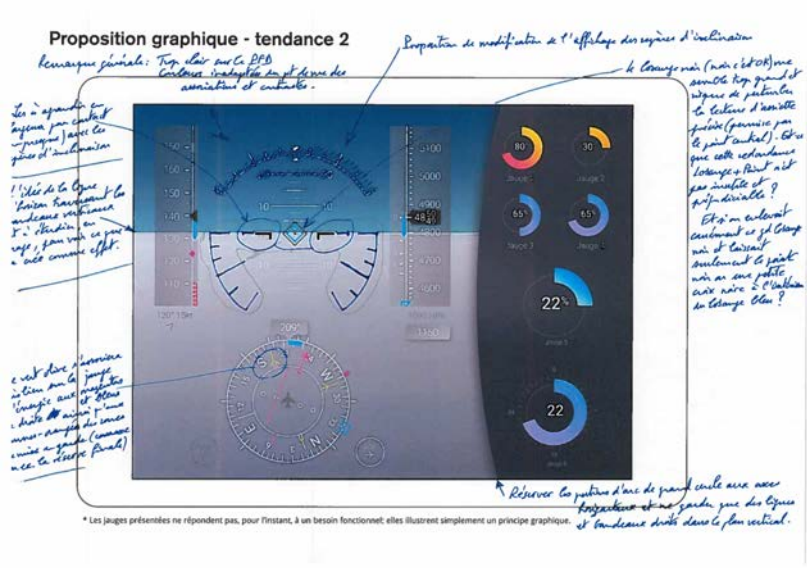



Figure 148: maquette papier de la solution proposée et transmise par le pilote instructeur expert

### VI.3.4.2. Description du processus

> *Observation de situations non prévues dans le système actuel mais gérées par les utilisateurs*

Nous avons parlé de l'idée de combiner les philosophies russe et occidentale à un ancien pilote d'hélicoptère : cela lui a remémoré l'accident d'avion du vol CrossAir 498 avec une perte de contrôle de l'avion au décollage. En effet, l'analyse réalisée par le Bureau Enquêtes Accident suisse (http://www.aaiu.ie/node/735) explique en partie la perte de contrôle par une mauvaise interprétation du sens de virage indiqué par l'instrument occidental, le pilote ayant été formé sur un instrument russe. Le rapport d'analyse pointe également la difficulté d'utiliser l'instrument occidental dans une situation de grande inclinaison. Nous avons commencé à utiliser le cycle de l'action pour analyser la séquence d'actions décrite dans le rapport d'analyse de l'accident du Crossair 498, en représentant les visualisations russe et occidentale côte à côte comme ressources pour l'action afin de mettre les différences en évidence et les potentielles confusions sur le sens du virage.

> *Génération et évaluation de solutions répondant aux situations non prévues*

Le problème est que la visualisation russe n'est pas disponible dans les simulateurs de vol : la domination commerciale des constructeurs occidentaux et la crainte de difficultés liées au changement des habitudes des pilotes ont contribué à la généralisation de la logique occidentale. Par conséquent, nous avons utilisé un dispositif simple, constitué de la visualisation et d'un manche, sans simulateur de vol. La visualisation, produite avec le langage djnn (Chatty et al. 2016), est pilotée directement par les événements envoyées par le manche. L'utilisation du langage djnn a permis de prototyper rapidement les trois visualisations (occidentale, russe et la nouvelle), avec la *dynamique* de l'affichage, notamment en forte inclinaison (voir Figure 149). Pour des visualisations temps-réel, les prototypes papier ont en effet leurs limites.

Le pilote-instructeur à l'origine du projet a pu constater la dynamique du nouvel instrument qu'il avait imaginé. Sur la base de ces prototypes, nous avons réalisé une séance d'idéation avec lui, explorant les capacités de djnn à gérer l'opacité des éléments graphiques, notamment pour un affichage partiel de l'échelle d'assiette selon la valeur courante (voir Figure 149, au centre de la photo c).


(a) (b) (c)

Figure 149: prototypage rapide de la dynamique des trois visualisations occidentale (a), russe (b) et combinée (c), à forte inclinaison, grâce à l'utilisation de djnn

De plus, la participation d'un chercheur en visualisation d'information a conduit à une décomposition de la visualisation en différents éléments graphiques (Figure 150). Cette décomposition génère une qualification précise des différences entre les trois visualisations (Hélène Gaspard-Boulinc, Louviot, et al. 2016) et suggère une extension de la gestion de l'opacité à tous les éléments graphiques.

Figure 150: éléments graphiques des indicateurs d'attitude : horizon, échelle d'assiette, maquette avion, pointeur, échelle d'inclinaison

Ainsi, grâce au développement de prototypes dynamiques et à une expertise en visualisation d'informations, l'idée initiale a été enrichie sur :

- la base de capacités technologiques du langage de programmation, non connues de l'utilisateur ;
- une maitrise de l'espace de conception de la visualisation, manipulée de façon empirique par l' utilisateur.

Ce sont les principes de visualisation d'information qui ont été mobilisés dans la génération de solution, plus que les principes de collaboration directe. L'instrument de vol n'est pas actuellement un outil interactif de collaboration entre les pilotes, mais cette idée reste à explorer.

> *Expression des exigences détaillées pour la gestion des situations non prévues*

Ces idées ont conduit à une demande de brevet, qui se traduit par une expression des exigences détaillées de l'invention (Louviot, Gaspard-Boulinc, et Conversy 2017), et une extension à des dispositifs d'affichage tels que les casques de réalité augmentée (voir Figure 151).

Cependant, à ce stade, nous ne pouvons pas affirmer que le système spécifié améliore la gestion de situations non prévues : à partir de l'idée initiale, nous avons rebondi sur le rapport d'analyse d'accident du Crossair, produit des premiers prototypes mais nous n'avons pas de scénarios bien définis, à part les virages à grande inclinaison. De plus, aucune évaluation n'a été conduite avec des utilisateurs pour tester leurs performances avec le nouveau système. Les exigences détaillées ont une valeur d'un point de vue innovation, mais leur valeur opérationnelle reste à démontrer.

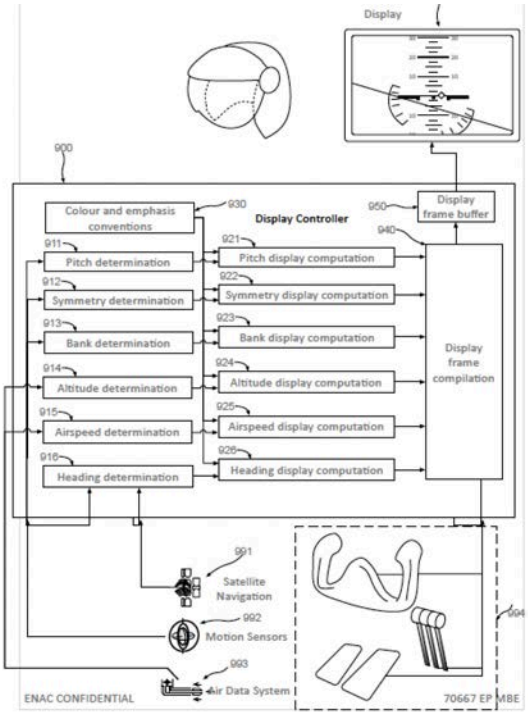



Figure 151: extrait du brevet décrivant une spécification détaillée des données en entrée de la visualisation

> *Observation de situations non prévues dans le système actuel mais gérées par les utilisateurs*

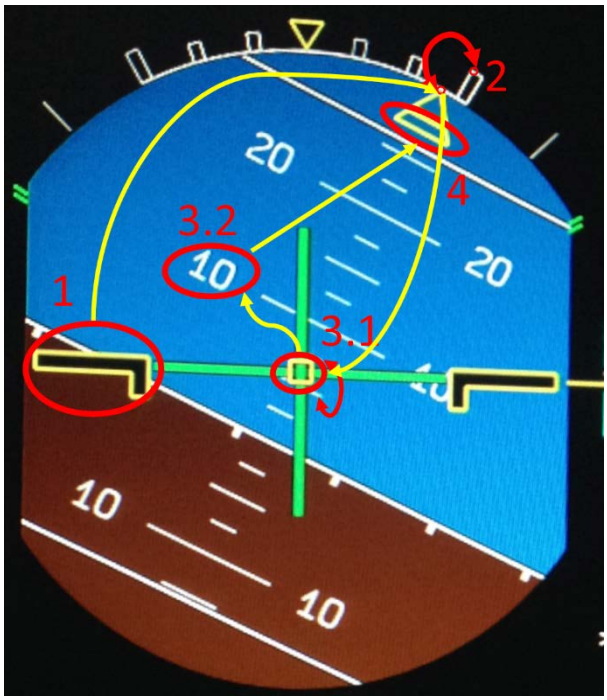
Nous avons décidé d'abandonner l'analyse de l'accident du Crossair 498, trop complexe à conduire et probablement synonyme d'un processus sans participation effective d'utilisateurs. L'expertise de notre pilote instructeur et concepteur de la progression pédagogique de la formation de pilote nous a permis d'identifier un scénario pertinent à étudier : l'apprentissage initial de la mise en virage. C'est un des exercices où l'alternance entre observation de l'extérieur de l'avion et lecture instrumentale est la plus exigeante pour l'apprenti-pilote. La rigueur de l'apprentissage est la garantie d'une « mécanisation » correcte du pilote, pour parvenir au final à une exécution automatique sécuritaire. Nous avons utilisé le cycle de l'action pour décomposer le scénario et conduire des interviews avec le pilote-instructeur. Trois cycles d'action successifs pour atteindre l'objectif de la mise en virage ont été identifiés et modélisés :

- 1er cycle : le but est de **débuter** le virage à gauche ;
- 2e cycle : le but est de **stabiliser** le virage ;
- 3e cycle : le but est de **calibrer** l'inclinaison du virage

Les deux premiers cycles se caractérisent par l'importance des perceptions du pilote et de l'extérieur pour l'évaluation des actions. Une consigne de l'instructeur est la non utilisation de l'instrument pour l'évaluation des actions. Lors d'interviews réalisées auprès d'instructeurs-pilotes (avion et hélicoptère), tous rapportent la tendance des élèves à regarder l'écran pour l'évaluation de leurs actions plutôt que l'extérieur. L'écran occupe de fait une place centrale devant l'élève, et attire l'attention par plusieurs animations (l'horizon bascule, la vitesse défile, et potentiellement l'altitude aussi). Pour contrer cette tendance à regarder l'écran, la pratique des instructeurs est de le cacher avec une feuille ou des post-it. L'utilisation de l'instrument n'est préconisée par l'instructeur que sur le 3e cycle, dans le but de calibrer l'inclinaison du virage. L'instructeur donne une consigne de

calibration du virage à 30°. Le cycle de l'action met en évidence une alternance d'observation extérieure et de lecture des valeurs instrumentales de la part du pilote, passant ainsi de la vision périphérique à la vision centralisée d'une indication précise au sein d'un environnement instrumental chargé.

Cependant, le cycle de l'action atteint ses limites pour décrire de façon fine la lecture de l'information sur l'instrument. En complément, nous avons utilisé le cadre de description du parcours visuel ScanVis (Conversy 2014b), afin de décomposer plus finement les actions de prise d'informations sur l'interface (voir Figure 152). La décomposition ScanVis, construite par interview du pilote-instructeur expert, révèle qu'il ne s'agit pas seulement de lire la valeur de l'inclinaison, mais de vérifier également les autres éléments de l'attitude de l'avion dans l'espace, l'assiette et la symétrie.




**1- L'utilisateur veut vérifier le sens du virage : il regarde la position du bout de l'aile de la maquette avion (chercher) par rapport à l'horizon artificiel (décoder et vérifier un prédicat).**
**2- L'utilisateur veut lire la valeur de l'inclinaison : il cherche la position du curseur triangulaire et le situe par rapport à la graduation la plus proche ainsi que la taille de cette graduation (chercher et naviguer parmi un sous-ensemble de marques).**
**3 - L'utilisateur veut lire la valeur de l'assiette :**
**3.1 il regarde la forme des deux graduations les plus proches du milieu de la croix (chercher et naviguer parmi un sous-ensemble de marques)**
**3.2 il regarde la position du milieu de la croix par rapport à la grande graduation la plus proche ainsi que la valeur de cette grande graduation (deviner une position sans marque)**
**4 - L'utilisateur veut lire l'écart de symétrie : il regarde le décalage entre le curseur triangulaire et le témoin trapézoïdale (décoder une marque et vérifier un prédicat)**

Figure 152: décomposition de la prise d'informations sur l'instrument avec ScanVis (Conversy 2014b)

> *Expression des exigences de haut niveau sur le système pour la gestion des situations non prévues*

Une analyse des cycles de l'action montre que l'instructeur-pilote apprend à l'élève à :

- d'abord utiliser d'autres signifiants pour contrôler l'avion que les symboles affichés sur l'instrument : repère capot et perceptions des mouvements ;
- utiliser l'instrument pour un contrôle plus fin de l'avion ;
- ne pas passer trop de temps sur l'instrument pour assurer la sécurité vis-vis de l'environnement extérieur.

Les conséquences sont que les opérations d'entrée-sortie de la visualisation vers l'extérieur sont continuelles et exigent une prise d'informations la plus performante possible. Une analyse plus précise des tâches de la prise d'informations montre que :

- pour passer de la **tâche 1** (vérifier le sens du virage) à la **tâche 2** (lire la valeur de l'inclinaison), le pilote doit parcourir la moitié de l'écran pour revenir dans la zone de la **tâche 1**.avec la **tâche 3** (lire la valeur de l'assiette).
- La **tâche 2** ne respecte pas le principe de compatibilité entre le contrôle et l'affichage : la quantité du virage à gauche se lit à droite.
- La **tâche 3.1** demande de deviner la position du centre de la croix par rapport aux traits horizontaux, avec des chiffres orientés dans le sens inverse à la rotation de l'avion.
- La méthode de visualisation de la **tâche 4,** suffisante pour les avions lourds, est beaucoup trop discrète pour une information qui est essentielle dans le pilotage des avions légers. En effet, les écarts de symétrie, liés à une mauvaise coordination de pilotage (en roulis et en lacet pour les variations d'inclinaison) ou à des turbulences extérieures auxquelles ils sont plus sensibles, peuvent conduire dans certaines phases de vol (notamment en vol lent, en montée ou en approche) au risque de perte de contrôle ou de départ en vrille dans les cas extrêmes.
- la plupart des tâches sont de type *seeking and navigating among a subset of marks,* qui est la plus consommatrice en temps et concentration.

Ces analyses nous permettent de formuler les exigences de haut niveau suivantes.

Afin de vérifier grossièrement l'attitude de l'avion lors de la mise en virage, l'instrument doit permettre à l'apprenti pilote de visualiser un horizon artificiel cohérent avec l'horizon réel visible à l'extérieur d'un simple coup d'œil.

Afin de vérifier finement l'attitude de l'avion, tout en assurant la sécurité vis-à-vis de l'environnement extérieur, l'instrument doit permettre à l'apprenti pilote de lire les informations d'attitude sur l'instrument le plus rapidement possible :

- lire la valeur de l'inclinaison du virage de façon cohérente avec le sens du virage ;
- lire la valeur de l'assiette de façon lisible à n'importe quelle inclinaison ;
- détecter un écart de symétrie le plus rapidement possible.

> *Expression des exigences détaillées pour la gestion des situations non prévues*

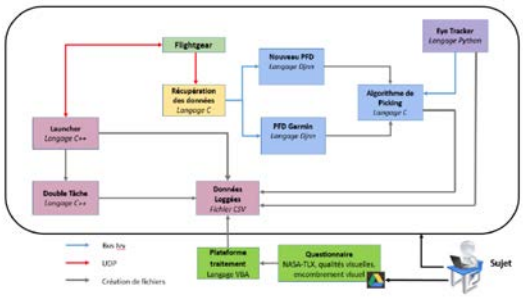
À partir de ce travail, nous pouvons reformuler les exigences détaillées de l'instrument et les justifier par rapport à une situation opérationnelle non prévue initialement pour l'instrument : le pilotage des avions légers en vol à vue.

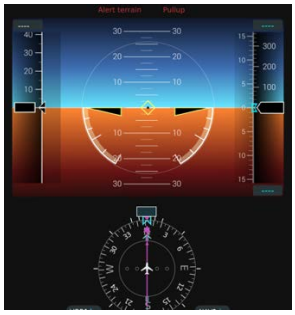
- L'horizon sera mobile, pour garder la cohérence avec l'environnement extérieure 
- L'échelle d'inclinaison sera composée de deux graduations courbes, symétriques de part et d'autre et en dessous de la maquette avion (à inclinaison nulle), dont la ligne d'inclinaison nulle, passant par le point central de la maquette avion, restera parallèle à l'horizon  ; quand l'avion s'inclinera, ces courbes effectueront donc une rotation autour du point central de la maquette avion  ; la valeur d'inclinaison se lira à l'extrémité de l'aile basse de la maquette  .
- L'échelle d'assiette restera fixe, perpendiculaire au plan des ailes de la maquette avion, permettant une lecture directe et centrale de la valeur d'assiette  .
- L'information de symétrie sera délivrée par un petit losange centré en vol symétrique sur le point central de la maquette avion et glissant à droite ou à gauche le long du plan des ailes en cas de dissymétrie  (et pouvant éventuellement devenir orange - warning - en cas de dépassement d'une valeur « acceptable »).

Une évaluation a priori du parcours visuel avec Scanvis sur ce nouvel instrument montre un parcours visuel réduit, avec une centralisation de l'information utile au contrôle de l'attitude de l'avion dans le contexte du vol à vue. Cependant, cette évaluation n'est pas suffisante : les performances de pilotage avec ce nouvel instrument restent à être démontrées. Pour cela, le réalisme de nos premiers prototypes n'est pas suffisant et la conception graphique de l'instrument reste à améliorer. Nous avons donc conduit en parallèle trois activités, illustrées en Figure 153 :

- L'intégration d'un simulateur de vol aux prototypes avec deux développeurs ;
- La conception graphique de l'instrument avec un graphiste ;
- La rédaction d'un protocole expérimental avec une psycho-ergonome.

Les tâches de pilotage ont été construites avec deux pilotes instructeurs, pour garantir leur pertinence opérationnelle. Nous avons demandé l'ajout de scénarios se rapprochant d'une perte de contrôle, afin de faire une évaluation dans cette situation délicate. Quinze sujets, jeunes pilotes, ont réalisé les 7 scénarios avec chacun des deux instruments, l'instrument occident actuel et le nouvel instrument. A la date de rédaction de ce manuscrit, les données n'ont pas été analysées. Les résultats devraient permettre d'évaluer la maturité les exigences de haut niveau et les exigences détaillées.



| Tâche de pilotage ( virage de 360°) | | | | | | |
|---|---|---|---|---|---|---|
| | Inclinaison(°) | Assiette (°) | Altitude | Vitesse (kt) | Vz (ft/min) | Dérapage |
| Fam1 | 30 | 0 | // | // | // | // |
| N1P | 45 | palier | stable | 100 | // | 0 |
| N2P | 40 | palier | stable | 100 | // | 0 |
| DT1P | 20 | // | // | 100 | // | 0 |
| DT2P | 20 | // | // | // | -500 | 0 |
| LC1P | Initialisation : virage 60°, assiette -20°, vitesse 130kt >> Ramener l'avion a cap constant en parlier | | | | | |
| LC2P | Initialisation : virage 60°, assiette +20°, vitesse 130kt >> Ramener l'avion a cap constant en parlier | | | | | |

Figure 153: intégration des prototypes avec un simulateur de vol, conception graphique et protocole expérimental pour évaluer le nouvel instrument de pilotage

### VI.3.4.3. Synthèse et obstacles à la validité

Les résultats du processus (synthèse en Tableau 12) montrent que l'ingénieur en exigences peut apporter une plus-value à une idée de solution émise par un utilisateur expert grâce aux techniques déjà identifiées dans l'ingénierie participative des exigences (observations, cycle de l'action, prototypage et idéation, évaluation), auxquelles nous avons adjoint un cadre supplémentaire de l'interaction : ScanVis, pour la décomposition du parcours visuel dans une représentation. La décomposition a été conduite à partir de l'interview utilisateur. Les principes de visualisation

d'information ont été mobilisés pour l'idéation, au détriment des principes de collaboration directe qui restent à explorer pour cet instrument. La validation des exigences détaillées passe par une évaluation contrôlée de la solution avec des utilisateurs (apprentis-pilotes), dans laquelle le scénario de perte de contrôle est intégré.

| **Activités** | **Résultats** | **Moyens de l'ingénierie participative des exigences mis en œuvre** |
|---|---|---|
| Observation de situations non prévues dans le système actuel mais gérées par les utilisateurs | Utilisation intensive de l'instrument de vol en condition de vol à vue et avion léger: apprentissage de la mise en virage | Rapport d'analyse d'accident Crossair 498<br>interview d'un pilotes instructeur hors contexte centrée sur les actions et ressources des actions (instrument de vol, capot, environnement)<br>Cycle de l'action pour conduire l'interview<br>ScanVis pour conduire l'interview sur le parcours visuel |
| Expression des exigences de haut niveau sur le système pour la gestion des situations non prévues | Exigences présentées dans le corps du document | Analyse des actions décomposées selon le cycle de l'action et ScanVis |
| Génération et évaluation de solutions répondant aux situations non prévues | Louviot, Denis, Hélène Gaspard-Boulinc, et Stéphane Conversy. Method and apparatus for controlling vehicle attitude display. European Patent Office 16306781.2, filed 2016, et issued 2017. | idéation et prototypage logiciel (djnn) avec un pilote instructeur et un chercheur en Visualisation d'Information<br>Conception graphique par un graphiste<br><br>Évaluation a priori avec ScanVis<br><br>Evaluation contrôlée sur la base d'une plate-forme légère de simulation<br>Protocole expérimental intégrant les scénarios de perte de contrôle |
| Expression des exigences détaillées pour la gestion des situations non prévues | Présentées dans le corps du document | Principes de visualisation d'informations<br><br>ScanVis |

Tableau 12: synthèse des résultats du processus d'ingénierie des exigences "nouvel instrument de vol"

# VI.4. Synthèse

Nous venons de décrire quatre processus d'ingénierie des exigences, avec des degrés de participation plus ou moins grands des utilisateurs. Les résultats montrent la possibilité d'exprimer des exigences de haut niveau et des scénarios à partir d'une approche ethnographique centrée sur le système actuel et les ressources pour l'action mobilisées par les utilisateurs.

A partir des scénarios et des principes de collaboration directe, les ingénieurs en exigences peuvent concevoir et évaluer avec les utilisateurs les interactions et les représentations du futur système. Le prototypage et l'idéation peuvent engager les utilisateurs dans une démarche d'innovation, avec cependant des écueils que les ingénieurs en exigences doivent connaître et maitriser. La maitrise passe par la mobilisation de cadres de l'interaction et de la représentation, tels que les principes de collaboration directe, d'utilisabilité et les variables visuelles.

A partir des prototypes, les ingénieurs en exigences peuvent transposer les scénarios initiaux en cas d'utilisation et affiner les exigences de haut niveau en exigences détaillées pour exprimer le comportement du système. Les contextes d'utilisation peuvent être utilisés pour structurer la spécification. Les ingénieurs en exigences peuvent avoir un degré de confiance plus grand dans les

exigences ainsi exprimées, car issues d'un travail conjoint avec les utilisateurs. Une évaluation semi-contrôlée ou contrôlée des prototypes avec les utilisateurs peut permettre de valider les exigences détaillées, avec des données objectives sur les performances et les bénéfices.

La description des quatre processus montre des séquences différentes dans les activités menées, dépendant notamment des données en entrée du processus : un nouveau dispositif technologique pour MAMMI, un cahier des charges exprimant une mission pour E-Fan 2.0, un rapport d'analyse d'accident et une solution conçue par un utilisateur pour le nouvel instrument. Il n'y a pas un processus d'ingénierie participative des exigences, mais plusieurs chemins possibles entre les parties prenantes de l'ingénierie des exigences (voir  Figure 154).

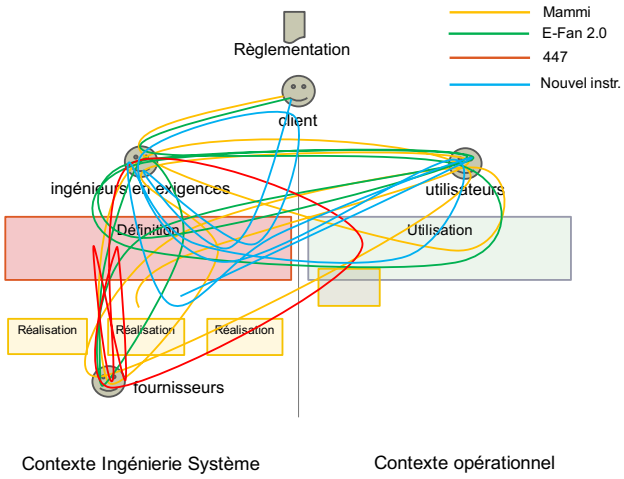



Figure 154: les chemins suivis entre parties prenantes des cas illustrant l'ingénierie participative des exigences

La première partie de cette thèse pourrait être considérée comme un cas illustrant l'ingénierie participative des exigences, dans lequel les utilisateurs sont les ingénieurs en exigences, et nous sommes dans la position d'un ingénieur en exigences. Cependant, nous considérons ce cas comme non significatif : la thèse est un travail personnel, sans client et sans fournisseurs et notre enjeu n'était pas de construire un système, mais de démontrer la contribution de l'utilisabilité des outils dans le processus d'ingénierie des exigences. Ce travail a conduit à une vision située de l'ingénierie des exigences.

Dans l'état actuel de nos connaissances, nous pouvons synthétiser le processus d'ingénierie participative des exigences dans notre vision située selon la Figure 155 : les ingénieurs en exigences pilotent un processus continu de définition du système futur informée par l'utilisation du système actuel, par la maitrise des techniques représentées dans les encadrés jaunes, avec une participation des utilisateurs, des fournisseurs et du client.

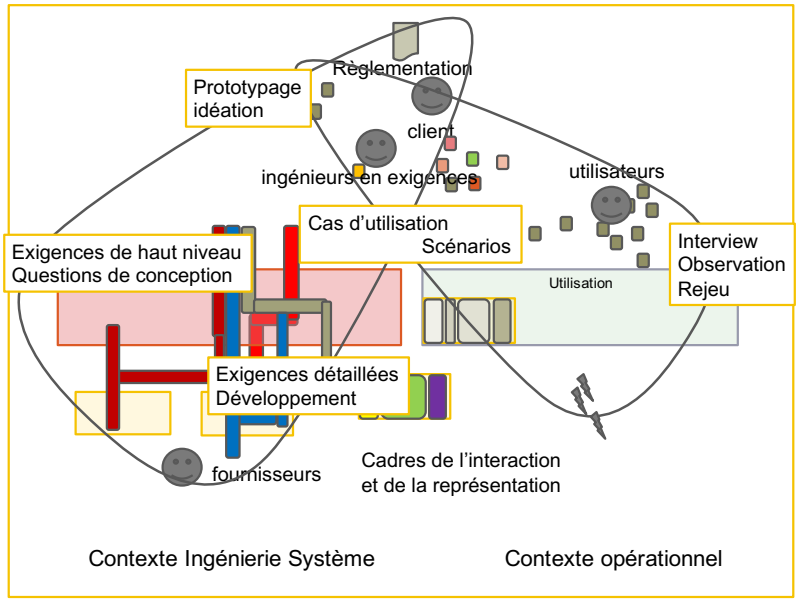



Figure 155: le processus d'ingénierie participative des exigences

Le lecteur averti notera que notre schéma est encadré : cela représente les cadres de l'interaction et de la représentation qui *guident* le pilotage du processus d'ingénierie participative des exigences. En effet, en énumérant les différentes techniques, nous proposons :

- Interview et observation centrées sur les *ressources pour l'action*, et l'identification de *contextes* différents selon les dimensions de l'utilisabilité : utilisateur, environnement, équipement ;
- Scénarios restituant les actions et les ressources utilisées ; analyse des scénarios selon les dimensions cognitives des notations, le cycle de l'action et ScanVis ;
- Prototypage et idéation mobilisant les principes d'utilisabilité, de collaboration directe, de visualisation d'information et des variables visuelles ;
- Cas d'utilisation structurés par *contexte* ;
- Exigences affinées détaillant les *interactions* entre acteur et système et les *représentations* ;
- Choix du formalisme d'expression des exigences pour les utilisateurs et pour les fournisseurs selon la *facilité de compréhension et de modification*.

En déplaçant l'ingénieur en exigences au centre de l'ingénierie participative des exigences, il devient l'assistant du client dans l'orientation et la prise de décision sur le budget et les priorités, à l'aide des visualisations interactives des exigences présentées dans le chapitre 5.

# Conclusion et Perspectives

## Contributions

Le processus de la thèse nous a conduit à une réflexion sur l'ingénierie des systèmes interactifs complexes et le métier de l'ingénieur en exigences. Les enjeux de sécurité sont cadrés par une réglementation imposant la production de preuves de conformité du système aux exigences spécifiées. Cette production de preuves est consommatrice d'énergie pour les ingénieurs, du fait d'outils inadaptés. La conséquence est moins de temps passé sur la compréhension des problèmes des utilisateurs et la recherche de solutions innovantes : on va choisir un équipement déjà certifié, même s'il n'est pas complètement satisfaisant pour les utilisateurs. Or comprendre les problèmes des utilisateurs et proposer de nouvelles solutions peut également contribuer à la sécurité. Nos travaux de thèse visent à rééquilibrer le métier de l'ingénieur : passer moins de temps sur la production de preuves, grâce à des outils supportant le processus d'ingénierie des exigences, et plus de temps sur la compréhension des problèmes des utilisateurs et la conception de solutions, grâce à l'ingénierie participative des exigences.

**RQ1 : Quelles sont les activités d'ingénierie des exigences réalisées par les ingénieurs en aéronautique ?**

**RQ2 : Comment les outils supportent-ils ces activités ?**

Durant ces travaux de thèse, nous avons réalisé une étude qualitative, à base d'interviews contextuelles et de prototypes, auprès de 15 praticiens industriels de quatre entreprises aéronautiques, afin d'enquêter sur les activités réellement effectuées par les ingénieurs en exigences et sur le support outillé de ces activités. Nous avons trouvé que les outils spécifiques à l'ingénierie des exigences contraignent les ingénieurs à un flux de travail rigide, qui est en conflit avec une exploration adaptative des problèmes de conception. Les ingénieurs commencent souvent par utiliser des outils à vocation générale pour favoriser l'exploration et la collaboration avec les fournisseurs, au détriment de la traçabilité. Quand les ingénieurs basculent sur le raffinement et la vérification des exigences, ils doivent utiliser des outils spécifiques pour garantir la traçabilité. L'insuffisance de l'utilisabilité de ces outils entraine une perte de temps significative et une insatisfaction.

Sur la base de nos observations et de nos scénarios, nous avons proposé une vision située de l'ingénierie des exigences afin de retranscrire la richesse des situations observées. Nous avons mis en valeur les parties prenantes (client, utilisateurs, ingénieur en exigences, fournisseurs) et leur champ de travail commun : le système, en utilisation et en définition. Les exigences sont un mécanisme de coordination entre les parties prenantes, et ne constituent pas la véritable base du travail collaboratif. Seuls les ingénieurs en exigences utilisent les exigences comme base de travail collaboratif entre eux, dans les activités de raffinement et de vérification. Nous avons formulé des exigences d'utilisabilité de support au travail collaboratif entre ingénieurs en exigences et fournisseurs, en considérant les exigences comme une articulation entre d'autres artefacts d'ingénierie, tels que les schémas d'architecture et les questions de conception (Hélène Gaspard-Boulinc et Conversy 2014).

**RQ3 : Comment améliorer l'utilisabilité des outils d'ingénierie des exigences, en exploitant les principes de la visualisation d'information ?**

Sur la base de scénarios et de prototypes, nous avons formulé des exigences d'utilisabilité pour les activités de raffinement et de vérification des exigences. Nous proposons plus particulièrement des visualisations interactives et coordonnées de texte structuré permettant aux ingénieurs de décorréler rigueur et rigidité dans le processus d'ingénierie des exigences, en rendant possible une souplesse pendant le processus tout en éliminant progressivement toute approximation en sortie du processus. L'ingénieur bénéfice de visualisations structurées des exigences, à partir desquelles il peut continuer à communiquer avec les parties prenantes, chercher du texte, voir l'état d'avancement des exigences, détecter et compléter les informations manquantes par une navigation et un filtrage interactifs sur les visualisations (H. Gaspard-Boulinc et Conversy 2017).

Les retours d'expérience sur l'utilisation des visualisations interactives sont prometteurs, mais restent à consolider. La difficulté est de mesurer de façon objective les bénéfices des visualisations dans une expérimentation contrôlée, le processus d'ingénierie des exigences mobilisant de nombreuses parties prenantes. Ce souci est reconnu et partagé dans la communauté de *Requirements Engineering* (Ivarsson et Gorschek 2009). Nous allons poursuivre le travail sur les visualisations interactives avec l'entreprise D, pour une mise à disposition sur un intranet à destination des différentes parties prenantes : client, sites opérationnels et fournisseurs de composants.

**RQ4 : Comment faire participer les utilisateurs à la spécification du système futur à partir de leurs usages du système actuel ?**

Au-delà des outils, nous proposons une nouvelle approche : l'ingénierie participative des exigences.

Elle est basée sur une articulation de techniques utilisées en conception participative pour impliquer les utilisateurs, avec un effort continu d'abstraction et de formalisation des exigences pour informer la définition du système. La finalité est une production d'exigences matures spécifiant dans le système futur la prise en compte de situations non prévues dans le système actuel, mais gérées par les utilisateurs. Nous avons illustré l'application de notre approche sur quatre projets aéronautiques : collaboration et contrôle aérien, cockpit d'avion-école électrique, analyse de rapport d'accident et nouvel instrument de vol. Selon le degré de formalisation et le niveau de détail des exigences, nous pouvons qualifier notre approche de conception participative plus que d'ingénierie participative des exigences (Contrôle aérien et collaboration). Selon le degré de participation des utilisateurs, nous pouvons qualifier l'ingénierie des exigences de centrée-utilisateur (Analyses d'incidents et d'accident : cas du rapport d'accident du vol 447 Rio-Paris ) plus que de participative (Avion électrique et gestion de l'énergie). Une participation effective, alliée à un effort continu de prototypage et de formalisation d'exigences, peut conduire à des exigences détaillées plus matures et moins susceptibles de changer. Elle peut constituer également une démarche d'innovation, par une réflexion autour de solutions alimentée par des problèmes réels (Pujos et al. 2016) (Hélène Gaspard-Boulinc, Conversy, et al. 2016) (Louviot, Gaspard-Boulinc, et Conversy 2017). Elle comporte des risques : des erreurs de conception dans les prototypes peuvent impliquer des discussions infructueuses avec les utilisateurs. Il faut maitriser ces risques par une mobilisation des principes de conception connus, et le positionnement des prototypes comme un support de discussion et non une proposition ferme de solution.

# Perspectives croisées entre IHM et Ingénierie des Exigences

Notre thèse est à la croisée de deux domaines : Ingénierie des Exigences (*Requirements Engineering*) et Interaction Humain-Machine (*Human-Computer Interaction*).

## L'IHM pour l'Ingénierie des Exigences

Nous avons utilisé une méthode d'étude de cas, enrichie par le design d'artefacts, pour comprendre le phénomène de l'ingénierie des exigences. Nous avons utilisé un cadre de travail coopératif (*Computer Supported Cooperative Work*) pour construire une vision située de l'ingénierie des exigences. Nous avons conçu des visualisations interactives et coordonnées de texte structuré, offrant aux ingénieurs en exigences un support à la communication entre parties prenantes et instrumentant la vérification des exigences. L'IHM peut contribuer à l'Ingénierie des Exigences, par une amélioration accrue de la lisibilité et de la modificabilité des différents modes d'expression des exigences (chapitre 2- partie Les différentes expressions des exigences).

Nous nous sommes centrés sur les exigences textuelles, mais les autres modes d'expression peuvent bénéficier d'une réflexion sur l'utilisabilité des outils les mettant en œuvre : l'adoption d'un mode d'expression d'exigences est en partie liée à son support outillé.

## L'ingénierie des Exigences pour l'IHM

Nous avons élaboré une ingénierie participative des exigences, visant à spécifier dans le système futur la prise en compte de situations non prévues dans le système actuel, mais gérées par les utilisateurs. Nous avons un projet en cours dans le domaine de la santé avec l'hôpital méthodiste de Houston dans lequel nous mettons en œuvre une ingénierie participative des exigences (Joerger et al. 2017). Ce projet se caractérise par une hétérogénéité des utilisateurs, allant du brancardier au chirurgien, en passant par les infirmières et les familles des patients. Les futurs résultats pourront illustrer l'application de l'ingénierie participative des exigences dans un domaine critique hors aéronautique, et accroitre nos connaissances sur les techniques à maitriser et les cadres de l'interaction et de la représentation.

En particulier, l'ingénierie participative des exigences constitue pour nous un cadre de réflexion et de mise en œuvre des principes de collaboration directe dans les systèmes interactifs critiques :

- des surfaces partagées interactives ;
- la réification des actions dans des instruments ;
- l'accomplissement partiel des actions ;
- un retour d'information continu pour une visibilité des actions.

Nous cherchons à démontrer leur contribution à la sécurité des systèmes interactifs complexes, au-delà de l'utilisabilité.

## Le support à l'abduction : un défi pour l'IHM et l'ingénierie des exigences

L'analyse du rapport d'accident du vol 447 Rio-Paris nous a conduit à mettre en évidence un raisonnement d'abduction mené par les pilotes pour poser un diagnostic sur la situation (Conversy et al. 2014). Nous avons identifié des questions de recherche sur l'IHM de support à l'abduction (partie VI.3.3) : de même que les médecins disposent d'instruments pour examiner un patient en minimisant l'impact sur son état de santé, il faut inventer les instruments de diagnostic sur un système pour ses utilisateurs en temps contraint et en situation dégradée.

Ces instruments existent pour les ingénieurs, en support aux phases d'essais et d'analyses a posteriori, dans un contexte non critique. Dans le domaine de l'Interaction Humain-Machine, les représentations et les interactions pour mener un raisonnement d'abduction sur une situation dégradée en temps contraint restent à inventer. Dans le domaine de l'Ingénierie des Exigences, un support à l'abduction pour les utilisateurs peut être considéré comme une technique de mitigation de l'incertitude à l'exécution du système, mais n'ajoute-t-elle pas de la complexité au système ? Les deux domaines ont un champ de travail commun : les systèmes interactifs complexes.

## L'ingénierie participative des exigences : un humanisme

Les valeurs que nous défendons à travers l'ingénierie participative des exigences sont portées par une vision humaniste du progrès technologique. A travers les cas illustrés que nous avons développés, nous voulons montrer que le progrès technologique n'est pas synonyme d'automatisation généralisée et de procéduralisation des interactions entre les êtres humains. Tout est une question de choix dans l'intégration des nouvelles technologies. L'ingénierie participative des exigences favorise la co-construction de ces choix entre utilisateurs et ingénieurs, en inventant les usages et en les traduisant en spécification sur les systèmes.

La technologie d'édition de l'ADN CRISPR-Cas9 (http://www.sciencemag.org/topic/crispr) fournit un contrôle sur les gênes et conduit au *Genome Engineering* (Cong et al. 2013). L'intelligence artificielle connaît de grandes avancées en apprentissage automatique et conduit à la construction de *systèmes autonomes* (Inria 2016). L'ingénierie participative des exigences peut être une approche de maitrise de ces nouvelles technologies au service de l'Humanité, et alimenter une réflexion sur leur règlementation.

# Bibliographie

Abad, Z. S. H., M. Noaeen, et G. Ruhe. 2016. « Requirements Engineering Visualization: A Systematic Literature Review ». In *2016 IEEE 24th International Requirements Engineering Conference (RE)*, 6‑15. https://doi.org/10.1109/RE.2016.61.

Abello, J., F. V. Ham, et N. Krishnan. 2006. « ASK-GraphView: A Large Scale Graph Visualization System ». *IEEE Transactions on Visualization and Computer Graphics* 12 (5): 669‑76. https://doi.org/10.1109/TVCG.2006.120.

Abrial, Jean-Raymond. 2006. « Formal Methods in Industry: Achievements, Problems, Future ». In *Proceedings of the 28th International Conference on Software Engineering*, 761‑68. ICSE '06. New York, NY, USA: ACM. https://doi.org/10.1145/1134285.1134406.

Abrial, J.-R. 1996. *The B-book: Assigning Programs to Meanings*. New York, NY, USA: Cambridge University Press.

Achour, Camille Ben, et Colette Rolland. 1997. « Introducing Genericity and Modularity of Textual Scenario Interpretation in the Context of Requirements Engineering ».

Alexander, I. s. d. « Requirements Management Tools - Vendors and Freeware Suppliers ». Consulté le 6 juin 2017. http://www.scenarioplus.org.uk/vendors.htm.

Alexander, I., S. Robertson, et N. Maiden. 2005. « What influences the requirements process in industry? A report on industrial practice ». In *13th IEEE International Conference on Requirements Engineering (RE'05)*, 411‑15. https://doi.org/10.1109/RE.2005.79.

Amyot, Daniel. 2003. « Introduction to the User Requirements Notation: learning by example ». *Computer Networks*, ITU-T System Design Languages (SDL), 42 (3): 285‑301. https://doi.org/10.1016/S1389-1286(03)00244-5.

Amyot, Daniel, Jennifer Horkoff, Daniel Gross, et Gunter Mussbacher. 2009. « A Lightweight GRL Profile for I* Modeling ». In *SpringerLink*, 254‑64. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-04947-7_31.

Amyot, Daniel, et Gunter Mussbacher. 2000. « On the Extension of UML with Use Case Maps Concepts ». In *Proceedings of the 3rd International Conference on The Unified Modeling Language: Advancing the Standard*, 16‑31. UML'00. Berlin, Heidelberg: Springer-Verlag. http://dl.acm.org/citation.cfm?id=1765175.1765178.

Amyot, D., L. Logrippo, R. J. A. Buhr, et T. Gray. 1999. « Use case maps for the capture and validation of distributed systems requirements ». In *Proceedings IEEE International Symposium on Requirements Engineering (Cat. No.PR00188)*, 44‑53. https://doi.org/10.1109/ISRE.1999.777984.

Anderson, David J. 2010. *Kanban: Successful Evolutionary Change for Your Technology Business*. Blue Hole Press.

« ANSI/EIA 632 - Processes for Engineering a System ». s. d. Consulté le 20 avril 2017. http://sebokwiki.org/wiki/ANSI/EIA_632.

Anwer, S., et M. El-Attar. 2014. « An Evaluation of the Statechart Diagrams Visual Syntax ». In *2014 International Conference on Information Science Applications (ICISA)*, 1‑4. https://doi.org/10.1109/ICISA.2014.6847358.

ASTM International. 2016. « ASTM F2245-16c, Standard Specification for Design and Performance of a Light Sport Airplane ». West Conshohocken, PA. http://www.astm.org/cgi-bin/resolver.cgi?F2245.

Aurum, Aybüke, et Claes Wohlin. 2005. « Requirements Engineering: Setting the Context », 1‑15. https://doi.org/10.1007/3-540-28244-0_1.

Barel, Y. 1971. « Prospective et analyse de système ». 1971. http://www.laprospective.fr/dyn/francais/memoire/trp/prospective-et-analyse-systeme-trp-14-barel-1971.pdf.

Beaudouin-Lafon, Michel. 2000. « Instrumental Interaction: An Interaction Model for Designing post-WIMP User Interfaces ». In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 446‑53. CHI '00. New York, NY, USA: ACM. https://doi.org/10.1145/332040.332473.

———. 2004. « Designing Interaction, Not Interfaces ». In *Proceedings of the Working Conference on Advanced Visual Interfaces*, 15‑22. AVI '04. New York, NY, USA: ACM. https://doi.org/10.1145/989863.989865.

Beaudouin-Lafon, Michel, et Wendy Mackay. 2002. « Prototyping tools and techniques ». In , 1006‑31. L. Erlbaum Associates Inc. http://dl.acm.org.proxy.lib.enac.fr/citation.cfm?id=772072.772136.

Beaudouin-Lafon, Michel, et Wendy E. Mackay. 2000. « Reification, Polymorphism and Reuse: Three Principles for Designing Visual Interfaces ». In *Proceedings of the Working Conference on Advanced Visual Interfaces*, 102‑9. AVI '00. New York, NY, USA: ACM. https://doi.org/10.1145/345513.345267.

Beller, Johannes, Matthias Heesen, et Mark Vollrath. 2013. « Improving the Driver–Automation Interaction: An Approach Using Automation Uncertainty ». *Human Factors* 55 (6): 1130‑41. https://doi.org/10.1177/0018720813482327.

Benkirane, Réda. 2006. *La complexité, vertiges et promesses*. Le Pommier. http://lefur.jean.free.fr/1jean/thesauru/complexite/NotesLectureBenkirane.JLF.htm.

Berry, Daniel, Ricardo Gacitua, Pete Sawyer, et Sri Fatimah Tjong. 2012. « The Case for Dumb Requirements Engineering Tools ». In *Proceedings of the 18th International Conference on Requirements Engineering: Foundation for Software Quality*, 211‑17. REFSQ'12. Berlin, Heidelberg: Springer-Verlag. https://doi.org/10.1007/978-3-642-28714-5_18.

Berry, Daniel M., et Erik Kamsties. 2004. « Ambiguity in Requirements Specification ». In *Perspectives on Software Requirements*, édité par Julio Cesar Sampaio do Prado Leite et Jorge Horacio Doorn, 7‑44. The Springer International Series in Engineering and Computer Science 753. Springer US. http://link.springer.com/chapter/10.1007/978-1-4615-0465-8_2.

Bertalanffy, Ludwig von. 1973. *Théorie générale des systèmes*. Dunod. https://www.panarchy.org/vonbertalanffy/systems.1968.html.

Bertin, Jacques. 1983. *Semiology of Graphics*. University of Wisconsin Press.

Blackwell, Alan, et Thomas Green. 2012. « A Cognitive Dimensions Questionnaire Optimised for Users ». In *PPIG*, 137‑54. A.F. Blackwell & E. Bilotta (Eds). https://www.researchgate.net/publication/2464079_A_Cognitive_Dimensions_Questionnaire_Optimised_for_Users.

Blanch, Renaud, et Eric Lecolinet. 2007. « Browsing Zoomable Treemaps: Structure-Aware Multi-Scale Navigation Techniques ». *IEEE Transactions on Visualization and Computer Graphics* 13 (6): 1248‑53. https://doi.org/10.1109/TVCG.2007.70540.

Bødker, S. 2000. « Scenarios in user-centred design—setting the stage for reflection and action ». *Interacting with Computers* 13 (1): 61‑75. https://doi.org/10.1016/S0953-5438(00)00024-2.

Bødker, Susanne, et Ole Sejer Iversen. 2002. « Staging a Professional Participatory Design Practice: Moving PD Beyond the Initial Fascination of User Involvement ». In *Proceedings of the Second Nordic Conference on Human-computer Interaction*, 11‑18. NordiCHI '02. New York, NY, USA: ACM. https://doi.org/10.1145/572020.572023.

Boehm, Barry. 2006. « Some Future Trends and Implications for Systems and Software Engineering Processes ». *Syst. Eng.* 9 (1): 1‑19. https://doi.org/10.1002/sys.v9:1.

Boehm, Barry, et Richard Turner. 2003. *Balancing Agility and Discipline: A Guide for the Perplexed*. 1re éd. Boston: Addison Wesley.

Bostock, M., V. Ogievetsky, et J. Heer. 2011. « D3 Data-Driven Documents ». *IEEE Transactions on Visualization and Computer Graphics* 17 (12): 2301‑9. https://doi.org/10.1109/TVCG.2011.185.

Boucher, Quentin. s. d. « Visually Effective TROPOS Models ». Consulté le 8 juin 2017. http://www.academia.edu/2870983/Visually_Effective_TROPOS_Models.

Bourgeois, Thomas, et Dominique Lecourt. 2006. *Dictionnaire d'histoire et philosophie des sciences*. 4e édition revue et augmentée. Paris: Presses Universitaires de France - PUF.

Braun, Edna, Daniel Amyot, et Timothy C. Lethbridge. 2015. « Generating Software Documentation in Use Case Maps from Filtered Execution Traces ». In *SpringerLink*, 177‑92. Springer, Cham. https://doi.org/10.1007/978-3-319-24912-4_13.

Brooks, F. P. J. 1987. « No Silver Bullet Essence and Accidents of Software Engineering ». *Computer* 20 (4): 10‑19. https://doi.org/10.1109/MC.1987.1663532.

———. 1995. *The Mythical Man-Month: Essays on Software Engineering*. Addison Wesley Longman. https://www.researchgate.net/publication/220689892_The_Mythical_Man-Month_Essays_on_Software_Engineering.

Buhr, R. J. A. 1998. « Use case maps as architectural entities for complex systems ». *IEEE Transactions on Software Engineering* 24 (12): 1131‑55. https://doi.org/10.1109/32.738343.

Burkhard, Remo Aslak. 2005. « Towards a Framework and a Model for Knowledge Visualization: Synergies Between Information and Knowledge Visualization ». In *Knowledge and Information Visualization*, édité par Sigmar-Olaf Tergan et Tanja Keller, 238‑55. Lecture Notes in Computer Science 3426. Springer Berlin Heidelberg. http://link.springer.com.proxy.lib.enac.fr/chapter/10.1007/11510154_13.

Burnay, C., J. Horkoff, et N. Maiden. 2016. « Stimulating Stakeholders' Imagination: New Creativity Triggers for Eliciting Novel Requirements ». In *2016 IEEE 24th International Requirements Engineering Conference (RE)*, 36‑45. https://doi.org/10.1109/RE.2016.36.

Bustard, D. 2012. « Beyond Mainstream Adoption: From Agile Software Development to Agile Organizational Change ». In *2012 IEEE 19th International Conference and Workshops on Engineering of Computer-Based Systems*, 90‑97. https://doi.org/10.1109/ECBS.2012.18.

Caire, P., N. Genon, P. Heymans, et D. L. Moody. 2013. « Visual notation design 2.0: Towards user comprehensible requirements engineering notations ». In *2013 21st IEEE International Requirements Engineering Conference (RE)*, 115‑24. https://doi.org/10.1109/RE.2013.6636711.

Card, Stuart K., Jock D. Mackinlay, et Ben Shneiderman, éd. 1999. *Readings in Information Visualization: Using Vision to Think*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Card, Stuart K., Allen Newell, et Thomas P. Moran. 1983. *The Psychology of Human-Computer Interaction*. Hillsdale, NJ, USA: L. Erlbaum Associates Inc.

Carrillo De Gea, Juan M., JoaquíN NicoláS, José L. FernáNdez AlemáN, Ambrosio Toval, Christof Ebert, et Aurora VizcaíNo. 2012. « Requirements Engineering Tools: Capabilities, Survey and Assessment ». *Inf. Softw. Technol.* 54 (10): 1142‑57. https://doi.org/10.1016/j.infsof.2012.04.005.

Carrillo de Gea, Juan M., JoaquíN NicoláS, José L. Fernández Alemán, et Aurora Vizcaíno. 2014. « Commonalities and Differences between Requirements Engineering Tools: A Quantitative Approach ». *Computer Science and Information Systems 12(1)*, 2014.

Carroll, John M. 2000. *Making Use: Scenario-Based Design of Human-Computer Interactions*. 1st éd. The MIT Press.

Castro, Jaelson, Manuel Kolp, Lin Liu, et Anna Perini. 2009. « Dealing with Complexity Using Conceptual Models Based on Tropos ». In *Conceptual Modeling: Foundations and Applications*, édité par Alexander T. Borgida, Vinay K. Chaudhri, Paolo Giorgini, et Eric S. Yu, 335‑62. Lecture Notes in Computer Science 5600. Springer Berlin Heidelberg. http://link.springer.com/chapter/10.1007/978-3-642-02463-4_18.

Chatty, Stéphane, Mathieu Magnaudet, Daniel Prun, Stéphane Conversy, Stéphanie Rey, et Mathieu Poirier. 2016. « Designing, developing and verifying interactive components iteratively with djnn ». In *8th European Congress on Embedded Real Time Software and Systems (ERTS 2016)*. TOULOUSE, France. https://hal.archives-ouvertes.fr/hal-01292291.

Chatzoglou, Prodromos D. 1997. « Factors affecting completion of the requirements capture stage of projects with different characteristics ». *Information and Software Technology* 39 (9): 627‑40. https://doi.org/10.1016/S0950-5849(97)00020-7.

Cheng, Betty H. C., et Joanne M. Atlee. 2007. « Research Directions in Requirements Engineering ». In , 285‑303. IEEE Computer Society. https://doi.org/10.1109/FOSE.2007.17.

Chen, X., J. Hosking, et J. Grundy. 2012. « Visualizing traceability links between source code and documentation ». In *2012 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 119‑26. https://doi.org/10.1109/VLHCC.2012.6344496.

Cleland-Huang, Jane, Orlena C. Z. Gotel, Jane Huffman Hayes, Patrick Mäder, et Andrea Zisman. 2014. « Software Traceability: Trends and Future Directions ». In *Proceedings of the on Future of Software Engineering*, 55‑69. FOSE 2014. New York, NY, USA: ACM. https://doi.org/10.1145/2593882.2593891.

Cleland-Huang, Jane, et Rafal Habrat. 2007. « Visual Support In Automated Tracing ». In *Proceedings of the Second International Workshop on Requirements Engineering Visualization*, 4 ‑ . REV ’07. Washington, DC, USA: IEEE Computer Society. https://doi.org/10.1109/REV.2007.7.

Cockburn, Alistair. 2002. *Agile Software Development*. Addison-Wesley.

Condamines, Anne, et Maxime Warnier. 2017. « Towards the Creation of a CNL Adapted to Requirements Writing by Combining Writing Recommendations and Spontaneous Regularities: Example in a Space Project ». *Language Resources and Evaluation* 51 (1): 221‑47. https://doi.org/10.1007/s10579-016-9368-1.

Cong, Le, F. Ann Ran, David Cox, Shuailiang Lin, Robert Barretto, Naomi Habib, Patrick D. Hsu, et al. 2013. « Multiplex Genome Engineering Using CRISPR/Cas Systems ». *Science* 339 (6121): 819‑23. https://doi.org/10.1126/science.1231143.

Conklin, E. Jeffrey, et K. C. Burgess Yakemovic. 1991. « A Process-oriented Approach to Design Rationale ». *Hum.-Comput. Interact.* 6 (3): 357‑91. https://doi.org/10.1207/s15327051hci0603&4_6.

Constantine, Larry. 2011. « Activity-Centered Interaction Design: A Model-Driven Approach ». In *Human-Computer Interaction – INTERACT 2011*, 696‑97. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-23768-3_120.

Constantine, Larry L. 2009. « Human Activity Modeling: Toward A Pragmatic Integration of Activity Theory and Usage-Centered Design ». In *Human-Centered Software Engineering*, 27‑51. Human-Computer Interaction Series. Springer, London. https://link-springer-com.proxy.lib.enac.fr/chapter/10.1007/978-1-84800-907-3_3.

Constantine, Larry L., et Lucy A. D. Lockwood. 1999. *Software for Use: A Practical Guide to the Models and Methods of Usage-centered Design*. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co.

———. 2003. « Usage-centered Software Engineering: An Agile Approach to Integrating Users, User Interfaces, and Usability into Software Engineering Practice ». In *Proceedings of the 25th International Conference on Software Engineering*, 746‑47. ICSE '03. Washington, DC, USA: IEEE Computer Society. http://dl.acm.org/citation.cfm?id=776816.776931.

Constantine, L. L., et L. A. D. Lockwood. 2003. « Usage-centered software engineering: an agile approach to integrating users, user interfaces, and usability into software engineering practice ». In *25th International Conference on Software Engineering, 2003. Proceedings.*, 746‑47. https://doi.org/10.1109/ICSE.2003.1201267.

Conversy, Stéphane. 2014a. « Unifying Textual and Visual: A Theoretical Account of the Visual Perception of Programming Languages ». In *Proceedings of the 2014 ACM International Symposium on New Ideas, New Paradigms, and Reflections on Programming & Software*, 201‑12. Onward! 2014. New York, NY, USA: ACM. https://doi.org/10.1145/2661136.2661138.

———. 2014b. « Unifying Textual and Visual: A Theoretical Account of the Visual Perception of Programming Languages ». In *Proceedings of the 2014 ACM International Symposium on New Ideas, New Paradigms, and Reflections on Programming & Software*, 201‑12. Onward! 2014. New York, NY, USA: ACM. https://doi.org/10.1145/2661136.2661138.

Conversy, Stéphane, Stéphane Chatty, Hélène Gaspard-Boulinc, et Jean-Luc Vinot. 2014. « The Accident of Flight AF447 Rio-Paris: A Case Study for HCI Research ». In *Proceedings of the 26th Conference on L'Interaction Homme-Machine*, 60‑69. IHM '14. New York, NY, USA: ACM. https://doi.org/10.1145/2670444.2670459.

Conversy, Stéphane, Hélène Gaspard-Boulinc, Stéphane Chatty, Stéphane Valès, Carole Dupré, et Claire Ollagnon. 2011. « Supporting Air Traffic Control Collaboration with a TableTop System ». In

*Proceedings of the ACM 2011 Conference on Computer Supported Cooperative Work*, 425‑34. CSCW ’11. New York, NY, USA: ACM. https://doi.org/10.1145/1958824.1958891.

Cooper, John R., Jr., Seok-Won Lee, Robin A. Gandhi, et Orlena Gotel. 2009. « Requirements Engineering Visualization: A Survey on the State-of-the-Art ». In *Proceedings of the 2009 Fourth International Workshop on Requirements Engineering Visualization*, 46‑55. REV ’09. Washington, DC, USA: IEEE Computer Society. https://doi.org/10.1109/REV.2009.4.

Cordeil, M., T. Dwyer, K. Klein, B. Laha, K. Marriott, et B. H. Thomas. 2017. « Immersive Collaborative Analysis of Network Connectivity: CAVE-style or Head-Mounted Display? » *IEEE Transactions on Visualization and Computer Graphics* 23 (1): 441‑50. https://doi.org/10.1109/TVCG.2016.2599107.

Damas, Christophe, Bernard Lambeau, et Axel van Lamsweerde. 2006. « Scenarios, Goals, and State Machines: A Win-win Partnership for Model Synthesis ». In *Proceedings of the 14th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, 197‑207. SIGSOFT ’06/FSE-14. New York, NY, USA: ACM. https://doi.org/10.1145/1181775.1181800.

Damm, Werner, et David Harel. 1999. « Lsc’s: Breathing Life Into Message Sequence Charts ». In *Formal Methods for Open Object-Based Distributed Systems*, 293‑311. IFIP — The International Federation for Information Processing. Springer, Boston, MA. https://link.springer.com/chapter/10.1007/978-0-387-35562-7_23.

Dardenne, Anne, Stephen Fickas, et Axel van Lamsweerde. 1991. « Goal-directed Concept Acquisition in Requirements Elicitation ». In *Proceedings of the 6th International Workshop on Software Specification and Design*, 14‑21. IWSSD ’91. Los Alamitos, CA, USA: IEEE Computer Society Press. http://dl.acm.org/citation.cfm?id=952786.952790.

Davis, A. M. 1982. « The Design of a Family of Application-Oriented Requirements Languages ». *Computer* 15 (5): 21‑28. https://doi.org/10.1109/MC.1982.1654021.

DeMarco, Tom. 1979. *Structured Analysis and System Specification*. Upper Saddle River, NJ, USA: Prentice Hall PTR.

———. 2002. « Structured Analysis: Beginnings of a New Discipline », 520‑27. https://doi.org/10.1007/978-3-642-59412-0_32.

Descartes, René, et Nicolas-Joseph Poisson. 1724. *Discours de la methode, pour bien conduire sa raison, & chercher la verité dans les sciences. Plus la dioptrique et les meteores, qui sont des essais de cette methode.* Paris: J.B. Mazuel.

Diaper, Dan. 2002. « Scenarios and task analysis ». *Interacting with Computers*, 379‑95.

Diaper, Dan, et Mark Addison. 1992. « Task analysis and systems analysis for software development ». *Interacting with Computers* 4 (1): 124‑39. https://doi.org/10.1016/0953-5438(92)90016-9.

Diderot, et D’Alembert. 1751a. « Esprit (par Voltaire) ». https://fr.wikisource.org/wiki/L%E2%80%99Encyclop%C3%A9die/1re_%C3%A9dition/ESPRIT.

———. 1751b. « Modèle ». https://fr.wikisource.org/wiki/L%E2%80%99Encyclop%C3%A9die/1re_%C3%A9dition/ESPRIT.

Dix, Alan, Janet E. Finlay, Gregory D. Abowd, et Russell Beale. 2003. *Human-Computer Interaction*. 3 edition. Harlow, England ; New York: Pearson.

Doherty, Gavin, Jose Campos, et Michael Harrison. 2008. « Resources for Situated Actions ». In *Interactive Systems. Design, Specification, and Verification*, 194‑207. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-70569-7_19.

Dorst, Kees, et Nigel Cross. 2001. « Creativity in the design process: co-evolution of problem–solution ». *Design Studies* 22 (5): 425‑37. https://doi.org/10.1016/S0142-694X(01)00009-6.

Doss, O., et T. P. Kelly. 2016. « Challenges and Opportunities in Agile Development in Safety Critical Systems: A Survey ». *SIGSOFT Softw. Eng. Notes* 41 (2): 30‑31. https://doi.org/10.1145/2894784.2894798.

Dourish, Paul. 1995. « Developing a Reflective Model of Collaborative Systems ». *ACM Trans. Comput.-Hum. Interact.* 2 (1): 40‑63. https://doi.org/10.1145/200968.200970.

———. 1997. « Accounting for System Behaviour: Representation, Reflection and Resourceful Action ». In *Computers and Design in Context*, édité par Morten Kyng et Lars Mathiassen, 145‑70. Cambridge, MA, USA: MIT Press. http://dl.acm.org/citation.cfm?id=270318.270324.

———. 1998. « Using Metalevel Techniques in a Flexible Toolkit for CSCW Applications ». *ACM Trans. Comput.-Hum. Interact.* 5 (2): 109‑55. https://doi.org/10.1145/287675.287676.

Doyon-Poulin, P., J. M. Robert, et B. Ouellette. 2012. « Review of visual clutter and its effects on pilot performance: A new look at past research ». In *2012 IEEE/AIAA 31st Digital Avionics Systems Conference (DASC)*, 2D1‑1 ‑ 2D1‑11. https://doi.org/10.1109/DASC.2012.6382290.

Duan, C., et J. Cleland-Huang. 2006. « Visualization and Analysis in Automated Trace Retrieval ». In *2006 First International Workshop on Requirements Engineering Visualization (REV'06 - RE'06 Workshop)*, 5‑5. https://doi.org/10.1109/REV.2006.6.

Elmqvist, N., et J. D. Fekete. 2010. « Hierarchical Aggregation for Information Visualization: Overview, Techniques, and Design Guidelines ». *IEEE Transactions on Visualization and Computer Graphics* 16 (3): 439‑54. https://doi.org/10.1109/TVCG.2009.84.

Elverum, Christer W., et Torgeir Welo. 2015. « On the use of directional and incremental prototyping in the development of high novelty products: Two case studies in the automotive industry ». *Journal of Engineering and Technology Management* 38 (octobre): 71‑88. https://doi.org/10.1016/j.jengtecman.2015.09.003.

Endert, Alex, M. Shahriar Hossain, Naren Ramakrishnan, Chris North, Patrick Fiaux, et Christopher Andrews. 2014. « The Human Is the Loop: New Directions for Visual Analytics ». *Journal of Intelligent Information Systems* 43 (3): 411‑35. https://doi.org/10.1007/s10844-014-0304-9.

Erzberger, Heinz. 2004. « (4) Transforming the NAS: The next generation air traffic control system ». *NASA*, n° TP–2004-212828. https://www.researchgate.net/publication/250066237_Transforming_the_NAS_The_next_generation_air_traffic_control_system.

EUROCAE, et RTCA. 2012. « ED-109/DO-278 Software Integrity Assurance Considerations for Communication, Navigation, Surveillance and Air Traffic Management (CNS/ATM) Systems ».

Faisandier, Alain. 2014. *Notions de système et d'ingénierie de système*. Sinergy'Com.

Fallman, Daniel. 2003. « Design-oriented Human-computer Interaction ». In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 225‑32. CHI '03. New York, NY, USA: ACM. https://doi.org/10.1145/642611.642652.

———. 2007. « Why Research-Oriented Design Isn't Design-Oriented Research: On the Tensions Between Design and Research in an Implicit Design Discipline ». *Knowledge, Technology & Policy* 20 (3): 193‑200. https://doi.org/10.1007/s12130-007-9022-8.

Feather, M. S., S. L. Cornford, J. D. Kiper, et T. Menzies. 2006a. « Experiences using Visualization Techniques to Present Requirements, Risks to Them, and Options for Risk Mitigation ». In *2006 First International Workshop on Requirements Engineering Visualization (REV'06 - RE'06 Workshop)*, 10‑10. https://doi.org/10.1109/REV.2006.2.

———. 2006b. « Experiences using Visualization Techniques to Present Requirements, Risks to Them, and Options for Risk Mitigation ». In *2006 First International Workshop on Requirements Engineering Visualization (REV'06 - RE'06 Workshop)*, 10‑10. https://doi.org/10.1109/REV.2006.2.

Fekete, Jean-Daniel, David Wang, Niem Dang, et Catherine Plaisant. 2003. « Overlaying Graph Links on Treemaps ». In . https://hal.inria.fr/hal-00875194.

Feyerabend, Paul, et Ian Hacking. 2010. *Against Method*. 4th ed. London ; New York: Verso.

Filion, L., N. Daviot, J. P. Le Bel, et M. Gagnon. 2017. « Using Atlassian tools for efficient requirements management: An industrial case study ». In *2017 Annual IEEE International Systems Conference (SysCon)*, 1‑6. https://doi.org/10.1109/SYSCON.2017.7934769.

Flemisch, Frank, Matthias Heesen, Tobias Hesse, Johann Kelsch, Anna Schieben, et Johannes Beller. 2012. « Towards a Dynamic Balance between Humans and Automation: Authority, Ability, Responsibility and Control in Shared and Cooperative Control Situations ». *Cognition, Technology & Work* 14 (1): 3‑18. https://doi.org/10.1007/s10111-011-0191-6.

Fleury, Lionel, Franck De Wit, et Hélène Gaspard-Boulinc. 2004. « Conduite du changement dans les services qualité-sécurité de la circulation aérienne : l'effet levier de la norme ». In *Communications V :Normalisation des systèmes complexes*. Genève.

Fuentes, José M., Anabel Fraga, Gonzalo Génova, Jose Álvarez, et Juan Llorens. 2016. « Analysis of the INCOSE Rules for Writing Good Requirement in Industry: A Tool Based Study ». In *Complex Systems Design & Management*, 283‑283. Springer, Cham. https://link.springer.com/chapter/10.1007/978-3-319-26109-6_21.

Gall, John. 1977. *Systemantics: How Systems Work and Especially How They Fail*. N edition. New York: Quadrangle.

Garron, J., J Journet, et Didier Pavet. 2006. « Vigiestrips: Toward a Paperless Tower Cab ». In , 224‑31. Toulouse: Cépaduès.

Gaspard-Boulinc, H., et S. Conversy. 2017. « Usability Insights for Requirements Engineering Tools: A User Study with Practitioners in Aeronautics ». In *2017 IEEE 25th International Requirements Engineering Conference (RE)*, 223‑32. https://doi.org/10.1109/RE.2017.20.

Gaspard-Boulinc, Hélène, et Stéphane Conversy. 2014. « Usability Requirements for Requirement Engineering Tools ». In *Proceedings of the 26th Conference on L'Interaction Homme-Machine*, 70‑79. IHM '14. New York, NY, USA: ACM. https://doi.org/10.1145/2670444.2670458.

Gaspard-Boulinc, Hélène, Stéphane Conversy, Mickaël Loubriat, Alexandre Duchevet, Clement Dupont, M. Riedinger, Matthieu Pujos, Raïlane Benhacene, et Denis Louviot. 2016. Activity Based Resource Management System, issued 15 décembre 2016. https://hal-enac.archives-ouvertes.fr/hal-01469780.

Gaspard-Boulinc, Helene, Yannick Jestin, et Fleury. 2002. « Epoques: Proposing tools and methods to treat Air Traffic Management Safety Occurences ». In *Workshop on Investigation and Reporting of Incidents and Accidents*. Glasgow.

Gaspard-Boulinc, Hélène, Denis Louviot, Thomas Paques, et Stéphane Conversy. 2016. « Analyse d'indicateur d'attitude pour l'instruction au pilotage d'avion ». In *Actes de la 28ième conférence francophone sur l'Interaction Homme-Machine*, 252‑58. https://hal.archives-ouvertes.fr/hal-01384299/.

Genon, Nicolas, Daniel Amyot, et Patrick Heymans. 2010. « Analysing the Cognitive Effectiveness of the UCM Visual Notation ». In *System Analysis and Modeling: About Models*, 221‑40. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-21652-7_14.

Ghazi, P., et M. Glinz. 2016. « An Exploratory Study on User Interaction Challenges When Handling Interconnected Requirements Artifacts of Various Sizes ». In *2016 IEEE 24th International Requirements Engineering Conference (RE)*, 76‑85. https://doi.org/10.1109/RE.2016.52.

Ghoniem, Mohammad, Jean-Daniel Fekete, et Philippe Castagliola. 2005. « On the Readability of Graphs Using Node-link and Matrix-based Representations: A Controlled Experiment and Statistical Analysis ». *Information Visualization* 4 (2): 114‑35. https://doi.org/10.1057/palgrave.ivs.9500092.

Goguen, Joseph. 1994. « Requirements Engineering as the Reconciliation of Technical and Social Issues ». In . http://citeseerx.ist.psu.edu/viewdoc/similar?doi=10.1.1.12.6434&type=ab.

Gotel, O. C. Z., et C. W. Finkelstein. 1994. « An analysis of the requirements traceability problem ». In *Proceedings of IEEE International Conference on Requirements Engineering*, 94‑101. https://doi.org/10.1109/ICRE.1994.292398.

Graaf, B., M. Lormans, et H. Toetenel. 2003. « Embedded software engineering: the state of the practice ». *IEEE Software* 20 (6): 61‑69. https://doi.org/10.1109/MS.2003.1241368.

Greenbaum, Joan, et Morten Kyng. 1991. *Design at Work: Cooperative Design of Computer Systems*. Taylor & Francis.

Green, T. R. G. 1989. « Cognitive Dimensions of Notations ». In *Proceedings of the Fifth Conference of the British Computer Society, Human-Computer Interaction Specialist Group on People and Computers V*, 443‑60. New York, NY, USA: Cambridge University Press. http://dl.acm.org/citation.cfm?id=92968.93015.

Guindon, Raymonde. 1990. « Designing the Design Process: Exploiting Opportunistic Thoughts ». *Hum.-Comput. Interact.* 5 (2): 305‑44. https://doi.org/10.1207/s15327051hci0502&3_6.

Guindon, R., et B. Curtis. 1988. « Control of Cognitive Processes During Software Design: What Tools Are Needed? » In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 263‑68. CHI '88. New York, NY, USA: ACM. https://doi.org/10.1145/57167.57211.

Hall, T., S. Beecham, et A. Rainer. 2002. « Requirements problems in twelve software companies: an empirical analysis ». *IEE Proceedings - Software* 149 (5): 153‑60.

Hanssen, Geir K., Gosse Wedzinga, et Martijn Stuip. 2017. « An Assessment of Avionics Software Development Practice: Justifications for an Agile Development Process ». In *Agile Processes in Software Engineering and Extreme Programming*, 217‑31. Lecture Notes in Business Information Processing. Springer, Cham. https://doi.org/10.1007/978-3-319-57633-6_14.

Harel, David. 1987. « Statecharts: a visual formalism for complex systems ». *Science of Computer Programming* 8 (3): 231‑74. https://doi.org/10.1016/0167-6423(87)90035-9.

Harel, David, et Hillel Kugler. 2000. « Synthesizing State-Based Object Systems from LSC Specifications ». In *Implementation and Application of Automata*, 1‑33. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-44674-5_1.

Harris, Jed, et Austin Henderson. 1999. « A Better Mythology for System Design ». In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 88‑95. CHI '99. New York, NY, USA: ACM. https://doi.org/10.1145/302979.303003.

Hayes-Roth, Barbara. 1985. « A blackboard architecture for control ». *Artificial Intelligence* 26 (3): 251‑321. https://doi.org/10.1016/0004-3702(85)90063-3.

Heer, Jeffrey, et Ben Shneiderman. 2012. « A taxonomy of tools that support the fluent and flexible use of visualizations ». *Commun. ACM* 55 (4): 45‑54. https://doi.org/10.1145/2133806.2133821.

Henry, Nathalie, et Jean-Daniel Fekete. 2007. « MatLink: Enhanced Matrix Visualization for Analyzing Social Networks ». In *Human-Computer Interaction – INTERACT 2007*, 288‑302. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-74800-7_24.

Henry, N., et J. d Fekete. 2006. « MatrixExplorer: a Dual-Representation System to Explore Social Networks ». *IEEE Transactions on Visualization and Computer Graphics* 12 (5): 677‑84. https://doi.org/10.1109/TVCG.2006.160.

Henry, N., J. D. Fekete, et M. J. McGuffin. 2007. « NodeTrix: a Hybrid Visualization of Social Networks ». *IEEE Transactions on Visualization and Computer Graphics* 13 (6): 1302‑9. https://doi.org/10.1109/TVCG.2007.70582.

Herman, Ivan, Guy Melançon, et M. Scott Marshall. 2000. « Graph Visualization and Navigation in Information Visualization: A Survey ». *IEEE Transactions on Visualization and Computer Graphics* 6 (1): 24‑43. https://doi.org/10.1109/2945.841119.

Hoarau, Raphaël, et Stéphane Conversy. 2012. « Augmenting the Scope of Interactions with Implicit and Explicit Graphical Structures ». In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 1937‑46. CHI '12. New York, NY, USA: ACM. https://doi.org/10.1145/2207676.2208337.

Hoffmann, M., N. Kuhn, M. Weber, et M. Bittner. 2004. « Requirements for requirements management tools ». In *Proceedings. 12th IEEE International Requirements Engineering Conference, 2004.*, 301‑8. https://doi.org/10.1109/ICRE.2004.1335687.

Hollan, James, Edwin Hutchins, et David Kirsh. 2000. « Distributed Cognition: Toward a New Foundation for Human-Computer Interaction Research ». *ACM Transactions on Computer-Human Interaction* 7: 174‑96.

Hollnagel, Erik. 2012. « Coping with Complexity: Past, Present and Future ». *Cognition, Technology & Work* 14 (3): 199‑205. https://doi.org/10.1007/s10111-011-0202-7.

Hollnagel, Erik, et David D. Woods. 2005. *Joint Cognitive Systems: Foundations of Cognitive Systems Engineering*. CRC Press.

Hollnagel, E., et D. Woods. 1999. « Cognitive Systems Engineering: New wine in new bottles ». *International Journal of Human-Computer Studies* 51 (2): 339‑56. https://doi.org/10.1006/ijhc.1982.0313.

Holten, D. 2006. « Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data ». *Visualization and Computer Graphics*, n° 12: 741‑48.

Holten, DHR. 2009. « Visualization of Graphs and Trees for Software Analysis ». Technische Universiteit Eindhoven. http://repository.tue.nl/642975.

Holtzblatt, Karen, Jessamyn Burns Wendell, et Shelley Wood. 2004. *Rapid Contextual Design: A How-to Guide to Key Techniques for User-Centered Design*. San Francisco: Morgan Kaufmann.

———. 2005a. « Chapter 4 - The Contextual Inquiry Interview ». In *Rapid Contextual Design*, 79‑100. Interactive Technologies. San Francisco: Morgan Kaufmann. http://www.sciencedirect.com/science/article/pii/B9780123540515500057.

———. 2005b. « Chapter 14 - Paper Prototype Interviews ». In *Rapid Contextual Design*, 261‑77. Interactive Technologies. San Francisco: Morgan Kaufmann. http://www.sciencedirect.com/science/article/pii/B978012354051550015X.

Hughes, John A., Dave Randall, et Dan Shapiro. 1992. « From Ethnographic Record to System Design ». *Computer Supported Cooperative Work (CSCW)* 1 (3): 123‑41. https://doi.org/10.1007/BF00752435.

Hurter, Christophe, Rémi Lesbordes, Catherine Letondal, Jean-Luc Vinot, et Stéphane Conversy. 2012. « Strip'TIC: Exploring Augmented Paper Strips for Air Traffic Controllers ». In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, 225‑32. AVI '12. New York, NY, USA: ACM. https://doi.org/10.1145/2254556.2254598.

Hutchins, Edwin. 1995. *Cognition in the Wild*. Cambridge, Mass: MIT Press.

———. 1996. « The Integrated Mode Management Interface ». https://ntrs.nasa.gov/search.jsp?R=19960047103.

Hutchins, Edwin L., James D. Hollan, et Donald A. Norman. 1985. « Direct Manipulation Interfaces ». *Hum.-Comput. Interact.* 1 (4): 311‑38. https://doi.org/10.1207/s15327051hci0104_2.

IEEE. 2005. « IEEE Std 1220-2005 Standard for Application and Management of the Systems Engineering Process ». *IEEE Std 1220-2005 (Revision of IEEE Std 1220-1998)*, 0_1‑87. https://doi.org/10.1109/IEEESTD.2005.96469.

———. 2006. « Requirements Engineering Visualization (REV), International Workshop on ». 2010 2006. http://ieeexplore.ieee.org/xpl/conhome.jsp?punumber=1001649.

INCOSE. 2015. *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*. John Wiley and Sons. USA. http://sebokwiki.org/wiki/INCOSE_Systems_Engineering_Handbook.

« Information technology – Modeling Languages–Part 1: Syntax and Semantics for IDEF0 ». 2012. *ISO/IEC/IEEE 31320-1:2012 (Adoption of IEEE Std 1320.1-1998 )*, octobre, 1‑120. https://doi.org/10.1109/IEEESTD.2012.6363476.

Inria. 2016. *Inria responds to the challenge of artificial intelligence*. https://www.inria.fr/en/news/news-from-inria/artificial-intelligence-current-challenges-and-inria-s-engagement.

ISO. 2010. « ISO 9241-210:2010(fr), Ergonomie de l'interaction homme-système — Partie 210: Conception centrée sur l'opérateur humain pour les systèmes interactifs ». https://www.iso.org/obp/ui/#iso:std:iso:9241:-210:ed-1:v1:fr.

ISO/IEC. 2001. « ISO/IEC 9126-1:2001 - Software engineering -- Product quality -- Part 1: Quality model ». https://www.iso.org/standard/22749.html.

———. 2007. « ISO/IEC 26702 IEEE Std 1220-2005 First edition 2007-07-15 Standard for Systems Engineering - Application and Management of the Systems Engineering Process ». *ISO/IEC 26702 IEEE Std 1220-2005 First edition 2007-07-15*, juillet, 1‑101. https://doi.org/10.1109/IEEESTD.2007.386502.

———. 2008. « ISO/IEC 12207:2008 - Systems and software engineering -- Software life cycle processes ». https://www.iso.org/standard/43447.html.

———. 2009. « ISO/IEC TR 24766:2009 Information Technology – Systems and Software Engineering – Guide for Requirements Engineering Tool Capabilities ».

———. 2011. « ISO/IEC 25010:2011 - Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models ». https://www.iso.org/standard/35733.html.

ISO/IEC/IEEE. 2011. « ISO/IEC/IEEE 29148:2011- Systems and software engineering – Life cycle processes – Requirements engineering ». *ISO/IEC/IEEE 29148:2011(E)*, 1‑94. https://doi.org/10.1109/IEEESTD.2011.6146379.

———. 2015. « ISO/IEC/IEEE 15288:2015 - Systems and software engineering -- System life cycle processes ». https://www.iso.org/standard/63711.html.

Ivarsson, Martin, et Tony Gorschek. 2009. « Technology Transfer Decision Support in Requirements Engineering Research: A Systematic Review of REj ». *Requirements Engineering* 14 (3): 155‑75. https://doi.org/10.1007/s00766-009-0080-1.

Jackson, Daniel. 2006. *Software Abstractions: Logic, Language, and Analysis*. The MIT Press.

Jackson, Daniel, Ilya Shlyakhter, et Manu Sridharan. 2001. « A Micromodularity Mechanism ». In *Proceedings of the 8th European Software Engineering Conference Held Jointly with 9th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, 62‑73. ESEC/FSE-9. New York, NY, USA: ACM. https://doi.org/10.1145/503209.503219.

Jackson, Michael. 2009. « Some Notes on Models and Modelling », 68‑81. https://doi.org/10.1007/978-3-642-02463-4_5.

Jacobson, Ivar. 1992. *Object-Oriented Software Engineering: A Use Case Driven Approach*. [New York], Wokingham, Eng, Reading, Mass: ACM Press.

Jacobson, Ivar, Grady Booch, et James Rumbaugh. 1999. *The Unified Software Development Process*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Jacobson, Ivar, Ian Spence, et Brian Kerr. 2016. « Use Case 2.0: the hub of software development ». *ACM Queue* 14 (1).

Jacquard, Albert, et Huguette Planès. 1999. *Petite philosophie à l'usage des non-philosophes*. Le Livre de Poche. Paris: Le Livre de Poche.

Joerger, G., J. Rambourg, M. Garbey, S. Conversy, et H. Gaspard-Boulinc. 2017. « Re-engineer operating room data acquisition and transmission for improving surgical suite awareness and management ». In *Biomedical & Health Informatics (BHI), 2017 IEEE EMBS International Conference on*, 205‑8. IEEE. http://ieeexplore.ieee.org/abstract/document/7897241/.

Jones, Sara, et Neil Maiden. 2005. « RESCUE: An Integrated Method for Specifying Requirements for Complex Sociotechnical Systems ». *Http://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.4018/978-1-59140-506-1.ch015*, 245‑65. https://doi.org/10.4018/978-1-59140-506-1.ch015.

Jones, S., P. Lynch, N. Maiden, et S. Lindstaedt. 2008. « Use and Influence of Creative Ideas and Requirements for a Work-Integrated Learning System ». In *2008 16th IEEE International Requirements Engineering Conference*, 289‑94. https://doi.org/10.1109/RE.2008.54.

Jung, Malte F., David Sirkin, Turgut M. Gür, et Martin Steinert. 2015. « Displayed Uncertainty Improves Driving Experience and Behavior: The Case of Range Anxiety in an Electric Car ». In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, 2201‑10. CHI '15. New York, NY, USA: ACM. https://doi.org/10.1145/2702123.2702479.

Jurca, G., T. D. Hellmann, et F. Maurer. 2014. « Integrating Agile and User-Centered Design: A Systematic Mapping and Review of Evaluation and Validation Studies of Agile-UX ». In *2014 Agile Conference*, 24‑32. https://doi.org/10.1109/AGILE.2014.17.

Juristo, N., A. Moreno, et M. I. Sanchez-Segura. 2007. « Guidelines for Eliciting Usability Functionalities ». *IEEE Transactions on Software Engineering* 33 (11): 744‑58. https://doi.org/10.1109/TSE.2007.70741.

Kamalrudin, Massila, John Hosking, et John Grundy. 2011. « Improving Requirements Quality Using Essential Use Case Interaction Patterns ». In *Proceedings of the 33rd International Conference on Software Engineering*, 531‑40. ICSE '11. New York, NY, USA: ACM. https://doi.org/10.1145/1985793.1985866.

Kamalrudin, M., J. Grundy, et J. Hosking. 2010. « Managing Consistency between Textual Requirements, Abstract Interactions and Essential Use Cases ». In *2010 IEEE 34th Annual Computer Software and Applications Conference*, 327‑36. https://doi.org/10.1109/COMPSAC.2010.40.

Karlsson, Lena, sa G. Dahlstedt, Björn Regnell, Johan Natt och Dag, et Anne Persson. 2007. « Requirements Engineering Challenges in Market-driven Software Development - An Interview Study with Practitioners ». *Inf. Softw. Technol.* 49 (6): 588‑604. https://doi.org/10.1016/j.infsof.2007.02.008.

Keim, Daniel A., Florian Mansmann, Jörn Schneidewind, Jim Thomas, et Hartmut Ziegler. 2008. « Visual Analytics: Scope and Challenges ». In *Visual Data Mining*, édité par Simeon J. Simoff, Michael H. Böhlen, et Arturas Mazeika, 76‑90. Lecture Notes in Computer Science 4404. Springer Berlin Heidelberg. http://link.springer.com/chapter/10.1007/978-3-540-71080-6_6.

Kennedy, J. D., et M. Towhidnejad. 2017. « Innovation and certification in aviation software ». In *2017 Integrated Communications, Navigation and Surveillance Conference (ICNS)*, 3D3‑1 ‑ 3D3‑15. https://doi.org/10.1109/ICNSURV.2017.8011916.

Klausen, Tove, et Ed Hutchins. 1996. « Distributed Cognition in an Airline Cockpit ». http://vbn.aau.dk/en/publications/distributed-cognition-in-an-airline-cockpit(d58fee90-9c2e-11db-8ed6-000ea68e967b)/export.html.

Kong, N., J. Heer, et M. Agrawala. 2010. « Perceptual Guidelines for Creating Rectangular Treemaps ». *IEEE Transactions on Visualization and Computer Graphics* 16 (6): 990‑98. https://doi.org/10.1109/TVCG.2010.186.

Kriesi, Carlo, Jørgen Blindheim, Øystein Bjelland, et Martin Steinert. 2016. « Creating Dynamic Requirements through Iteratively Prototyping Critical Functionalities ». *Procedia CIRP*, 26th CIRP Design Conference, 50 (Supplement C): 790‑95. https://doi.org/10.1016/j.procir.2016.04.122.

Kunz, W., et H. Rittel. 1970. « Issues as elements of information systems ». *Working Paper n°131*, 1970, Institute of Urban and Regional Development édition.

Lamsweerde, A. van. 2001. « Goal-oriented requirements engineering: a guided tour ». In *Proceedings Fifth IEEE International Symposium on Requirements Engineering*, 249‑62. https://doi.org/10.1109/ISRE.2001.948567.

Lamsweerde, Axel van. 2000. « Requirements Engineering in the Year 00: A Research Perspective ». In *Proceedings of the 22Nd International Conference on Software Engineering*, 5‑19. ICSE '00. New York, NY, USA: ACM. https://doi.org/10.1145/337180.337184.

———. 2009. *Requirements Engineering: From System Goals to UML Models to Software Specifications*. 1st éd. Wiley Publishing.

Lano, Kevin. 1996. *The B Language and Method: A Guide to Practical Formal Development*. 1st éd. Secaucus, NJ, USA: Springer-Verlag New York, Inc.

Laplante, Phillip A., et Colin J. Neill. 2004. « The Demise of the Waterfall Model Is Imminent ». *Queue* 1 (10): 10‑15. https://doi.org/10.1145/971564.971573.

Lauesen, S. 1998. « Usability requirements in a tender process ». In *Computer Human Interaction Conference, 1998. Proceedings. 1998 Australasian*, 114‑21. https://doi.org/10.1109/OZCHI.1998.732203.

Lauesen, Soren, et Houman Younessi. 1998. « Six Styles for Usability Requirements ». In . /paper/Six-Styles-for-Usability-Requirements-Lauesen-Younessi/a5abb93b8cbb096bf60a2ce92826e4cc4120d428.

Laurent, P., P. Mader, J. Cleland-Huang, et A. Steele. 2010. « A Taxonomy and Visual Notation for Modeling Globally Distributed Requirements Engineering Projects ». In *2010 5th IEEE International Conference on Global Software Engineering*, 35‑44. https://doi.org/10.1109/ICGSE.2010.55.

Lawson, Bryan R. 2006. *How Designers Think -The Design Process Demystified*. 2nd Edition. https://www.elsevier.com/books/how-designers-think/lawson/978-0-7506-0268-6.

Le Moigne, Jean-Louis. 2006. *Théorie du système général: théorie de la modélisation*. Les classiques du réseau Intelligence de la Complexité. www.mcxapc.org.

Letondal, Catherine, Christophe Hurter, Rémi Lesbordes, Jean-Luc Vinot, et Stéphane Conversy. 2013. « Flights in My Hands: Coherence Concerns in Designing Strip'TIC, a Tangible Space for Air Traffic Controllers ». In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2175‑84. CHI '13. New York, NY, USA: ACM. https://doi.org/10.1145/2470654.2481300.

Leveson, Nancy. 2002. *System Safety Engineering: Back To The Future*. sunnyday.mit.edu/book2.pdf.

Lévi-Strauss, Claude. 1957. *Tristes tropiques*. Pocket. Paris: Pocket.

Lohmann, Steffen, Jürgen Ziegler, et Philipp Heim. 2008. « Involving End Users in Distributed Requirements Engineering ». In *Engineering Interactive Systems*, 221‑28. Springer, Berlin, Heidelberg. https://link-springer-com.proxy.lib.enac.fr/chapter/10.1007/978-3-540-85992-5_20.

Lormans, Marco, Arie van Deursen, et Hans-Gerhard Gross. 2008. « An Industrial Case Study in Reconstructing Requirements Views ». *Empirical Software Engineering* 13 (6): 727‑60. https://doi.org/10.1007/s10664-008-9078-4.

Lormans, M., et A. van Deursen. 2006. « Can LSI help reconstructing requirements traceability in design and test? » In *Conference on Software Maintenance and Reengineering (CSMR'06)*, 10 pp. ‑56. https://doi.org/10.1109/CSMR.2006.13.

Louviot, Denis, Hélène Gaspard-Boulinc, et Stéphane Conversy. 2017. Method and apparatus for controlling vehicle attitude display. European Patent Office 16306781.2, filed 2016 2016, et issued 2017.

Lubars, M., C. Potts, et C. Richter. 1993. « A review of the state of the practice in requirements modeling ». In *[1993] Proceedings of the IEEE International Symposium on Requirements Engineering*, 2‑14. https://doi.org/10.1109/ISRE.1993.324842.

LUZEAUX, Dominique, Jean-René RUAULT, et Jean-Luc WIPPLER. 2011. *Maîtrise de l'ingénierie des systèmes complexes et des systèmes de systèmes : étude de cas*. Hermès. https://www.lavoisier.fr/livre/informatique/maitrise-de-l-ingenierie-des-systemes-complexes-et-des-systemes-de-systemes-etude-de-cas/luzeaux/descriptif-9782746224681.

Mackay, Wendy E. 1990. « Users And Customizable Software: A Co-Adaptive Phenomenon ».

———. 1999. « Is Paper Safer? The Role of Paper Flight Strips in Air Traffic Control ». *ACM Trans. Comput.-Hum. Interact.* 6 (4): 311‑40. https://doi.org/10.1145/331490.331491.

———. 2003. « Educating multi-disciplinary design teams ». In *In Proceedings of Tales of the Disappearing Computer, Santorini*. ACM Press.

Mackay, Wendy E., et Anne-Laure Fayard. 1997. « HCI, Natural Science and Design: A Framework for Triangulation Across Disciplines ». In *Proceedings of the 2Nd Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques*, 223‑34. DIS '97. New York, NY, USA: ACM. https://doi.org/10.1145/263552.263612.

Mackay, Wendy E., Anne-Laure Fayard, Laurent Frobert, et Lionel Médini. 1998. « Reinventing the Familiar: Exploring an Augmented Reality Design Space for Air Traffic Control ». In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 558‑65. CHI '98. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co. https://doi.org/10.1145/274644.274719.

MacLean, Allan, Richard M. Young, Victoria M. E. Bellotti, et Thomas P. Moran. 1991. « Questions, Options, and Criteria: Elements of Design Space Analysis ». *Hum.-Comput. Interact.* 6 (3): 201‑50. https://doi.org/10.1207/s15327051hci0603&4_2.

Maiden, N., et S. Robertson. 2005. « Integrating creativity into requirements processes: experiences with an air traffic management system ». In *13th IEEE International Conference on Requirements Engineering (RE'05)*, 105‑14. https://doi.org/10.1109/RE.2005.34.

Mancini, Roberta, Alan J. Dix, et Stefano Levialdi. 1997. « Dealing with Undo ». In *Human-Computer Interaction INTERACT '97*, 703‑5. IFIP — The International Federation for Information Processing. Springer, Boston, MA. https://link-springer-com.proxy.lib.enac.fr/chapter/10.1007/978-0-387-35175-9_140.

Martinie, Célia, Philippe Palanque, Marco Winckler, et Stéphane Conversy. 2010. « DREAMER: A Design Rationale Environment for Argumentation, Modeling and Engineering Requirements ». In *Proceedings of the 28th ACM International Conference on Design of Communication*, 73‑80. SIGDOC '10. New York, NY, USA: ACM. https://doi.org/10.1145/1878450.1878463.

Mattessich, Richard. 1978a. « Introduction ». In *Instrumental Reasoning and Systems Methodology*, 1‑16. Theory and Decision Library 15. Springer Netherlands. http://link.springer.com.proxy.lib.enac.fr/chapter/10.1007/978-94-010-9431-3_1.

———. 1978b. « Systems Analysis as a Tool of Philosophical Investigation ». In *Instrumental Reasoning and Systems Methodology*, 17‑52. Theory and Decision Library 15. Springer Netherlands. http://link.springer.com.proxy.lib.enac.fr/chapter/10.1007/978-94-010-9431-3_2.

McGuirl, John M., et Nadine B. Sarter. 2006. « Supporting Trust Calibration and the Effective Use of Decision Aids by Presenting Dynamic System Confidence Information ». *Human Factors* 48 (4): 656‑65. https://doi.org/10.1518/001872006779166334.

Merten, T., D. Jüppner, et A. Delater. 2011. « Improved representation of traceability links in requirements engineering knowledge using Sunburst and Netmap visualizations ». In *2011 4th International Workshop on Managing Requirements Knowledge*, 17‑21. https://doi.org/10.1109/MARK.2011.6046557.

Mertz, Christophe, Stéphane Chatty, et Jean-Luc Vinot. 2000a. « Pushing the limits of ATC user interface design beyond S&M interaction: the DigiStrips experience ». In *Proc. ATM'2000 R&D seminar*. https://pdfs.semanticscholar.org/f99a/a3c972bbe38c848d2ce4391422ed8c32b101.pdf.

Mertz, Christophe, Stéphane Chatty, et Jean-luc Vinot. 2000b. *The influence of design techniques on user interfaces: the DigiStrips experiment for air traffic control*.

Miller, G.A. 1956. « The Magical Number Seven, Plus Or Minus 2: Some Limits On Our Capacity for Processing Information ». *ResearchGate*, 1956. https://www.researchgate.net/publication/10255186_The_Magical_Number_Seven_Plus_Or_Minus_2_Some_Limits_On_Our_Capacity_for_Processing_Information.

Moacdieh, Nadine, et Nadine Sarter. 2015. « Display Clutter: A Review of Definitions and Measurement Techniques ». *Human Factors* 57 (1): 61‑100. https://doi.org/10.1177/0018720814541145.

Moacdieh, N. M., et N. Sarter. 2017. « The Effects of Data Density, Display Organization, and Stress on Search Performance: An Eye Tracking Study of Clutter ». *IEEE Transactions on Human-Machine Systems* PP (99): 1‑10. https://doi.org/10.1109/THMS.2017.2717899.

Moody, D. 2009. « The Physics of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering ». *IEEE Transactions on Software Engineering* 35 (6): 756‑79. https://doi.org/10.1109/TSE.2009.67.

Moody, Daniel, et Jos van Hillegersberg. 2008. « Evaluating the Visual Syntax of UML: An Analysis of the Cognitive Effectiveness of the UML Family of Diagrams ». In *Software Language Engineering*, 16‑34. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-00434-6_3.

Moody, Daniel L., Patrick Heymans, et Raimundas Matulevičius. 2010. « Visual Syntax Does Matter: Improving the Cognitive Effectiveness of the I* Visual Notation ». *Requirements Engineering* 15 (2): 141‑75. https://doi.org/10.1007/s00766-010-0100-1.

Morin, Edgar. 2005. *Introduction à la pensée complexe*. Seuil. Points.

Mylopoulos, John, Lawrence Chung, et Eric Yu. 1999. « From Object-oriented to Goal-oriented Requirements Analysis ». *Commun. ACM* 42 (1): 31‑37. https://doi.org/10.1145/291469.293165.

Nair, Sunil, Jose Luis de la Vara, Mehrdad Sabetzadeh, et Davide Falessi. 2015. « Evidence management for compliance of critical systems with safety standards: A survey on the state of practice ». *Information and Software Technology* 60: 1‑15. https://doi.org/10.1016/j.infsof.2014.12.002.

Neill, C. J., et P. A. Laplante. 2003. « Requirements engineering: the state of the practice ». *IEEE Software* 20 (6): 40‑45. https://doi.org/10.1109/MS.2003.1241365.

Nguyen, L., J. Carroll, et P. A. Swatman. 2000. « Supporting and monitoring the creativity of IS personnel during the requirements engineering process ». In *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, 9 pp. vol.1 ‑ . https://doi.org/10.1109/HICSS.2000.926899.

Nguyen, L., et P. A. Swatman. 2000. « Essential and incidental complexity in requirements models ». In *Proceedings Fourth International Conference on Requirements Engineering. ICRE 2000. (Cat. No.98TB100219)*, 130‑39. https://doi.org/10.1109/ICRE.2000.855599.

Nielsen, Jakob. 1993. *Usability Engineering*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Niu, N., S. Reddivari, et Z. Chen. 2013. « Keeping requirements on track via visual analytics ». In *2013 21st IEEE International Requirements Engineering Conference (RE)*, 205‑14. https://doi.org/10.1109/RE.2013.6636720.

Norman, Donald. 2014. *The Design of Everyday Things*. Revised and expanded edition. https://mitpress.mit.edu/books/design-everyday-things.

OMG. 2015. « OMG Unified Modeling Language version 2.5 ». http://www.omg.org/spec/UML/.

« p28: sur les principes de la pensée complexe (par E. Morin) ». s. d.

« p371: sur la complexité: à la fois nombre et pli ». s. d.

Palanque, Philippe, Rémi Bastide, et Fabio Paternò. 1997. « Formal Specification as a Tool for Objective Assessment of Safety-Critical Interactive Systems ». In *Human-Computer Interaction INTERACT '97*, 323‑30. IFIP — The International Federation for Information Processing. Springer, Boston, MA. https://link.springer.com/chapter/10.1007/978-0-387-35175-9_53.

Pavet, Didier. 2001. « Use of paper strips by Tower Air Traffic Controllers and promises offered by new design techniques on user interface — ATM Seminar ». In . http://atmseminar.eurocontrol.fr/past-seminars/4th-seminar-santa-fe-nm-usa-december-2001/papers/paper_087/view.

Peirce, Charles S. 1955. *Philosophical Writings of Peirce*. Édité par Justus Buchler. New York, NY: Dover Publications.

Pérez, F., et P. Valderas. 2009. « Allowing End-Users to Actively Participate within the Elicitation of Pervasive System Requirements through Immediate Visualization ». In *2009 Fourth International Workshop on Requirements Engineering Visualization*, 31‑40. https://doi.org/10.1109/REV.2009.1.

Plaisant, Catherine, Jesse Grosjean, et Benjamin B. Bederson. 2002. « SpaceTree: Supporting Exploration in Large Node Link Tree, Design Evolution and Empirical Evaluation ». In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis'02)*, 57 ‑ . INFOVIS '02. Washington, DC, USA: IEEE Computer Society. http://dl.acm.org/citation.cfm?id=857191.857752.

Pohl, Klaus. 1994. « The three dimensions of requirements engineering: A framework and its applications ». *Information Systems*, Fifth International Conference on Advanced Information Systems Engineering (CAISE '93), 19 (3): 243‑58. https://doi.org/10.1016/0306-4379(94)90044-2.

Pons, Alain. s. d. « Ingenium: définition ». Consulté le 4 avril 2017. http://robert.bvdep.com/public/vep/Pages_HTML/INGENIUM.HTM.

Popper, Karl Raimund. 1979. *Objective Knowledge: An Evolutionary Approach*. Clarendon Press.

Poppleton, Michael R. 2007. « Towards Feature-Oriented Specification and Development with Event-B ». In *Requirements Engineering: Foundation for Software Quality*, 367‑81. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-73031-6_28.

Pujos, Matthieu, Raïlane Benhacene, Hélène Gaspard-Boulinc, Denis Louviot, et Jean-Luc Vinot. 2016. Energy management system for vehicles. US 15/178,762, issued juin 2016. http://www.google.com/patents/US20160363456.

Ramesh, Balasubramaniam. 1998. « Factors Influencing Requirements Traceability Practice ». *Commun. ACM* 41 (12): 37‑44. https://doi.org/10.1145/290133.290147.

Rapezzi, Claudio, Roberto Ferrari, et Angelo Branzi. 2005. « White coats and fingerprints: diagnostic reasoning in medicine and investigative methods of fictional detectives ». *BMJ : British Medical Journal* 331 (7531): 1491‑94.

Rashid, A., D. Meder, J. Wiesenberger, et A. Behm. 2006. « Visual Requirement Specification In End-User Participation ». In *2006 First International Workshop on Multimedia Requirements Engineering (MERE'06 - RE'06 Workshop)*, 6‑6. https://doi.org/10.1109/MERE.2006.7.

Rasmussen, J. 1985. « The role of hierarchical knowledge representation in decisionmaking and system management ». *IEEE Transactions on Systems, Man, and Cybernetics* SMC-15 (2): 234‑43. https://doi.org/10.1109/TSMC.1985.6313353.

Reddivari, S. 2013. « Visual analytics for software requirements engineering ». In *2013 21st IEEE International Requirements Engineering Conference (RE)*, 389‑92. https://doi.org/10.1109/RE.2013.6636762.

Reddivari, S., Z. Chen, et N. Niu. 2012. « ReCVisu: A tool for clustering-based visual exploration of requirements ». In *2012 20th IEEE International Requirements Engineering Conference (RE)*, 327‑28. https://doi.org/10.1109/RE.2012.6345828.

Rekimoto, Jun, et Mark Green. 1993. « The Information Cube: Using Transparency in 3D Information Visualization ». In *In Proceedings of the Third Annual Workshop on Information Technologies & Systems (WITS'93*, 125‑32.

Rittel, H. 1987. « The reasoning of Designers ». In *International Congress on Planning and Design Theory*. Boston.

Robertson, George G., Jock D. Mackinlay, et Stuart K. Card. 1991. « Cone Trees: Animated 3D Visualizations of Hierarchical Information ». In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 189‑94. CHI '91. New York, NY, USA: ACM. https://doi.org/10.1145/108844.108883.

Robertson, Suzanne, et James Robertson. 2006. *Mastering the Requirements Process (2Nd Edition)*. Addison-Wesley Professional.

Rognin, Laurence, Pascal Salembier, et Moustapha Zouinar. 2000. « Cooperation, reliability of socio-technical systems and allocation of function ». *International Journal of Human-Computer Studies* 52 (2): 357‑79. https://doi.org/10.1006/ijhc.1999.0293.

Ross, D. T. 1977. « Structured Analysis (SA): A Language for Communicating Ideas ». *IEEE Transactions on Software Engineering* SE-3 (1): 16‑34. https://doi.org/10.1109/TSE.1977.229900.

Ross, D. T., et K. E. Schoman. 1977. « Structured Analysis for Requirements Definition ». *IEEE Transactions on Software Engineering* SE-3 (1): 6‑15. https://doi.org/10.1109/TSE.1977.229899.

Rosson, Mary Beth, et John M. Carroll. 2009. « Scenario based design ». *Human-computer interaction. Boca Raton, FL*, 145‑62.

RTCA, et EUROCAE. 2011. « DO-178C Software Considerations in Airborne Systems and Equipment Certification ».

RTCA SC-167, et WG-12 EUROCAE. 1992. « DO-178B ». *Wikipedia*. https://en.wikipedia.org/w/index.php?title=DO-178B&oldid=776348564.

Sage, A. P. 2000. « Systems Engineering Education ». *Trans. Sys. Man Cyber Part C* 30 (2): 164‑74. https://doi.org/10.1109/5326.868437.

Salah, Dina, Richard F. Paige, et Paul Cairns. 2014. « A Systematic Literature Review for Agile Development Processes and User Centred Design Integration ». In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, 5:1‑5:10. EASE '14. New York, NY, USA: ACM. https://doi.org/10.1145/2601248.2601276.

Savio, D., P. Anitha, A. Patil, et O. Creighton. 2012. « Visualizing requirements in distributed system development ». In *2012 Second IEEE International Workshop on Requirements Engineering for Systems, Services, and Systems-of-Systems (RESS)*, 14‑19. https://doi.org/10.1109/RES4.2012.6347690.

Sawyer, Pete, Ian Sommerville, et Stephen Viller. 1997. « Improving the Requirements Process ». In *Proceedings of the Fourth International Workshop on Requirements Engineering: Foundations of for Software Quality, (REFSQ'98), Presses Universitaires de Namur*, 71‑84.

Schmidt, Kjeld, et Carla Simonee. 1996. « Coordination Mechanisms: Towards a Conceptual Foundation of CSCW Systems Design ». *Computer Supported Cooperative Work (CSCW)* 5 (2-3): 155‑200. https://doi.org/10.1007/BF00133655.

Schon, Donald A. 1983. *The Reflective Practitioner How Professionals Think in Action*. Basic Books. https://www.abebooks.co.uk/book-search/title/the-reflective-practitioner-how-professionals-think-in-action/author/schon-donald-a/.

Schon, Donald A., et Glenn Wiggins. 1992. « Kinds of seeing and their functions in designing ». *Design Studies* 13 (2): 135‑56. https://doi.org/10.1016/0142-694X(92)90268-F.

Schwaber, Ken, et Mike Beedle. 2001. *Agile Software Development with Scrum*. 1 edition. Upper Saddle River, NJ: Pearson.

Seddon, Peter B., et Rens Scheepers. 2012. « Towards the Improved Treatment of Generalization of Knowledge Claims in IS Research: Drawing General Conclusions from Samples ». *European Journal of Information Systems* 21 (1): 6‑21. https://doi.org/10.1057/ejis.2011.9.

Serres, Michel. 2015. *Le système de Leibniz et ses modèles mathématiques: Étoiles, schémas, points* -. Paris: PUF.

Seyff, N., F. Graf, et N. Maiden. 2010. « End-user requirements blogging with iRequire ». In *2010 ACM/IEEE 32nd International Conference on Software Engineering*, 2:285‑88. https://doi.org/10.1145/1810295.1810355.

Seyff, N., G. Ollmann, et M. Bortenschlager. 2011. « iRequire: Gathering end-user requirements for new apps ». In *2011 IEEE 19th International Requirements Engineering Conference*, 347‑48. https://doi.org/10.1109/RE.2011.6051669.

Shneiderman, B. 1996. « The eyes have it: a task by data type taxonomy for information visualizations ». In *Proceedings 1996 IEEE Symposium on Visual Languages*, 336‑43. https://doi.org/10.1109/VL.1996.545307.

———. 2003. « Why not make interfaces better than 3D reality? » *IEEE Computer Graphics and Applications* 23 (6): 12‑15. https://doi.org/10.1109/MCG.2003.1242376.

Shneiderman, Ben. 1992. « Tree Visualization with Tree-maps: 2-d Space-filling Approach ». *ACM Trans. Graph.* 11 (1): 92‑99. https://doi.org/10.1145/102377.115768.

———. 1996. « The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations ». In *Proceedings of the 1996 IEEE Symposium on Visual Languages*, 336‑. VL '96. Washington, DC, USA: IEEE Computer Society. http://dl.acm.org/citation.cfm?id=832277.834354.

Sikora, Ernst, Bastian Tenbergen, et Klaus Pohl. 2011. « Requirements Engineering for Embedded Systems: An Investigation of Industry Needs ». In *Requirements Engineering: Foundation for Software Quality*, 151‑65. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-19858-8_16.

Sire, Stéphane. 2000. « La collaboration directe d'un paradigme d'interaction pour le travail assisté par ordinateur ». Toulouse 1. http://www.theses.fr/2000TOU10005.

Sire, Stéphane, Stéphane Chatty, Hélène Gaspard-Boulinc, et François-Régis Colin. 1999. « How Can Groupware Preserve our Coordination Skills? Designing for Direct Collaboration. » In *INTERACT*, 304‑12.

Sohaib, O., et K. Khan. 2010. « Integrating usability engineering and agile software development: A literature review ». In *2010 International Conference On Computer Design and Applications*, 2:V2‑32‑V2‑38. https://doi.org/10.1109/ICCDA.2010.5540916.

Solis, C., et N. Ali. 2010. « Distributed Requirements Elicitation Using a Spatial Hypertext Wiki ». In *2010 5th IEEE International Conference on Global Software Engineering*, 237‑46. https://doi.org/10.1109/ICGSE.2010.35.

Sommerville, Ian, et Pete Sawyer. 1997. *Requirements Engineering: A Good Practice Guide*. 1st éd. New York, NY, USA: John Wiley & Sons, Inc.

Sommerville, I., T. Rodden, P. Sawyer, R. Bentley, et M. Twidale. 1993. « Integrating ethnography into the requirements engineering process ». In *[1993] Proceedings of the IEEE International Symposium on Requirements Engineering*, 165‑73. https://doi.org/10.1109/ISRE.1993.324821.

Spanoudakis, George, et Andrea Zisman. 2004. « Software Traceability: A Roadmap ». In *Handbook of Software Engineering and Knowledge Engineering*, 395‑428. World Scientific Publishing.

Spitzer, Cary R. 2000. *Digital Avionics Handbook*. CRC Press.

Spivey, J. M. 1989. *The Z Notation: A Reference Manual*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc.

Srinivasan, J., R. Dobrin, et K. Lundqvist. 2009. « “State of the Art” in Using Agile Methods for Embedded Systems Development ». In *2009 33rd Annual IEEE International Computer Software and Applications Conference*, 2:522‑27. https://doi.org/10.1109/COMPSAC.2009.186.

Suchman, Lucy A. 1983. « Office Procedure As Practical Action: Models of Work and System Design ». *ACM Trans. Inf. Syst.* 1 (4): 320‑28. https://doi.org/10.1145/357442.357445.

———. 1987. *Plans and Situated Actions: The Problem of Human-machine Communication*. New York, NY, USA: Cambridge University Press.

Sumner, Tamara. 1995. « The High-tech Toolbelt: A Study of Designers in the Workplace ». In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 178‑85. CHI ’95. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co. https://doi.org/10.1145/223904.223927.

Supakkul, S., et L. Chung. 2010. « Visualizing non-functional requirements patterns ». In *2010 Fifth International Workshop on Requirements Engineering Visualization*, 25‑34. https://doi.org/10.1109/REV.2010.5625663.

Sutcliffe, Alistair G., Neil A. M. Maiden, Shailey Minocha, et Darrel Manuel. 1998. « Supporting Scenario-Based Requirements Engineering ». *IEEE Trans. Softw. Eng.* 24 (12): 1072‑88. https://doi.org/10.1109/32.738340.

Svahnberg, Mikael, Tony Gorschek, Thi Than Loan Nguyen, et Mai Nguyen. 2015. « Uni-REPM: A Framework for Requirements Engineering Process Assessment ». *Requirements Engineering* 20 (1): 91‑118. https://doi.org/10.1007/s00766-013-0188-1.

Svennevig, Jan. 2001. « Abduction as a methodological approach to the study of spoken interaction ». *Norskrift* 103 (janvier).

Tesler, Larry. 2012. « A Personal History of Modeless Text Editing and Cut/Copy-paste ». *interactions* 19 (4): 70‑75. https://doi.org/10.1145/2212877.2212896.

Uchitel, Sebastian, Jeff Kramer, et Jeff Magee. 2003. « Synthesis of Behavioral Models from Scenarios ». *IEEE Trans. Softw. Eng.* 29 (2): 99‑115. https://doi.org/10.1109/TSE.2003.1178048.

Ugai, T., S. Hayashi, et M. Saeki. 2010. « Visualizing stakeholder concerns with anchored map ». In *2010 Fifth International Workshop on Requirements Engineering Visualization*, 20‑24. https://doi.org/10.1109/REV.2010.5625662.

Uninski, Hélène. 1998. « Vers la réalisation d un outil collecticiel à collaboration directe pour le contrôle aérien ». http://www.lii-enac.fr/~helene/Site/Recherche_files/resume.html.

US Air Force. 2009. « HUMAN SYSTEMS INTEGRATION REQUIREMENTS POCKET GUIDE ». www.dtic.mil/get-tr-doc/pdf?AD=ADA517632.

VanderLeest, S. H., et A. Buter. 2009. « Escape the waterfall: Agile for aerospace ». In *2009 IEEE/AIAA 28th Digital Avionics Systems Conference*, 6.D.3‑1 ‑ 6.D.3‑16. https://doi.org/10.1109/DASC.2009.5347438.

Vicente, Kim J. 1999a. *Cognitive Work Analysis: Toward Safe, Productive, and Healthy Computer-Based Work*. https://www.crcpress.com/Cognitive-Work-Analysis-Toward-Safe-Productive-and-Healthy-Computer-Based/Vicente/p/book/9780805823974.

———. 1999b. *Cognitive Work Analysis: Toward Safe, Productive, and Healthy Computer-Based Work*. https://www.crcpress.com/Cognitive-Work-Analysis-Toward-Safe-Productive-and-Healthy-Computer-Based/Vicente/p/book/9780805823974.

———. 2000. « HCI in the Global Knowledge-based Economy: Designing to Support Worker Adaptation ». *ACM Trans. Comput.-Hum. Interact.* 7 (2): 263‑80. https://doi.org/10.1145/353485.353489.

Vicente, Kim J., et Jens Rasmussen. 1990. « The Ecology of Human-Machine Systems II: Mediating “Direct Perception” in Complex Work Domains ». *Ecological Psychology* 2 (3): 207‑49. https://doi.org/10.1207/s15326969eco0203_2.

Vicente, K. J., et J. Rasmussen. 1992. « Ecological interface design: theoretical foundations ». *IEEE Transactions on Systems, Man, and Cybernetics* 22 (4): 589‑606. https://doi.org/10.1109/21.156574.

Visser, Willemien. 1990. « More or less following a plan during design: opportunistic deviations in specification ». *International Journal of Man-Machine Studies* 33 (3): 247‑78. https://doi.org/10.1016/S0020-7373(05)80119-1.

———. 2006. « Designing as Construction of Representations: A Dynamic Viewpoint in Cognitive Design Research ». *Human-Computer Interaction* 21 (1): 103‑52.

Whitehead, J. 2007. « Collaboration in Software Engineering: A Roadmap ». In *Future of Software Engineering, 2007. FOSE ’07*, 214‑25. https://doi.org/10.1109/FOSE.2007.4.

Whittle, Jon, et Johann Schumann. 2000. « Generating Statechart Designs from Scenarios ». In *Proceedings of the 22Nd International Conference on Software Engineering*, 314‑23. ICSE ’00. New York, NY, USA: ACM. https://doi.org/10.1145/337180.337217.

Wils, Andrew, Stefan Van Baelen, Tom Holvoet, et Karel De Vlaminck. 2006. « Agility in the Avionics Software World ». In *Proceedings of the 7th International Conference on Extreme Programming and Agile Processes in Software Engineering*, 123‑32. XP’06. Berlin, Heidelberg: Springer-Verlag. https://doi.org/10.1007/11774129_13.

Winkler, S. 2008. « On Usability in Requirements Trace Visualizations ». In *2008 Requirements Engineering Visualization*, 56‑60. https://doi.org/10.1109/REV.2008.4.

Winkler, Stefan, et Jens von Pilgrim. 2010. « A Survey of Traceability in Requirements Engineering and Model-Driven Development ». *Software & Systems Modeling* 9 (4): 529‑65. https://doi.org/10.1007/s10270-009-0145-0.

Wohlin, Claes, et Aybüke Aurum. 2015. « Towards a Decision-Making Structure for Selecting a Research Design in Empirical Software Engineering ». *Empirical Software Engineering* 20 (6): 1427‑55. https://doi.org/10.1007/s10664-014-9319-7.

Woodcock, Jim, et Jim Davies. 1996. *Using Z: Specification, Refinement, and Proof*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc.

Woodman, Mark. 1988. « Yourdon dataflow diagrams: A tool for disciplined requirements analysis ». *Information and Software Technology* 30 (9): 515‑33. https://doi.org/10.1016/0950-5849(88)90131-0.

Yang, D., D. Wu, S. Koolmanojwong, A. W. Brown, et B. W. Boehm. 2008. « WikiWinWin: A Wiki Based System for Collaborative Requirements Negotiation ». In *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008)*, 24‑24. https://doi.org/10.1109/HICSS.2008.502.

Yin, Robert K. 2009. *Case Study Research: Design and Methods*. SAGE.

Yourdon, Edward. 1989. *Modern Structured Analysis*. Upper Saddle River, NJ, USA: Yourdon Press.

———. 2006. *Just Enough Structured Analysis*. https://docs.google.com/file/d/0B42Cu1mD9Z7seVVHLUdqb1Q1SlU/preview?usp=embed_facebook.

Yourdon, Edward, et Larry L. Constantine. 1979. *Structured Design: Fundamentals of a Discipline of Computer Program and Systems Design*. 1st éd. Upper Saddle River, NJ, USA: Prentice-Hall, Inc.

Yu, E. S. K. 1997. « Towards modelling and reasoning support for early-phase requirements engineering ». In *, Proceedings of the Third IEEE International Symposium on Requirements Engineering, 1997*, 226‑35. https://doi.org/10.1109/ISRE.1997.566873.

Zave, Pamela. 1985. « A Distributed Alternative to Finite-state-machine Specifications ». *ACM Trans. Program. Lang. Syst.* 7 (1): 10‑36. https://doi.org/10.1145/2363.2365.

Zave, Pamela, et Michael Jackson. 1997. « Four Dark Corners of Requirements Engineering ». *ACM Trans. Softw. Eng. Methodol.* 6 (1): 1‑30. https://doi.org/10.1145/237432.237434.

AUTHOR : **Hélène UNINSKI**
TITLE : **Participatory Requirements Engineering for Complex Interactive Systems in Aeronautics**
SUPERVISOR : Stéphane CONVERSY
DEFENCE : Toulouse, Ecole Nationale de l'Aviation Civile, December, 20th 2017

ABSTRACT

Requirements Engineering plays a crucial role in coordinating the different stakeholders needed for safe aeronautics systems engineering. The notion of requirement is a key concept in system definition and represents the cornerstone of communication among stakeholders: customer, user, systems engineers, suppliers. The certification process in critical systems requires traceability documents as a means of demonstrating safety, showing links between the several artifacts (high-level requirements, refined requirements, components and interfaces).
We conducted a qualitative study, using interviews and mockups, with fifteen industrial practitioners from four aeronautics companies, to investigate which tasks are actually performed by requirements engineers and how current tools support these tasks. We found that RE-specific tools
constrain engineers to a rigid workflow, which conflicts with the adaptive exploration of the problem. Engineers often start by using general-purpose tools to foster exploration and collaborative work with suppliers, at the expense of traceability. When engineers shift to requirements refinement and verification, they must use RE-specific tools to grant traceability. But the lack of tool usability yields significant time loss and dissatisfaction.
Based on observed RE practices, we devise a situated vision of Requirements Engineering, rendering its role in between the operational context and the engineering context of the system. Based on scenarios and prototypes, we formulate usability insights for RE-specific tools. In particular, we propose interactive coordinated visualizations of structured text, allowing engineers to decorelate rigor from rigidity throughout the RE process, by providing flexibility during the process while gradually tackling approximation to the end of the process. The requirements engineer takes advantage of structured visualizations of requirements, from which he can communicate with stakeholders, search for text, get the progress status of requirements, detect and fill missing information thanks to interactive navigation and filtering.
Beyond interactive tools supporting a situated vision of RE, we propose a new approach to RE: participatory requirements engineering (PRE). Its purpose is the production of mature requirements specifying in the future system the coverage of unforeseen situations in the current system, yet managed by users. The approach is based on the combined use of participatory design techniques to involve the users, with a continuous effort of abstraction and requirements statement to inform the system definition. We apply the approach on four projects in aeronautics: collaboration and air traffic control, cockpit of an electric training aircraft, analysis of accident report and new flight instrument. We present the engineering artifacts resulting from the application of the approach in order to evaluate its benefits.

AUTEUR : **Hélène UNINSKI**
TITRE : **Une ingénierie participative des exigences pour les systèmes interactifs complexes en aéronautique**
DIRECTEUR DE THESE : Stéphane CONVERSY
LIEU ET DATE DE SOUTENANCE : Toulouse, Ecole Nationale de l'Aviation Civile, 20 décembre 2017

RESUME

L'ingénierie des exigences joue un rôle crucial dans la construction de systèmes aéronautiques sûrs. La notion d'exigence constitue la brique de base de la communication entre les différentes parties prenantes du système : client, utilisateurs, ingénieur système, fournisseurs. Non seulement les exigences sont structurantes pour la définition de système, mais le processus de certification lui-même est basé sur la démonstration de la conformité du système avec les exigences spécifiées, notamment par la traçabilité, c'est-à-dire un maintien des liens entre les différents artefacts d'ingénierie (exigences, exigences détaillées, composants, interfaces).
Nous avons réalisé une étude qualitative, à base d'interviews contextuelles et prototypes, auprès de 15 praticiens industriels de quatre entreprises aéronautiques, afin d'enquêter sur les activités réellement effectuées par les ingénieurs en exigences et sur le support outillé de ces activités. Nous avons trouvé que les outils spécifiques à l'ingénierie des exigences contraignent les ingénieurs à un flux de travail rigide, qui est en conflit avec une exploration adaptative des problèmes de conception. Les ingénieurs commencent souvent par utiliser des outils à vocation générale pour favoriser l'exploration et la collaboration avec les fournisseurs, au détriment de la traçabilité. Quand les ingénieurs basculent sur le raffinement et la vérification des exigences, ils doivent utiliser des outils spécifiques pour garantir la traçabilité. Le manque d'utilisabilité de ces outils entraine une perte de temps significative et une insatisfaction.
Sur la base de nos observations, nous développons une vision située de l'ingénierie des exigences, retranscrivant son rôle entre contexte d'ingénierie et contexte opérationnel du système. Sur la base de scénarios et de prototypes, nous formulons des exigences d'utilisabilité pour les outils spécifiques d'ingénierie des exigences. Nous proposons plus particulièrement des visualisations interactives et coordonnées de texte structuré permettant de décorréler rigueur et rigidité dans le processus d'ingénierie des exigences, en rendant possible une souplesse pendant le processus tout en éliminant progressivement toute approximation en sortie du processus. L'ingénieur bénéfice de visualisations structurées des exigences, à partir desquelles il peut communiquer avec les parties prenantes, chercher du texte, voir l'état d'avancement des exigences, détecter et compléter les informations manquantes par une navigation et un filtrage interactifs sur les visualisations.
Au-delà des outils supportant une vision située de l'ingénierie des exigences, nous proposons une nouvelle approche : l'ingénierie participative des exigences. La finalité est la production d'exigences matures spécifiant dans le système futur la prise en compte de situations non prévues dans le système actuel mais gérées par les utilisateurs. Elle est basée sur une articulation de techniques utilisées en conception participative pour impliquer les utilisateurs, avec un effort continu d'abstraction et de formalisation des exigences pour informer la définition du système. Nous appliquons notre approche sur quatre projets aéronautiques : collaboration et contrôle aérien, cockpit d'avion-école électrique, analyse de rapport d'accident et nouvel instrument de vol. Nous présentons les artefacts d'ingénierie issues de l'application de notre approche afin d'en évaluer ses bénéfices.

MOTS-CLES : ingénierie, exigence, participatif, système, interactif, utilisabilité, outil

DISCIPLINE ADMINISTRATIVE : Réseaux, Telecom, Système et Architecture

INTITULE ET ADRESSE DE L'U.F.R. OU DU LABORATOIRE : UMR 5505 IRIT