Conception des chaînes éditoriales : documentariser l'activité et structurer le graphe documentaire pour améliorer la maîtrise de la rééditorialisation

Thibaut Arribe

▶ **To cite this version:**

Thibaut Arribe. Conception des chaînes éditoriales : documentariser l'activité et structurer le graphe documentaire pour améliorer la maîtrise de la rééditorialisation. Informatique [cs]. Université de Technologie de Compiègne, 2014. Français. NNT : 2014COMP2146 . tel-01127476

HAL Id: tel-01127476

https://theses.hal.science/tel-01127476

Submitted on 7 Mar 2015

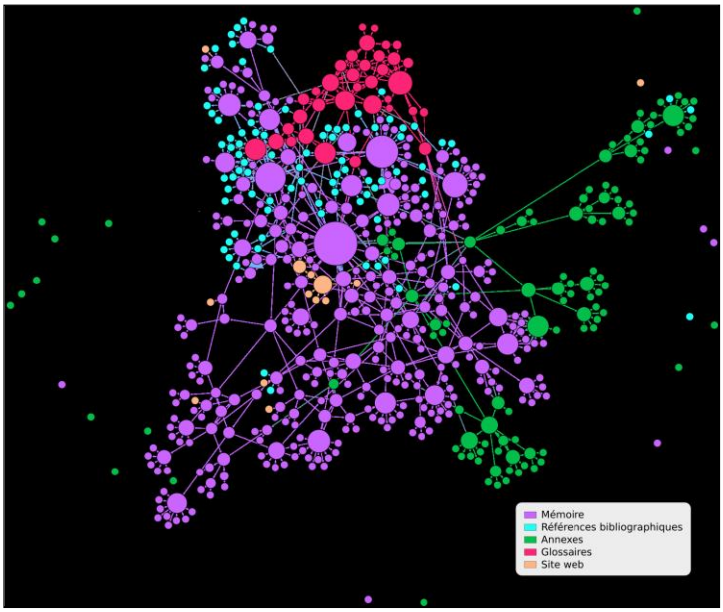
HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Par **Thibaut ARRIBE**

Conception des chaînes éditoriales : documentariser l'activité et structurer le graphe documentaire pour améliorer la maîtrise de la rééditorialisation

Thèse présentée
pour l'obtention du grade
de Docteur de l'UTC



Soutenue le 24 novembre 2014
Spécialité : Technologies de l'Information et des Systèmes

D2146

Thèse
Pour l'obtention du grade de

Docteur de l'Université de Technologie de Compiègne
Spécialité : Technologies de l'Information et des Systèmes

Conception des chaînes éditoriales

Documentariser l'activité et structurer le graphe documentaire pour améliorer la maîtrise de la rééditorialisation

par
Thibaut ARRIBE

Soutenue le 24 novembre 2014 devant un jury composé de :

M. Yannick PRIÉ	Professeur Université de Nantes *Rapporteur*
M. Jean CHARLET	Chargé de mission recherche (HDR) AP-HP & INSERM U1142 *Rapporteur*
M. Khaldoun ZREIK	Professeur Université Paris 8 *Examinateur*
M. Pierre MORIZET	Professeur Université de Technologie de Compiègne *Examinateur*
M. Bruno BACHIMONT	Enseignant-Chercheur (HDR) Université de Technologie de Compiègne *Directeur*
M. Stéphane CROZAT	Enseignant-Chercheur Université de Technologie de Compiègne *Directeur*
M. Sylvain SPINELLI	Directeur Technique Kelis *Invité - encadrant entreprise*
M. Dominique SAINT-MARTIN	Président de l'association Scenari *Invité*

Résumé

Les travaux présentés dans ce mémoire traitent de la *conception des chaînes éditoriales* numériques XML : des logiciels de production documentaire qui outillent la rédaction de *fragments* et l'assemblage de ces fragments pour former des documents. La publication des documents s'opère par transformation de fragments XML en documents numériques aux formats standards. La composition des fragments permet d'instrumenter la *rééditorialisation documentaire* soit l'usage de contenus existants dans la rédaction de documents originaux.

La production de documents rééditorialisés implique une modification de la représentation logique des documents de la forme classique d'arbre à celle de graphe de fragments liés entre eux. Ces travaux s'intéressent aux limites rencontrées lorsque la *complexité* du graphe devient un frein à la production de documents cohérents.

Nous avons élaboré une théorie pour la conception des chaînes éditoriales qui s'appuie sur deux approches complémentaires s'articulant autour de la figure de l'atelier - soit l'environnement d'écriture des fragments - pour accompagner les rédacteurs dans l'édition de graphes complexes.

La première approche est une *articulation interne* à l'atelier. Nous proposons de modéliser des *fragments pragmatiques* dont l'enjeu est d'assister les rédacteurs dans la gestion des fragments tout en enregistrant l'activité sur le graphe. Ces fragments participent à la *documentarisation de l'activité* puisqu'ils produisent et mettent à jour une documentation des actions opérées sur le graphe.

La seconde approche est une *articulation externe* à l'atelier. Nous proposons de *structurer le graphe* documentaire dans plusieurs ateliers en fonction des projets éditoriaux (projets de documents à publier) et auctoriaux (projets d'organisation de la production). L'enjeu est d'éclater le graphe dans plusieurs ateliers afin d'en *simplifier la perception* et donc la manipulation.

Nous avons participé aux développements de la suite Scenari afin d'expérimenter notre approche théorique. Les modèles de fragments pragmatiques de la tâche, du fragment commenté, et les responsabilités et cycles de vie sur les fragments ont été développés afin de documentariser l'activité. Les ateliers calques et fragments proxys ont été conçus afin de structurer le graphe. Nos expérimentations nous ont permis de synthétiser une méthodologie pour la conception de chaînes éditoriales exploitant des graphes complexes. Nos travaux se concluent sur une réflexion épistémologique sur la recherche technologique et plus particulièrement sur ses modalités de validité.

Mots clés

Ingénierie documentaire, chaîne éditoriale, rééditorialisation, graphe, complexité, modélisation

Abstract

Design of publishing chains: documenting the activity and structuring the graph of document fragments to enhance the mastery of repurposing

This thesis deals with the design of digital XML publishing chains : document production software which allows writing fragments and composing them in order to publish documents. Document publication is computed by transforming XML fragments into digital documents in standard formats. The composition of the fragments allows the repurposing of existing contents into new documents.

The writing of repurposed documents involves a change in the logical representation of documents from the classical form of the tree to a graph of linked fragments. This work is dedicated to the limitations encountered when the complexity of the graph becomes an obstacle to the writing of consistent documents.

We developed a theory for the design of publishing chains which is based on two complementary approaches around the concept of the writing workspace - the space where the authors write the fragments - to assist writers in the editing of complex graphs.

The first approach is internal to the workspace. We propose to model pragmatic fragments intended to assist authors in fragment management while recording graph activity. These fragments are involved in the activity documentarisation since they produce and maintain a documentation of the actions performed on the graph.

The second approach is external to the workspace. We propose to structure the graph as several workspaces based on publishing and authorial projects. The challenge is to distribute the graph over several workspaces in order to simplify their perception and handling.

We participated in the development of Scenari in order to experiment our theoretical approach. The pragmatic fragment models of the task, of the commented fragment, and of the responsibilities and life cycles on fragments were developed to document the activity. The layer workspaces and proxy fragments were designed to structure the graph.

Our experiments allowed us to summarize a methodology for designing publishing chains which handle complex graphs. Our work is concluded with an epistemological approach in order to qualify and evaluate this technological research.

Key words

Document engineering, publishing chain, repurposing, graph, complexity, modeling

Remerciements

En premier lieu, je tiens à remercier les trois encadrants qui ont collaboré avec moi tout au long de ces trois années de recherche. Merci Stéphane, Bruno et Sylvain pour l'apport de vos expertises respectives et complémentaires. Votre appui théorique, technique et technologique m'ont permis de façonner ma recherche au carrefour entre plusieurs disciplines et technologies. Merci également, par votre investissement et votre bonne humeur constante, d'avoir jalonné ces trois années de recherche de moments de collaboration conviviaux, motivants et intellectuellement féconds.

Je tiens également à remercier Jean Charlet et Yannick Prié d'avoir accepté de rapporter ce mémoire. Merci également à Pierre Morizet, Khaldoun Zreik et Dominique Saint-Martin d'avoir accepté d'être membres du jury.

Merci aux différentes équipes parmi lesquelles j'ai pu travailler pendant ces trois années. Merci à tous les membres de la société Kelis pour l'accompagnement et le soutien, avec une mention spéciale à Xavier, Julie et Léonard, l'équipe de coworking de Paris. Merci à l'unité ICS (et tout particulièrement à Martine) pour le soutien tant dans l'organisation que pour les expérimentations. Merci enfin au laboratoire Heudiasyc pour l'accompagnement scientifique et les débats fructueux lors des journées de doctorants et des séminaires.

Merci aux clients de Kelis, la DILA et Quick, d'avoir accepté de laisser apparaître leur contexte de production documentaire dans ce mémoire.

Des remerciements également pour les doctorants avec qui j'ai pu échanger durant ces trois années. Je pense notamment à Camille, Ben, Kevin, Antoine et Léo.

Merci à mes relecteurs, Ziad, Diane et Élisabeth pour votre patience et votre disponibilité.

Merci à ma famille pour toutes ces années riches qui m'ont conduit à accomplir ces travaux. Bien entendu, une mention spéciale pour Diane : merci pour tes conseils, tes relectures attentives et ton soutien.

Table des matières

Introduction 11
Partie I : Hypothèses et problématique 15

Chapitre 1 : La rééditorialisation documentaire, du document au graphe de fragments 17
1.1 La documentarisation du monde 17
1.2 Le document numérique et la redocumentarisation du monde 20
1.3 La rééditorialisation documentaire 21
1.4 Les chaînes éditoriales : technologie de rééditorialisation documentaire 23
1.5 Le graphe, structure de représentation des documents fragmentés 27
1.6 La modélisation des chaînes éditoriales 32

Chapitre 2 : La rééditorialisation, exemples de mise en œuvre 39
2.1 Enaction Series 40
2.2 Enseignements universitaires 43
2.3 Direction de l'Information Légale et Administrative 47
2.4 Quick 50

Chapitre 3 : Problématique 57
3.1 Notion de cohérence 57
3.2 Complexité du graphe documentaire 59
3.3 Artisanat et industrialisation 60
3.4 L'élaboration des documents rééditorialisés, entre complexité et cohérence 61

Partie II : État de l'art 63

Chapitre 4 : Assistance au développement informatique 65
4.1 Commentaires 66
4.2 Gestion des versions 67

Chapitre 5 : Travail coopératif assisté par ordinateur 71
5.1 Le pragmatisme ou l'action du discours 71
5.2 La perspective Langage/Action 73
5.3 Les documents pour l'action 73

Chapitre 6 : Analyse technologique des chaînes éditoriales 75
6.1 Rédaction Technique 75
6.2 Contenus pédagogiques 78
6.3 Synthèse 79

Partie III : Propositions 83

Chapitre 7 : Préambule 85
7.1 À propos de nos propositions 85
7.2 À propos de notre contribution 86

Chapitre 8 : Accompagner la cohérence du graphe par la documentarisation de l'activité 87
8.1 Fragments de document et discours pragmatique 88
8.2 Éléments pour la modélisation de fragments pragmatiques 91
8.3 Exemples de modèles 103

Chapitre 9 : Structurer le graphe pour en simplifier la manipulation 111
9.1 Projet de rééditorialisation, entre convergence et portée 112
9.2 Éléments pour la structuration des projets de rééditorialisation 118
9.3 Exemples 131
9.4 Distribution des projets de rééditorialisation 134

Chapitre 10 : Vers une industrialisation de la rééditorialisation documentaire 143
10.1 Méthodologie 144
10.2 Exemples 147

Partie IV : Positionnement épistémologique et évaluation 165

Chapitre 11 : Épistémologie et recherche technologique 167
11.1 Épistémologie : critères de scientificité et programmes de recherche 167
11.2 Épistémologie normative interne des programmes de recherche empirique 169
11.3 Épistémologie normative interne des programmes de recherche technologique 171

Chapitre 12 : Positionnement de notre recherche 173
12.1 Épistémologie descriptive 173
12.2 Positionnement : notre contribution au programme 177

Chapitre 13 : Évaluation 181
13.1 Évaluation technique 182
13.2 Évaluation des usages 186
13.3 Synthèse 189

Conclusion 191
Bibliographie 193
Annexes 199
Glossaire 201
Système développé 219
Contribution technique 231
Méthodologie pour la conception de chaînes éditoriales 253

Introduction

> « *L'étude des humanités devra un puissant foyer de lumières à cette nouvelle espèce de libraires sortis de la Germanie comme un cheval de Troie pour se répandre sur tous les points du monde civilisé. On raconte un peu partout que c'est aux environs de Mayence que vivait ce Jean, dit Gutenberg, qui le premier a inventé l'art de l'imprimerie, grâce auquel sans emploi de roseau ni de plume mais au moyen de caractères métalliques, des livres sont fabriqués rapidement, correctement et élégamment [...]. L'invention de Gutenberg [...] nous a donné des caractères à l'aide desquels tout ce qui se dit ou se pense peut être immédiatement écrit, récrit et livré à la postérité...* »
>
> *Guillaume Fichet, 1471*

C'est par ces mots que Guillaume Fichet, associé de la première société d'imprimerie parisienne hébergée dans les locaux de l'université La Sorbonne décrit l'invention de l'imprimerie (Marichal, 1987).

La mobilisation du support graphique, accentuée par la diffusion des imprimés, a considérablement modifié la manière dont les civilisations conçoivent et transmettent les connaissances. Le rythme effréné de production de connaissances de la révolution industrielle fera dire à Bush (1945), un des théoriciens du métier de documentaliste, que l'accès aux documents devait se concevoir « *as we may think* », par une simple association d'idées.

Au même titre que le support graphique s'est substitué à une tradition orale en ouvrant de nouveaux possibles pour l'expression des connaissances, le support numérique se substitue au papier et ajoute une dimension dynamique pour leur conception et leur transmission. Le caractère manipulatoire du numérique, ses tendances à la de-sémantisation et l'interprétation, à la fragmentation et la recombinaison transforment fondamentalement nos rapports à la connaissance (Bachimont, 2007).

Nos travaux s'inscrivent dans un programme de recherche technologique sur l'étude des modalités de l'écriture numérique ordinaire. Plus particulièrement, elle s'intéresse à la rééditorialisation que nous définissons d'une part comme un principe de production documentaire permettant à l'auteur d'embrasser le rôle d'éditeur ; de l'autre, comme une exploitation des propriétés numériques du support pour optimiser la production des documents. Le principe fondateur de cette optimisation repose sur l'accompagnement de la copie entre documents. L'objectif est d'outiller l'écriture afin d'agencer des fragments de documents existants en complément d'autres, originaux. Reprenant l'expression de Bush, Crozat (2012) décrit les possibles de l'écriture numérique derrière l'expression « *as we may write* ».

Notre recherche fait suite aux développements des chaînes éditoriales comme technologie de mise en œuvre de la rééditorialisation et à leurs usages expérimentaux et industriels.

Problématique et objectif

En exploitant des fonctions permettant la production de documents rééditorialisés, les chaînes éditoriales segmentent leur représentation interne des documents en fragments. Ces fragments sont liés les uns aux autres formant ainsi un vaste graphe documentaire.

La production de documents rééditorialisés tend à complexifier le graphe documentaire exploité par les chaînes éditoriales en augmentant à la fois sa taille et sa densité. La mise à jour d'un fragment devient alors une opération complexe qui nécessite de nombreux contrôles afin d'éviter d'introduire des incohérences au sein d'un de ses contextes de publication.

Au delà d'une certaine échelle, les rédacteurs n'ont pas les moyens de maintenir de vastes graphes documentaires complexes. Les rédacteurs ont alors le choix entre réduire la complexité du graphe en dupliquant certains fragments afin de diminuer le nombre de liens au sein du graphe ou alors accepter un manque de contrôle sur les documents produits et donc une baisse de la qualité de ceux-ci avec l'introduction d'incohérences éditoriales.

L'enjeu de notre recherche consiste à assister les concepteurs de chaînes éditoriales afin de simplifier la perception que des rédacteurs peuvent avoir d'un graphe documentaire tout en maintenant la qualité du contrôle de sa cohérence. Notre objectif est d'assister la conception de chaînes éditoriales outillant une production documentaire dont la rééditorialisation est opérée à une échelle industrielle, soit répétable indéfiniment sans altérer la qualité de la documentation produite.

Contributions

Notre contribution s'établit selon quatre perspectives.

Une contribution théorique qui propose deux approches complémentaires pour permettre l'industrialisation de la rééditorialisation. Il s'agit de projeter le graphe documentaire dans différents espaces de travail simplifiés afin de donner une section du graphe à voir et à manipuler aux rédacteurs. Il s'agit ensuite de documentariser le discours pragmatique des rédacteurs de chacun des espaces de travail tout en proposant des fonctions d'assistance de l'activité.

Une contribution méthodologique qui permet d'assister les concepteurs de chaînes éditoriales. Il s'agit d'une méthode de structuration et de modélisation en deux temps. Dans un premier temps, nous proposons de structurer un graphe documentaire en plusieurs espaces en tenant compte des projets éditoriaux et auctoriaux d'une organisation. Dans un second temps, nous proposons d'accompagner les besoins d'assistance des rédacteurs et de documentarisation des organisations au sein de chacun des espaces préalablement mis en place.

Une contribution technique qui s'inscrit dans les développements de la suite logicielle de chaînes éditoriales Scenari. Nos développements ont été menés en lien avec l'unité Ingénierie des Connaissances et des Savoirs de l'UTC et avec la société Kelis, éditrice de Scenari.

Une contribution épistémologique qui constitue l'élément le plus inattendu de nos travaux. En cherchant à caractériser la validité scientifique de notre démarche appliquée, nous nous sommes intéressés au concept de programme de recherche technologique de Theureau (2009). Dans ce mémoire, nous proposons une adaptation des propositions de Theureau afin de constituer une épistémologie normative interne et une épistémologie descriptive du programme de recherche technologique sur la rééditorialisation documentaire.

Organisation du mémoire

Ce mémoire présente les travaux réalisés au cours de notre thèse de doctorat. Il se structure en quatre parties.

La première partie est dédiée à la définition des hypothèses et de notre problématique de recherche. Le chapitre 1 présente la rééditorialisation documentaire, de l'histoire des supports d'inscription des connaissances jusqu'à la modélisation des chaînes éditoriales. Le chapitre 2 présente la structure du graphe comme objet de représentation des documents rééditorialisés. Il introduit et illustre la notion de graphe documentaire. Le chapitre 3 expose la problématique.

La seconde partie est dédiée à une complétion de l'état de l'art. Nos deux chapitres d'hypothèses constituent un véritable ancrage de notre recherche. Cette seconde partie complète cet ancrage en étudiant les diverses solutions mobilisées pour accompagner la rééditorialisation au sein des chaînes éditoriales. Le chapitre 4 s'intéresse ainsi à l'assistance au développement informatique et plus particulièrement, à la gestion des versions des fichiers de code. Le chapitre 5 traite des différentes propositions du domaine du Travail Coopératif Assisté par Ordinateur pour organiser le partage d'une ressource par un collectif. Le chapitre 6 mène une étude technologique des chaînes éditoriales et la met en perspective avec les différentes propositions exposées dans les chapitres précédents.

La troisième partie expose nos contributions théoriques, méthodologiques et techniques. Le chapitre 7 est un avant-propos à la partie III. Les chapitres 8 et 9 présentent la contribution théorique, son implémentation et des exemples d'usages associés. Ils spécifient la part de l'implémentation constituant notre contribution technique. Le chapitre 10 reprend notre contribution méthodologique et la met en application dans deux exemples fictifs inspirés de nos contextes d'usage.

La quatrième partie traite de l'évaluation de nos propositions et contient en outre notre contribution épistémologique. Le chapitre 11 fait un rapide tour d'horizon des différentes

théories épistémologiques et s'intéresse plus particulièrement aux propositions de Theureau. Le chapitre 12 exploite ces propositions pour formuler une épistémologie descriptive du programme de recherche « rééditorialisation documentaire » et positionner notre recherche au sein de ce programme. Le dernier chapitre s'appuie sur ce positionnement et sur l'épistémologie descriptive du programme de recherche pour mener une évaluation de notre recherche.

Partie I
Hypothèses et problématique

Cette première partie expose l'ancrage de notre recherche. Elle est composée d'un premier chapitre dédié aux hypothèses sur lesquelles nos travaux se fondent. Nous y introduisons la notion de rééditorialisation et la technologie des chaînes éditoriales. Le second chapitre montre plusieurs instrumentations des chaînes éditoriales : de contextes relativement simples à des usages industriels. Ce chapitre met en exergue les difficultés dans la mise en œuvre de la production de documents rééditorialisés. Le troisième et dernier chapitre s'appuie sur ces difficultés pour formaliser notre problématique de recherche autour de la tension entre la fragmentation des contenus et la cohérence documentaire du graphe.

Chapitre 1

La rééditorialisation documentaire, du document au graphe de fragments

1.1 La documentarisation du monde 17
1.2 Le document numérique et la redocumentarisation du monde 20
1.3 La rééditorialisation documentaire 21
1.4 Les chaînes éditoriales : technologie de rééditorialisation documentaire 23
1.5 Le graphe, structure de représentation des documents fragmentés 27
1.5.1 Théorie des graphes 27
1.5.2 Graphes documentaires 28
1.5.3 Représentation des graphes documentaires 30
1.6 La modélisation des chaînes éditoriales 32

1.1 La documentarisation du monde

Du manuscrit au document moderne

Notre recherche s'ancre dans une histoire des supports et des techniques de production des écrits. Cette discipline est jalonnée d'inventions techniques facilitant à la fois la production et la diffusion des écrits. Dans cette section, nous rappellerons les étapes clés de la discipline en nous appuyant sur l'ouvrage de Barbier (2012).

Dans le monde occidental, les premiers supports conçus pour l'inscription d'écrits sont les *volumen* : des rouleaux fabriqués à partir de bandes de papyrus (ibid., p. 42). Le *volumen* est inventé par les égyptiens au troisième millénaire avant J.-C. Il constituera ensuite le support de prédilection pour l'inscription des écrits au sein des civilisations grecques et romaines.

La fin de l'antiquité voit l'apparition d'un nouveau matériau, le parchemin, et d'une nouvelle technique d'assemblage de feuillets, le *codex*. Première forme de livre à proprement parler, le *codex* est un ensemble de feuillets, pliés ou reliés entre eux. Dans la civilisation romaine, le *codex* est réservé à des écrits moins nobles que le *volumen*. « L'invention du codex est fondamentale pour l'avenir de la civilisation écrite, parce qu'elle ouvre à tous les développements futurs du travail intellectuel sur des documents écrits. Le codex, divisé en éléments semblables (les feuillets, chacun composé de deux pages), se prête bien à la consultation partielle et on pourra, à terme, lui superposer un système de références facilitant

cette consultation (la foliotation, puis la pagination) (ibid., p. 62). » L'élaboration des parchemins, l'écriture et l'enluminure des livres a été l'apanage des monastères durant tout le Moyen-Âge (ibid., p. 68).

Deux inventions majeures, nécessaires à l'ère de l'imprimerie, sont diffusées depuis la Chine vers l'Occident : le papier, inventé au premier siècle après J.-C. et progressivement diffusé en Europe entre le XII et XVème siècle (ibid., p. 90) et la gravure sur bois (xylographie) inventée au VIIème siècle et diffusée en Europe au cours du XIVème (ibid., p. 91). C'est au cours du XVème siècle que Henne Gensfleisch zur Laden (dit Gutenberg) invente l'imprimerie à caractères mobiles, ouvrant ainsi la voie à un considérable essor dans la diffusion des imprimés (ibid., p. 95). Son invention marque, suite au *codex*, la seconde grande révolution du livre.

La raison graphique

Dans la mesure où l'esprit de l'humain contemporain doit être le même que celui des hommes de la préhistoire, Goody s'interroge sur les raisons pour lesquelles « plusieurs millénaires de stagnation s'intercalent comme un palier, entre la révolution néolithique et la science contemporaine » (1979). Il montre ainsi qu'avec l'écrit, soit avec l'usage d'un nouveau support pour inscrire et transmettre les connaissances naît une raison propre qu'il nomme la « raison graphique ». Il fait en outre émerger trois structures de construction et représentation des connaissances propres à l'écrit.

La liste (ibid., p. 140). Elle est la transposition sur du papier d'une énumération orale. La structure de la liste permet d'ordonnancer des éléments les uns par rapport aux autres, de les classer dans des catégories différentes. C'est de cette figure logique que naissent les projets scientifiques de classification de l'époque moderne comme la classification des espèces de la faune ou de la flore.

Le tableau (ibid., p. p. 108). Le tableau constitue une exploitation de l'espace graphique. La position d'une case influe directement sur sa valeur. L'usage d'un tableau pour représenter des connaissances ouvre une approche systématique dans la recherche de nouveaux éléments. Cet aspect systématique peut assister le chercheur dans la constitution de connaissances. Un exemple typique de connaissance directement impactée par sa forme de représentation est le tableau périodique des éléments (voir figure 1). L'exploitation graphique apporte plus de lisibilité au tableau (il n'est pas nécessaire de lire un traité de chimie pour situer les différents éléments les uns par rapport aux autres) et a permis de créer de nouvelles connaissances en mettant en évidence lors de sa création les différents atomes encore non découverts par l'intermédiaire de cases vides.

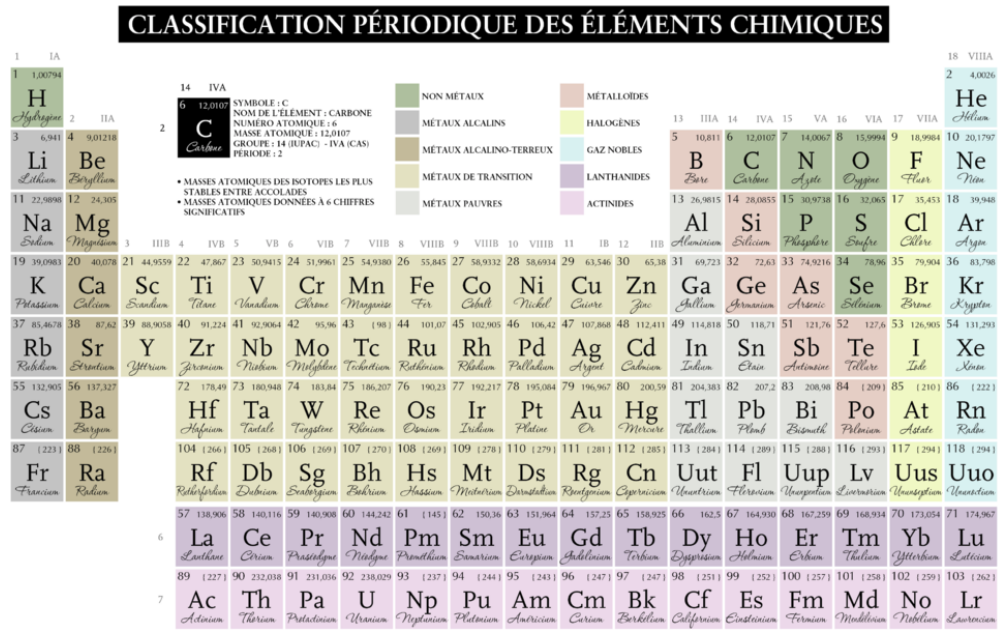


Figure 1 : tableau périodique des éléments (CC BY-SA 3.0, Poke2001)

La formule (ibid., p. 97). La notion de formule désigne des éléments permettant de formaliser la connaissance. Goody montre que la formule orale est mobilisée pour structurer l'énonciation du discours. « Elles font partie de ce [qu'il] appelle "formes orales standardisées" ; leur standardisation prend la forme d'une articulation rythmique, souvent avec l'accompagnement de musique » (ibid., p. 199). La formule écrite est constituée de signes graphiques au sens formel permettant de représenter des connaissances. Ce sont ces formules écrites qui permettent l'existence des mathématiques. Notons que la transposition d'une formule écrite dans un énoncé oral est un exercice difficile qui ne peut restituer l'ensemble des informations contenues par l'enchaînement de signes.

L'histoire des dispositifs techniques rappelée plus haut est à mettre en perspective avec la rationalité propre à l'écrit. On peut ainsi noter que la première forme technique exploitée pour la rédaction d'ouvrages est directement issue de la tradition orale. Le *volumen* est un rouleau qui conserve la linéarité du discours et ne facilite pas un accès désordonné au contenu. Il faudra attendre de nouvelles techniques, plus éloignées de la tradition orale comme la pagination et les index des *codices*, pour permettre de nouvelles modalités d'accès à une section d'un écrit. De nouvelles formes de livres aux contenus non linéaires arriveront également par la suite comme, par exemple, « l'Encyclopédie ou Dictionnaire raisonné des sciences, des arts et des métiers » de Diderot et D'Alembert.

La documentarisation du monde industriel

L'imprimerie a rendu possible la prolifération et la diffusion des imprimés. Le collectif R.T. Pédauque explique ainsi que ces imprimés sont « directement associé[s] à la première modernisation, celle qui a permis l'esprit scientifique, la rupture avec les traditions de l'Ancien Régime, l'expérimentation et sa validation à travers des comptes-rendus détaillés comme critère de la scientificité (Pedauque, 2006, p. 3) ». Le collectif Pédauque utilise le terme de *documentarisation* pour désigner l'omniprésence et la participation des documents à la construction du monde moderne : autant dans la construction des États que dans le soutien des révolutions industrielles et scientifiques.

De la prolifération des documents naît la documentation comme discipline de gestion des documents. Son principal objectif est de gérer la masse documentaire produite pour en faciliter l'accès. Ses principaux théoriciens sont Paul Otlet, Suzanne Briet ou encore Vannevar Bush. Paul

Otlet est un juriste Belge à l'origine de l'Office International de Bibliographie (un organisme de coopération internationale entre bibliothèques) et de la Classification Décimale Universelle (un système de classification des documents utilisés par les bibliothèques). Otlet est également à l'origine du Mundaneum, un centre gigantesque dédié à stocker et répertorier l'ensemble du savoir humain (Levie, 2006 ; Otlet, 1934). Suzanne Briet est une documentaliste française qui a travaillé pendant vingt ans à la salle des catalogues et bibliographies de la bibliothèque nationale. En 1951, elle publie un manifeste sur la documentation : « Qu'est ce que la documentation ? » (Briet, 1951) dans lequel elle définit le métier de documentaliste. Elle y donne également une définition très englobante du document : « tout indice concret ou symbolique, conservé ou enregistré, aux fins de représenter, de reconstituer ou de prouver un phénomène ou physique ou intellectuel » (ibid., p. 7), qu'elle illustre par l'exemple de l'antilope. Dès lors qu'une nouvelle espèce est cataloguée et exposée dans un zoo, elle devient un document. Tous les documents annexes (dessin, peinture, photographie, enregistrement du cri) n'en sont que des documents dérivés. Vannevar Bush est connu pour son article, « As we may think » (1945), rédigé au lendemain de la guerre de 1945 qu'il a vécu en tant que coordinateur de la recherche américaine. Il y publie ses idées pour améliorer l'accès à l'information et théorise notamment un système de liens entre les documents en fonction des idées qu'ils véhiculent afin de permettre leur accès *as we may think*.

Dans ce mémoire, nous partageons la définition du document de Bachimont, plus restrictive que celle de Briet, qui s'arrête aux inscriptions faites sur un support, persistantes, délimitées temporellement et spatialement et contenant une intentionnalité (2007, pp. 178-179).

1.2 Le document numérique et la redocumentarisation du monde

Qu'est ce qu'un document numérique ?

Bachimont distingue trois dimensions dans la manipulation des documents : « la dimension de l'expression », « la dimension de la conservation » et « la dimension de la restitution » (2007, p. 174). Il montre que le document papier fait figure d'exception en faisant intervenir le même dispositif pour l'expression, la conservation et la restitution. Les documents temporels introduits au cours du XXème siècle (vidéo, audio) rompent cette unité en nécessitant des supports d'enregistrement et de restitution différents. Le numérique généralise cette différence en nécessitant systématiquement des dispositifs de restitution particuliers. Bachimont montre que l'essence du numérique réside dans « la discrétisation des contenus à des énoncés exprimés à l'aide d'un langage composé de symboles vides de sens » et « la manipulation des symboles soumis à des règles formelles » (ibid., p. 32). Il en déduit un noème du numérique : « ça a été manipulé » (ibid., p. 33) ; en extension au noème de la photographie de Roland Barthes (1980) : « ça a été ».

Un document numérique est un contenu discrétisé et désémantisé, stocké dans une forme binaire, et nécessairement manipulé en vue de le reconstruire pour sa restitution. Dans ces termes, un document numérique ne peut respecter la définition du document introduite dans la section précédente. Les critères de persistance et de délimitation spatiale et temporelle ne sont plus clairement respectés. Comme le souligne Crozat (2012, p. 187), « le document numérique finalement n'existe pas, la locution est oxymorique. Il ne peut exister que des constructions numériques dont le traitement calculatoire permet de simuler un ordre documentaire. »

La raison computationnelle

Dans ce même ouvrage, Bachimont s'appuie sur les travaux de Goody présentés dans la section précédente sur une raison propre au support graphique pour poser les bases d'une raison computationnelle, propre au support numérique. « L'hypothèse que nous formulons est que l'informatique, sous la forme des systèmes formels automatiques, fournit précisément un nouveau type de support, les supports dynamiques, auquel doit correspondre un type spécifique de synthèse, et par conséquent une rationalité spécifique, que nous proposons de baptiser "raison computationnelle" » (ibid., p. 16). Bachimont fait ainsi évoluer les trois structures logiques de Goody vers trois nouvelles structures propres au support numérique : le programme, le réseau et la couche (ibid., pp. 73-76).

Le programme. Le programme représente l'évolution de la liste. Il permet de spécifier a priori des logiques d'exécution d'une liste d'instructions. Il projette dans le temps de l'exécution informatique une liste d'instructions logiques et formelles.

Le réseau. Bachimont propose le terme réseau pour désigner l'évolution du tableau. Les cases deviennent indépendantes les unes des autres et communiquent entre elles par des relations pré-établies.

La couche. La formule théorisée par Goody constitue l'élément de formalisation de l'écrit. Au même titre, la couche informatique permet de structurer les manipulations de la machine en différents niveaux indépendants les uns des autres. Chaque couche peut effectuer des actions en faisant abstraction des calculs réalisés par les couches sous-jacentes. C'est ce système de couches qui permet toujours plus d'automatisation et d'abstraction au sein des systèmes informatisés.

La redocumentarisation

Le numérique, par son essence, favorise grandement la production et la transmission des documents. Le collectif Pédauque (2006, p. 4) parle de nouvelle révolution qu'il nomme *redocumentarisation*. Ce terme « désigne à la fois un retour sur une documentarisation ancienne et une révolution documentaire ». Le terme redocumentarisation est également mobilisé par Salaün et Zacklad pour désigner des réalités différentes. Zacklad le mobilise pour désigner les éléments permettant un meilleur accès aux contenus, tant à l'extérieur d'un document (indexation, utilisation de moteurs de recherche) qu'à l'intérieur (ré-agencement, liens internes). Salaün (2007) désigne les étapes de numérisation préalables à une exploitation numérique des documents : « dans un premier temps, il s'agit de traiter à nouveau des documents traditionnels qui ont été transposés sur un support numérique en utilisant les fonctionnalités de ce dernier. [...] Il s'agit alors d'apporter toutes les métadonnées indispensables à la reconstruction à la volée de documents et toute la traçabilité de son cycle. » C'est bien dans une reconfiguration de l'ordre documentaire existant tel que montré par Pédauque que nous inscrivons notre recherche.

La première appropriation du numérique pour l'élaboration et la gestion des documents a consisté en une transposition des dispositifs existants sur un nouveau support. Pour l'exemple, on pourra citer les outils de traitement de texte qui simulent l'inscription d'un contenu sur une feuille de papier ou les plates-formes de gestion électronique de documents (GED) qui permettent d'informatiser le centre de documentation et d'archive d'une organisation, rendant ainsi possibles les projets d'indexation des théoriciens de la documentation que sont Bush, Briet et Otlet. Par hypothèse, nous considérons ces outils comme le balbutiement de l'élaboration et la gestion des contenus numériques. Au même titre que le *volumen* est un objet conçu selon une tradition orale d'accès aux contenus que le *codex* a revisité en abandonnant la continuité de l'inscription au profit d'un système de pagination, notre recherche vise à mieux outiller la production des documents en s'appuyant sur les spécificités du support numérique.

1.3 La rééditorialisation documentaire

Tendance technique et numérique

Dans *L'homme et la matière*, Leroi-Gourhan (1971, p. 27) met en évidence le concept de tendance technique : « la tendance a un caractère inévitable, prévisible, rectiligne ; elle pousse le silex tenu à la main à acquérir un manche, le ballot traîné sur deux perches à se munir de roues. [...] La roue entraîne l'apparition de la manivelle, de la courroie de transmission, de la démultiplication. » Ce concept de tendance technique indépendante des civilisations est également partagé par Stiegler (1994, p. 57) : « le concept de tendance technique s'oppose à cette illusion ethnocentrique [...] il n'y a pas de génie de l'invention, ou du moins, il ne joue qu'un rôle mineur dans l'évolution technique. »

Les techniques suivraient donc une évolution déterminée par la nature intrinsèque de l'objet et du milieu dans lequel il est mobilisé plutôt que par le milieu ethnologique dans lequel il est manipulé. Bachimont (2007, p. 37) propose ainsi une formulation de la tendance du numérique : une « fragmentation et recombinaison » d'un côté, une « désémantisation » de l'autre. Crozat (2012, p. 183) exploite ce concept de tendance technique pour définir l'enjeu de notre recherche en ingénierie du document. Il s'agit « d'inventer - pour les exhumer - des formes documentaires qui ouvrent un nouveau champ du possible, pour tenter d'en anticiper les

incidences cognitives et sociétales en genèse ». À titre d'exemple, la carte heuristique de l'écriture numérique, réalisée dans le cadre du projet PRECIP (Crozat, Bachimont, Cailleau, Bouchardon & Gaillard, 2011) tente d'explorer la nature du numérique pour *exhumer* les techniques fondamentales et particulières de l'écriture numérique.

La rééditorialisation

En continuité des travaux de Bachimont et dans cet objectif d'exhumer les techniques d'écriture numérique, Crozat propose le terme de rééditorialisation qu'il définit comme suit.

« Le terme de rééditorialisation est un néologisme qui a émergé dans le domaine du document numérique pour désigner le processus consistant à reconstruire un nouveau document à partir d'archives. La construction de ce mot tente une première synthèse entre les concepts d'édition au sens de publication d'une œuvre, d'éditorialisation au sens d'expression d'un point de vue propre, de réédition au sens de nouvelle proposition de lecture. Elle tente une seconde synthèse entre les fonctions d'éditeur, celui qui met en forme et diffuse, et d'auteur, celui qui écrit, fonctions qui tendent à se mêler dans le contexte du numérique. La rééditorialisation est donc la publication d'une œuvre originale dans son point de vue, sa forme, sa scénarisation, à partir de contenus qui ne le sont pas tous. » (Crozat, 2012, p. 189)

La rééditorialisation de contenus numériques peut s'opérer suivant différentes modalités. Nous parlerons de **fonctions de rééditorialisation** pour désigner les fonctions éditoriales permettant l'insertion de contenus existants dans l'élaboration de documents originaux. La fonction de rééditorialisation la plus simple est la fonction de **transclusion** (Nelson, 1981). Elle consiste à insérer un contenu par une simple référence à un fragment de document existant. Par exemple, la construction d'un document A peut se faire comme illustré dans la figure 2 en mobilisant un fragment A0 et des fragments A1, A2, A3 et A4 référencés au sein d'un arbre dont A0 est la racine. Lors de la construction de l'affichage par la machine, l'affichage de A3 et A4 est inséré dans l'affichage de A2, qui avec A1, est également inséré dans l'affichage de A0. Cette technique de fragmentation et transclusion est par exemple permise par la technologie *LaTeX*. Rééditorialiser par transclusion consiste dans cet exemple à produire un document B comme illustré dans la figure 3. Certains fragments issus de l'arbre dont B0 est la racine référencent par transclusion les fragments mobilisés par le document A.

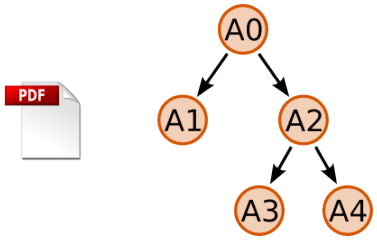


Figure 2 : du document à l'arbre de fragments

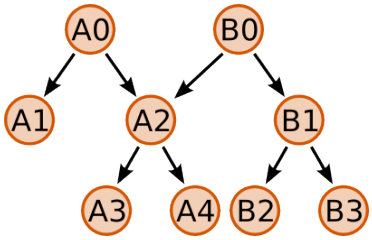


Figure 3 : rééditorialisation par transclusion

Des fonctions de rééditorialisation plus spécifiques s'appuient sur la transclusion mais opèrent en outre une surcharge des fragments référencés. Cette surcharge peut être manuelle : nous parlerons alors de **dérivation** ; ou programmée : nous parlerons de **déclinaison**. Dérivation et déclinaison ont de nombreux usages. Parmi ceux-ci, on pourra citer la localisation en règle générale et la traduction en particulier, la documentation de familles de produits aux caractéristiques proches ou encore la rédaction d'écrits sur un sujet unique et pour des cibles différentes. D'un point de vue plus large, chaque action de « copier/coller » dans un éditeur de documents classique peut être instrumentée soit par une transclusion si le fragment copié est réutilisé sans modification, soit une déclinaison ou dérivation si le fragment est modifié pour correspondre à son contexte d'usage.

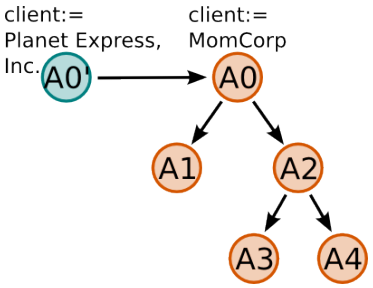


Figure 4 : déclinaison d'un arbre de fragments

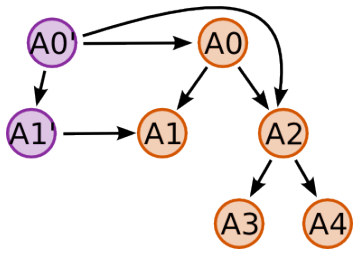


Figure 5 : dérivations d'un arbre de fragments

Par exemple : sur la figure 4, la valeur « Planet Express, Inc. » remplace l'ensemble des valeurs « MomCorp » sur l'ensemble du document ; sur la figure 5, les fragments A0 et A1 ont manuellement été surchargés par le concepteur du document.

1.4 Les chaînes éditoriales : technologie de rééditorialisation documentaire

Définition

Les chaînes éditoriales numériques sont des outils qui accompagnent la production documentaire de masse. Pour y parvenir, elles s'appuient sur une mise en évidence des structures documentaires présentes dans un corpus. Ces structures sont représentées dans un modèle documentaire qui contrôle la validité des documents. La publication des documents s'opère par des algorithmes de transformation qui s'appuient sur le modèle pour publier des documents dans des formats standards comme PDF ou HTML.

En travaillant uniquement sur des structures et en automatisant la mise en forme, les chaînes éditoriales permettent la séparation entre le fond et la forme - ou entre le fonds documentaire et ses formes (Bachimont & Crozat, 2004). Elles facilitent ainsi une automatisation des manipulations documentaires.

Les documents numériques contrôlés par un modèle documentaire sont appelés des documents structurés (André, Futura & Quint, 1989). Leur édition ne s'opère plus en fonction de la transformation de restitution proposée dès l'édition, soit selon le paradigme WYSIWYG (*What You See Is What You Get*), mais selon la sémantique des structures proposées, soit selon le paradigme WYSIWYM (*What You See Is What You Mean*) (Crozat, 2007, p. VI).

Sur le plan technologique, les chaînes éditoriales s'appuient sur les technologies de marquages que sont le SGML et son successeur le XML. Les briques technologiques élémentaires sont donc :

- un modèle de document XML (DTD (W3C, 2008), XML-Schema (W3C, 1999) ou RelaxNG (OASIS, 2001)) ;
- un éditeur de fichiers XML (par exemple, Oxygen, ou un simple éditeur de texte avec contrôle du schéma) ;
- un algorithme de transformation permettant une publication dans un format standard.

Exemple : chaîne éditoriale pour l'écriture et la publication d'un dictionnaire

Prenons l'exemple d'une chaîne éditoriale minimale pour l'édition d'un dictionnaire. Les structures qui composent un dictionnaire sont très semblables les unes aux autres. Elles comprennent le nom d'un terme, sa qualification grammaticale ainsi qu'une ou plusieurs définitions. Ces structures sont formalisées dans le schéma suivant exprimé dans le standard RelaxNG.

```
<?xml version="1.0" encoding="UTF-8"?>
<element name="dictionnaire" xmlns=
 "http://relaxng.org/ns/structure/1.0">
  <attribute name="titre"/>
  <oneOrMore>
    <element name="entree">
      <element name="terme">
        <text/>
      </element>
      <element name="qualifGram">
        <text/>
      </element>
      <oneOrMore>
        <element name="definition">
          <text/>
        </element>
      </oneOrMore>
    </element>
  </oneOrMore>
</element>

```

À partir de ce schéma, un éditeur XML permet l'édition de dictionnaires. L'exemple ci-dessous est un dictionnaire contenant deux définitions.

```
<?xml version="1.0" encoding="UTF-8"?>
<dictionnaire titre="Le Petit Nibbler">
  <entree>
    <terme>Rééditorialisation</terme>
    <qualifGram>Nom féminin</qualifGram>
    <definition>Le terme de rééditorialisation est un néologisme qui
 a émergé dans le domaine du document numérique pour désigner le
 processus consistant à reconstruire un nouveau document à partir
 d'archives. La construction de ce mot tente une première synthèse entre
 les concepts d'édition au sens de publication d'une œuvre,
 d'éditorialisation au sens d'expression d'un point de vue propre, de
 réédition au sens de nouvelle proposition de lecture. Elle tente une
 seconde synthèse entre les fonctions d'éditeur, celui qui met en forme
 et diffuse, et d'auteur, celui qui écrit, fonctions qui tendent à se
 mêler dans le contexte du numérique. La rééditorialisation est donc la
 publication d'une œuvre originale dans son point de vue, sa forme, sa
 scénarisation, à partir de contenus qui ne le sont pas tous.
 </definition>
  </entree>
  <entree>
    <terme>Redocumentarisation</terme>
    <qualifGram>Nom féminin</qualifGram>
    <definition>Désigne à la fois un retour sur une
 documentarisation ancienne et une révolution documentaire. (Pédauque)
 </definition>
    <definition>Processus de transfert de documents existants sur le
 support numérique pour leur permettre de nouvelles manipulations.
 (Salaun)</definition>
  </entree>
</dictionnaire>
```

Un algorithme de transformation permet ensuite de transformer un dictionnaire ainsi formé en un document encodé dans un format standard. L'exemple ci-dessous transforme le dictionnaire XML en document HTML. Le rendu HTML affiché par un navigateur est illustré dans la figure 6.

1. Transformation

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl=
 "http://www.w3.org/1999/XSL/Transform" xmlns:xs=
 "http://www.w3.org/2001/XMLSchema" exclude-result-prefixes="xs"
  version="1.0">
  <xsl:output method="html" encoding="UTF-8"/>
  <xsl:template match="/">
    <html>
      <head><title><xsl:value-of select="dictionnaire/@titre"
 /></title></head>
      <body>
        <h1><xsl:value-of select="dictionnaire/@titre"/></h1>
        <xsl:apply-templates/>
      </body>
    </html>
  </xsl:template>

  <xsl:template match="entree">
    <h2><xsl:value-of select="terme"/></h2>
    <p class="formGram"><xsl:value-of select="qualifGram"/></p>
    <xsl:for-each select="definition">
      <p class="def"><xsl:value-of select="."/></p>
    </xsl:for-each>
  </xsl:template>

</xsl:stylesheet>
```

2. Fichier HTML produit

```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">

      <title>Le Petit Nibbler</title>
  </head>
  <body>
    <h1>Le Petit Nibbler</h1>

    <h2>R&eacute;&eacute;ditorialisation</h2>
    <p class="formGram">Nom f&eacute;minin</p>
    <p class="def">Le terme de r&eacute;&eacute;ditorialisation
est un n&eacute;ologisme qui a &eacute;merg&eacute; dans le
domaine du document num&eacute;rique pour d&eacute;signer le
processus
      consistant &agrave; reconstruire un nouveau document
&agrave; partir d'archives. La construction de ce mot tente
une premi&egrave;re synth&egrave;se entre
      les concepts d'&eacute;dition au sens de publication
d'une &#339;uvre, d'&eacute;ditorialisation au sens
d'expression d'un point de vue propre,
      de r&eacute;&eacute;dition au sens de nouvelle
proposition de lecture. Elle tente une seconde synth&egrave;se
entre les fonctions d'&eacute;diteur, celui
      qui met en forme et diffuse, et d'auteur, celui qui
&eacute;crit, fonctions qui tendent &agrave; se m&ecirc;ler
dans le contexte du num&eacute;rique.
      La r&eacute;&eacute;ditorialisation est donc la
publication d'une &#339;uvre originale dans son point de vue,
sa forme, sa sc&eacute;narisation, &agrave; partir
      de contenus qui ne le sont pas tous.
    </p>

```

```
21      <h2>Redocumentarisation</h2>
22      <p class="formGram">Nom f&eacute;minin</p>
23      <p class="def">D&eacute;signe &agrave; la fois un retour
        sur une documentarisation ancienne et une r&eacute;volution
        documentaire. (P&eacute;dauque)</p>
24      <p class="def">Processus de transfert de documents
        existants sur le support num&eacute;rique pour leur permettre
        de nouvelles manipulations. (Salaun)</p>
25
26    </body>
27  </html>
```

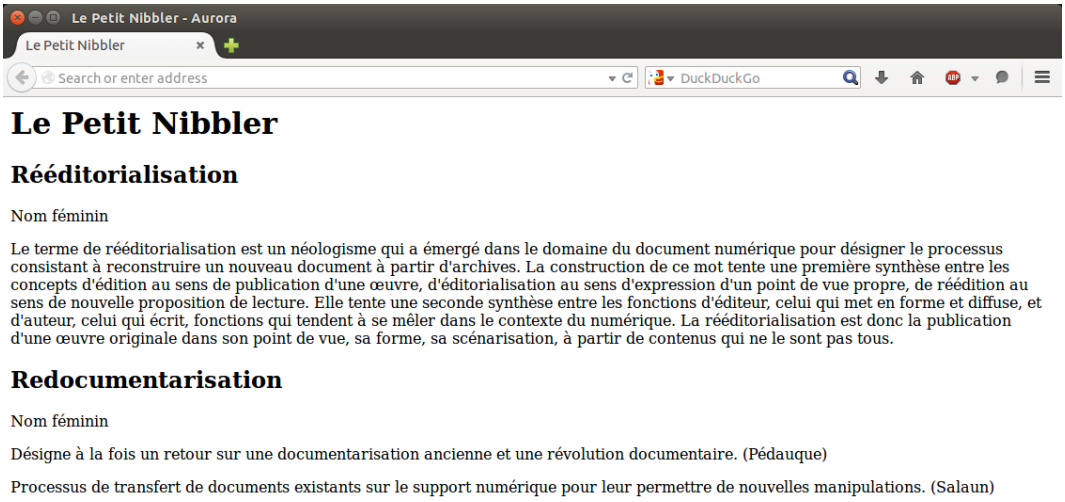


Figure 6 : fichier HTLM affiché dans un navigateur

Chaînes éditoriales et rééditorialisation

Les chaînes éditoriales disposent de deux atouts leur permettant d'être des outils adaptés pour l'expérimentation de la rééditorialisation.

Les documents qu'elles manipulent sont structurés et les structures sont conçues selon la sémantique des contenus. Cela permet de prévoir a priori des fragments de contenus pouvant être réutilisables et surchargeables. Par exemple, au sein du schéma de dictionnaire, une entrée est sémantiquement indépendante des autres tandis qu'un terme est intrinsèquement lié à ses définitions. Fragmenter et partager une entrée entre plusieurs dictionnaires est donc plus logique.

Les chaînes éditoriales assument la séparation entre le dispositif d'enregistrement et le dispositif de restitution. La rédaction des contenus s'opère dans un environnement graphique différent de la restitution (principe au cœur des environnements WYSIWYM). L'édition séparée, le référencement et la surcharge manuelle ou programmée des fragments ne constituent alors qu'une abstraction supplémentaire dans un processus d'abstraction déjà entamé.

La suite Scenari

Notre recherche est financée par la société Kelis assistée par un dispositif CIFRE. Elle s'inscrit dans le cadre industriel et technologique de la société. Kelis est l'éditeur de la suite logicielle de chaînes éditoriales Scenari (Crozat, 2007).

La suite Scenari est composée :

- de SCENARIbuilder, un logiciel de conception des modèles documentaires, des éditeurs XML associés et des algorithmes de transformation ;
- d'un ensemble de logiciels permettant un usage de bureau ou client - serveur (SCENARIclient, SCENARIserver, SCENARIserver-lite, SCENARIchain) exploitant un modèle produit par SCENARIbuilder pour instancier un espace pour la rédaction de documents ;
- de SCENARIstyler, un logiciel permettant de produire des modèles documentaires

alternatifs dont le stylage des documents produits est modifié.

L'ensemble des logiciels de la suite Scenari est publié sous licence libre. Ils sont soutenus par une communauté d'utilisateurs rassemblés administrativement dans une association loi 1901 présente en ligne sur le site scenari-plateforme.org *(http://scenari-platform.org)*.

1.5 Le graphe, structure de représentation des documents fragmentés

En outillant le principe de fragmentation des contenus et de référencement des fragments, les chaînes éditoriales modifient l'objet technique qu'elles manipulent. Il ne s'agit plus d'une archive regroupant l'ensemble des contenus utilisés pour un document, mais d'un réseau d'archives liées les unes aux autres. La structure logique informatique pour penser et opérer un tel réseau est le graphe. Dans cette section, nous introduisons la notion de graphe en particulier et la théorie des graphes en général. Nous montrons ensuite comment cette structure est mobilisable pour représenter les réseaux de fragments des chaînes éditoriales.

1.5.1 Théorie des graphes

L'histoire de la théorie des graphes débute avec les travaux d'Euler sur le problème devenu célèbre des ponts de Königsberg (Sachs, 1988). Euler cherchait à déterminer s'il existait un chemin empruntant les sept ponts Königsberg une seule fois. Pour résoudre son problème, Euler n'a pas réellement tracé un graphe, mais il a théorisé le problème de la même façon en nommant par une lettre les différentes terres séparées par les rivières et représenté les ponts par des séquences de lettres (AB relie ainsi la terre A à la terre B).

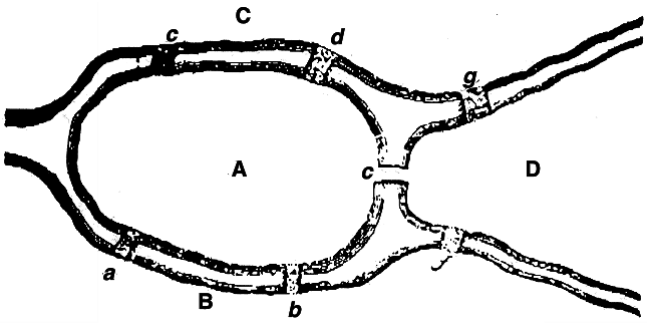


Figure 7 : les sept ponts de Königsberg (Hopkins, 2007)

L'objectif de la théorie des graphes est de résoudre des problèmes en simplifiant leur représentation à une série de sommets et d'arêtes reliant ces sommets. La résolution du problème peut alors être traitée de façon logique en opérant des algorithmes particuliers cherchant à optimiser le graphe, ou à déterminer le plus court chemin, ou juste à caractériser la structure du graphe. Un des problèmes les plus connus adressé par la théorie des graphes est le problème du voyageur de commerce qui cherche à visiter un certain nombre de villes par le chemin le plus court (Kruskal, 1956 ; Lin, 1965)

Nota Bene

La théorie des graphes est une discipline largement adressée tant par la recherche que dans la formation en logistique et informatique. Notre recherche mobilise des aspects rudimentaires de la discipline que nous définissons ci-après. Ces rudiments sont consensuels au sein de la discipline. La mobilisation de ces notions étant cependant novatrice dans l'analyse des chaînes éditoriales, nous rappelons ci-dessous leurs définitions (toutes issues de l'ouvrage « Éléments de théorie des graphes » (Bretto, Faisant & Hennecard, 2012)).

Définitions

Graphe. « Un graphe est un triplet $\Gamma = (V ; E, N)$ où V est l'ensemble des sommets du graphe ; il sera commode d'utiliser la notion $V(\Gamma)$ pour désigner l'ensemble des sommets du

graphe ; N est un ensemble qui sert à étiqueter les arêtes (par exemple N = {1, 2, ..., p}, N = {bleu, rouge, vert, ..., violet}, N = ℕ...) ; E ⊂ P_2(V) × N est l'ensemble des arêtes ; notation E = E(Γ) » (Bretto et *al.*, 2012, p. 1).

Degré. « Le degré d'un sommet x ∈ V est le nombre d'arêtes incidentes à x » (ibid., p. 6).

Chaîne. « Soit un graphe Γ = (V ; E, N) et x, y ∈ V ; une chaîne C de x à y est une succession finie d'arêtes » (ibid., p. 7). La longueur de la chaîne k correspond à la somme des arrêtes reliant x à y. « Une chaîne est simple si elle ne contient pas deux fois une même arrête. elle est élémentaire si elle ne contient pas deux fois un même sommet » (ibid., p. 8).

Cycle. « Un cycle est une chaîne fermée simple » (ibid., p. 8).

Graphe connexe. « Un graphe est dit connexe si pour toute paire de sommets x, y ∈ V, il existe une chaîne entre x et y : on dit alors que les sommets x et y sont connectés » (ibid., p. 8).

Sous-graphes. « Un graphe Γ' = (V' ; E', N') est appelé sous-graphe de Γ si V' c V et E' ⊂ E » (ibid., p. 9).

Graphe orienté. « Un graphe orienté ou digraphe $\vec{\Gamma}$ (ou simplement Γ) est un triplet $\vec{\Gamma}$ = (V ;$\vec{E}$, N) défini de la manière suivante : V est l'ensemble des sommets ; notation V = V($\vec{\Gamma}$) ; $\vec{E}$ ⊂ V × V × N est l'ensemble des arcs ; notation $\vec{E}$ = $\vec{E}$(Γ) ; N est un ensemble servant à étiqueter les arcs. Un arc a ∈ $\vec{E}$ sera noté ((x, y), n) : l'arc va de x vers y » (ibid., p. 14).

Chemin. « Un chemin C de x à y est une suite finie [d'arcs] ; l'entier k est la longueur du chemin. » « Un chemin est simple s'il ne contient pas deux fois le même arc ; il est élémentaire s'il ne contient pas deux fois le même sommet » (ibid., p. 15).

Circuit. « Un circuit est un chemin simple dont les extrémités coïncident » (ibid., p. 15).

1.5.2 Graphes documentaires

Fragment documentaire

Dans les premières sections de ce chapitre, nous utilisons le terme *fragment* pour désigner indifféremment un ensemble de données mobilisées pour la construction de la restitution du document sans constituer l'exhaustivité des données mobilisées. Notre recherche s'intéresse aux fragments manipulés par les chaînes éditoriales. Dans ce contexte, la notion peut être définie comme un ensemble de données ayant une identité propre dans le gestionnaire d'identifiants du système. Contrairement à une sélection dans un fichier XML que l'on pourrait réaliser avec la technologie XPATH (W3C, 1999), l'identité d'un fragment est indépendante de son contenu. Cette propriété permet de concevoir des systèmes robustes d'adressage des fragments.

Par exemple, une chaîne éditoriale peut isoler les fragments dans des fichiers distincts. Le gestionnaire des identifiants peut ainsi s'appuyer sur le chemin du système de fichiers. Quel que soit le contenu d'un fichier, son identifiant reste inchangé.

Le contrôle et la validité de la structure documentaire d'un fragment impliquent qu'un modèle fournisse une classe de fragments que l'on pourra instancier. Par exemple avec les technologies XML le modèle de chaînes éditoriales pour la rédaction de dictionnaires abordé dans la section précédente peut fournir un schéma XML pour des fragments de types *dictionnaire* et un autre schéma pour des fragments de types *entrée*. Une chaîne éditoriale manipule donc un ensemble d'objets fragments étiquetables selon leur classe.

On dira que deux fragments sont liés lorsqu'un fragment A fait référence à un fragment B. Ces liens sont orientés depuis le fragment contenant l'adresse vers le fragment adressé.

Le graphe documentaire

Soit V, l'ensemble des fragments manipulés par la chaîne éditoriale, E, l'ensemble des liens, orientés depuis le fragment référençant vers le fragment référencé, et N, l'ensemble des catégories de liens (soit ici uniquement un lien de référencement). Nous appelons *graphe documentaire* le graphe noté D = (V ; E, N) ainsi formé.

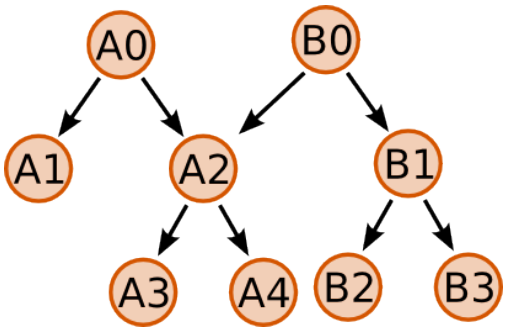


Figure 8 : graphe documentaire simple

Le graphe documentaire de la figure 8, noté D = (V ; E, N) est composé de l'ensemble des sommets V = {A0, A1, A2, A3, A4, B0, B1, B2, B3}, l'ensemble des étiquettes des arcs N = { référence} et l'ensemble des arcs orientés E = {(A0, A1), (A0, A2), (A2, A3),..., (B1, B3)}

Caractérisation des graphes documentaires

La figure 8 montre bien la déconstruction du document opérée par les chaînes éditoriales. Nous proposons d'enrichir les concepts permettant de caractériser les graphes documentaires avec deux nouvelles définitions.

Sommet racine. Soit un graphe documentaire D = (V ; E, N). Un sommet issu de V est appelé racine lorsque le modèle documentaire associe un algorithme de transformation de document au fragment qu'il représente. Nous noterons l'ensemble des sommets racines Vr. Notons que par définition, Vr ∈ V.

Sous-graphe document. Soit un graphe documentaire D = (V ; E, N) et un sommet racine Vr_1 tel que $Vr_1 \in Vr$, un sous-graphe documentaire est noté D'd = (V'd ; E'd, N) avec

- V'd : l'ensemble des sommets de V tel que quelque soit le sommet x ∈ V'd, il existe un chemin entre Vr_1 et x.
- E'd : l'ensemble des arcs de E dont les extrémités appartiennent à l'ensemble V'd.

Un sous-graphe document est donc constitué de l'ensemble des sommets à disposition de l'algorithme de transformation pour lui permettre de générer un document. Attention, la présence d'une information dans un sommet issu de V'd n'implique pas nécessairement sa publication. L'algorithme de publication peut volontairement exclure certaines informations. Par exemple, ce processus est à l'œuvre dans un fragment à profondeur variable. En filtrant le contenu pour une publication ou une autre, le rédacteur fait volontairement le choix d'exclure une information d'une publication.

Graphe documentaire et perception de l'auteur

La notion de graphe documentaire telle que nous la proposons représente la structure de l'agencement des fragments les uns avec les autres. Pour se repérer dans le graphe, les chaînes éditoriales exploitent d'autres stratégies que la typologie des liens. Une solution assez commune, appelée « arbre de gestion », consiste à simuler un arbre de dossiers dans lesquels les fragments sont rangés. Cette solution est par exemple retenue dans les chaînes éditoriales Scenari. Nous parlons ici de simulation car peu de chaînes éditoriales s'appuient réellement le système de fichiers pour stocker les fragments de documents.

Le graphe documentaire donne une vue représentative de l'objet sur lequel travaillent les rédacteurs. En revanche, cette vue, souvent assez lourde et peu lisible comme nous le verrons dans les prochains exemples, est souvent peu perçue. Ce manque de perception est un des fondements de notre recherche.

1.5.3 Représentation des graphes documentaires

Gephi, outil de manipulation de graphe

Gephi (Bastian, Heymann & Jacomy, 2009) est un outil conçu pour manipuler des graphes représentant des pages présentes sur le web et les liens hypertextes les liant entre elles. L'objet de telles visualisations est de faire ressortir les pages les plus populaires (les plus citées par d'autres pages) afin de faire ressortir des tendances du réseau.

Procédé de mise en forme des graphes

Nous avons développé un module complémentaire à Gephi permettant de se connecter à une chaîne éditoriale Scenari afin d'extraire les données nous permettant de reconstituer le graphe documentaire. Ce module extrait pour chaque sommet :

- les liens vers d'autres sommets,
- la classe du fragment,
- le titre du fragment (si défini par le modèle documentaire),
- les dossiers de premier et second niveau dans lesquels le fragment a été rangé par le rédacteur (par exemple, un fragment rangé dans « /procedures/machine-Outil-A72/ressources/ » aura « procedures » pour dossier de premier niveau et « procedures/machine-Outil-A72 » comme dossier de second niveau).

L'algorithme de spatialisation utilisé est ForceAtlas2 (Jacomy, Venturini, Heymann, Bastian, 2014). Il met en place un principe d'attirance et répulsion des sommets en fonction de leurs connexions. Un ensemble de sommets très fortement connectés les uns avec les autres s'attireront tandis que deux sommets non connectés se repousseront.

Les sommets sont colorés en fonction des dossiers dans lesquels ils sont rangés. La couleur permet donc de comparer la perception de rangement du rédacteur par rapport à la structure réelle du graphe. La taille des sommets varie en fonction du nombre d'arcs. Contrairement aux graphes permettant de visualiser les relations entre pages du web, nous avons préféré faire varier la taille des fragments en fonction des arcs sortants et non entrants. Les représentations des pages web cherchent à mettre en valeur les pages faisant figure de référence au sein d'un ensemble, donc les pages pointées par de nombreuses autres pages. À l'inverse, nos représentations mettent en exergue les fragments mobilisant de nombreux autres pour la construction de leur restitution.

Les visualisations obtenues ont été exportées en image pour une intégration dans la version papier de ce mémoire et en pages HTML interactives pour une manipulation par le lecteur dans la version HTML du mémoire.

Il est à noter que pour l'ensemble des graphes présentés, seuls les liens de transclusion et de référence ont été exportés, sans distinctions entre les deux statuts.

Exploitations

Afin d'accompagner la représentation d'un graphe, nous proposerons systématiquement une fiche d'identité compilant quelques données :

- le nombre de sommets présents sur le graphe ;
- le nombre de liens ;
- le degré moyen (soit le nombre moyen de liens par sommet) ;
- le degré sortant maximum (soit le nombre de liens sortants du sommet référençant le plus de fragments) ;
- le degré entrant maximum (soit le nombre de liens entrant du sommet le plus référencé par d'autres fragments) ;
- le nombre de sommets orphelins (soit le nombre de sommets de degré 0) ;
- le nombre de sommets racines.

Ces données nous permettent de réaliser plusieurs interprétations. Par exemple, le degré

moyen donne une indication sur le type de documents produits. La plupart des graphes mobilisés pour écrire des documents issus d'une tradition de l'imprimé (document linéaire à imprimer) sont rédigés dans des graphes au degré moyen proche de 1. Au contraire, les graphes mobilisés pour produire des bases documentaires en ligne tendent vers des degrés moyens plus élevés (le plus souvent, entre 2 et 4).

Le nombre de fragments orphelins est un indicateur de bonne prise en main de la chaîne éditoriale par les rédacteurs. Il est rare qu'un modèle documentaire permettent de produire un document à partir d'un seul et unique fragment. Dans ces cas, les fragments orphelins sont nécessairement des fragments inutilisés dans le graphe.

Outre les chiffres, la structure du graphe permet également de tirer des conclusions sur les contenus rédigés. On pourra ainsi observer des mutualisations de contenus entre plusieurs sous-graphes document ou la place centrale ou périphérique d'un fragment dans la construction d'un document.

Exemple, ce mémoire

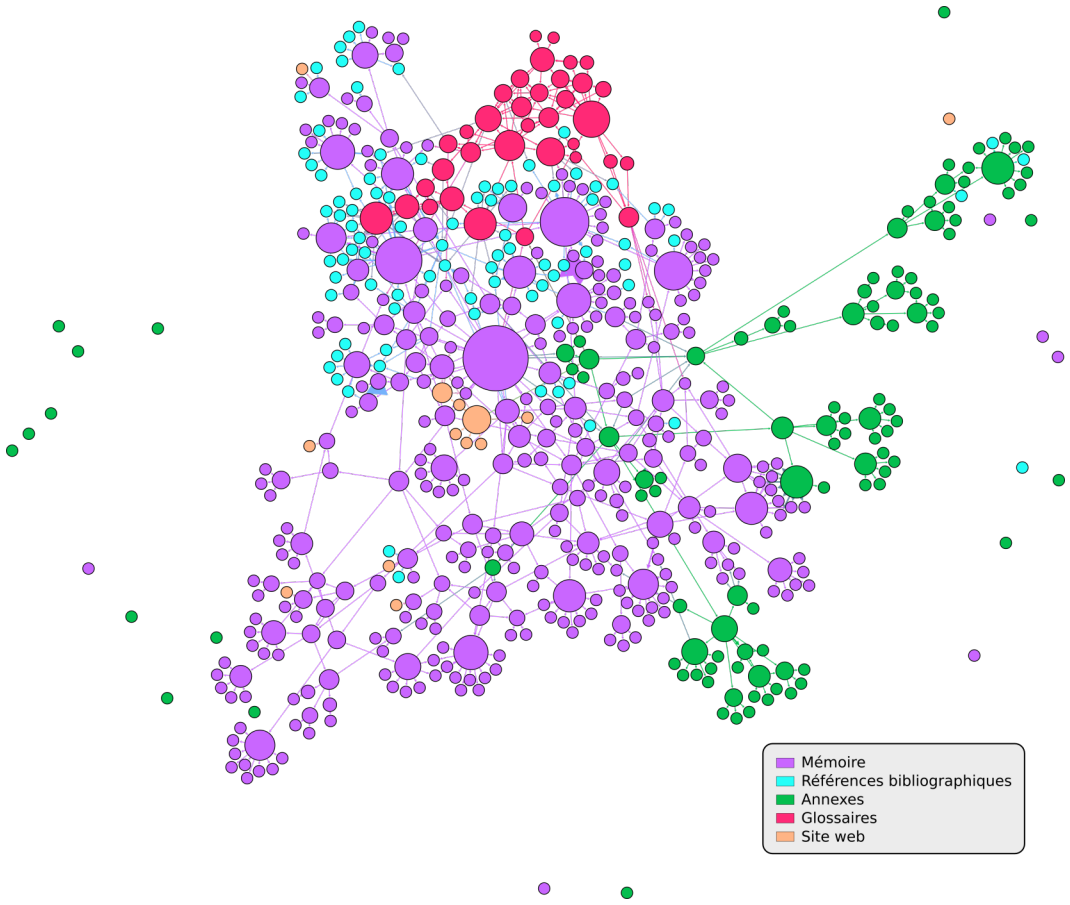


Figure 9 : graphe documentaire - ce mémoire

Fiche d'identité

Nombre de sommets : 574

Nombre de liens : 796

Degré moyen : 1.39

Degré sortant maximum : 26

Degré entrant maximum : 10

Nombre de sommets orphelins : 24

Nombre de fragments racines : 2

À 1.39, le degré moyen est légèrement supérieur à ce qui peut être observé sur ce type de graphe. Cette valeur s'explique d'une part par la structure de la partie violette du graphe, qui correspond au corps du mémoire. Nous avons volontairement inséré des liens entre fragments à chaque fois qu'une partie du mémoire est citée. Dans la version imprimée, ces liens sont transformés en numéros de pages et, dans la version web, par des hyperliens. D'autre part, cette valeur plutôt élevée s'explique également par la structure du glossaire qui s'appuie sur de nombreux liens entre chacune des entrées.

La forme globale du graphe est intéressante quant à la construction de ce mémoire. Dans la partie centrale et systématiquement entourés de références bibliographiques se trouvent les contenus des parties I, II et IV correspondant respectivement aux hypothèses, à l'état de l'art et à l'évaluation. La partie III, soit nos propositions, est construite en prolongement de ce noyau (partie violette en bas de la figure 9) et substitue les références bibliographiques par un ancrage au noyau théorique. Les annexes se positionnent en prolongement de la partie III avec une série de sommets verts encore plus extérieurs (en bas et à droite).

1.6 La modélisation des chaînes éditoriales

Les modèles de chaînes éditoriales

Les modèles de documents exploités par les chaînes éditoriales peuvent se concevoir selon deux paradigmes : ils peuvent être conçus pour s'adapter au mieux à un contexte d'usage donné, on parlera alors de modèles dédiés ; à l'inverse, ils peuvent être conçus pour gérer l'exhaustivité d'une famille de contextes, on parlera de modèles universels (Arribe, Crozat, Bachimont & Spinelli, 2012).

Historiquement portées par SGML, les approches par modèles dédiés sont aujourd'hui ancrées dans les technologies XML. L'intérêt du modèle dédié est par construction son adéquation au contexte adressé. C'est la solution juste et nécessaire au problème, permettant de traiter des structures documentaires métiers comme des tableaux comptables, des scénarios pédagogiques ou encore des procédures d'intervention, sans scories héritées de fonctions liées à d'autres contextes d'usage.

Un modèle universel est au contraire un modèle à forte valeur de généralité visant à circonscrire l'ensemble des usages pour une famille de contextes éditoriaux. Généralement portés par un organisme de standardisation comme le W3C ou OASIS, les modèles universels visent l'intégration d'un très large ensemble de besoins, et misent sur la mutualisation des développements autour du standard. Les exemples les plus utilisés sont DITA (OASIS, 2010) et DocBook (OASIS, 2009). La raison d'être du modèle universel, une fois celui-ci standardisé et les développements associés mûris, est la possibilité de disposer de chaînes éditoriales prêtes à l'emploi.

L'enjeu des méthodes de conception de chaînes éditoriales est d'être le plus générique possible afin de permettre l'adressage d'un maximum de contextes. Par construction, la conception d'une chaîne éditoriale exploitant un modèle universel est générique. L'objectif de la modélisation des chaînes éditoriales est d'atteindre un niveau de généricité équivalent pour la conception de chaînes éditoriales exploitant des modèles dédiés.

Généricité par déclinaison ou génération

Dans son ouvrage, Cassirer (1977) s'intéresse aux différentes théories de la notion de concept pour mettre en relief les notions de déclinaison et génération. Il y distingue deux approches : d'un côté la logique formelle forgée par Aristote, de l'autre celle des sciences modernes et contemporaines. L'objet de la logique formelle est l'étude de la métaphysique : « l'essence et l'articulation de l'être », dit autrement, ce qui est. Du côté des sciences modernes, la notion de concept ne s'appuie pas uniquement sur l'existence mais également sur la preuve, ce qui est vérifiable.

Le concept vu par la logique est « un rassemblement par similitude d'essence », c'est-à-dire un rassemblement d'individus par ressemblance. Par exemple, l'hirondelle, le moineau et l'aigle ont tous des plumes, des ailes et un bec. Ces caractéristiques constituent l'essence du concept d'oiseau. La généralisation d'un concept vers un concept de niveau

supérieur se fait en procédant au rassemblement des concepts de niveau inférieur. Un animal serait un mammifère, un oiseau, un amphibien, un poisson ou un reptile. Le concept universel serait alors une liste de toutes les essences possibles de ce qui est. L'universalité sera ici appelée abstraite car il n'existe pas de relation entre un concept et un sous-concept. Le passage de concept au sous-concept se fait par une déclinaison de l'ensemble des propriétés du concept.

Pour Cassirer, le concept scientifique est une abstraction de la liste des propriétés, permettant ainsi la réunion de sous-concepts qui ne se ressemblent pas dans un même concept. Ces sous-concepts sont obtenus par une fonction génératrice attachée au concept. En partant du nombre 0 et avec la loi successeur, il est possible de générer l'ensemble des entiers naturels. La généralisation de plusieurs concepts scientifiques se fait en changeant les fonctions génératrices. L'universalité est ici appelée concrète car les fonctions du concept universel permettent la génération de l'ensemble des individus qui le composent.

Déclinaison en ingénierie documentaire

Le modèle DITA (OASIS, 2010) est un exemple typique de modèle universel déclinable. Sa définition est relativement simple et se compose d'éléments génériques (*topic*, *task*, *map*, *reference*, *concept*). Un outil universel peut donc proposer l'utilisation de ces éléments et leur transformation en documentation technique. Le standard DITA inclut un système d'héritage qui permet la spécialisation d'un nouvel élément qui hérite des propriétés d'un des éléments génériques et inclut de nouvelles propriétés.

Un tel système de déclinaison permet en effet d'approcher l'universalité des usages. Néanmoins, chaque nouvelle déclinaison demande la production de code source spécifique ce qui est laborieux et onéreux. En outre, l'objectif d'interopérabilité du standard n'est que partiellement atteint. Il est bel et bien possible d'échanger des contenus entre n'importe quelle chaîne éditoriale construite autour du standard DITA mais un contenu spécialisé perdra l'ensemble de ses propriétés particulières et sera rapporté à l'élément générique dont il est issu.

L'ingénierie dirigée par les modèles

L'ingénierie logicielle apporte une théorie opérationnelle au concept de généricité par génération de Cassirer. Il s'agit de l'ingénierie dirigée par les modèles (IDM). La notion de modèle est ici à entendre selon la définition consensuelle de Rosenberg (1989, p. 75) : « Modeling, in the broadest sense, is the cost-effective use of something in place of something else for some cognitive purpose. It allows us to use something that is simpler, safer or cheaper than reality instead of reality for some purpose. A model represents reality for the given purpose; the model is an abstraction of reality in the sense that it cannot represent all aspects of reality. » Un modèle est donc une vue simplifiée de quelque chose afin d'en simplifier son usage. Bézivin (2005) propose d'illustrer la notion de modèle avec l'exemple de la cartographie. Une carte est effectivement une vue simplifiée permettant de montrer certaines réalités. Une carte sans simplification, soit à l'échelle 1:1 et représentant exactement un territoire, n'est d'aucune utilité.

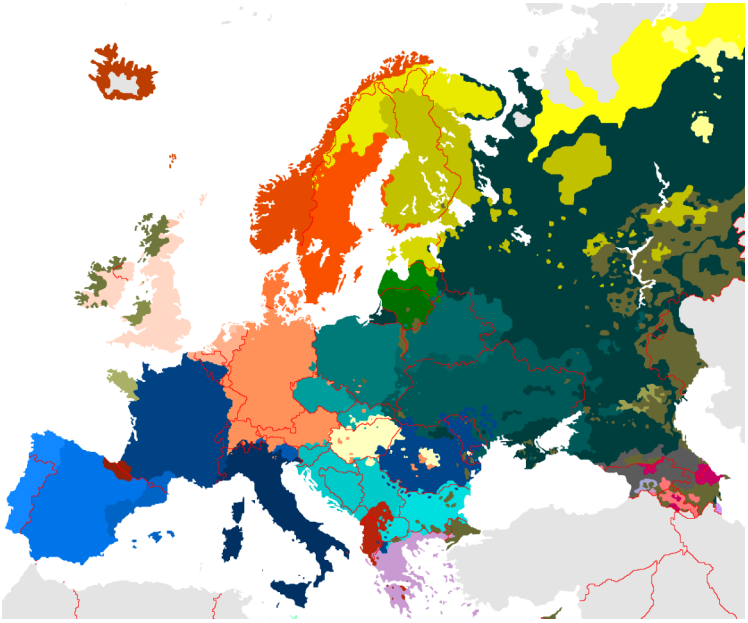


Figure 10 : une carte comme modèle de représentation du territoire

Pour dessiner et interpréter une carte, il est nécessaire d'avoir l'usage d'une légende. Sans légende, un observateur de la carte illustrée sur la figure 10 reconnaît la carte de l'Europe. Une observation attentive peut lui faire deviner que le code couleur est lié aux cultures européennes mais à moins d'être un spécialiste des données modélisées, il n'est pas possible d'aller plus loin. La légende montrée sur la figure 11 nous indique exactement le code couleur et les différentes cultures représentées sur la carte.

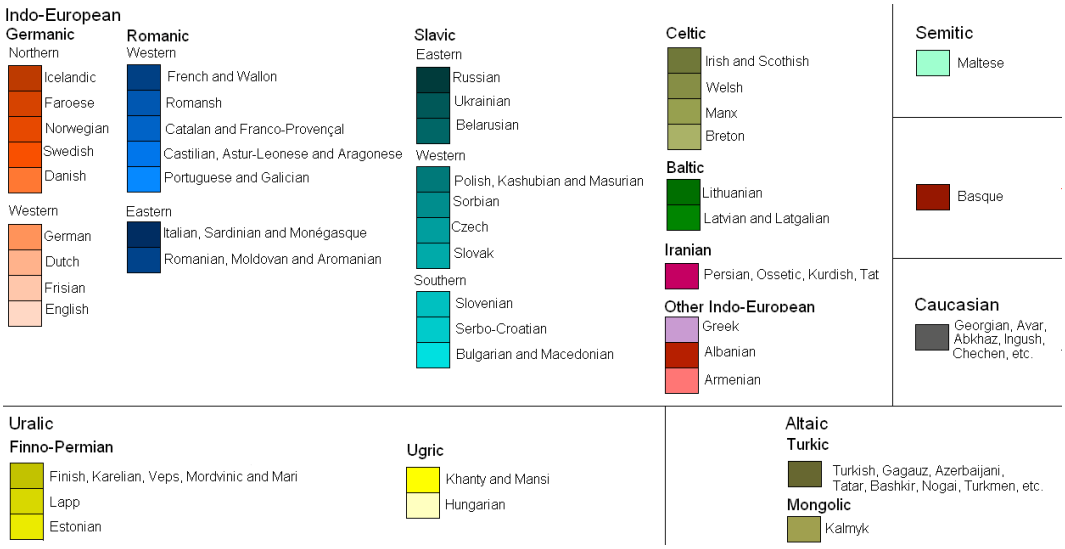


Figure 11 : légende de la carte de la figure 10

La légende est un méta-modèle de carte (ibid.). En posant une unique légende, il est possible de modéliser un nombre infini de cartes à toutes sortes d'échelles ou représentant des régions différentes.

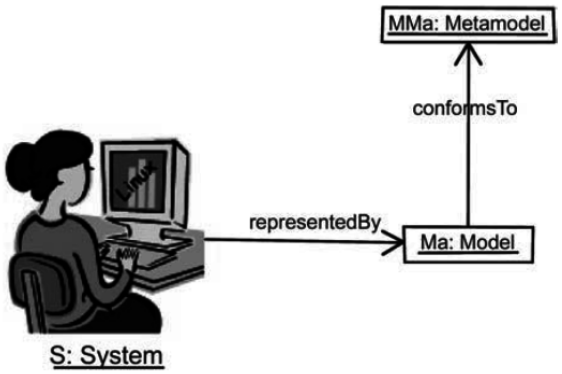


Figure 12 : système, modèle et méta-modèle (Bézivin, 2005)

Le concept clé de l'ingénierie dirigée par les modèles consiste à mobiliser les modèles, non plus comme une simple perspective assistant à la conception, mais comme une véritable ressource opérationnelle pour la production d'un système (Brambilla, Cabot & Wimmer, 2012). Ce changement dans l'usage des modèles se fait en utilisant un algorithme de transformation qui génère le système final à partir du modèle. Pour pouvoir être transformé, le modèle doit être valide. Cette validité et l'ensemble des possibles à modéliser sont définis par le méta-modèle. Algorithme de transformation et méta-modèle sont donc les deux éléments nécessaires à une approche de conception dirigée par les modèles.

Appliquée à la conception des chaînes éditoriales, la généricité préalablement visée est alors atteignable à condition de concevoir un méta-modèle permettant l'expression de l'ensemble des modèles documentaires et leurs générations ultérieures en chaîne éditoriale.

Modèle, IDM et raison computationnelle

Le passage de la modélisation sur papier à la modélisation dans un logiciel contrôlé par un méta-modèle constitue à notre sens une bonne illustration de la raison computationnelle. Outre le programme, le réseau et la couche, Bachimont (2010, pp. 169-171) complète l'analogie par la figure de la maquette numérique en comparaison au schéma. Les notions de modèles et d'ingénierie dirigée par les modèles sont à notre sens une généralisation de cet exemple. Le schéma est un modèle classique et la maquette numérique est un modèle manipulable et contrôlé.

SCENARIbuilder, logiciel de conception de chaînes éditoriales dirigé par les modèles

Le logiciel SCENARIbuilder s'inscrit dans le paradigme d'ingénierie dirigée par les modèles. Il permet la modélisation d'une chaîne éditoriale et la génération d'une archive de code source qui, utilisée avec SCENARIchain ou SCENARIserver, permet d'instancier une chaîne éditoriale au modèle dédié.

La modélisation s'appuie sur des primitives de documents et des primitives de transformations. Trois familles de primitives de documents sont à disposition des modélisateurs, les primitives de composition qui définissent les liens méréologiques entre un ensemble et ses parties, des primitives de métadonnées et des primitives de ressources. La combinaison de ces trois familles permet l'expression de n'importe quelle structure de document. Les primitives de transformation permettent de spécifier les algorithmes de transformation des structures documentaires valides vers des formats standards (PDF, OpenDocument, HTML).

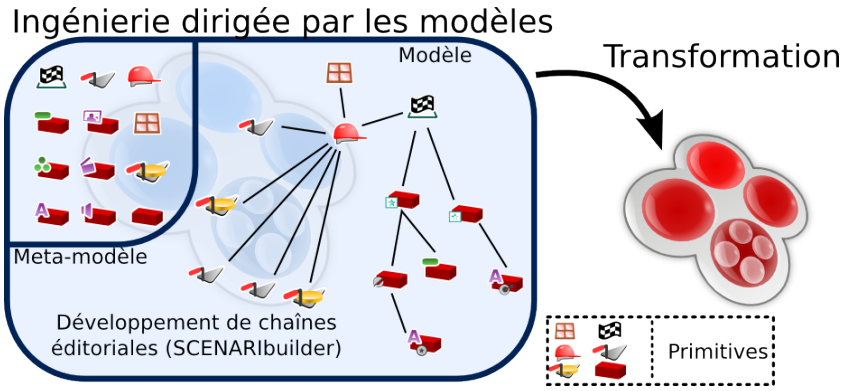


Figure 13 : ingénierie dirigée par les modèles pour la conception de chaînes éditoriales

Exemple

Nous utilisons le standard UML (OMG, 2011) afin de représenter un modèle Scenari. Chaque classe représente une primitive documentaire. Nous utilisons les stéréotypes pour donner le type de primitive choisie (composition, meta et ressource). L'exemple de dictionnaire de la section précédente peut être modélisé comme illustré sur la figure 14.

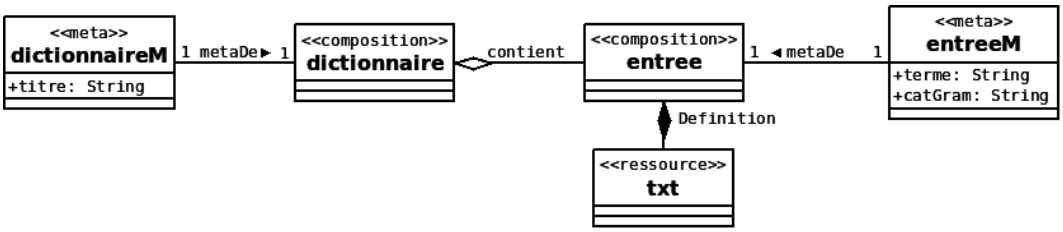


Figure 14 : modèle documentaire pour la rédaction d'un dictionnaire

Dans cet exemple, les primitives de méta-données sont utilisées pour définir des champs de données (*titre* pour le dictionnaire, *terme* et *catGram* pour les entrées). Un *dictionnaire* référence des fragments *entree* qui contiennent des *définitions*.

Les différentes primitives sont rédigées dans SCENARIbuilder, l'environnement de modélisation et de génération des modèles documentaires (figures 15 et 16).

```
sm:compositionPrim name="Dictionnaire" category="" info=""
  sm:help...
  sm:identification targetNamespace="isc.utc.fr:dic" targetPrefix="dic" code="dic" itemExtension="xml"
  sm:structure
    sm:meta title.model usage="required"
    sm:set usage="required"
      sm:part code="entree" name="Entrée" family="" internalized="never" linkTraits=""
        sm:allowedModel entree.model restrictUserDep=""
```

Figure 15 : primitive Dictionnaire définie dans SCENARIbuilder

```
sm:compositionPrim name="Entrée" category="" info=""
  sm:help...
  sm:identification targetNamespace="isc.utc.fr:dic" targetPrefix="dic" code="entree" itemExtension="xml"
  sm:structure
    sm:meta entreeM.model usage="required"
    sm:set usage="required"
      sm:part code="def" name="Définition" family="" internalized="always" linkTraits=""
        sm:allowedModel txt.model restrictUserDep=""
```

Figure 16 : primitive Entrée définie dans SCENARIbuilder

Une fois transformé par SCENARIbuilder et interprété par SCENARIchain, le modèle

permet l'instanciation de fragments *dictionnaire* et de fragments *entrée* (figures 17 et 18).

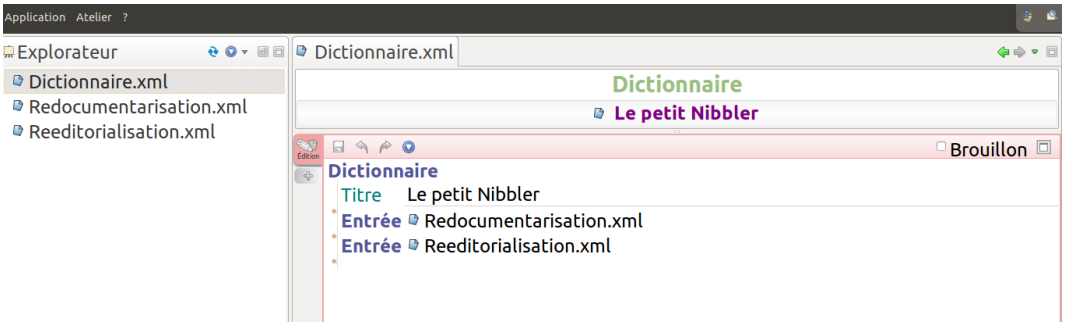


Figure 17 : capture d'écran de l'éditeur d'un fragment dictionnaire

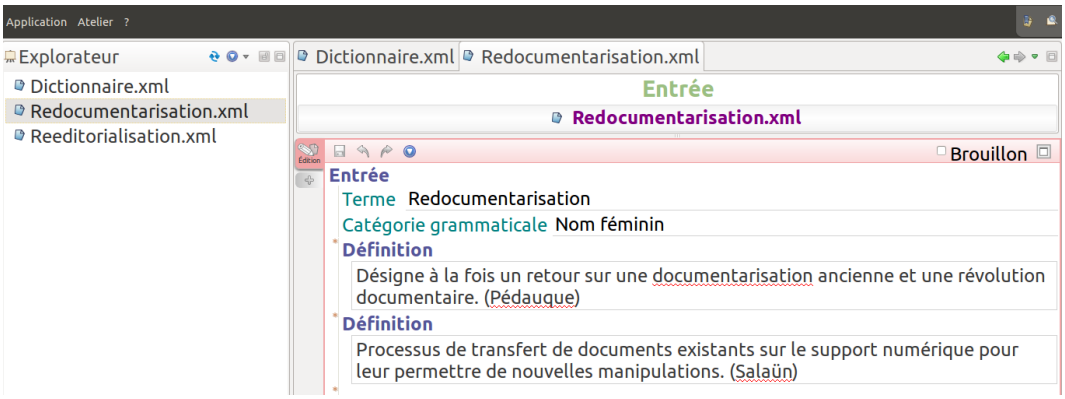


Figure 18 : capture d'écran de l'éditeur d'un fragment entree

Chapitre 2

La rééditorialisation, exemples de mise en œuvre

2.1 Enaction Series 40
2.1.1 Le modèle Enaction 40
2.1.2 Graphe documentaire 42
2.2 Enseignements universitaires 43
2.2.1 Le modèle Opale 43
2.2.2 Exemple de graphe documentaire 46
2.3 Direction de l'Information Légale et Administrative 47
2.3.1 Le modèle Copéria 47
2.3.2 Graphe documentaire 49
2.4 Quick 50
2.4.1 Modèle documentaire 50
2.4.2 Graphe documentaire 52

Le premier chapitre nous a permis de circonscrire l'ancrage théorique de notre recherche. Ce second chapitre s'inscrit dans son prolongement en donnant à voir quatre contextes d'exploitation des chaînes éditoriales mises en place par l'unité ICS de l'UTC *(http://ics.utc.fr)* ou par la société Kelis à l'aide de la suite Scenari. Il s'agit de :

- la documentation interne de la société Quick *(www.quick.fr)*,
- la rédaction du site service-public.fr *(http://service-public.fr/)* par la Direction de l'Information Légale et Administrative,
- l'édition scientifique en ligne avec la collection Enaction Series *(http://www.enactionseries.com/)*
- la production de contenus dans l'enseignement supérieur (au sein de l'UTC *(http://www.utc.fr/)*, ou tel qu'expérimentée par le projet investissement d'avenir SUP E-educ *(http://www.universites-numeriques.fr/SUP-E-educ/)*).

Ces contextes représentent une large diversité d'usages, tant dans le type de documents produits (documents pédagogiques, scientifiques, techniques et administratifs) que dans

l'organisation de la production (du simple rédacteur isolé à plusieurs équipes collaborant entre elles).

Ces différents contextes seront tous mobilisés au cours de ce mémoire afin d'illustrer nos propositions.

2.1 Enaction Series

Contexte

Enaction Series est une édition scientifique en ligne dédiée au paradigme de l'énaction. Elle est portée par le laboratoire Costech *(http://www.utc.fr/costech)* de l'UTC, plus particulièrement par Olivier Gapenne et Bruno Bachimont. Son objectif est de permettre à des chercheurs expérimentés de capitaliser leurs travaux dans un ouvrage accessible librement sur le web, et d'instaurer un échange savant - appelé dans ce contexte, de la glose - entre de jeunes chercheurs et les auteurs.

2.1.1 Le modèle Enaction

Le modèle Enaction Series permet l'écriture d'un livre contenant une ou des préfaces, une introduction, des chapitres, une conclusion, une discussion entre l'auteur et ses glossateurs et des annexes. Chacune de ces parties peut contenir trois niveaux de chapitrage (section, sous-section et sous-sous-section). En outre, un objet livre peut être associé à un ou plusieurs objets auteurs (contenant la bibliographie de l'auteur) et des métadonnées classiques.

Un soin particulier a été apporté à l'esthétique de la publication web. Une publication PDF est utilisée pour effectuer un dépôt légal des ouvrages et éventuellement réaliser des impressions à la demande des auteurs.

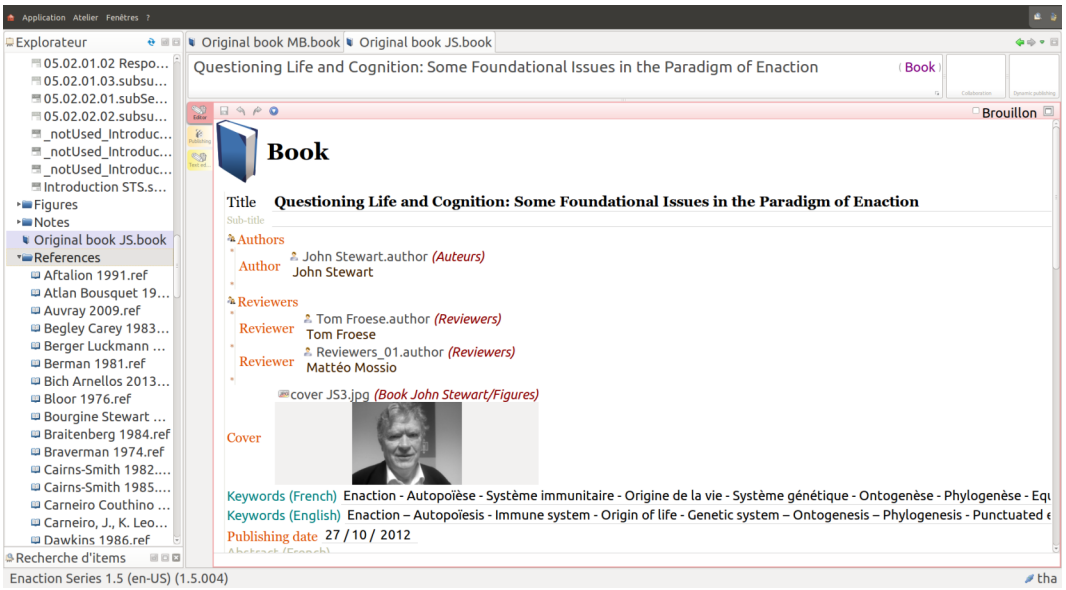


Figure 19 : éditeur du modèle Enaction Series

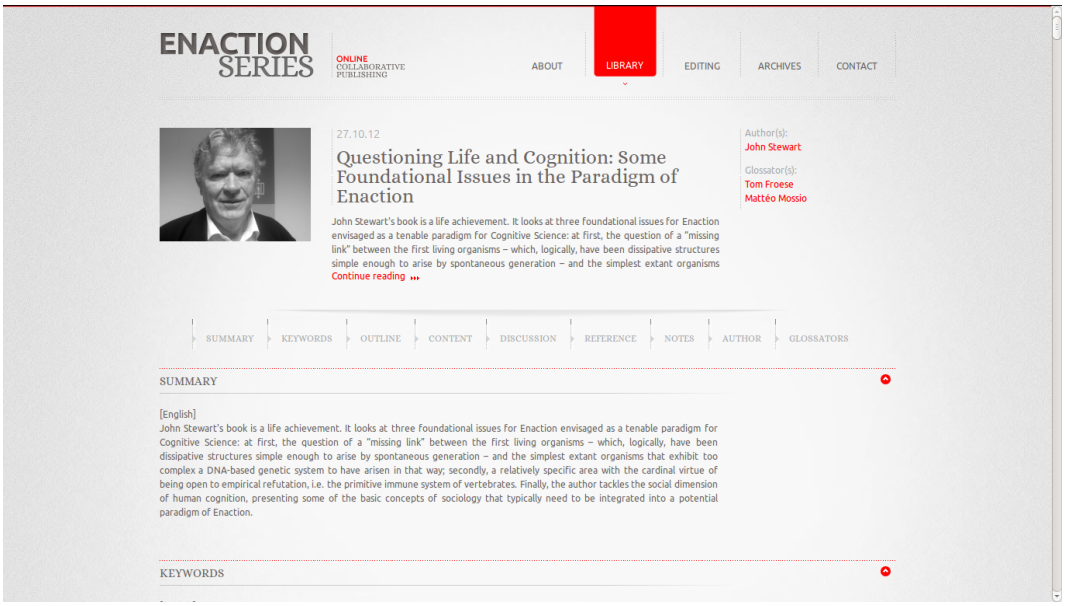


Figure 20 : publication en ligne - Enaction Series

2.1.2 Graphe documentaire

Figure 21 : graphe documentaire - Enaction Series

Fiche d'identité

Nombre de sommets : 939
Nombre de liens : 1021
Degré moyen : 1,09
Degré sortant maximum : 30
Degré entrant maximum : 5
Nombre de sommets orphelins : 15
Nombre de fragments racines : 2

Interprétations

Le nombre de sommets total du graphe est raisonnable et permet une bonne lecture. Le degré moyen, très proche de un, montre que les fragments sont majoritairement mobilisés une unique fois pour la publication. Ce chiffre est classique dans la production de documents linéaires issus d'une tradition de l'imprimé. Seuls huit fragments sont réutilisés quatre ou cinq

fois. Il s'agit majoritairement de références bibliographiques ou de sections plusieurs fois rappelées dans le déroulé du livre. Le degré sortant maximal montre que les fragments contiennent une quantité de contenu importante. Il est à mettre en relation avec le nombre relativement faible de fragments par rapport à la taille des ouvrages. Il est logique qu'un fragment contenant plusieurs pages, voire plusieurs dizaines de pages, référence un nombre important d'entrées bibliographiques.

Du point de vue de la structure du graphe, les fragments violets correspondent à l'ouvrage de Michel Bitbol (2013), les verts à celui de John Stewart (2013), les rouges aux fragments de description des auteurs et les jaunes aux fragments de description des glossateurs. On peut noter que la partie violette du graphe contient très peu de références entre chapitres. Les différents chapitres, sections, sous-sections et sous-sous-sections sont relativement indépendants et les références bibliographiques sont rarement utilisées à deux endroits différents. À l'inverse l'ouvrage représenté par la partie verte du graphe contient plusieurs références entre fragments de contenu où les mêmes références sont utilisées à partir de plusieurs fragments.

2.2 Enseignements universitaires

Contexte

Cette section est dédiée à la mobilisation des chaînes éditoriales dans des contextes de production pédagogiques universitaires. Nous y présentons le modèle de chaînes éditoriales libres Opale *(http://scenari-platform.org/projects/opale/fr/pres/co/index.html)* ainsi qu'un graphe documentaire exploité par l'unité ICS de l'UTC.

2.2.1 Le modèle Opale

Le modèle Opale est construit autour de modules qui scénarisent des activités d'apprentissage et des activités d'évaluation. Ces activités sont elles-mêmes composées de grains de contenu ou d'exercices rédactionnels. Un grain de contenu se compose d'un enchaînement de blocs titrés et typés (par exemple avec un balisage sémantique indiquant un contenu important, un contenu juridique ou encore une définition).

Le modèle Opale permet de publier des supports de présentation à utiliser en séance, des cours à imprimer et distribuer aux étudiants et des modules de cours à diffuser sur le web. En outre, la chaîne éditoriale Opale prévoit un système de texte de filtrage des contenus programmables : par défaut chaque bloc de contenu est mobilisé pour les publications dites *standards* et *courtes*. Par un jeu de filtres, il est possible de restreindre un bloc uniquement à une publication standard ou à une publication courte. Lors de la publication d'un support, il est possible de choisir quel jeu de filtres activer (standard ou court). Ce système est souvent utilisé pour rédiger côte à côte des contenus destinés aux cours de référence et aux supports de présentation : un premier bloc complet sera filtré (donc, non publié) sur la publication courte tandis qu'un second bloc résumé sera filtré sur la publication standard.

Figure 22 : éditeur du modèle Opale

Figure 23 : publication PDF du modèle Opale

Figure 24 : publication web du modèle Opale

2.2.2 Exemple de graphe documentaire

Le graphe documentaire proposé dans cette section est issu d'un atelier de rédaction de contenus pédagogiques mis à disposition des enseignants de l'UTC par l'unité ICS. Il contient cinq cours rédigés par un seul enseignant.

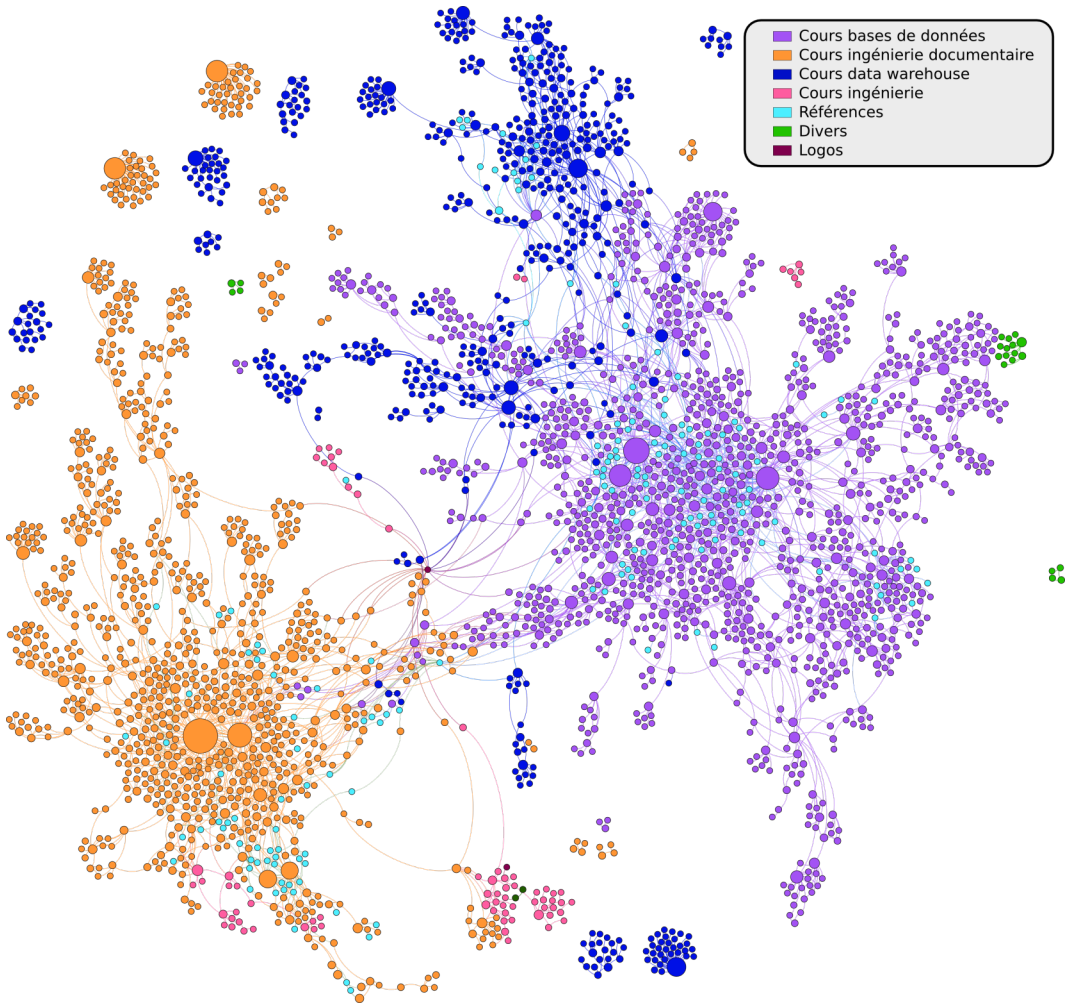


Figure 25 : graphe documentaire - Opale

Fiche d'identité

Nombre de sommets : 4135

Nombre de liens : 4753

Degré moyen : 1,15

Degré sortant maximum : 130

Degré entrant maximum : 49

Nombre de sommets orphelins : 802

Nombre de fragments racines : 158

NB : pour simplifier la lisibilité du graphe, les 802 sommets orphelins ne sont pas représentés.

Remarque

Le graphe contient 19% de sommets orphelins. Mis en relation avec le modèle

documentaire qui ne permet pas la publication de tels fragments, ce chiffre est relativement important. Il convient de corriger le degré moyen en conséquence afin d'obtenir une donnée plus proche de la forme du graphe formé par les fragments mobilisés pour la publication des documents. Le degré moyen corrigé est donc égal à 4753/(4135-802), soit une valeur de 1,43.

Interprétations

Le degré moyen corrigé est d'une valeur supérieure au graphe d'Enaction Series (+20%). Cette augmentation s'observe par des agglomérats de fragments plus denses et moins ordonnés. Au niveau du contenu, cela indique un nombre plus important de fragments mobilisés dans plusieurs contextes que dans les ouvrages scientifiques d'Enaction Series. Cette diversité dans les usages d'un même fragment s'explique par la nature du contenu : un contenu important pourra être rappelé plusieurs fois dans un même cours et un contenu générique pourra être mutualisé entre plusieurs cours. Le degré sortant maximal est à relativiser. Seuls seize sommets, correspondant à des modules ou exercices mobilisant de nombreuses ressources, disposent de plus de vingt liens sortants. 110 fragments sont mobilisés dans quatre contextes ou plus dont environs 60% représentent des ressources statiques (acronymes, ressources bibliographiques, images, etc.) et 40% des sommets référencent des ressources rédigées (activités d'apprentissage, grains de contenu ou exercices rédactionnels). Le graphe est composé de 158 fragments racines, soit 158 sous-graphes documents différents qui s'entrecroisent entre eux.

Nous constatons sur ce graphe une proximité entre les contenus dédiés au cours sur les bases de données et celui sur les *data warehouse*. Le second peut ainsi être considéré comme une extension du premier : sa partie du graphe (fragments bleus foncés) référence toute une série de fragments limitrophes issus du cours sur les bases de données. À l'inverse, le cours sur l'ingénierie documentaire partage très peu de contenus avec le reste du graphe. Seule une poignée de fragments dédiés à des concepts génériques sont mutualisés (il s'agit de cours sur la modélisation objet et sur les technologies HTML, XHTML).

2.3 Direction de l'Information Légale et Administrative

Contexte

La Direction de l'Information Légale et Administrative (DILA) est une structure dépendant directement du premier ministre. Elle est notamment en charge de la rédaction, de la maintenance et de la publication de la diffusion légale (journal officiel, textes législatifs et réglementaires, bulletins officiels), de l'information administrative (service-public.fr et service de renseignement par téléphone au 3939) et de l'édition publique (édition des différents codes, conventions collectives, périodiques des collectivités locales).

La DILA utilise une chaîne éditoriale numérique (appelée Copéria) pour la rédaction du site service-public.fr et de la base documentaire fournie au service assurant le renseignement administratif par téléphone. Une particularité de cette chaîne éditoriale réside dans le statut intermédiaire de ses publications. La transformation des documents issus de la chaîne éditoriale se fait vers un schema XML propre à la DILA. Les documents intermédiaires sont utilisés par un moteur de publication pour produire le site web service-public.fr. Ces contenus peuvent également être réutilisés par les collectivités locales et leurs groupements (mairies, préfectures, communautés de communes, etc.) pour la production de documents ou bases documentaires particulières.

2.3.1 Le modèle Copéria

Le modèle Copéria

Les fragments mobilisés dans le modèle Copéria peuvent à nouveau être segmentés en trois catégories : les satellites, les fiches et la structure. Les satellites désignent de petits fragments qui sont ré-appelés dans de nombreuses pages. Par exemple, sur la page du site service-public.fr dédiée aux fondations reconnues d'utilité publique *(http://vosdroits.service-public.fr/associations/F31023.xhtml)*, les blocs en bas de page sont issus de satellites (« Services en ligne et formulaires », « Où s'adresser » et « Références »). Les fiches forment les pages de

contenus du site. La page sur les fondations d'utilité publique précédemment citée est rédigée dans un fragment de type fiche. Enfin la structure permet d'organiser les fiches entre elles. Elle se compose de différents niveaux permettant de répertorier les fiches les unes avec les autres : les canaux (Téléphone, Web), les audiences (Associations, Professionnels, Particuliers), les thèmes (par exemple pour les particuliers : Argent, Étranger - Europe, Famille, etc.) et enfin les sous-thèmes (par exemple pour le thème argent : Assurance, Banque, Consommation,etc.). Cette structure est directement visible dans la hiérarchie du site web final.

Il est à noter qu'à chacun des niveaux, certains fragments sont directement fléchés pour la publication web, d'autres pour la publication des fiches dédiées aux réponses téléphoniques.

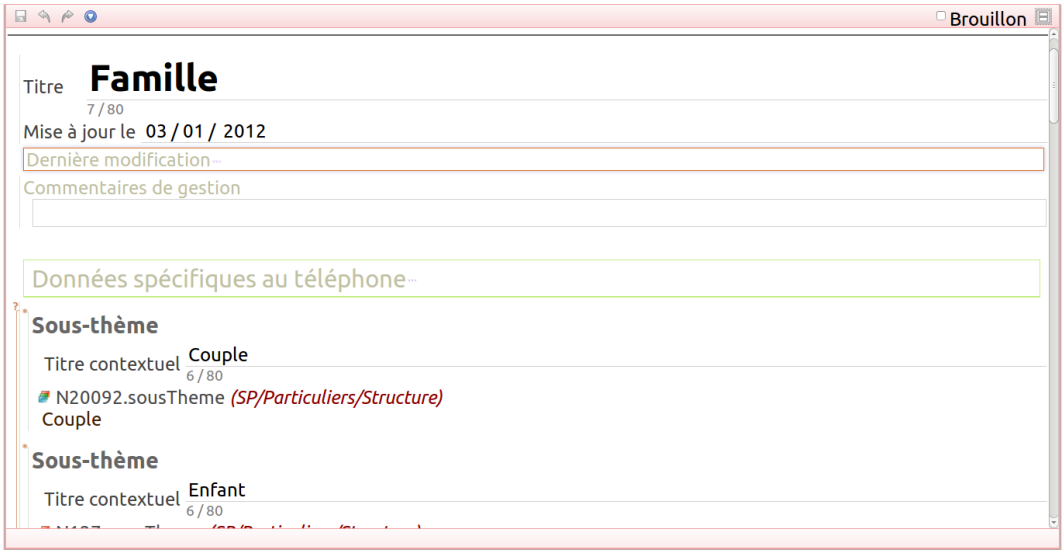


Figure 26 : éditeur du modèle Copéria

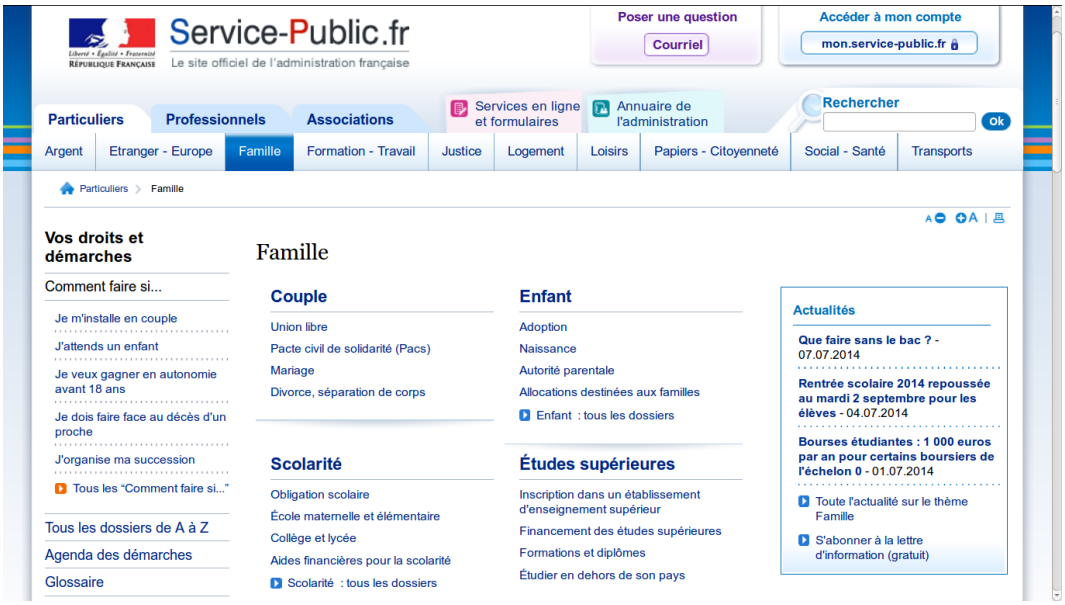


Figure 27 : publication modèle Copéria (NB : la DILA effectue un post traitement sur la publication de la chaîne éditoriale)

2.3.2 Graphe documentaire

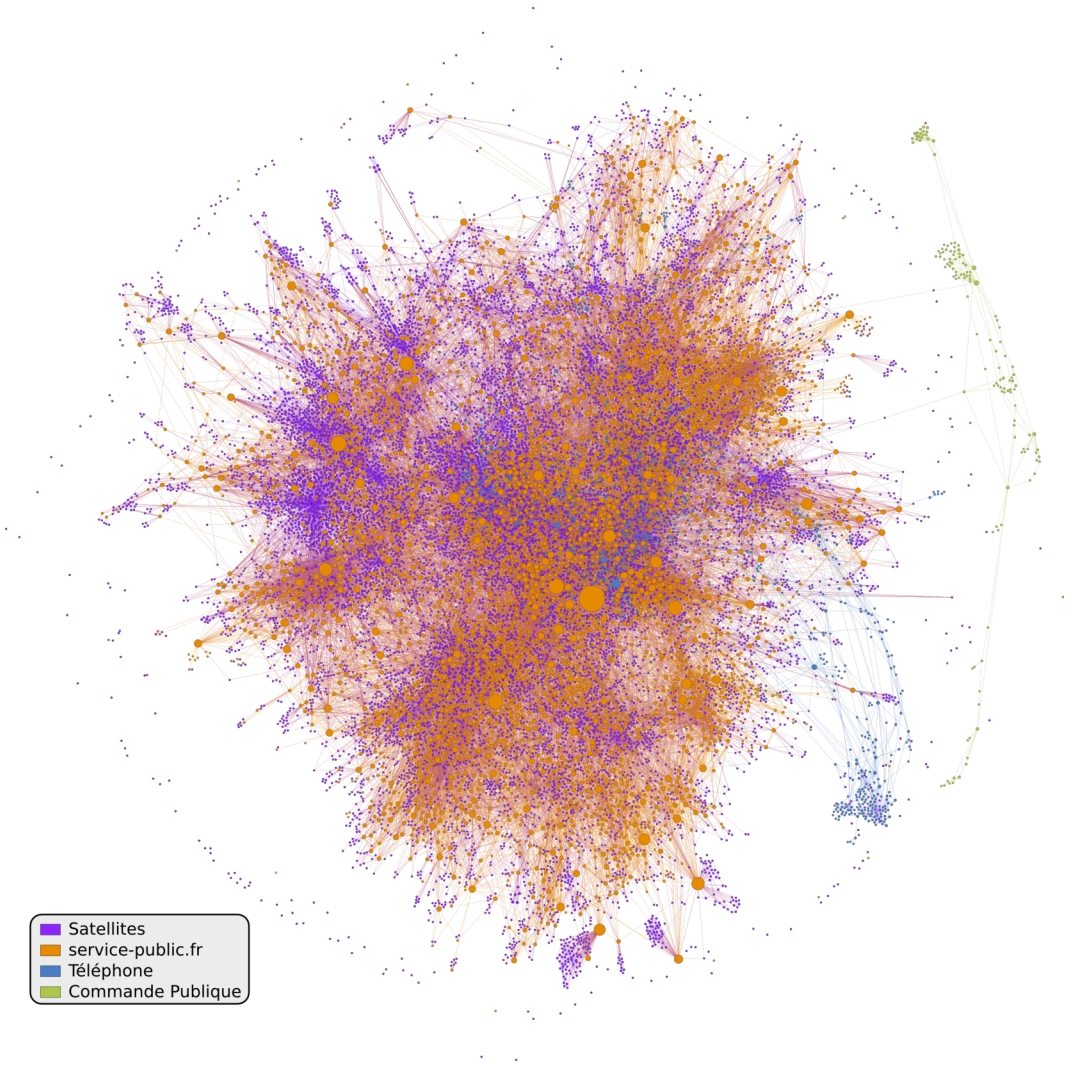


Figure 28 : graphe documentaire - DILA

Fiche d'identité

Nombre de sommets : 17999
Nombre de liens : 53969
Degré moyen : 3,00
Degré sortant maximum : 188
Degré entrant maximum : 1116
Nombre de sommets orphelins : 222
Nombre de fragments racines :2

Interprétations

Le graphe documentaire mobilisé pour la DILA est d'envergure importante. Avec plus de quinze mille sommets, il fait partie des usages industriels de la rééditorialisation. Le degré moyen est ici très élevé. Il correspond à une documentation où de très nombreux renvois internes sont effectués. Typiquement, cela correspond à la rédaction de sites web dont les pages sont fortement interconnectées plutôt que d'écrits de tradition imprimée (comme des articles, mémoires, livres ou rapports). Ce degré a un impact sur la lisibilité de la visualisation du graphe.

Hormis quelques exceptions, peu de sommets sortent du regroupement principal.

La réutilisation de contenus est ici élevée comme le montre le tableau suivant.

Nombre de réutilisations	2 ou plus	5 ou plus	10 ou plus	1116
Proportion de fragments	42% Plus de 7500 fragments	12% Plus de 2000 fragments	4% Plus de 700 fragments	1 fragment

Les fragments *service public* sont mobilisés pour la publication du site web et les fragments *téléphone* pour la publication des fiches d'aide à la réponse téléphonique. Parmi les groupement de fragments peu connectés à la structure principale, notons une traîne des fragments *téléphone* sur le côté droit et les fragments *commandes publiques*, pratiquement déconnectés du reste de la structure.

En parcourant le graphe, on constate que le fragment d'audience *Particuliers*, le fragment le plus volumineux en plein cœur du graphe, se retrouve au centre d'un vaste agglomérat de fragments oranges, soit de fragments de structures ou de fiches. À l'inverse, les fragments d'audiences *Professionnels* et *Associations*, situés en haut à gauche du graphe sont principalement entourés de fragments satellites. De cette forme, nous déduisons que la partie des contenus dédiés aux particuliers est plus importante que celle dédiée aux professionnels ou aux associations.

2.4 Quick

Contexte

La société Quick est une enseigne de restauration rapide présente en France, en Belgique et au Luxembourg. Elle s'appuie sur un réseau d'un peu moins de 500 restaurants dont 80% en franchise. Quick a d'abord été un partenaire de recherche de la cellule ICS avant de devenir un client de Kelis.

Avant de diffuser une nouvelle recette (par exemple, pour une opération promotionnelle), ou de déployer de nouveaux équipements, la société Quick rédige auparavant un dossier d'homologation qui est validé par un expert de la société. L'essentiel du contenu de ce dossier est ensuite copié dans le « référentiel des normes et méthodes » utilisé dans les restaurants (anciennement appelé, la « Bible », en interne). Le référentiel correspond à la documentation exhaustive de l'ensemble des techniques et méthodes en usage dans les restaurants. Ses fiches sont mobilisées à la fois comme référence dans les restaurants, et comme ressource pour la constitution de contenus de formation des nouveaux collaborateurs.

En complément de ce processus documentaire, la société Quick est confrontée à des problématiques de localisation avec l'adaptation du référentiel pour les restaurants belges. Il est d'abord nécessaire d'adapter la version française afin de mettre les contenus en conformité, tant au niveau linguistique que réglementaire. Ensuite, il convient de traduire ces contenus préalablement adaptés en néerlandais.

2.4.1 Modèle documentaire

Le modèle documentaire du référentiel des normes et méthodes est composé de trois catégories de fragments de documents : les fragments de publication, les fiches et les ressources. Les ressources représentent des fichiers binaires (images, vidéos) ou des fragments structurant des références communes (par exemple, les valeurs et unités de mesure de certains ingrédients). Les fiches constituent le cœur de la documentation. Elles sont dédiées soit à une connaissance théorique, soit à une matière première (composition, conditionnement, stockage, préparation), soit à un équipement (identification, description, manuel d'utilisation), soit à un produit cuisiné dans le restaurant (description, référencement des procédures de fabrication), soit enfin à une procédure (liste des étapes pour l'entretien d'un équipement ou la fabrication d'un produit). Le

dernier niveau de fragment vise uniquement à agréger des fiches pour la publication d'un document comme par exemple le référentiel qui agrège toutes les fiches en cours d'usage ou les documents dédiés à toutes les fiches concernant les produits vendus pendant une campagne thématique (comme par exemple la campagne de l'été 2014 : Giant Max ; ou la campagne Jedi/Dark Burger).

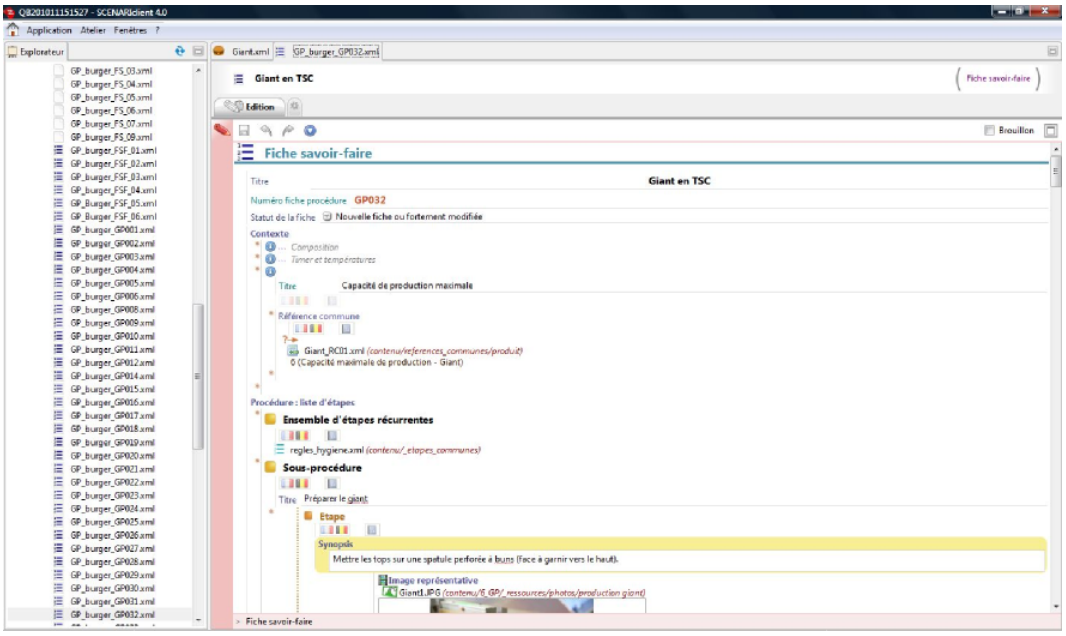


Figure 29 : éditeur du modèle documentaire de Quick

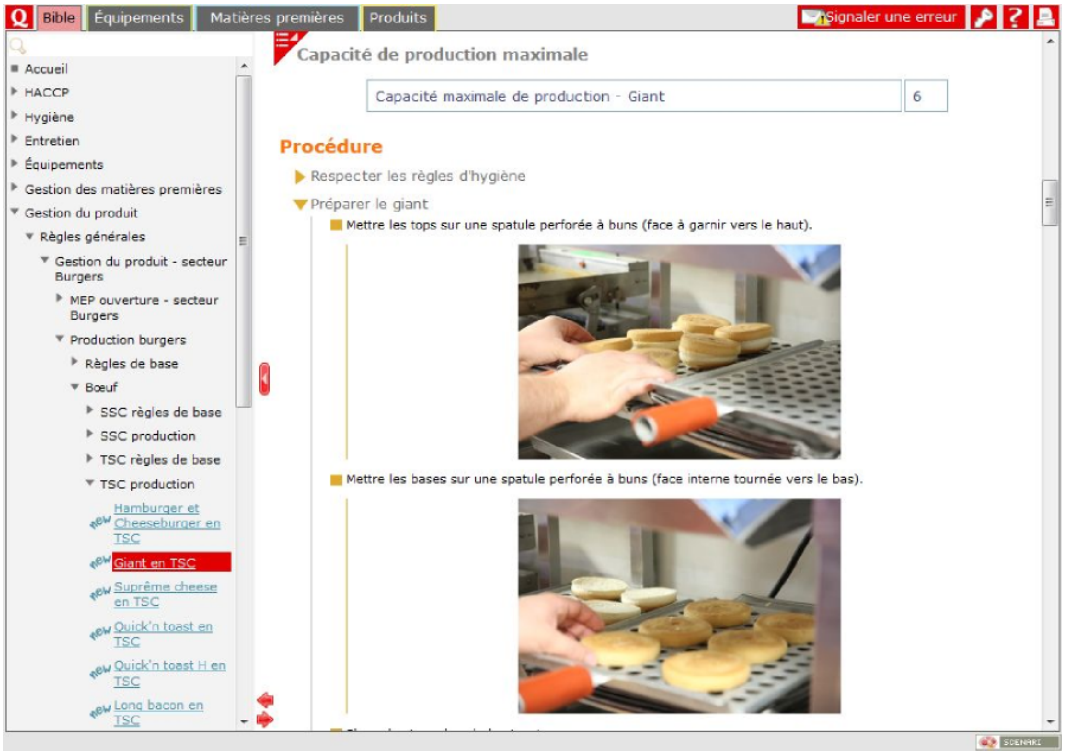


Figure 30 : publication web du référentiel Quick

2.4.2 Graphe documentaire

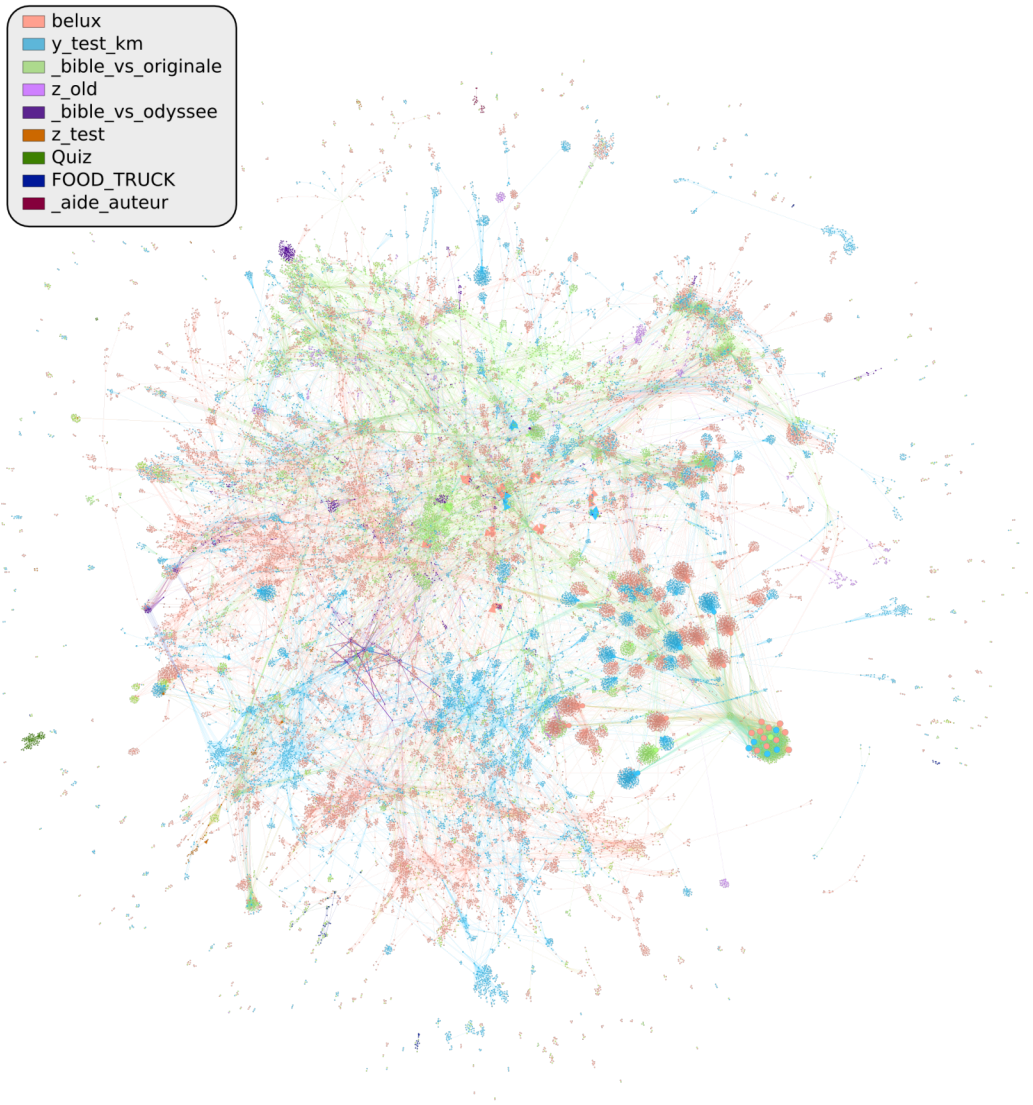


Figure 31 : graphe documentaire - Le référentiel Quick

Fiche d'identité

Nombre de sommets : 52244

Nombre de liens : 73601

Degré moyen : 1.41

Degré sortant maximum : 188

Degré entrant maximum : 375

Nombre de sommets orphelins : 13107

Nombre de fragments racines :406

Remarques

Ce graphe contient un nombre très important de fragments. Afin d'être commenté, ce nombre appelle trois remarques.

Le nombre de fragments orphelins est très élevé. À l'instar de la situation observée dans un contexte de production de cours universitaire, ces fragments ne peuvent pas être exploités en l'état (le modèle documentaire ne permet pas la publication de fragments non liés au reste du graphe). Ces fragments sont donc inutilisés et il convient de corriger le degré moyen en conséquence. Le degré moyen corrigé obtenu est d'une valeur de 1.88.

Seules deux versions du référentiel sont produites à l'aide de ce graphe, il s'agit du référentiel pour les restaurants français et d'une version complémentaire pour les restaurants belges. Ces deux documentations peuvent partager certains fragments mais une partie des fragments mobilisés dans le contexte belge doit être dupliquée pour être surchargée.

Il convient enfin de comparer ces données avec le nombre de fragments mobilisés pour la publication d'une version du référentiel. Le référentiel des normes et méthodes exploité dans les restaurants français et produit à partir du graphe visible en illustration de cette section mobilise un sous-graphe composé de moins de 3500 fragments.

Interprétations

En faisant abstraction des fragments orphelins (composés à 95% d'illustrations), le nombre de fragments présents dans ce graphe est du même ordre de grandeur que dix occurrences du sous-graphe mobilisé pour la production du référentiel. De nombreux fragments ont été dupliqués par les rédacteurs ce qui empêche toute interprétation pertinente de la topologie du graphe. Ces copies ne sont pas toujours effectuées de la même façon comme en témoigne la partie du graphe agrandie sur la figure 32.

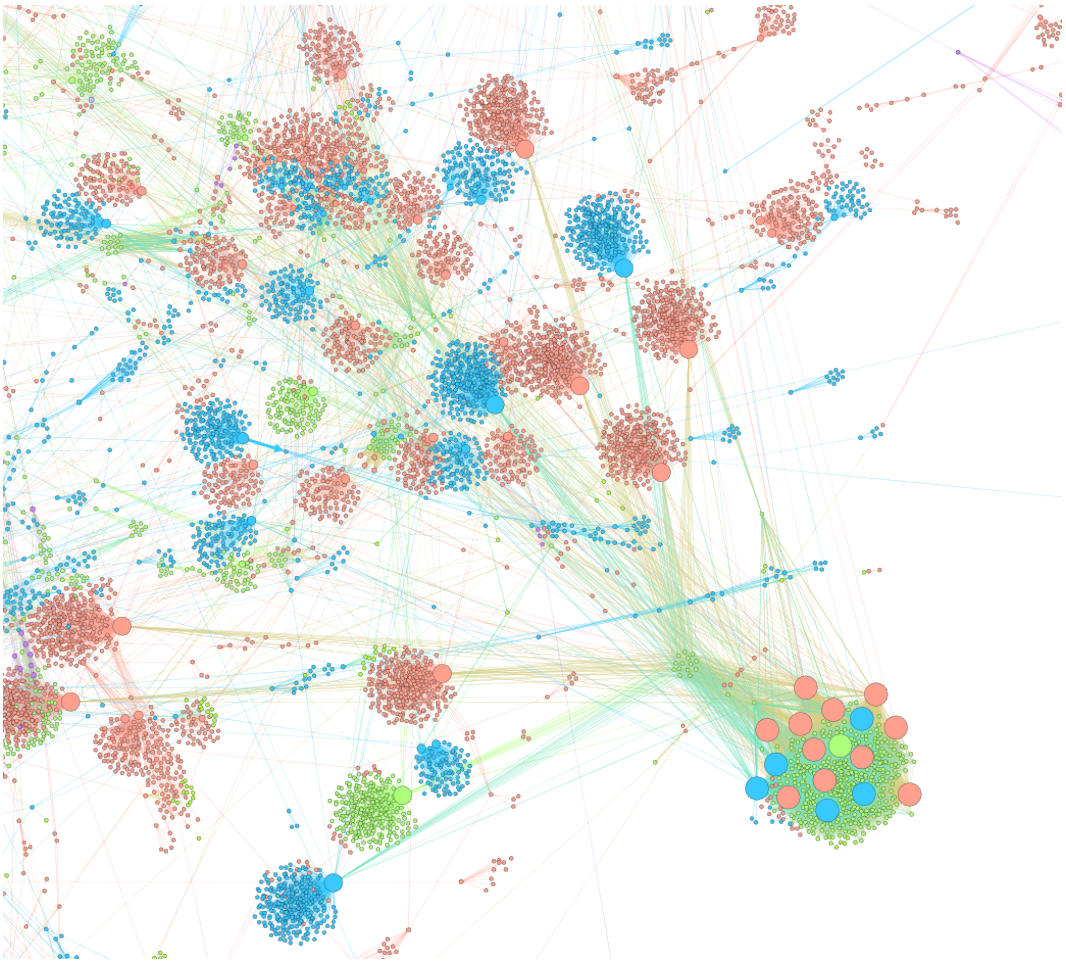


Figure 32 : section du graphe mobilisé pour produire le référentiel de Quick

Chacun des sommets de taille plus importante représente le fragment racine d'un dossier d'homologation importé depuis un autre graphe dédié à la production de ces dossiers. Le conglomérat volumineux en bas à droite est composé de 17 copies d'un même dossier d'homologation. Seuls les fragments racines ont été copiés et l'ensemble du sous-graphe est resté dans son espace d'origine (« _bible_vs_originale »). Les différentes grappes situées au dessus et à gauche de ce conglomérat représentent un autre dossier d'homologation, copié quasiment dans son intégralité. L'objectif est le même (dupliquer un dossier d'homologation) mais la technique varie puisque ici la majeure partie du sous-graphe est dupliquée. Une quinzaine de fragments mobilisés dans chacun des dossiers se retrouvent au milieu des différents conglomérats.

La légende représente les espaces de premier niveau dans lesquels sont rangés les différents fragments. On y constate que l'organisation des espaces semble peu pertinente à la vue de l'organisation métier. En effet, plusieurs espaces ne contiennent pas de nom signifiant et l'espace principalement utilisé par les équipes de Quick France est « y_test_km ».

En observant le graphe dans son ensemble, il paraît difficile d'isoler un sous-graphe document, rangé dans unique espace et complètement isolé du reste du graphe. Mis à part la structure mise en exergue ci-dessus, aucune structure particulière ne se dégage. Contrairement à l'organisation décidée par les rédacteurs (soit la centralisation du référentiel dans le dossier « y_test_km »), les contenus mobilisés pour la production du référentiel sont répartis dans de nombreux espaces au gré des copies pas toujours maîtrisées.

Problèmes

Cet exemple met en exergue plusieurs problèmes dans la manipulation de la chaîne éditoriale :

1. la mobilisation de l'arbre de gestion ne semble pas pertinente dans l'objectif d'assister la maîtrise de la rééditorialisation ;
2. les contenus sont copiés à de nombreuses reprises ;
3. la copie des contenus est réalisée selon des techniques différentes, ajoutant au sentiment d'entropie générale du graphe ;
4. il est difficile de circonscrire clairement le sous-graphe mobilisé pour la production de la documentation pour laquelle ce graphe est constitué.

Ces différents problèmes soulignent une inadéquation de l'outil pour ce contexte de production. Maîtriser un graphe documentaire est complexe lorsque le degré moyen du graphe est élevé et que le nombre de sommets est important. La copie de certains fragments plutôt que leur référencement permet alors d'isoler des fragments pour les réutiliser en sécurité (sans risque qu'un autre contexte d'utilisation ne motive l'ajout de scories). L'opération de copie n'étant pas triviale (la copie d'un fragment racine ne duplique pas l'ensemble du sous-graphe), le graphe peut alors évoluer de façon aberrante par rapport à l'objectif de production documentaire poursuivi.

Chapitre 3

Problématique

3.1 Notion de cohérence 57
3.2 Complexité du graphe documentaire 59
3.3 Artisanat et industrialisation 60
3.4 L'élaboration des documents rééditorialisés, entre complexité et cohérence 61

Notre recherche vise à accompagner la rééditorialisation dans des contextes de forte production. Le chapitre précédent nous a permis de montrer certaines difficultés dans la production de documents rééditorialisés. Concrètement, au-delà d'un certain volume de fragments, la rééditorialisation est freinée soit volontairement par les rédacteurs qui ne s'y retrouvent plus et dupliquent des contenus au lieu de maintenir des liens entre fragments, soit involontairement par des copies non voulues et des liens non maîtrisés ce qui peut donner lieu à des incohérences dans les documents publiés.

Ce chapitre s'appuie sur ces difficultés afin de formuler notre problématique de recherche. Dans les trois premières sections, nous définissons les termes de cohérence éditoriale, de complexité du graphe documentaire et d'industrialisation. La dernière section définit la problématique en agençant ces différentes notions.

3.1 Notion de cohérence

Définition

Le terme *cohérence* est défini dans le *Trésor de la Langue Française informatisé* (TLFI) comme suit : « harmonie, rapport logique, absence de contradiction dans l'enchaînement des parties de ce tout ». Rapporté à notre objet, la cohérence désigne le bon enchaînement et l'absence de contradictions dans le contenu d'un document. On parlera de cohérence éditoriale.

Production documentaire et contrôle de la cohérence

En simulant un ordre documentaire classique c'est-à-dire en approchant le rendu d'un document à l'écran tel qu'il pourrait être une fois imprimé sur papier, les suites bureautiques n'ont que peu d'impact sur le contrôle de la cohérence éditoriale.

La rééditorialisation documentaire s'appuie sur le découpage d'un document en fragments. En isolant un fragment du contexte dans lequel son contenu est publié, les chaînes éditoriales ont un impact négatif sur le contrôle de la cohérence.

Par exemple, un fragment peut contenir des références à des contenus qui le précédent ou le suivent. L'évolution du contexte dans lequel ce fragment est publié peut réordonnancer l'enchaînement des parties et casser la cohérence du document.

Cet impact est en partie résorbé par les modèles documentaires. En définissant des classes de fragments à instancier et les possibles associations entre classes, le modèle permet d'assister les rédacteurs dans le maintien de la cohérence éditoriale.

Par exemple, le modèle de document peut définir des classes de fragments ayant une relative autonomie quant à la cohérence éditoriale. Le réordonnancement des fragments ne pourra donc pas introduire d'incohérences dans le document. Ainsi, le modèle Quick définit trois catégories de classes de fragments (scénarisations, fiches, ressources). Les fiches constituent des entités relativement autonomes qui n'ont pas besoin de s'appuyer sur les fragments précédents ou suivants pour assurer la cohérence du document. Les fragments de scénarisation peuvent donc ordonnancer indifféremment les fiches sans que cela ne produise d'incohérence éditoriale.

Cohérence du graphe

Par extension, nous parlerons dans ce mémoire de cohérence du graphe. Nous considérons qu'un graphe est cohérent si l'ensemble des documents pour lequel il est constitué sont eux-mêmes cohérents. Le principe de la rééditorialisation, soit le fait de mobiliser un fragment existant pour la publication de nouveaux documents, est à l'origine de nouvelles difficultés dans le contrôle de la cohérence du graphe.

Par exemple, prenons le graphe documentaire utilisé pour la publication du site service-public.fr. Imaginons qu'un même exemple (comme, une famille type constituée d'un couple marié ayant trois enfants) soit mobilisé dans deux contextes différents (comme pour le calcul des allocations familiales et celui de la prime à la naissance). Les trois fragments obtenus sont illustrés sur la figure 33.

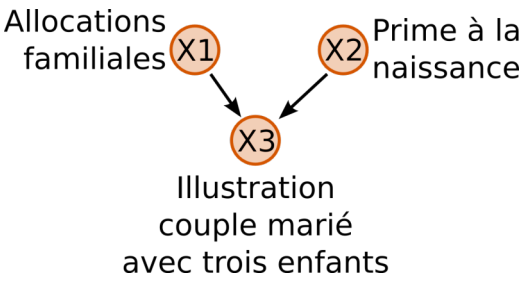


Figure 33, deux fragments mobilisant un même exemple

Imaginons que, dans un objectif de réduction du déficit budgétaire, le projet de loi de finance 2015 intègre des conditions de ressources dans l'accès aux allocations familiales. Les rédacteurs de la DILA ont pour mission de maintenir le site service-public.fr soit dans le cas présent, de mettre à jour les fragments concernés après l'adoption et la promulgation de la loi. Ainsi, le fragment X1 doit être modifié pour contenir les nouvelles réglementions et le fragment X3 doit être mis à jour pour ajuster les ressources de la famille prise en exemple (sur la figure 34, les fragments modifiés sont représentés en vert).

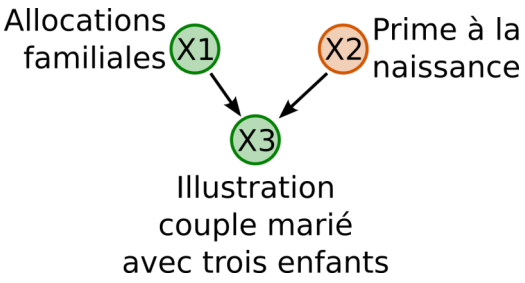


Figure 34 : erreur dans la cohérence suite à la mise à jour des fragments

En ajustant les ressources de la famille type prise en exemple, la modification peut rendre invalide la mobilisation de X3 par X2 car les seuils de ressources de la prime à la

naissance ne sont (très probablement) pas identiques à ceux des allocations familiales. L'incohérence éditoriale n'aura pas été apportée par une erreur dans le raisonnement du rédacteur, mais par une erreur de manipulation du graphe documentaire en rédigeant une modification d'un fragment contrôlée selon un seul de ses différents contextes de publication. Au cours d'un mois de maintenance, les rédacteurs de la DILA peuvent effectuer plusieurs centaines d'opérations de maintenance du graphe qui sont autant de sources d'incohérences éditoriales possibles.

3.2 Complexité du graphe documentaire

Notion de complexité

Le sens exact du terme *complexité* varie en fonction des disciplines dans lesquelles il est utilisé. Plutôt que la définition de la complexité algorithmique classiquement utilisée en informatique (soit le classement d'un problème selon la fonction permettant d'estimer le temps de traitement ou l'espace maximal utilisé par l'algorithme qui le résout, exprimé O(n), O(log(n)), $O(n^2)$, etc.), nous nous référons dans ce mémoire à la caractérisation plus générique des systèmes complexes.

Un système est considéré de plus en plus complexe lorsqu'il contient de plus en plus de composantes qui réagissent indépendamment les unes des autres tout un conservant un impact sur le comportement global du système.

Complexité du graphe documentaire

Le graphe documentaire est un ensemble de fragments liés les uns avec les autres. Les fragments entretiennent entre eux des relations d'interdépendance selon les fonctions de rééditorialisation qui les lient, soit selon les arcs du graphe.

Le terme interdépendance implique que la modification d'un fragment peut avoir un impact sur les fragments voisins. Par exemple, en modifiant le fragment d'introduction d'un document, il peut être nécessaire de modifier le fragment de conclusion du même document. Cette conclusion peut elle même être la source d'une dérivation qu'il conviendra également de mettre à jour.

La modification d'un fragment peut ainsi avoir un impact sur *n* autres fragments qui lui sont directement liés. Chaque fragment modifié peut à nouveau entraîner *n* autres opérations de maintenance et ainsi de suite. Les rédacteurs se retrouvent ainsi confrontés à un graphe où chaque modification peut entraîner plusieurs dizaines de contrôles et d'opérations de maintenance en suivant les liens du graphe.

Facteurs de complexité

Deux facteurs sont sources de complexité au sein du graphe documentaire :

- le nombre de fragments ;
- la densité du graphe.

Un graphe très volumineux et peu dense est peu complexe car la modification d'un fragment aura peu de répercussions sur le graphe. Au même titre, un graphe dense mais peu volumineux est peu complexe car le nombre global de fragments à maintenir facilite les contrôles.

Perception du graphe

Si ce n'est dans des visualisations comme celles proposées dans le chapitre précédent, le graphe documentaire n'est jamais manipulé ou visualisé dans sa globalité. Il est systématiquement réduit à un sous-graphe sélectionné selon un critère donné. Nous distinguons trois critères pour la réduction du graphe à un sous-ensemble de fragments plus intelligible.

- Selon un critère éditorial : cela revient à isoler l'ensemble des fragments mobilisés pour la production d'un document. C'est ce que nous avons appelé en chapitre 2 les sous-graphes documents. Il est à noter qu'un même fragment peut faire partie de nombreux

sous-graphes documents au sein d'un même graphe.

- Selon un critère de gestion : cela revient à isoler l'ensemble des fragments rangés par le rédacteur au sein d'un même espace de l'arbre de gestion.
- Selon un critère généalogique : cela revient, à partir d'un fragment, à isoler l'ensemble des fragments dont il dérive ou qui en sont des dérivés, directement ou indirectement.

En définitive, non seulement le graphe documentaire est un objet complexe où la modification d'un fragment peut entraîner plusieurs dizaines d'opérations de maintenance mais ces opérations peuvent intervenir sur des fragments masqués dans la vue partielle du graphe exploitée par un rédacteur.

3.3 Artisanat et industrialisation

Définitions

Bachimont définit les concepts d'ingénierie artisanale et d'ingénierie industrielle comme suit.

« L'ingénierie artisanale où le monde est pensé sur le modèle de l'atelier. L'artisan réinvente à chaque fois l'usage de ses outils et improvise pour s'adapter à la nature singulière des matériaux qu'il utilise (les nœuds dans le bois) et la commande particulière qui lui est faite (un meuble unique) » (Bachimont, 2007, p. 12).

« L'ingénierie industrielle où le monde est pensé sur le modèle de l'usine, qui n'est que le passage à l'échelle du laboratoire. Dans ce contexte, le monde se présente de manière cadrée et enrégimentée dans le système de production : les matériaux ne sont plus bruts, mais préfaçonnés en vue de la production. On passe de l'improvisation artisanale à la répétabilité et la standardisation industrielles » (ibid., p. 12).

Approches artisanale et industrielle

Nous avons donc d'une part une **approche artisanale** où chaque action est pensée dans un **cadre unique** et où l'opérateur est contraint de **s'adapter** aux particularités de son objet. D'autre part, une approche industrielle qui se démarque par une **uniformisation** des objets traités et par une **répétabilité** des actions à entreprendre.

Approche industrielle et rééditorialisation documentaire

La notion d'approche industrielle a accompagné le développement des chaînes éditoriales (Bachimont, Cailleau, Crozat, Majada & Spinelli, 2002). L'enjeu était alors de montrer qu'en combinant la séparation d'un fonds documentaire et de ses formes à un principe de rééditorialisation documentaire, les chaînes éditoriales sont des outils permettant de passer d'une production documentaire artisanale à une production industrielle.

Cette opposition entre artisanal et industriel nous semble également pertinente afin de qualifier l'usage de la rééditorialisation au sein d'un contexte de production. La rééditorialisation peut être considérée comme artisanale lorsque la mutualisation de fragments entre documents ou la dérivation de fragments existants se fait à la marge de la production documentaire. Le graphe constitué peut alors être volumineux mais sa complexité est réduite puisque l'interdépendance entre les fragments reste maîtrisable par les rédacteurs. Par exemple, le graphe exploité par Enaction Series, voire l'exemple de graphe mobilisé dans un contexte pédagogique, représentent des contextes de rééditorialisation artisanaux. À l'inverse, on pourra parler de rééditorialisation à l'échelle industrielle pour désigner un graphe dans lequel la mutualisation des fragments est la règle de production et le contrôle de la cohérence sur des fragments interdépendants entre eux est une étape systématique. Les graphes exploités par Quick et par la DILA en sont de bons exemples.

3.4 L'élaboration des documents rééditorialisés, entre complexité et cohérence

Hypothèse

L'élaboration des documents rééditorialisés conduit à des difficultés dans le maintien de la cohérence éditoriale. Ces difficultés sont induites par l'outil. Pour les résoudre, les rédacteurs doivent manipuler et contrôler un objet complexe : le graphe documentaire. Cette situation est soutenable lorsque la complexité du graphe est faible : soit avec un volume de fragments et une densité de liens qui restent dans des proportions artisanales.

Tension

L'écriture des documents rééditorialisés se retrouve prise en tension entre d'un côté la fragmentation des contenus pour améliorer l'efficacité de la production et la maintenance des documents et de l'autre la nécessité de garantir la cohérence éditoriale du graphe.

Sans éléments d'assistance adéquats, le seul moyen de simplifier le contrôle de la cohérence consiste à supprimer des fragments mutualisés en dupliquant leurs contenus en lieu et place de leurs référencements. Cette pratique de la copie va à l'encontre de l'objet de la rééditorialisation.

Impact

Cette tension limite l'usage de la rééditorialisation et l'enferme dans une pratique artisanale où le passage à l'échelle est contraint par la complexité de l'objet manipulé - le graphe - et la nécessité de maîtriser la cohérence éditoriale des documents produits.

Objet de notre recherche

Notre recherche s'intéresse à cette tension en proposant différents éléments permettant d'assister les rédacteurs dans le contrôle de la cohérence du graphe. L'enjeu est d'ouvrir une approche industrielle dans l'élaboration et le contrôle des documents rééditorialisés. Nos propositions se structurent en deux parties : une première dédiée à la documentarisation et la régulation des actions menées sur le graphe et une seconde dédiée à la structuration du graphe en plusieurs ateliers afin de réduire le nombre de fragments perçus et d'assister les rédacteurs dans la rééditorialisation de volumes de fragments importants.

L'objet final poursuivi est de faciliter une production documentaire que Crozat (2012) définit comme la façon dont nous pourrions écrire avec le numérique (« As we may write »). C'est-à-dire une production documentaire où chaque action de copie de contenus par un rédacteur est remplacée et contrôlée par une fonction de rééditorialisation.

Partie II
État de l'art

Introduction

Les deux premiers chapitres de ce mémoire nous ont permis d'ancrer notre recherche. Dans cette partie, nous menons une analyse des solutions mobilisées usuellement par les chaînes éditoriales pour accompagner les rédacteurs dans le contrôle de la cohérence du graphe documentaire. Les deux premiers chapitres sont dédiés aux deux paradigmes usuellement mobilisés : la gestion de code source et les systèmes de travail coopératif assisté par ordinateur. Le troisième mène un état de l'art technologique des chaînes éditoriales numériques.

Chapitre 4

Assistance au développement informatique

4.1 Commentaires 66
4.2 Gestion des versions 67
4.2.1 Systèmes centralisés 67
4.2.2 Systèmes distribués 68

Les chaînes éditoriales ont été conçues dans l'objectif de tirer partie du support numérique. Elles s'appuient sur une propriété fondamentale du numérique à savoir la manipulabilité de l'information afin d'optimiser le processus d'élaboration des documents. Ce faisant, elles rapprochent le procédé de l'élaboration des documents et celui du développement informatique.

Prenons pour l'exemple le procédé de développement d'un langage compilé comme le C ou le C++ : le code source est composé principalement de fichiers de sources, de fichiers d'en-têtes et de fichiers de bibliothèques déjà pré-compilées. Un fichier peut en référencer un autre afin de pouvoir appeler des fonctions qui y sont définies. Le code source d'une application peut ainsi réunir plusieurs milliers voire dizaines de milliers ou centaines de milliers de fichiers distincts (à titre d'exemple, l'ordre de grandeur du nombre de fichiers de sources mobilisés pour compiler les applications Scenari est entre quarante et cinquante mille). Pour compiler un programme, le compilateur prend les différents fichiers de code source et produit en sortie, un fichier binaire d'instructions au processeur. Chaque bloc de code source a été transformé par l'algorithme de compilation en un jeu d'instructions interprétables et exécutables pour le processeur de la machine.

Ainsi, au même titre qu'un ensemble de fragments de documents peut être représenté sous la forme d'un graphe, on pourra constituer un graphe de code source. Chaque sommet y représente un fichier de source et chaque lien une référence depuis un fichier de code vers un autre. Ce type de représentation du code source est par ailleurs mobilisé pour des études bien précises comme celle de Zanetti & Schweitzer (2012) sur la modularité du code source ou celle de Kabbedijk & Jansen (2011) sur les écosystèmes *open source*.

Les solutions que nous présentons dans ce chapitre ont été exploitées pour le développement de plusieurs chaînes éditoriales, notamment en raison de la proximité du problème de cohérence globale d'un graphe de ressources. Néanmoins, la gestion du code source comporte des différences fondamentales avec celle des fragments de documents. En premier lieu, un code peut être exécuté par la machine. En opérant les instructions telles qu'elles sont écrites dans les fichiers de sources, la machine opère un vaste contrôle de cohérence sur la validité des algorithmes. Une erreur importante dans l'écriture générera une erreur dans la

compilation ou l'exécution. Lorsque ce contrôle montre ses limites, les développeurs peuvent mettre au point des contrôles complémentaires appelés *test unitaires* (Zhu, Hall & May, 1997). Ces tests valident l'exécution d'une portion de code bien ciblée. Enfin pour clore ce tableau des différences, notons que le code informatique est écrit par des développeurs qui sont formés pour maîtriser les aspects techniques du contrôle de leur code. Cette différence est notable avec un projet de diffusion de la rééditorialisation au plus grand nombre, soit à toute personne souhaitant produire des documents.

L'assistance aux développeurs s'opère selon deux modalités différentes. Au sein des fichiers de code source avec l'usage de commentaires et d'annotations ou de façon externe à ces fichiers avec la gestion de quantités importantes de fichiers et de leurs versions respectives.

4.1 Commentaires

Principe

Chaque langage de programmation propose un moyen d'insérer des commentaires au sein du code. Il s'agit de caractères spéciaux reconnus par le programme qui traite le code (compilateur ou interpréteur) et qui agissent comme des bornes d'exclusion. L'ensemble des caractères contenu entre la chaîne de caractères de départ et la chaînes de caractères de fin sont ignorés à la compilation ou l'interprétation.

Par exemple, au sein des langages C ou C++, deux méthodes coexistent pour assurer l'insertion de commentaires :

- la chaîne « // » pour débuter le commentaire et le retour chariot pour le terminer ;
- la chaîne « /* » pour débuter le commentaire et la chaîne « */ » pour le terminer.

Pour l'exemple, on pourra consulter un fichier de sources en accès libre tel que ceux du système d'exploitation Linux disponible sur la plateforme github *(https://github.com/torvalds/linux)* comme le fichier cpu_rmap.c *(https://github.com/torvalds/linux/blob/master/lib/cpu_rmap.c)*. On constate ainsi que le fichier contient des informations en en-tête (licence, objet du fichier de sources) et que chaque fonction contient son propre commentaire.

Exploitation

Le premier usage des commentaires au sein du code source consiste à documenter le fonctionnement d'un fragment de code pour permettre à d'autres développeurs de bien comprendre son usage et faciliter ainsi sa maintenance et ses évolutions.

Certains environnements de développement proposent d'exploiter la logique des commentaires pour assurer d'autres fonctions d'assistance aux développeurs dans la maintenance de leur code source. Par exemple, le logiciel Eclipse *(http://eclipse.org/)* propose un système de tâches s'appuyant sur les commentaires. En ajoutant les commentaires « TODO », « FIXME » ou « XXX », un développeur ajoute une tâche dont la description est définie à la suite du mot-clé.

Un autre usage courant des commentaires est leur exploitation pour la production de documentations externes au code. La plupart des langages intègrent ainsi une logique de commentaires particulière ; des logiciels externes d'analyse de code source prennent le relais lorsque ce n'est pas le cas (comme Doxygen *(http://doxygen.org)* pour les langages C et C++). Des chaînes de caractères spécifiques sont insérées dans les commentaires et utilisées par l'algorithme de génération de la documentation pour déterminer quels commentaires correspondent à quels fragments de la documentation. La sortie du processus génère un site HTML ou un document PDF reprenant la description de l'ensemble des fonctions.

Étude sur l'usage

Ying, Wrigth & Abrams (2005) ont mené une étude sur l'usage des commentaires, et plus particulièrement du système de tâches d'Eclipse, au sein d'un dépôt de code source interne à la société IBM. Ils ont ainsi établi toute une typologie d'usage (communication entre équipes, indicateur de portion à changer dans une tâche plus importante, mémorisation de tâches passées, tâches en cours, tâches futures) et répertorié manuellement les différents commentaires au sein de ces catégories. Ying et *al* constatent ainsi que le système de commentaires en général, et celui

de tâches en particulier offre un cadre permettant à la fois de communiquer entre équipes de développement et plus généralement, facilite la maintenance du code source en permettant l'inscription d'indications de maintenance au sein même du code.

4.2 Gestion des versions

La gestion des versions est un terme désignant toute une catégorie d'outils utilisés pour simplifier à la fois la mutualisation de code source par des équipes de développements distribuées et la gestion de l'historique ou de versions dérivées des fichiers de sources.

Les systèmes de gestion de versions s'appuient sur un principe d'entrepôt de code source accessible sur le réseau et de copies locales. Un entrepôt définit un répertoire principal (le *trunk*) et des dérivations (des *branches*). Lorsqu'un développeur veut participer à un projet, il télécharge et stocke des copies locales des fichiers de l'entrepôt. Une fois modifiés, ces fichiers sont reversés à l'entrepôt. Leurs mise à jour peut être accompagnée d'une description s'inscrivant dans un journal de suivi des versions (système de *log*).

Les systèmes de gestion de version cherchent à encadrer le mieux possible les conflits entre versions (modifications concurrentes de deux développeurs) en proposant le plus souvent un système de marquage des version locales conflictuelles et en mobilisant des algorithmes de différentiels pour assister un développeur dans la résolution de son conflit local. Il existe deux paradigmes d'outils : les systèmes centralisés et les systèmes distribués.

4.2.1 Systèmes centralisés

Principe

Dans un système de gestion de versions centralisé, seule une application serveur est déployée. Les copies locales réalisées par les développeurs en sont directement issues et toute mise à jour réintègre l'entrepôt central. Les systèmes centralisés les plus connus et utilisés sont Concurrent Version System *(https://savannah.nongnu.org/projects/cvs)* (CSV) et Subversion *(http ://subversion.apache.org/)* (SVN).

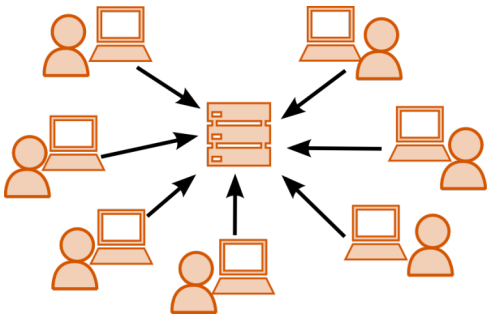


Figure 35 : système de gestion de versions centralisé

En fonctionnant ainsi directement entre copie locale et serveur, un développeur peut travailler facilement sur un sous-ensemble de l'entrepôt : les fichiers locaux peuvent être récupérés à partir de n'importe quel dossier de l'arborescence du serveur.

Limites

Les entrepôts de nombreux logiciels ont migré de systèmes de gestion de versions centralisés vers des systèmes distribués. Déjà en 2009, De Alwis et Sillito constatent ce mouvement de migration. Ils mènent ainsi une analyse des faiblesses des systèmes centralisés et anticipent les plus-values de leurs pendants distribués (De Alwis & Sillito, 2009).

- Par la nature des informations qui sont enregistrées à chaque mise à jour de fichiers sur l'entrepôt, les systèmes centralisés gèrent souvent très mal la fusion d'une branche avec une autre.
- Lorsque des développements sont expérimentaux, il est d'usage de les isoler du dépôt principal pour éviter de rendre temporairement les sources non compilables ou non

interprétables. En raisons des difficultés dans la fusion de branches, il est courant que les organisations responsables du développement n'ouvrent pas de branches pour cet usage, rendant ainsi délicate la gestion des droits d'écriture sur le dépôt principal. Sans droit d'écriture, un développeur ne peut pas reverser ses modifications sur l'entrepôt, avec ce droit, des modifications contenant des erreurs peuvent endommager la cohérence du code et nuire à la productivité des autres développeurs.

- La centralisation du dépôt principal crée une faiblesse sur le système qui doit être sauvegardé en temps réel.
- Les développeurs doivent nécessairement être connectés au réseau du serveur pour pouvoir mettre à jour des contenus.

4.2.2 Systèmes distribués

Principe

Dans les systèmes distribués, le téléchargement ou la mise à jour des sources ne peuvent pas être réalisés sur un serveur distant. Pour travailler sur un entrepôt, un développeur doit au préalable le dupliquer intégralement sur sa machine. Le téléchargement de copies de travail et la mise à jour se font ainsi directement sur un même poste de travail.

Le mécanisme utilisé pour la duplication d'un entrepôt est en réalité la création d'une branche. Chaque développeur travaille sur sa propre branche, sans nécessité de mettre en place un entrepôt principal. Pour envoyer du contenu d'une branche à l'autre, les systèmes distribués s'appuient sur une comparaison de l'historique des mises à jour au lieu d'effectuer un comparatif des arborescences.

Les systèmes distribués les plus connus et utilisés sont Git *(http://www.git-scm.com/)*, Bazaar *(http://bazaar.canonical.com/en/)* et Mercurial *(http://mercurial.selenic.com/)*.

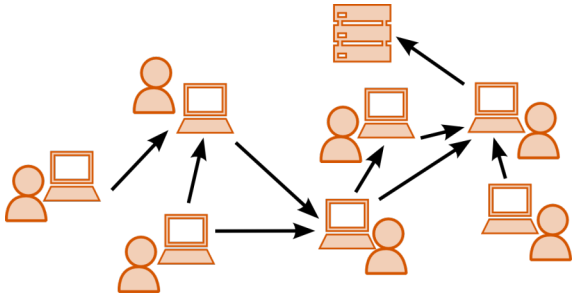


Figure 36 : système de gestion de versions distribué

Les systèmes de gestion de versions distribués lèvent les principales limites vues dans la section précédente.

- Le système de fusion de versions a été revu pour être massivement utilisé. Il exploite l'historique des modifications en plus du contenu des fichiers.
- Chaque développeur dispose de son propre serveur de gestion de versions et peut travailler directement dessus. Lorsque des développements sont prêts, il envoie une demande de mise à jour à un autre serveur (version de production, version d'un autre utilisateur en charge des tests, etc.)
- Le serveur de sources est dupliqué sur le poste de chaque utilisateur. Si une machine tombe en panne, seules les modifications locales, propre à la branche hébergée par cette machine seront perdues.
- Les développeurs disposent d'un serveur de gestion de sources sur leur propre machine. Ils peuvent donc travailler localement et sans accès au réseau.

Limite

Une pratique courante dans l'usage des systèmes de gestion de sources centralisés consiste à importer, au sein d'un entrepôt, un sous-ensemble de sources issues d'un serveur tiers. En imposant la copie complète d'un entrepôt pour y travailler dans une branche séparée, les systèmes décentralisés ne sont pas conçus pour ce type d'usage.

Chapitre 5

Travail coopératif assisté par ordinateur

5.1 Le pragmatisme ou l'action du discours 71
5.2 La perspective Langage/Action 73
5.3 Les documents pour l'action 73

Le travail coopératif assisté par ordinateur (TCAO) (en anglais, *Computer Supported Cooperative Work (CSCW)*) est une vaste discipline de recherche qui s'intéresse à la mobilisation de technologies informatiques pour assister les interactions au sein de collectifs. À titre d'exemple de la diversité des travaux menés au sein des TCAO, on pourra observer les différentes sessions de la conférence CSCW 2014 (Fussel & Lutters, 2014). On y retrouve des sessions dédiées aux média sociaux, aux communautés en ligne, au travail à distance, aux travaux participatifs et distribués, à l'usage des technologies collaboratives en milieu familial, professionnel ou hospitalier, au développement collaboratif de logiciel, aux algorithmes permettant le développement d'outils collaboratifs, aux formations en ligne ouvertes à tous ou encore, aux usages des applications mobiles collaboratives.

Travailler à plusieurs nécessite une répartition du travail, altérant ainsi la vue globale qu'un membre du collectif peut avoir sur la production. Le domaine des TCAO s'est très tôt retrouvé confronté à la problématique de la structuration de la collaboration afin de maintenir une organisation commune, comprise de tous les participants. Il s'agit de procédés permettant d'expliciter toutes actions faites sur une production commune. En formalisant ainsi l'action sur la production, le domaine de recherche des TCAO a produit des outils largement réutilisés, notamment au sein de certaines chaînes éditoriales afin d'assister les rédacteurs dans la maintenance de leurs graphes documentaires.

5.1 Le pragmatisme ou l'action du discours

Austin et le discours performatif

Austin est un philosophe Anglais de la première moitié du XXème siècle. Convaincu qu'une analyse ne pouvait démarrer par une simple énonciation de faits, il a développé toute une théorie d'analyse du langage ordinaire qu'il a nommé « Phénoménologie Linguistique » (Austin, 1979, p. 130). À sa mort en 1960, ses articles et conférences ont été publiés sous la forme de trois ouvrages (ibid. ; Austin, 1962a ; Austin, 1962b/1991). Dans *How to do things with words*, Austin introduit la notion d'énonciation performative : une énonciation ne pouvant être considérée comme juste ou fausse et exécutant une action par sa simple locution. Il cite ainsi par exemple, l'énonciation « Oui, [je le veux] (c'est-à-dire je prends cette femme comme épouse

légitime) » ou « Je vous parie six pence qu'il pleuvra demain. » (Austin, 1991, p. 41).

Dans ses dernières conférences (ibid., pp. 109-162), Austin introduit la notion d'*acte de langage*. N'arrivant pas à établir un moyen efficace de déterminer si une énonciation est réellement performative ou affirmative, il propose une étude plus approfondie des éléments constitutifs d'un discours performatif. Il différencie ainsi l'acte de locution (le simple fait de dire quelque chose, par exemple : « tire sur elle » (ibid., p. 114)), l'acte d'illocution (la façon de prononcer des paroles, par exemple : « il me pressa (ou me conseilla, ou m'ordonna, etc.) de tirer sur elle » (ibid., p. 114)) et l'acte de perlocution (action réalisée en parlant, par exemple : « Il parvint à me faire (ou me fit, etc.) taire »). De son propre aveu, dans sa douzième et dernière conférence sur le sujet (ibid., pp. 151-162), Austin n'est toujours pas satisfait de la formulation finale des catégories d'actes.

Searle et les actes de langage

Searle reprend les travaux d'Austin et s'intéresse principalement aux actes illocutionnaires (Searle, 1969). Il propose ainsi quatre actes de langage élémentaires : les actes assertifs pour une description du monde, les actes directifs pour une demande de modification du monde (par exemple, une demande ou un ordre), les actes promissifs pour une proposition de changer le monde (par exemple, une promesse ou une offre), les actes déclaratifs qui changent le monde par leur énonciation (par exemple, lorsqu'un maire déclare un couple marié) et les actes expressifs qui expriment un état psychologique et n'ont aucun impact sur le monde (par exemple, des excuses ou une prière).

Les conversations pour l'action

Winograd et Flores (Winograd & Flores, 1986 ; Winograd, 1986) ont montré que les actes de langage n'étaient pas simplement utilisés de façon décorrélée les uns des autres pour énoncer une action unitaire pour chaque acte. Il est cependant fréquent qu'ils soient ajustés entre eux au sein d'un discours ou d'une conversation. Cet exemple de la conversation est structurant chez Winograd, qui le représente par le schéma représenté sur la figure 37.

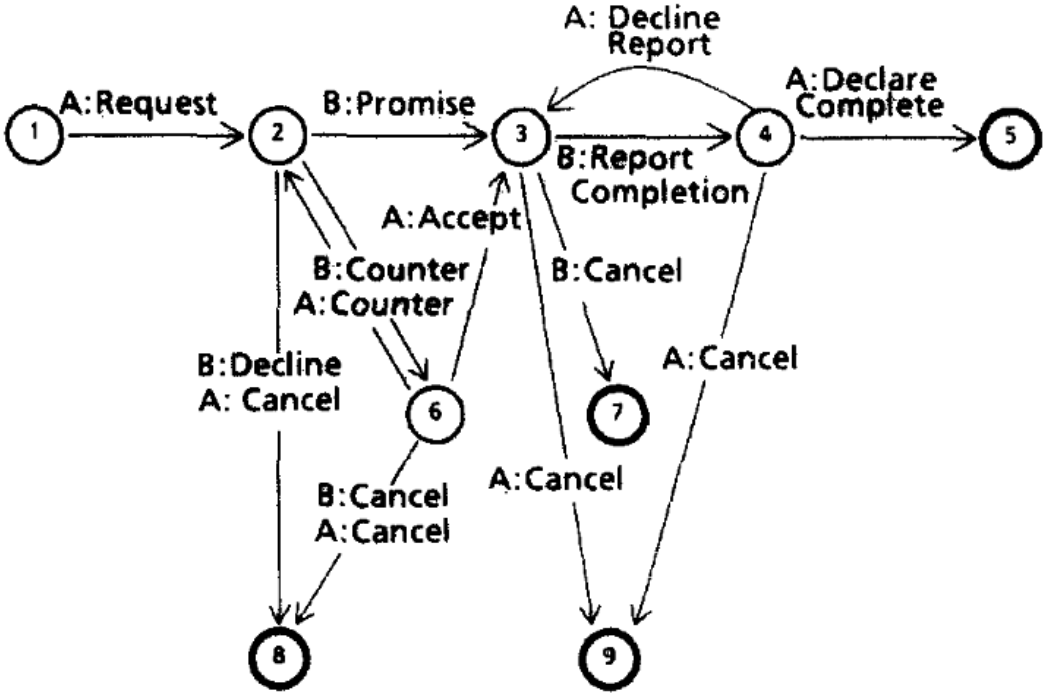


Figure 37 : Conversation pour l'Action (Winograd, 1986)

Le schéma montre une conversation qui se déroule en parallèle de la réalisation d'une action. La lecture du schéma commence par l'état 1 dans lequel le locuteur A demande quelque chose au locuteur B. Chaque état représente une étape dans la discussion. Le passage d'une étape à l'autre se fait par l'énonciation d'un acte de langage. À la fin de la conversation, l'action parallèle a soit été accomplie (état 5) soit été annulée (état 6, 7, 8 et 9).

5.2 La perspective Langage/Action

Principe

Winograd et Flores (1986) proposent de prendre en compte les conversations pour l'action en usage dans les collectifs afin de concevoir leur système d'information. Winograd liste ainsi les différentes perspectives existantes pour la conception de systèmes d'information : l'implémentation, les contraintes entre un système et son environnement, le processus de traitement de l'information, les rôles, espaces et ressources, les responsabilités des utilisateurs, les conflits ou encore les relations entre usagers (Winograd, 1986, pp. 25-27). Chacune de ces perspectives donne à penser et concevoir certains aspects du système.

Winograd et Flores proposent d'ajouter une nouvelle perspective : la perspective « Langage/Action ». Cette perspective de développement a été largement reprise, implémentée, critiquée ou plébiscitée dans le domaine de recherche des TCAO (De Michelis, 1994). Elle a notamment donné naissance aux systèmes de pilotage de l'activité par les procédures et les systèmes de *workflow* et est à l'œuvre dans la plupart des systèmes de gestion documentaire (Dumas, Van Der Aalst & Ter Hofstede, 2005).

Exemple

Au sein des chaînes éditoriales, la perspective Langage/Action peut être mobilisée pour concevoir la façon dont les différents usagers interagissent à travers le logiciel. Par exemple, au sein de la société Quick, les dossiers d'homologation sont rédigés par un rédacteur et validés par un expert métier. On peut étendre cette procédure comme illustré sur la figure 38 en supposant qu'un chef d'équipe répartit la rédaction des différents dossiers aux rédacteurs.

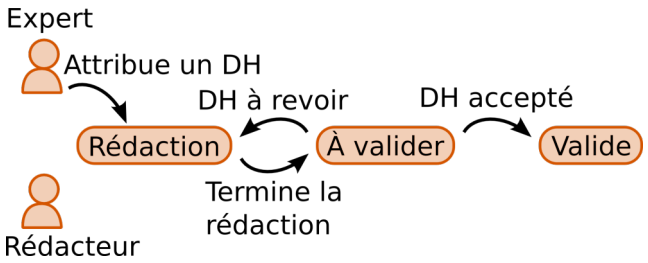


Figure 38 : procédure de validation d'un DH chez Quick

Dénombrabilité de l'activité

Dourish (2001) prend l'exemple d'une société d'impression ayant adopté un système de gestion de l'activité par les procédures. Lorsqu'un client fait une commande, une procédure est ajoutée au système d'information. Lorsqu'un employé effectue la prestation commandée, la procédure est marquée comme réalisée, puis livrée, puis encaissée. Dourish constate que malgré de nombreuses critiques (réduction de la diversité des tâches à une routine répétable (Suchman, 1985 ; Button & Harper, 1993), critique de la mobilisation des actes de langage et du principe de conversation pour l'action (Button, 1995), critique du pouvoir accordé aux concepteurs de systèmes de *workflow*, produisant souvent des systèmes non fonctionnels (Suchman, 1993), les systèmes de pilotage de l'activité restent massivement utilisés. Il conclut que certains avantages des systèmes de gestion par la procédure n'ont pas encore été théorisés. Son hypothèse réside dans le caractère dénombrable de l'activité : en instanciant des objets informatiques pour chaque action planifiée ou réalisée, les systèmes de gestion par la procédure transforment une activité en liste d'actions dénombrables, simplifiant ainsi considérablement la gestion de ladite activité.

5.3 Les documents pour l'action

Plus récemment, Zacklad s'est intéressé aux façons dont les collectifs organisaient leurs productions collaboratives (Zacklad, 2005 ; Zacklad, 2007). Il a mis en exergue l'usage de documents particuliers qu'il nomme des Documents Pour l'Action (DopA), dont l'objet est d'assister le collectif dans sa production. Il s'agit d'une autre exploitation des conversations pour

l'action de Winograd. Au lieu d'être implémentées dans un système d'information, elles sont inscrites dans des documents.

Zacklad définit les documents pour l'action par leur inachèvement prolongé (« ils possèdent un statut d'inachèvement prolongé pendant la phase active de la production sémiotique collective, phase durant laquelle nous les appellerons des DopA en évolution (vs DopA stabilisés) » (2005, p. 20)), leur pérennité, leur fragmentation (« au moins pendant leur phase évolutive ils articulent plusieurs fragments plus ou moins liés sémantiquement (notamment des annotations) qui ne peuvent être intégrés mécaniquement ou de manière organique à l'ensemble du document ; » (ibid., p. 20)), leur rapport aux auteurs complexes (« les différentes parties du DopA sont le plus souvent produites par différents réalisateurs (plurivocité, pluri textualité...) » (ibid., p. 20)) et leur rapport argumentatif non trivial aux autres parties du document (« chaque partie possède une relation potentiellement complexe aux autres - modalisation, incertitude, relation argumentative vis-à-vis des autres fragments. » (ibid., p. 20).

Exemples

Par exemples, les documents usuellement mobilisés pour organiser une action collective, comme l'ordre du jour d'une réunion, son compte rendu ou encore une répartition des tâches à réaliser, sont des documents pour l'action classiques. Zacklad va plus loin en incluant l'ensemble des documents mobilisés par un collectif distribué pour l'assister dans sa communication. On retrouve ainsi la figure du forum, des mails ou encore celui des blogs.

Documents pour l'action et documentarisation de l'activité

En produisant une activité documentaire de soutien de la production collective, l'usage des documents pour l'action s'inscrit dans un mouvement de documentarisation de l'activité dans le sens où il produit une documentation de l'ensemble des actions réalisées par le collectif.

Chapitre 6

Analyse technologique des chaînes éditoriales

6.1 Rédaction Technique 75
6.1.1 AuthorIT 76
6.1.2 PTC ArborText 76
6.1.3 MadCap 76
6.1.4 FrameMaker et RoboHelp 77
6.1.5 Calenco 77
6.1.6 Autres chaînes éditoriales 77
6.2 Contenus pédagogiques 78
6.3 Synthèse 79

Outre des projets expérimentaux ou isolés à des contextes d'usages très précis, il existe deux grandes familles de chaînes éditoriales sur le marché du document structuré : les chaînes éditoriales dédiées à la production de ressources pour l'enseignement et les chaînes éditoriales conçues pour la documentation technique de produits industriels ou de logiciels.

Dans ce chapitre, nous présentons quelques logiciels représentatifs des solutions couvrant l'ensemble du spectre fonctionnel des chaînes éditoriales soit de l'écriture de documents structurés et rééditorialisés à la publication en passant par l'assistance des rédacteurs dans la maintenance de la cohérence du graphe.

6.1 Rédaction Technique

Les chaînes éditoriales conçues pour la rédaction technique, également identifiées dans le milieu anglo-saxon sous la dénomination de *Components Content Management System* (CCMS), sont des outils de rééditorialisation avancés. Souvent basées sur des modèles génériques standards comme DITA (OASIS, 2010), DocBook (OASIS, 2009) ou S1000D (ASD, 2008), ces chaînes éditoriales prévoient systématiquement la possibilité dériver manuellement ou de programmer des déclinaisons des fragments. Elles intègrent toutes des modalités de contrôle des modifications apportées sur le graphe documentaire, soit par un système de procédures inspiré de la perspective Langage/Action, soit par un système de gestion de versions centralisé.

6.1.1 AuthorIT

AuthorIT *(http://www.author-it.com/)* (Scandura, 2005) est une chaîne éditoriale éditée par la société Néo-Zélandaise Author-it Software Corporation. La société vise trois marchés principaux : la documentation technique, les documents pour la santé et les documents pédagogiques. AuthorIT est aujourd'hui principalement disponible en offre de service via internet : la société éditrice héberge directement les serveurs et délivre des accès à ses clients.

AuthorIT s'appuie sur un modèle documentaire générique, spécialisé pour ses différentes cibles (documentation technique, éducation, contenus médicaux). Ce modèle prend en charge le principe de fragmentation et de mutualisation de fragments entre documents. Le logiciel permet également la gestion de fragments incluant des contenus calculés (variables ou filtres dans la publication des fragments). En revanche, la solution ne permet pas de dériver des fragments ni de contrôler simplement l'état des dérivations. Afin d'assister les rédacteurs dans la mutualisation des fragments, AuthorIT propose un module dédié à l'analyse d'un graphe documentaire et la suggestion de fragments à mutualiser.

La structuration des contenus écrits dans AuthorIT se fait dans des bibliothèques. Une bibliothèque représente l'ensemble des contenus stockés dans une même base de données accessible aux rédacteurs. Les fragments stockés dans une base de données se retrouvent donc tous membres du même graphe documentaire. Pour maîtriser et contrôler la complexité du graphe exploité dans une bibliothèque, AuthorIT propose la mise en place de procédures afin d'augmenter les contrôles effectués sur les fragments modifiés avant leur publication.

6.1.2 PTC ArborText

PTC ArborText *(http://www.ptc.com/products/arbortext/)* (Dunn, 2003) est une chaîne éditoriale éditée par la société PTC, spécialisée dans la production de logiciels pour la conception mécanique (PTC Creo pour la conception assistée par ordinateur (CAO), PTC Windchill pour la gestion des cycles de vie des produits ou encore PTC Mathcad pour les calculs de structures mécaniques). ArborText est un logiciel complémentaire de l'offre de PTC afin de permettre la documentation technique des produits conçus avec Creo, Windchill et Mathcad. Le logiciel se structure en plusieurs modules : ArborText Editor pour l'écriture des fragments, AborText isoDraw pour la création d'illustrations issues de logiciels de CAO, ArborText Content manager pour la gestion des fragments et enfin ArborText Publishing Engine pour la publication des documents.

ArborText s'appuie directement sur des modèles documentaires fondés sur des standards de la documentation technique (DITA et S1000D). Le modèle gère nativement la fragmentation des documents et la mutualisation des fragments. En outre, il prend en charge la dérivation manuelle des fragments et le contrôle de la validité des fragments dérivés. Il est cependant impossible de programmer des déclinaisons de fragments.

Pour gérer les entrepôts de contenus, les cycles de vie des fragments et plus généralement, la cohérence des graphes documentaires manipulés, ArborText s'appuie sur le module de gestion des fragments. Ce module est en réalité un connecteur entre le logiciel ArborText Editor et la plate-forme de gestion des cycles de vie des produits Windchill. Le paradigme de gestion des fragments est donc complètement calqué sur celui des fichiers de CAO composant un assemblage.

Le problème de contrôle est donc à la fois géré par la possibilité de mettre en place des procédures pour un contrôle strict des modifications et par la structuration des projets de documentation, organisée et limitée au périmètre d'un projet de conception Windchild.

6.1.3 MadCap

MadCap *(http://www.madcapsoftware.com)* est une chaîne éditoriale développée par la société américaine MadCap Software. MadCap est principalement utilisée dans le milieu de la documentation technique de logiciels. Le modèle documentaire est conçu pour un usage générique et seuls les supports de publication différencient un usage de MadCap pour la documentation logicielle (son cœur de métier) d'autres cibles dérivées comme la formation par exemple.

Le modèle documentaire de MadCap est conçu pour supporter la fragmentation des documents et la mutualisation des fragments. Il est possible de décliner les contenus à partir d'une programmation par insertion de variable ou de filtrer les contenus utilisés pour une publication donnée. En revanche, MadCap ne permet pas de dériver des fragments.

La gestion des modifications apportées au graphe documentaire se place dans le paradigme de la gestion de versions du code source (centralisée). Un serveur contenant un entrepôt par projet est mis à disposition des rédacteurs qui peuvent ainsi récupérer des copies locales, travailler localement et reverser leurs modifications sur le serveur. Le système de reversement demande un message de description à ajouter au journal de l'entrepôt, permettant ainsi de tracer l'historique des modifications. En outre, MadCap propose un module d'analyse de la cohérence du graphe documentaire. Ce module analyse un projet de documentation, détermine l'ensemble des fragments mutualisés entre plusieurs documents afin de faciliter les contrôles par l'usager. Il propose également plusieurs actions de traitement des fragments par lots (conversion, remplacement, etc.).

6.1.4 FrameMaker et RoboHelp

FrameMaker *(http://www.adobe.com/products/framemaker.html)* et RoboHelp *(http://www.adobe.com/products/robohelp.html)* sont deux produits complémentaires de la suite Adobe Technical Communication. FrameMaker est une chaîne éditoriale directement issue de la tradition de l'imprimé. Elle est conçue pour la publication de fichiers au format PDF. Le rédacteur a la possibilité de créer des fragments, dans un format XML ou non, contenant de la mise en forme ou non. Ces contenus sont ensuite liés à des patrons de publication définissant ainsi une surcharge de la mise en forme. RoboHelp est une chaîne éditoriale conçue pour la rédaction d'aide en ligne. Plus moderne dans ses fonctions, RoboHelp ne supporte que les contenus structurés et propose, outre la fragmentation des documents et la mutualisation des fragments, l'insertion de variables dans certains fragments pour permettre leur déclinaison.

FrameMaker et RoboHelp sont complémentaires car il est possible de transposer des projets issus d'une chaîne dans l'autre. Cette technique est utilisée pour améliorer le capacités de publication de la chaîne complète.

Malgré une place très importante dans le marché des chaînes éditoriales, nous considérons que FrameMaker et RoboHelp font plus partie du domaine des chaînes éditoriales pour les éditeurs que pour les rédacteurs. Les possibilités de manipulation du graphe documentaire ne sont pas à la hauteur des autres outils tandis que les possibilités de mise en page sont beaucoup plus travaillées. Il en résulte un décalage dans la complexité de l'usage : l'enjeu est plus de maîtriser les liens entre fragments, patrons et règles de publication plutôt que de maîtriser la cohérence du graphe.

6.1.5 Calenco

Calenco *(http://www.neodoc.biz/)* est une chaîne éditoriale libre développée par la société NéoDoc. Le modèle documentaire de Calenco est fondé sur le standard DocBook. Il permet la fragmentation des documents et la mutualisation des fragments. Le logiciel permet en outre de décliner des fragments par un système de filtres. Il n'est en revanche pas possible de réaliser des dérivations manuelles. Un principe de gestion par procédure permet d'assister le ou les rédacteurs dans le maintien de la cohérence de leur graphe.

6.1.6 Autres chaînes éditoriales

Le marché des chaînes éditoriales pour la rédaction technique contient également de nombreuses solutions partielles qui doivent nécessairement s'interfacer avec d'autres produits pour être pleinement fonctionnelles. On notera notamment : Componize *(http://www.componize.com/)*, qui exploite la solution de gestion électronique de documents (GED) Alfresco *(http://www.alfresco.com/)* comme module de stockage des fragments ; Schema ST4 *(http://www.schema.de/en/software/schema-st4.html)* qui propose uniquement un module de gestion par les procédures et de contrôle de versions à interfacer avec un éditeur et des algorithmes de publication externes ; et TCToolBox *(http://www.ovidius.com/xml-content-management.html)* édité par la société Ovidius, au positionnement similaire.

6.2 Contenus pédagogiques

Les chaînes éditoriales pour la production de contenus pédagogiques, également identifiées dans le milieu anglo-saxon sous la dénomination de *Learning Content Management System* (LCMS), sont des outils partageant à la fois des modèles documentaires plus spécialisés que les précédents exemples et des outils de gestion du graphe documentaire moins avancés.

Chaînes éditoriales simples

Parmi les chaînes éditoriales les plus simples conçues pour la production de documents pédagogiques, nous pouvons citer ChainEdit *(http://www.chainedit.fr/)* et CreaLearning *(http://www.tree-learning.fr/crea-learning.php)*.

ChainEdit est une chaîne éditoriale éditée par l'université de Rennes 1. Dédiée à la rédaction de contenus pédagogiques, son modèle documentaire est clairement orienté avec des structures documentaires de type *cours*, *travail pratique* ou encore *exercice*. ChainEdit permet la fragmentation des contenus mais ne propose ni déclinaison, ni dérivation, ni système de contrôle du graphe documentaire.

Crea Learning est une chaîne éditoriale éditée par l'entreprise française Logipro. Son positionnement est similaire à ChainEdit avec la possibilité d'éditer des ressources, de les assembler dans des modules pédagogiques et de déployer ces modules vers des plate-formes d'apprentissage en ligne.

Chaînes éditoriales avancées

Les chaînes éditoriales plus avancées bénéficient d'un spectre fonctionnel plus avancé pour la gestion et l'assistance à la maintenance des contenus. Il s'agit d'outils des milieux anglo-saxons mobilisés pour la production de très larges documentations de formation professionnelle.

Xyleme LCMS *(http://www.xyleme.com/)* est une chaîne éditoriale éditée par la société américaine Xyleme, Inc. Le logiciel combine un contrôle du graphe par système de gestion de versions et par procédures. Un système centralisé de gestion des versions peut être utilisé par les rédacteurs pour mettre en commun le graphe, gérer les mises à jour et enregistrer des messages dans le journal. Le système de procédures permet quant à lui de faire intervenir plusieurs usagers dans la validation et la publication de nouvelles ressources.

Kenexa LCMS *(http://www-03.ibm.com/software/products/en/category/SW333)* est une chaîne éditoriale éditée par la société IBM. Son spectre fonctionnel pour la gestion et l'assistance à la maintenance du graphe documentaire est similaire à Xyleme : mise en place de procédures de validation de modification et système de contrôle de versions.

On pourra également citer Exact LCMS *(http://www.learnexact.com)* est une chaîne éditoriale éditée par la société Exact Learning Solutions dont le positionnement est similaire.

6.3 Synthèse

Ingé Doc. CE	Modélisation	Fragmentation	Dérivation	Déclinaison	Éléments de contrôle du graphe
Author IT	✗	✓	✗	✓	Procédures Système d'analyse
Arbor Text	✗	✓	✓	✗	Système de gestion de cycle de vie Gestion par procédure
MadCap	✗	✓	✗	✓	Système de gestion de versions centralisé Journal des mises à jour Système d'analyse
Calenco	✗	✓	✗	✓	Gestion par procédure
Chaînes éditoriales pédagogiques simples	✗	✓	✗	✗	Aucun
Chaînes éditoriales pédagogiques avancées	✗	✓	✗	✗	Système de gestion de versions centralisé Journal des mises à jour Gestion par procédure

Tableau de synthèse - chaînes éditoriales

Les réponses apportées par les différentes chaînes éditoriales à la maintenance d'un graphe documentaire sont similaires. Elles s'inscrivent dans les deux paradigmes présentés dans les chapitres quatre et cinq.

Chaînes éditoriales pour la presse

Nous excluons volontairement de ce chapitre les logiciels conçus pour la presse. Cette branche des chaînes éditoriales est certes existante et les outils qui la composent sont utilisés par de nombreuses sociétés de presses. Cependant, l'objectif et la problématique de ces outils sont différents. Un organisme de presse cherche à faciliter la production d'un article et sa publication sur un ou plusieurs supports. Dans son contexte éditorial, maintenir une base documentaire signifie maintenir l'accès et la lecture aux archives. Il n'y a donc aucun enjeu de rééditorialisation et la fragmentation des documents est issue d'une tradition de l'imprimé et de l'organisation des rédactions. La problématique de ces outils est en définitive plus tournée sur la qualité de la mise en forme, automatiquement ou manuellement reprise, que sur la pertinence et le contrôle des fonctions de rééditorialisation.

Contrôle des modifications par des procédures

La mise en place de procédures pour faire valider les modifications sur le graphe par un ou plusieurs usagers est mobilisée par la majorité des chaînes éditoriales.

Un système de procédures permet d'augmenter le contrôle des modifications réalisées sur le graphe. En instaurant un contrôle par un ou plusieurs usagers supplémentaires, des incohérences dans l'écriture des contenus ou la rééditorialisation de contenus existants peuvent être évitées. En outre, un tel système permet, comme souligné par Dourish (2001) et rappelé dans le chapitre précédent, de donner des mesures de l'activité des rédacteurs pour les organisations en ayant besoin.

Un tel système ne peut être efficace qu'à la condition d'être utilisé à plusieurs, ce qui n'est pas toujours le cas des chaînes éditoriales. En outre, la gestion par les procédures est complexe à mettre en œuvre correctement : si les procédures sont trop fermées dans leur exécution, elles empêchent le bon fonctionnement du système au moindre imprévu (absence du gestionnaire, processus de validation non prévu au préalable) ; au contraire si les procédures sont trop ouvertes dans leurs modalités d'exécution, elles pourront facilement être détournées et seront inopérantes.

Système de gestion de versions

Les systèmes de gestion de versions sont moins utilisés que les procédures mais restent présents dans la moitié des chaînes éditoriales que nous avons analysées. En permettant de stocker l'ensemble de l'historique de chacun des fragments, les systèmes de gestion de versions sont des outils puissants pour la maintenance des graphes documentaires. Lorsque des erreurs de cohérences sont observées par un rédacteur, il lui est possible d'observer l'historique d'un ou plusieurs fragments, de consulter le journal des modifications et de revenir dans un état préalablement enregistré.

Le principal inconvénient de ce type de système réside dans leur technicité. Le comportement du système, notamment pour la gestion des conflits entre versions locales et distantes ou entre plusieurs branches, demande une bonne compréhension technique qui ne fait pas nécessairement partie du bagage d'un rédacteur.

Systèmes d'analyse de graphe

La troisième solution mise en œuvre par les systèmes les plus complets consiste en un module d'analyse du graphe et d'assistance aux rédacteurs comme par exemple les modules dédiés d'AuthorIT et de MadCap. Le principal enjeu de tels modules est d'inciter les rédacteurs à mobiliser des fonctions de rééditorialisation dans leur écriture en leur montrant les mutualisations possibles dans AuthorIT ou en donnant une meilleure visualisation du graphe dans MadCap.

Discussions

Rapporté à notre problème de maintien de la cohérence du graphe, les solutions mises en œuvre dans les chaînes du marché sont insuffisantes.

- L'enjeu des systèmes de gestion par les procédures est de contrôler et quantifier les modifications réalisées par les rédacteurs et non de contrôler la cohérence du graphe. Apporter plus de contrôle ne réduit pas la complexité du graphe et donc la difficulté de maintenance en fonction du nombre de fragments et du nombre de publications maintenues. Un relecteur aura les mêmes difficultés, voire encore plus s'il ne manipule pas le graphe tous les jours, à valider la mise à jour d'un fragment vis-à-vis de l'ensemble des fragments auxquels il est lié et non simplement du document pour lequel il est modifié.
- Les systèmes de gestion de versions apportent un moyen efficace d'annuler des modifications non cohérentes en annulant les dernières mises à jour. En revanche, ils ne proposent aucune assistance pour détecter de nouvelles incohérences.
- Les systèmes d'analyse du graphe permettent d'assister les auteurs dans leur compréhension de l'objet manipulé. Ce type de systèmes constitue une proposition intéressante pour améliorer la maîtrise du graphe. Il est néanmoins insuffisant en l'état car

trop annexe dans le procédé d'écriture. Nos travaux visent le même objectif d'assistance à la maîtrise du graphe. La méthode est en revanche intrinsèque à l'outil de production de sorte à généraliser les apports pour l'ensemble des rédacteurs au lieu de les restreindre à l'usage d'un outil d'analyse annexe.

Partie III
Propositions

Chapitre 7

Préambule

7.1 À propos de nos propositions 85
7.2 À propos de notre contribution 86

7.1 À propos de nos propositions

La première partie de ce mémoire se conclut sur l'objet de notre recherche. Il s'agit de gérer la tension entre cohérence du graphe et écriture rééditorialisée. Concrètement, au delà d'un certain volume de fragments, la rééditorialisation est freinée soit volontairement par les rédacteurs qui ne s'y retrouvent plus et dupliquent des contenus au lieu de maintenir des liens entre fragments, soit involontairement par l'apparition d'incohérences à la maintenance du graphe.

La seconde partie de ce mémoire évalue les solutions classiquement mises en place par les chaînes éditoriales pour faciliter le contrôle du graphe. Il s'agit essentiellement d'un accroissement du contrôle de l'activité des rédacteurs par l'usage d'une gestion par procédures et de la mise en place de systèmes de traçabilité de l'ensemble des versions des fragments d'un graphe. Nous y montrons que les systèmes de gestion par les procédures augmentent les contrôles sur l'activité et donc, sur la cohérence du graphe d'une part. D'autre part, nous montrons que les systèmes de gestion de versions sont des outils efficaces pour corriger un graphe où une incohérence a été remarquée mais assistent peu les rédacteurs au cours de leurs interventions sur le graphe.

Nous proposons deux approches complémentaires à mettre en œuvre dans la conception des chaînes éditoriales afin d'assister les rédacteurs dans la maîtrise de leur graphe. Un premier chapitre de propositions montre comment mobiliser les différentes exploitations du discours pragmatique par le domaine de recherche des TCAO afin de documentariser l'activité au sein du graphe.

Dans un second chapitre, nous montrons comment structurer le graphe en fonction des différents projets de rééditorialisation afin de créer des environnements de rédaction simplifiés pour les rédacteurs tout en automatisant le contrôle liens entre les projets de rééditorialisation.

Dans un troisième et dernier chapitre de propositions, nous montrons comment agencer ces deux approches dans une méthodologie pour la conception de chaînes éditoriales manipulant des graphes complexes.

7.2 À propos de notre contribution

Au cours de notre recherche s'est posée la question de la plate-forme mobilisée pour expérimenter nos propositions. Une approche classique de la recherche appliquée en informatique consiste à développer un prototype de laboratoire *ex nihilo*. Une seconde possibilité s'est manifestée avec la possibilité de développer directement nos propositions au sein de la suite Scenari. Chacune de ces alternatives dispose d'avantages et d'inconvénients orientant fortement notre recherche.

Développer un prototype de laboratoire *ex nihilo* donne beaucoup plus de latitude dans les choix techniques et technologiques. Sans lien avec les usages industriels de Scenari, il n'est pas nécessaire de s'intégrer dans son contexte technologique ni de s'astreindre à sa qualité de choix techniques. Les prototypes proposés peuvent alors disposer d'une grande amplitude fonctionnelle. L'inconvénient réside cependant dans l'expérimentation elle-même. En raison d'une maturité moins aboutie (autant en terme de couverture fonctionnelle que de robustesse), l'expérimentation peut difficilement se faire dans un contexte d'usage nécessitant réellement une chaîne éditoriale.

Développer nos propositions directement dans la suite Scenari est plus complexe. Il est nécessaire de s'inscrire dans le contexte technologique du logiciel et de s'astreindre à sa qualité technique pour ne pas pénaliser ses usages industriels. Développer l'intégralité de nos propositions dans la temporalité d'une thèse de doctorat n'y est pas réaliste. En revanche, la maturité de la solution facilite l'expérimentation des prototypes et le cycle agile de développement de la société Kelis permet une transition rapide des développements expérimentaux de laboratoire vers des tests en contextes industriels.

L'enjeu de notre recherche consiste à outiller les contextes de production documentaire où la rééditorialisation est mobilisée à une échelle industrielle et d'en améliorer la maîtrise par les rédacteurs. L'expérimentation de tels contextes de production n'est pas possible dans l'hypothèse d'un prototype de laboratoire. Le développement de nos propositions au sein de la suite Scenari est donc la meilleure solution dans l'objectif de confronter nos propositions à ses contextes d'usages réels.

Par conséquent, les développements dirigés par nos propositions théoriques ont été réalisés en équipe avec les développeurs de la société Kelis. L'état d'avancement de ces développements (p. 219) et notre contribution technique (p. 231) sont documentés au sein de deux annexes dédiées.

Chapitre 8

Accompagner la cohérence du graphe par la documentarisation de l'activité

8.1 Fragments de document et discours pragmatique 88
8.2 Éléments pour la modélisation de fragments pragmatiques 91
8.2.1 Moteur de tâches 91
Modélisation 91
Exploitation 94
8.2.2 Système de commentaires 97
Modélisation 100
Exploitation 100
8.2.3 Enrichissement des fragments documentaires 101
Modélisation 101
Exploitation 102
8.3 Exemples de modèles 103
8.3.1 DILA 103
8.3.2 Enaction Series 109

Introduction

Dans ce chapitre, nous nous intéressons aux exploitations du discours pragmatique par le domaine de recherche des TCAO. Nous extrapolons les propositions de Winograd (1986) et Zacklad (2005 ; 2007) aux chaînes éditoriales et à leur objet : le graphe documentaire. L'enjeu est d'inscrire le discours pragmatique directement dans le graphe afin de le documentariser. Nous cherchons donc à documenter l'action au sein du graphe pour faciliter sa gestion et le contrôle de sa cohérence.

8.1 Fragments de document et discours pragmatique

Objectif

Le domaine de recherche des TCAO s'est approprié le discours pragmatique et les conversations pour l'action de deux façons différentes : la perspective langage/action et les documents pour l'action. Nous proposons une approche exploitant ces deux propositions. Il s'agit de proposer une inscription du discours support à l'action dans le graphe documentaire, participant ainsi à la documentarisation de l'activité, et d'exploiter ces informations pour construire des fonctions d'assistance et de pilotage de l'activité telles que des tâches ou des conversations entre rédacteurs.

Le fragment pour l'Action

Rapporté au graphe documentaire, permettre l'instanciation et l'usage de documents pour l'action revient à modéliser, en fonction de l'organisation du ou des rédacteurs, des documents mobilisés comme support pour encadrer les actions sur le graphe.

Posons comme exemple théorique un rédacteur utilisant une liste de tâches pour organiser sa production. Au début d'une séquence de travail, le rédacteur effectue la liste des modifications sur le graphe qu'il souhaite opérer.

Modéliser un tel document dans une chaîne éditoriale est aisé. Nous proposons dans la figure 39 deux classes de fragments afin de rédiger ces listes de tâches.

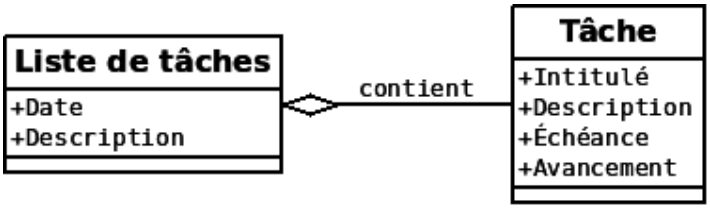


Figure 39 : modèle d'une liste de tâches

Une instance XML d'un fragment de tâche pourrait alors être comme suit.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <tache>
3   <intitule>Mise à jour de la procédure de fabrication Giant Burger
  </intitule>
4   <description>
5     Suite au changement des équipements de cuissons des steaks, la
  procédure de fabrication des Giant Burger doit être mise à jour. Le
  temps de cuisson passe de 9 minutes à 7 et l'entretien de la machine
  entre deux séries doit être modifié.
6   </description>
7   <echeance>13-10-2014</echeance>
8   <avancement>Tache non démarrée</avancement>
9 </tache>
```

Dans ce modèle, les tâches sont agrégées dans des listes de tâches dont l'instance XML pourrait être comme suit.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <listeTache>
3   <date>16-08-2014</date>
4   <description>
5     Liste des taches de maintenance avant le déploiement des
  nouveaux équipements en restaurant.
6   </description>
7   <tache refUri="tacheGiant.xml"/>
8   <tache refUri="tacheSupreme.xml"/>
9   <tache refUri="tacheSodas.xml"/>
```

```
10   <tache refUri="tacheEntretien.xml"/>
11 </listeTache>
```

À ce stade, le principal apport de tels fragments consiste à réimporter dans la chaîne éditoriale l'activité documentaire mobilisée pour soutenir l'action. En faisant des liens dans chaque tâche vers le ou les fragments mobilisés, les fragments pour l'action s'intègrent pleinement dans le graphe documentaire. En consultant les fragments de type *tâche* ayant un lien vers un fragment à modifier un rédacteur pourra facilement consulter l'historique des tâches qui lui sont liées.

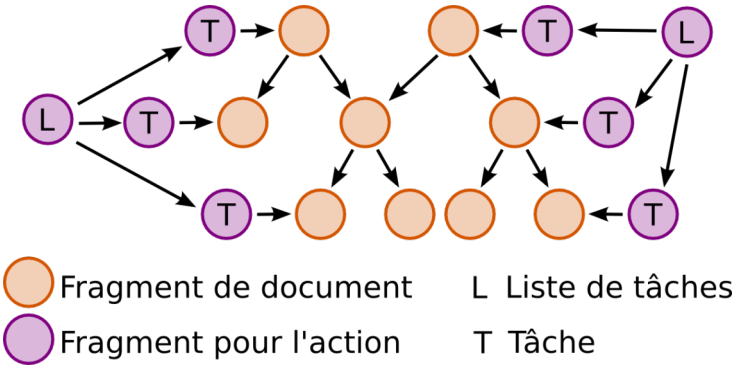


Figure 40 : fragments tâche et liste de tâche dans un graphe documentaire

Exploitations des fragments pour l'action

La manipulabilité offerte par le support numérique permet d'aller plus loin dans l'exploitation des fragments pour l'action. Au même titre que les environnements de développement intégrés comme Eclipse exploitent les commentaires pour instancier un moteur de tâches, il est possible d'utiliser ces fragments comme représentations informatiques de fonctions de support de l'action. On pourrait ainsi imaginer une fenêtre dédiée à la gestion de tâches dont l'affichage est construit à partir des fragments pour l'action comme illustré sur la figure 41. Toute interaction avec la vue ainsi proposée modifierait directement les objets documentaires.

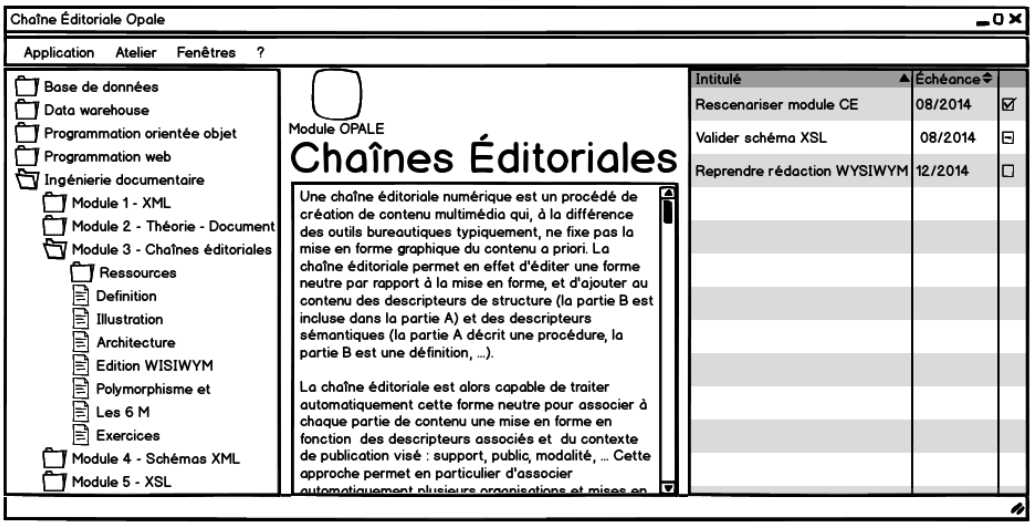


Figure 41 : maquette - chaîne éditoriale avec liste de tâches intégrée

De nombreuses possibilités de modélisation et d'exploitation peuvent ainsi être réfléchies, il suffit pour cela de concevoir des fragments dont les informations enregistrées sont mobilisées par des fonctions dédiées à la gestion de l'activité.

Le fragment, entre production de documents et organisation de l'action

La notion de fragment pour l'action constitue une transposition du concept de

Documents pour l'Action rapporté aux chaînes éditoriales numériques. Cependant cette délimitation entre fragments dont l'objet est la production de documents et fragments dont l'objet est le support de l'action n'a pas de raison d'être en tenant compte de l'unicité de l'objet considéré : un fragment de document encodé selon un schéma XML. En outre, cette distinction n'a pas nécessairement de sens au niveau des usages. Il peut ainsi être nécessaire d'ajouter des informations de support de l'action au cœur des fragments classiques. À l'inverse, les fragments pour l'action peuvent mobiliser des structures documentaires complexes et embarquer une partie de la documentation produite par une chaîne éditoriale.

Par exemple, les dossiers d'homologation rédigés par la société Quick permettent la mise en œuvre de nouvelles procédures au sein des restaurants. Les informations assistant l'action d'un tel document sont importantes pour déterminer son statut (par exemple, pour différencier un projet de nouvelle procédure d'une procédure validée et à respecter dans les restaurants). Ces informations sont publiées sur le pied de la page de garde des dossiers.

Figure 42 : pied de la page de garde d'un dossier d'homologation de Quick

Sur la figure 42, on distingue l'identité du rédacteur à gauche, le statut du document et la date de publication en bas à gauche, la liste des validations sur la droite - constituée de la nature des validations, des identités des experts ayant validé et des dates de validation - et de la date de la dernière modification en bas à droite. En remettant en perspective la production des documents d'homologation vis à vis de la maintenance du référentiel des normes et méthodes, on peut considérer un nouveau dossier validé comme une fiche de demande de maintenance du référentiel. La validation d'un dossier est en effet préalable à une mise à jour des procédures en usage dans les restaurants. Les dossiers d'homologation ont donc un statut hybride avec un premier objectif de production documentaire en soi et un second objectif d'organisation de l'activité documentaire sur le référentiel.

La notion de fragment pour l'action construite en opposition à des fragments documentaires classiques perd ainsi de son sens. L'enjeu est plutôt d'outiller des situations hybrides comme l'exemple des dossiers d'homologation de Quick où le support de l'action et la production documentaire sont intimement entremêlés. Nous proposons ainsi le concept de *fragment pragmatique* pour désigner ces fragments au double enjeu. En lieu et place de la distinction claire entre fragment pour l'action et fragment documentaire, les fragments pragmatiques peuvent être projetés sur un axe allant du support de l'action à la production documentaire. La place sur l'axe dépend ainsi de l'importance accordée à chacune des dimensions.

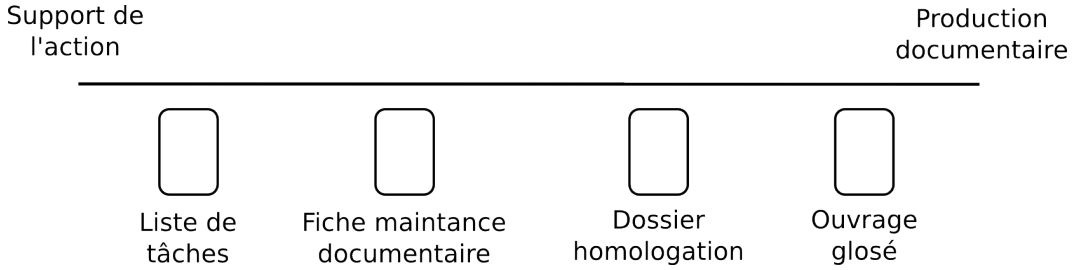


Figure 43 : axe du support de l'action à la production documentaire

Sur la figure 43, nous avons représenté quatre documents mobilisant des fragments pragmatiques. La liste de tâches est le plus proche du support de l'action, elle permet d'assister les rédacteurs dans leur travail journalier mais n'a pas d'enjeu de publication ni d'archivage par l'organisation. Nous proposons un second document appelé fiche de maintenance permettant à

un rédacteur de faire la synthèse d'une opération de maintenance sur le graphe. Nous avons placé cette fiche relativement plus proche de la production documentaire car un système d'agrégation des fiches et de publication permettraient de constituer des rapports d'activités à exploiter par l'organisation. L'exemple du dossier d'homologation de Quick est plus proche de la production documentaire car son enjeu premier est la production de ces nouveaux dossiers. Il comporte en revanche un second enjeu d'assistance à la maintenance du référentiel comme évoqué précédemment. Le dernier exemple correspond aux ouvrages glosés d'Enaction Series. Dans ce contexte l'ensemble de l'activité est tournée vers la production documentaire. Les remarques des glossateurs ont la vocation de critiquer le contenu afin de le faire évoluer ou de donner un éclairage particulier à un lecteur. Dans ce contexte, c'est le processus éditorial de relecture qui vient rejoindre l'objectif de publication documentaire.

Modèles d'activité

La documentarisation de l'activité s'appuie sur des structures particulières insérées dans les fragments. Ces structures font partie intégrante du modèle exploité par les chaînes éditoriales. Nous proposons de distinguer le modèle documentaire tel qu'il peut être partagé par l'ensemble d'une organisation pour la production de ses documents, de *modèles d'activité* complémentaires enrichissant le modèle documentaire par des structures d'enregistrement de l'activité propres aux habitudes d'une équipe de travail donnée. Ces structures peuvent être stockées soit directement dans les fragments du modèle documentaire, soit en proposant de nouveaux fragments.

8.2 Éléments pour la modélisation de fragments pragmatiques

Au sein de la suite Scenari, nous avons expérimenté trois types de structures, à inclure dans des fragments ou constituant des fragments dédiés, sur lesquelles nous avons construit des fonctions d'assistance aux rédacteurs : un moteur de tâches, un système de commentaires et un enrichissement des fragments documentaires.

8.2.1 Moteur de tâches

Principe

Le moteur de tâches constitue notre expérimentation du principe de liste de tâches pris en exemple dans la section précédente. Nous proposons de modéliser des fragments représentant des tâches. L'enjeu est de modéliser les documents de travail mobilisés par une organisation et d'inclure dans cette modélisation des champs d'informations particuliers qui seront exploités par la chaîne éditoriale pour opérer des fonctions de gestion de tâches. Par exemple, un modèle de tâches peut définir un champ définissant une responsabilité d'un usager dans l'exécution de la tâche. À l'exécution par la chaîne éditoriale, ce champ d'information contrôlé est exploité afin d'associer des fragments tâches à des rédacteurs (un rédacteur peut ainsi consulter sa liste de tâches personnelles).

Modélisation

Éléments constitutifs d'une tâche

Le modèle minimal d'une tâche est composé d'un cycle de vie et d'un titre. Un modèle de tâche correspond à une classe de fragments dont le titre de la tâche est l'intitulé. Par exemple, il est possible de modéliser des tâches à s'auto-attribuer qui s'intituleraient « post-it » ou « notes personnelles ». Le cycle de vie est constitué d'un ensemble à définir d'états et de transitions. Chaque état est associé à un statut de tâche qui peut être soit *à venir*, soit *en cours*, soit *close*. Ce statut est directement exploité par la chaîne éditoriale. Par exemple, si une note personnelle prévoit un état intitulé *note ouverte*, associé au statut *en cours*, et un état *note fermée*, associé au statut *close*, la note personnelle apparaîtra avec toutes les tâches en cours lorsque son état sera *note ouverte* et avec toutes les tâches closes lorsque son état sera *note fermée*.

Les définitions d'une tâche et de son cycle de vie se font respectivement dans une primitive dédiée au sein de SCENARIbuilder. Le schéma et le comportement de cette primitive font partie du méta-modèle de chaînes éditoriales de SCENARIbuilder (appelé *modeling*). Comme l'ensemble de la suite logicielle Scenari, les sources sont libres et consultables en ligne *(http://scenari-platform.org/svn/dev-core/versions/4.1.008/Wsp_Modeling/)*.

Une instance de la primitive de définition du cycle de vie d'une tâche est illustrée sur la figure 44.

Figure 44 : définition d'un cycle de vie

Éléments facultatifs

En complément, un modèle de tâches peut définir une date de planification et une date d'échéance. Ces propriétés permettent d'ordonnancer les tâches entre elles. Un modèle peut également définir un fil de discussion et référencer des primitives documentaires classiques (primitive de composition ou de ressources). C'est par un lien depuis la partie documentaire du modèle de la tâche qu'un fragment lui est associé. Enfin, il est possible de définir des responsabilités. Une responsabilité correspond à un rôle joué par un utilisateur au sein d'une tâche. Par exemple, une tâche de validation d'un contenu par un expert pourra contenir une responsabilité « expert » et une responsabilité « rédacteur ». La modélisation des responsabilités permet d'associer une responsabilité à un état du cycle de vie. Ainsi, quand la tâche de validation par l'expert sera dans l'état « en attente de validation », elle se retrouvera associée à l'usager ayant la responsabilité « expert ».

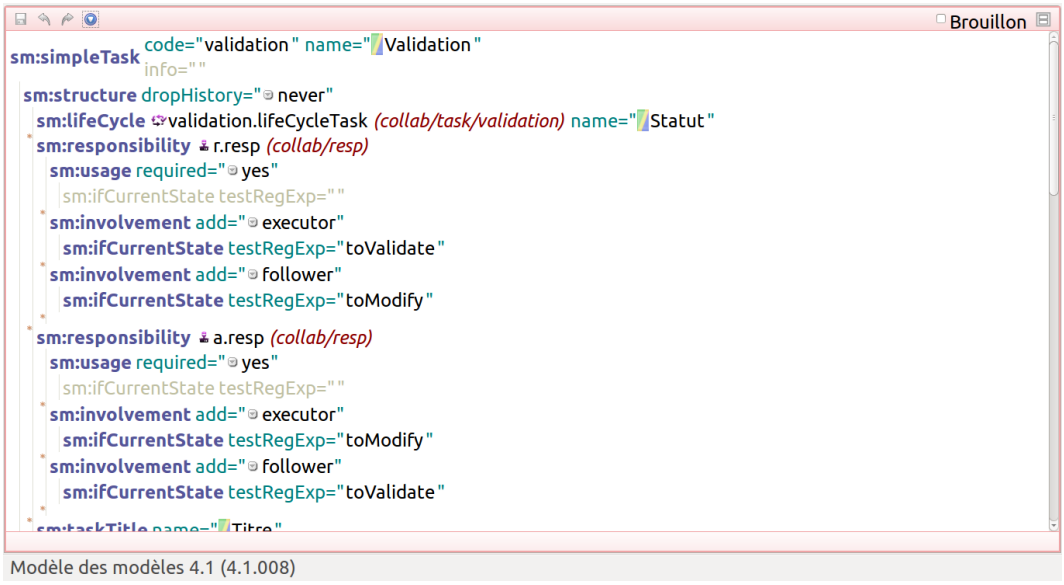


Figure 45 : définition des responsabilité dans une tâche

Conventions de modélisation

Dans ce mémoire, nous utilisons les conventions suivantes.

Pour représenter un cycle de vie, nous proposons une notation inspirée des diagrammes d'activité UML (OMG, 2011, section 12, pp. 303-434).

- Un état initial sera représenté par : ●
- Un état sera représenté par : État
- Une transition sera représentée par : Transition →
- Un état final sera représenté par : ◉

La notion d'état final correspond à un état du cycle de vie duquel aucune transition ne permet de sortir. Par définition, un cycle de vie pourra donc avoir plusieurs états finaux différents ou ne pas avoir d'état final.

À chaque état du cycle de vie correspond un statut (à venir, en cours, clos). Nous proposons de faire varier la couleur de fond d'un état pour spécifier le statut.

- Un état avec statut à venir sera représenté par un fond de couleur orangé : État
- Un état avec statut en cours sera représenté par un fond de couleur blanc : État
- Un état avec statut terminé sera représenté par un fond de couleur mauve : État

Ainsi, un cycle de vie pourrait se modéliser comme illustré sur la figure 46 :

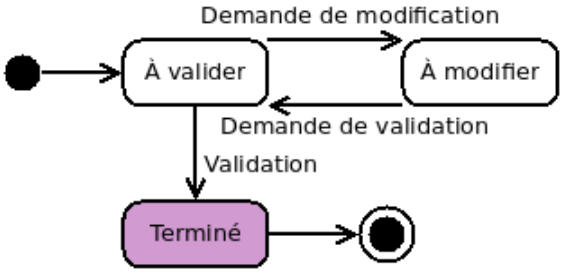


Figure 46 : exemple de cycle de vie

Un fragment de tâche sera modélisé dans une classe au sein d'un diagramme de classes de le standard cycleUML (OMG, 2011, section 7, pp. 21-144). Un stéréotype dédié, différent

des trois déjà introduits (composition, meta et ressource) au sein de la section dédiée à la modélisation du premier chapitre (p. 32), indiquera qu'il s'agit d'une tâche. En outre, afin de prendre en compte le caractère plus fermé de la définition de tâche (mis à part la description qui pointe une primitive documentaire classique, la tâche ne peut contenir qu'un ensemble fini de propriétés), nous proposons de lister les champs pouvant constituer une tâche avec une valeur booléenne pour indiquer son usage ou non dans un modèle donné. Ainsi, dans l'exemple suivant, la tâche de validation comporte un titre, une date d'échéance, un fil de discussion et deux responsabilités intitulées *Relecteur* et *Rédacteur*.

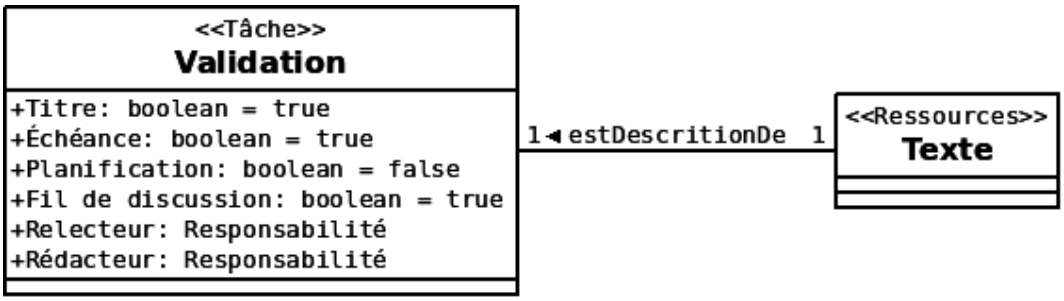


Figure 47 : modèle de tâche de validation

La dernière information nécessaire pour représenter l'ensemble du modèle d'une tâche correspond à l'association entre une responsabilité et l'état d'une tâche. Cette association, si elle existe, peut prendre la valeur *suiveur*, la valeur *exécutant* ou les deux en mêmes temps. Nous proposons de représenter ces valeurs dans un tableau à double entrée : les responsabilités d'un côté et les états de l'autre.

	À valider	**À modifier**	**Terminé**
Rédacteur	Suiveur	Exécutant	Aucun
Relecteur	Exécutant	Suiveur	Aucun

Exploitation

Interface

Lorsqu'un modèle documentaire définit au moins un modèle de tâche, deux nouveaux panneaux sont proposés aux utilisateurs : un panneau de gestion des tâches associées à un utilisateur et un panneau de recherche de tâches.

Figure 48 : paneau de gestion des tâches

Le panneau de gestion des tâches permet de créer de nouvelles tâches, d'afficher les tâches où l'utilisateur connecté est associé en tant qu'exécutant (onglet *À faire*) ou associé en tant que suiveur (onglet *À suivre*). Les onglets *Closes* et *À venir* permettent d'afficher les tâches dans lesquelles un utilisateur est impliqué (exécutant ou suiveur) et dont le statut correspond à l'onglet.

Lorsqu'une tâche est sélectionnée, le fragment tâche est automatiquement transformé en document HTML et affiché dans la partie basse du gestionnaire de tâches.

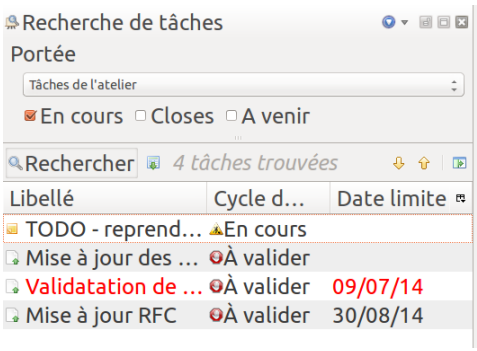


Figure 49 : moteur de recherche de tâches

Le moteur de recherche permet de rechercher une tâche en fonction : du statut de la tâche ou des utilisateurs qui y sont impliqués. Les résultats obtenus peuvent être triés par date d'échéance ou de planification, par état de cycle de vie ou par responsabilité sur une tâche.

Outre ces deux panneaux, une partie du bandeau associé à l'édition d'un fragment est dédié à la visualisation et la gestion des tâches. On peut ainsi y retrouver la liste des tâches associées au fragment en cours d'édition (les tâches dont le modèle documentaire répertorie ce fragment). Il est également possible de créer des tâches directement depuis ce bandeau ou d'afficher la publication HTML d'une tâche, comme depuis le gestionnaire de tâches.

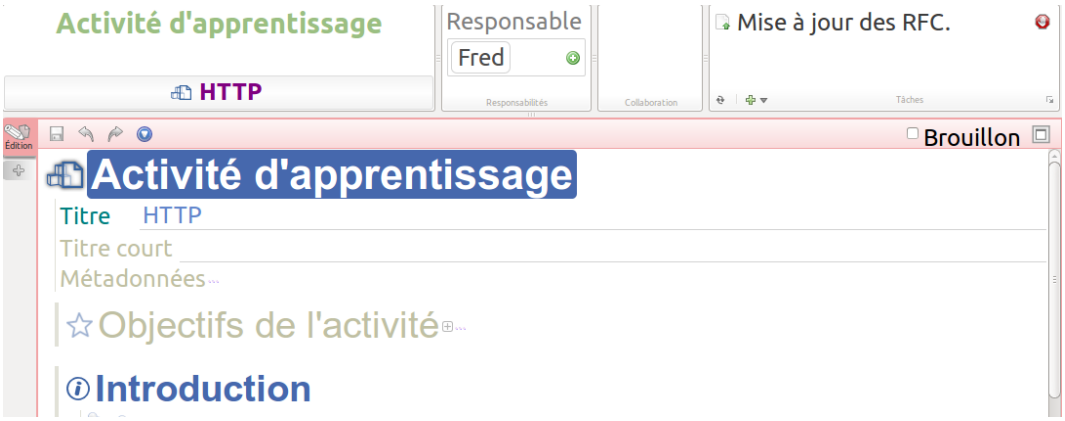


Figure 50 : bandeau associé à un fragment de la chaîne éditoriale Opale

Stockage

Le moteur de tâches est conçu pour accompagner les rédacteurs dans la production documentaire. Les modifications opérées dans les fragments de tâches permettent ainsi de retracer l'activité des rédacteurs sur le graphe. Afin d'exploiter ces informations, le format d'enregistrement des fragments de tâches est conçu comme une liste de modifications.

Ainsi, lorsqu'un fragment de tâche est créé, son enregistrement contient une unique mise à jour.

```
<stk:updateTask>
  <stk:execTransition transition="initToDo"/>
  <stk:putUser account="fred" resp="p"/>
  <stk:setTitle newTitle="Formulations à revoir"/>
  <stk:setDescription>
    <op:sTxtPostIt>
      <sc:para xml:space="preserve" sc:id="t2">
Item : <sc:objectLeaf role="linkedItem" sc:id="t3" sc:refUri=
"id:015tG2z2raTCqMTeW6ps9k"/>
      </sc:para>
    </op:sTxtPostIt>
  </stk:setDescription>
</stk:updateTask>
```

Après plusieurs mises à jour, le contenu évolue en une suite d'enregistrements, comme suit :

```
<stk:task>
  <stk:update by="fred" date="1411900511360">
    <stk:execTransition transition="initToDo" newState="toDo"
newActStage="pending"/>
    <stk:putUser account="fred" resp="p"/>
    <stk:setTitle newTitle="Formulations à revoir"/>
    <stk:setDescription>
      <op:sTxtPostIt>
        <sc:para xml:space="preserve" sc:id="t2">
Item : <sc:objectLeaf role="linkedItem" sc:id="t3" sc:refUri=
"id:015tG2z2raTCqMTeW6ps9k"/>
        </sc:para>
      </op:sTxtPostIt>
    </stk:setDescription>
  </stk:update>
  <stk:update by="fred" date="1411907251849">
    <stk:execTransition transition="goToPlan" newState="toPlan"
newActStage="forthcoming"/>
```

```
  </stk:update>
  <stk:update by="fred" date="1411907258762">
    <stk:setScheduledDt newDt="1413376052008"/>
  </stk:update>
  <stk:update by="fred" date="1411907272532">
    <stk:setDescription>
      <op:sTxtPostIt>
        <sc:para xml:space="preserve" sc:id="t2">
Item : <sc:objectLeaf role="linkedItem" sc:id="t3" sc:refUri=
"id:015tG2z2raTCqMTeW6ps9k"/>
        </sc:para>
       <sc:para xml:space="preserve">
La formulation de l'introduction est à revoir
       </sc:para>
     </op:sTxtPostIt>
    </stk:setDescription>
  </stk:update>
</stk:task>
```

8.2.2 Système de commentaires

Principe

Le système de commentaires constitue une expérimentation inspirée à la fois des commentaires mobilisés par les outils de bureautique classique et des systèmes de commentaires de code source évoqués dans le chapitre quatre (p. 66). Leur enjeu est de permettre l'ajout de contenus au sein de n'importe quel fragment, à n'importe quel niveau, quel que soit le modèle. Cette indépendance du modèle permet de construire des fonctions d'exploitation de ces commentaires qui pourront fonctionner de manière universelle quel que soit le modèle.

Le contenu de ces commentaires a pour objet la documentarisation du fragment. Les informations qu'il contient sont donc exclusivement à destination des rédacteurs et des autres usagers intervenant au cours du processus de rédaction (comme par exemple le responsable d'une équipe ou un expert métier). Elles sont expurgées des publications finales des chaînes éditoriales.

Il est possible d'imaginer de multiples usages. Par exemple, l'ajout de commentaires pour négocier le contenu d'un fragment entre rédacteurs, l'ajout d'indications personnelles sur des éléments à revoir ou encore la contextualisation d'un fragment.

Conception

Pour permettre l'ajout de commentaires dans n'importe quel fragment, à n'importe quel niveau, quel que soit le modèle, nous exploitons le principe de commentaire du langage XML. Le XML permet l'ajout de balises de commentaires qui démarrent par les caractères « <!-- » et terminent par les caractères « --> ». Entre ces deux chaînes de caractères, il est possible d'insérer n'importe quel contenu non pris en compte lors des contrôles de validité d'un schéma XML. Nous exploitons ce principe pour insérer du contenu formaté afin d'y enregistrer des fils de discussions entre rédacteurs.

Un fil de discussion est une succession de commentaires. Chaque commentaire a une date de création, de dernière édition, est associé à un usager et dispose d'un contenu. Le fil de discussion peut être ouvert ou clos. La structuration de ces discussions se fait par un formalisme XML. Cela correspond donc à des contenus XML insérés dans les commentaires XML d'un fragment de document. Le schéma de ces fils de discussion est reproduit ci dessous.

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0" xmlns:a=
"http://relaxng.org/ns/compatibility/annotations/1.0"
datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes">
    <start>
      <element xmlns="http://relaxng.org/ns/structure/1.0" name=
"comment" ns="scenari.eu:comment:1.0">
```

```
5         <attribute name="type"><value>thread</value></attribute>
6         <optional><attribute name="threadClosed"><data type=
  "boolean"/></attribute></optional>
7         <oneOrMore>
8           <element name="comment">
9             <optional><attribute name="author"
  ><text/></attribute></optional>
10            <attribute name="creationTime"><data type=
  "positiveInteger "/></attribute>
11            <optional>
12              <attribute name="updateTime"><data type=
  "positiveInteger "/></attribute>
13              <attribute name="updateBy"><text/></attribute>
14            </optional>
15            <text/>
16          </element>
17        </oneOrMore>
18      </element>
19    </start>
20 </grammar>
```

Les fils de discussion sont directement insérables à n'importe quel niveau dans l'éditeur XML de Scenari.

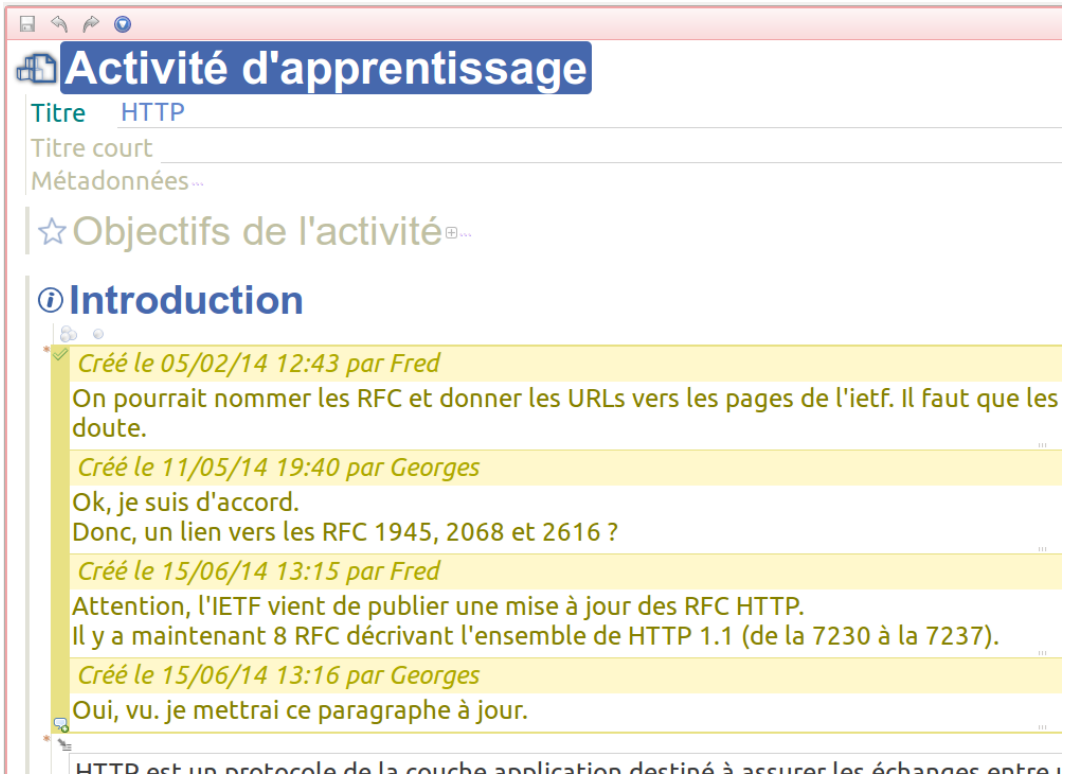


Figure 51 : éditeur Scenari - affichage des commentaires

La figure 51 est une vue de l'éditeur Scenari. Le code XML correspondant au fragment commenté est visible dans l'encadré ci-dessous.

```
1 <sc:item xmlns:sc="http://www.utc.fr/ics/scenari/v3/core">
2   <op:courseUa xmlns:sp="http://www.utc.fr/ics/scenari/v3/primitive"
    xmlns:op="utc.fr:ics/opale3">
3     <op:uM>
4       <sp:title>HTTP</sp:title>
5     </op:uM>
6     <sp:intro>
7       <op:res>
```

```
<!--<comment xmlns="scenari.eu:comment:1.0" type="thread">
  <comment author="Fred" creationTime="1391600612399" updateTime="1405617354829" updateBy="Arthur">
    On pourrait nomdmer les RFC et donner les URLs vers les pages de l'ietf. Il faut que les étudiants  prennent l'habitude d'aller regarder le standard quand ils ont un doute.
  </comment>
  <comment author="Georges" creationTime="1399830009263">
    Ok, je suis d'accord.
    Donc, un lien vers les RFC 1945, 2068 et 2616 ?
  </comment>
  <comment author="Fred" creationTime="1402830904557">
    Attention, l'IETF vient de publier une mise à jour des RFC HTTP.
    Il y a maintenant 8 RFC décrivant l'ensemble de HTTP 1.1 (de la 7230 à la 7237).
  </comment>
  <comment author="Georges" creationTime="1402830960289">Oui, vu. je mettrai ce paragraphe à jour.</comment>
</comment>-->
<        sp:txt>
          <op:txt>
            <sc:para xml:space="preserve" sc:id="t3">HTTP est un protocole de la couche application destiné à assurer les échanges entre un serveur et un client dans l'Internet.</sc:para>
          </op:txt>
        </sp:txt>
        <sp:res sc:refUri="id:02stG2z2raTCqMTeW6ps9k">
          <op:resInfoM/>
        </sp:res>
        <sp:txt>
          <op:txt>
            <sc:para xml:space="preserve" sc:id="t6">Le protocole HTTP, normalisé par des RFC, a été défini en trois versions. La version 0.9 est le protocole défini à l'origine par Tim Berners-Lee (l'idéateur du WEB). La version 1.0 apporte de très nombreuses fonctionnalités (typage des documents, codage du contenu, identification, ...). La dernière, la version 1.1, est la plus avancée et permet de réaliser différentes négociations, mais surtout d'obtenir des sessions persistantes.</sc:para>
          </op:txt>
        </sp:txt>
      </op:res>
    </sp:intro>
    <sp:courseUc sc:refUri="id:02utG2z2raTCqMTeW6ps9k"/>
    <sp:courseUc sc:refUri="id:02vtG2z2raTCqMTeW6ps9k"/>
    <sp:courseUc sc:refUri="id:02wtG2z2raTCqMTeW6ps9k"/>
    <sp:courseUc sc:refUri="id:02xtG2z2raTCqMTeW6ps9k"/>
  </op:courseUa>
</sc:item>
```

Publication, édition et indépendance du modèle

Sans aucun moyen de publier les commentaires en dehors de l'interface d'édition de la chaîne éditoriale, le système de commentaires reste assez neutre quant à la possibilité de véritablement documentariser l'activité sur les fragments, soit de créer sa documentation. Une seconde exploitation des commentaires réside dans leur publication.

Les chaînes éditoriales peuvent permettre la génération de publications statiques (les documents sont générés puis copiés par les usagers pour un usage externe au logiciel) ou dynamique (les documents sont générés à chaque demande de l'usager). Cette différence technique est assimilable aux vues (pour les publications dynamiques) et aux vues matérialisées (pour les publications statiques) des bases de données. En produisant à la volée chaque page HTML, une chaîne éditoriale se comporte comme un serveur web classique.

Le principe de la publication dynamique et l'indépendance du modèle permettent de

concevoir de nouvelles modalités d'édition des fragments à moindre coût. Le lien dynamique permet d'avoir constamment une version à jour d'un document transformé au sein d'un navigateur tandis que l'indépendance du modèle permet de concevoir un service d'édition des commentaires sans étendre la mécanique de génération et d'interprétation des modèles documentaires à de nouvelles composantes des chaînes éditoriales.

Ainsi, l'indépendance du modèle nous permet de développer un service de visualisation et d'édition des commentaires au sein des publications HTML dynamiques. Cette expérimentation permet d'étudier à moindre coût l'opportunité de sortir une partie de la gestion de la production documentaire des interfaces graphiques des chaînes éditoriales et de leur complexité d'usage dans l'objectif de la déporter dans un navigateur web sur des documents HTML classiques. En fonction des retours des contextes d'usage, cette expérimentation pourrait gagner à être étendue à la gestion de l'ensemble des fragments pragmatiques.

Problèmes techniques

Un tel projet de publication pose plusieurs problèmes techniques.

- Lorsqu'un usager commente un élément HTML, il s'attend à ce que son commentaire soit attaché à l'élément XML du fragment mobilisé pour générer l'élément HTML. Il est donc nécessaire de développer un système d'adressage d'un fragment et des éléments XML au sein de ce fragment depuis la publication HTML.
- De nouvelles modalités d'édition des fragments posent la question de la synchronisation du document HTML visualisé. Si le contenu d'une publication dynamique est modifié au sein de la chaîne éditoriale, cela peut remettre en cause à la fois la validité technique des liens entre élément HTML et élément XML ainsi que la validité du contenu du commentaire (le contenu commenté ayant pu évoluer).
- Proposer un document HTML à commenter dont l'usage édite directement les fragments au sein de la chaîne éditoriale nécessite de penser le comportement du système lors d'une édition simultanée par plusieurs usagers différents.

Les détails de la conception et le traitement de ces trois contraintes sont décrits dans l'annexe dédié à notre contribution technique (p. 231).

Modélisation

L'indépendance du modèle est valable pour les modalités d'édition et d'accès des commentaires entre la publication dynamique et le serveur Scenari. En revanche, comme toute publication de fragments, la publication dynamique est dépendante du modèle. Afin d'héberger le système d'édition des commentaires, elle doit en outre contenir quelques éléments supplémentaires par rapport à une publication dynamique classique.

- Un paramètre du générateur permet d'ajouter la publication des liens élément HTML - élément XML au sein du code HTML.
- Le générateur dynamique doit embarquer des fichiers de scripts permettant de positionner et d'éditer les commentaires dans la page HTML en lien avec le serveur Scenari.

Exploitation

L'exploitation du système de commentaires permet l'affichage et l'édition des commentaires dans un navigateur web comme illustré sur la figure 52.

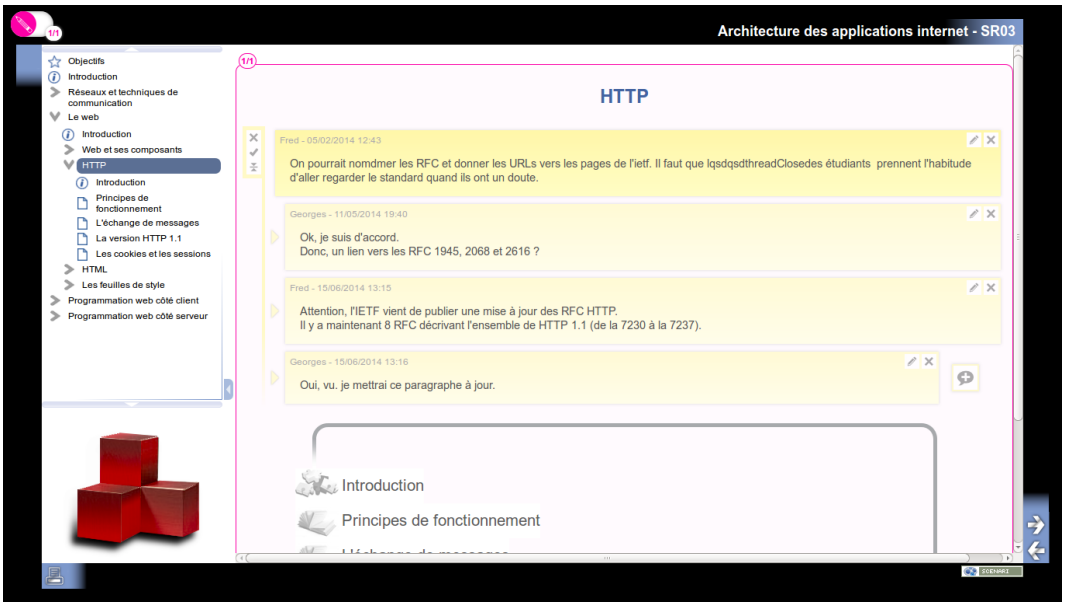


Figure 52 : commentaires vus et éditables dans un navigateur web

Dans ce modèle, un commentaire s'ajoute en double cliquant sur une zone rose. L'édition se fait soit en appuyant sur un des boutons à gauche (supprimer, clore ou cacher le fil) ou en haut à droite de chaque commentaire (éditer ou supprimer le commentaire). Le dernier commentaire de la discussion est suivi d'un bouton permettant l'ajout d'une réponse.

Il est à noter que les interactions (boutons, interactions à la souris) et la vue sont intégralement dépendantes du modèle. D'autres modalités d'affichage et d'édition peuvent être réfléchies au sein des fichiers de scripts et de style développés pour chaque modèle.

8.2.3 Enrichissement des fragments documentaires

Principe

Dans cette section, nous présentons les explorations que nous avons menées pour enrichir les fragments documentaires classiques avec des informations de support de l'action. L'enjeu est d'amener certains éléments conçus pour le moteur de tâches vers des fragments documentaires plus classique. Les fragments documentaires ainsi enrichis contiendront des informations de support à l'action (contrairement aux tâches dont l'enjeu principal est le support de l'action et dont une partie pointe un modèle documentaire).

Un fragment peut être enrichi avec des structures documentaires permettant de suivre son évolution. Par exemple, un document d'homologation de chez Quick contient des structures permettant d'enregistrer la date de sa validation et le nom de l'expert ayant validé le document. Dans cette section, nous présentons des champs d'information à ajouter aux fragments et sur lesquels nous avons développé des fonctions d'assistance à la gestion du graphe. Ces expérimentations permettent d'enrichir les fragments selon deux champs : la gestion de cycles de vie et l'ajout de responsabilités. D'autres expérimentations similaires pourront être entreprises pour outiller des contextes d'usage plus exigeants.

Modélisation

Cycle de vie

Au sein d'un fragment, un cycle de vie correspond à un champ *état* pouvant prendre une valeur parmi une liste définie dans le modèle. Les transitions d'un état à un autre sont également modélisées et peuvent être déclenchées soit manuellement par un rédacteur, soit automatiquement suite à une action du rédacteur (par exemple, lors de l'écriture sur le fragment courant ou sur un des fragments mobilisés par le fragment courant). Les transitions peuvent être restreintes en fonction des classes des fragments, permettant ainsi l'écriture de cycles de vie

différents en fonction de chaque classe. L'ensemble des états et transitions acceptés au sein d'un même modèle documentaire sont définis dans un même fichier de définition.

Figure 53 : modélisation d'un cycle de vie

Responsabilité

Une responsabilité constitue un champ spécifique d'un fragment pouvant contenir un ou plusieurs usagers de la chaîne éditoriale. Plusieurs responsabilités peuvent être définies pour un même fragment et l'intitulé de la responsabilité est à définir à la modélisation.

Par exemple, certaines classes de fragments pourraient avoir un *responsable éditorial*, d'autres un *expert métier* et d'autres encore un *rédacteur* et un *relecteur*. L'enjeu est d'inclure dans le fragment les responsabilités des différents usagers de la chaîne éditoriale.

Conventions de modélisation

Contrairement aux cycles de vie des tâches, les cycles de vie des fragments n'ont pas de statut associé (à venir, en cours ou passé). Un simple diagramme d'activité suffit donc à les modéliser.

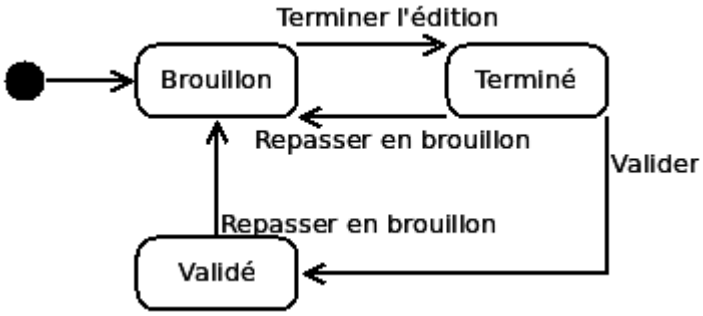


Figure 54 : modélisation du cycle de vie d'un fragment

Exploitation

Lorsqu'un cycle de vie est associé à un fragment, deux éléments graphiques permettent de visualiser son état actuel. Une partie du bandeau est dédiée au cycle de vie et permet l'affichage de l'état courant et le passage d'un état à un autre. Le second élément est directement inclus dans l'arbre de gestion. Les fragments y sont affichés avec une couleur différente en fonction de leur état (couleur dépendante du modèle).

La visualisation des responsabilités se fait également dans le bandeau comme illustré dans la figure 55.

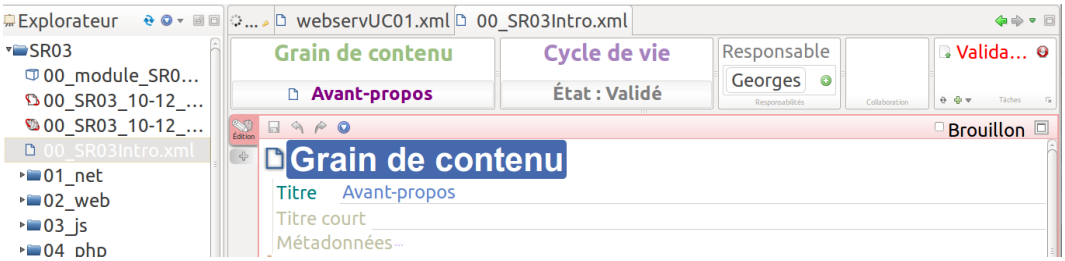


Figure 55 : bandeau d'un fragment avec cycle de vie et responsabilité

Outre la visualisation, les cycles de vie et responsabilités peuvent être mobilisés dans le moteur de recherche via des critères de recherche dédiés. Il est ainsi possible de chercher un fragment par son état ou par le rédacteur y assurant une responsabilité.

8.3 Exemples de modèles

8.3.1 DILA

Contexte

Les équipes travaillant à la rédaction du site service-public.fr ont une organisation très structurée. Des rédacteurs en chef ont la responsabilité de valider les contenus produits pour le site. Une intervention sur le graphe documentaire se fait soit à l'initiative d'un rédacteur, soit à la demande d'un rédacteur en chef. Étant donnée l'importance légale des documents publiés, toute modification doit à la fois être contrôlée pour valider son contenu et documentée pour faciliter la compréhension de son origine et ainsi, faciliter sa maintenance.

Modélisation

La modélisation de ce contexte s'appuie sur le système de tâches. Deux fragments tâches ont été modélisés : un fragment de commande et un fragment de demande de validation. Ces deux fragments sont exploités par le moteur de tâches. L'intégralité de l'activité est à la fois enregistrée (qui a fait quoi et quand) et documentée (dans quel contexte et pour répondre à quel objectif).

Modélisation - fragment de commande

Le fragment de commande est initié par un rédacteur en chef ou un simple rédacteur. Il dispose de trois champs *responsabilité* (responsable pour le rédacteur, suiveur pour un rédacteur ayant initié une commande et souhaitant être informé de l'évolution de la commande et responsable de la validation pour le rédacteur en chef), d'un titre, d'une date de planification, d'une date d'échéance et d'un fil de discussion. Une commande est le plus souvent effectuée suite à un retour fait par des usagers de la documentation (les lecteurs de la documentation). Le modèle documentaire associé à une tâche contient des informations dédiées à la remontée de l'erreur (date et provenance), une description textuelle du problème et une synthèse des modifications à apporter.

Le fragment de commande peut ainsi être modélisé comme illustré sur la figure 56.

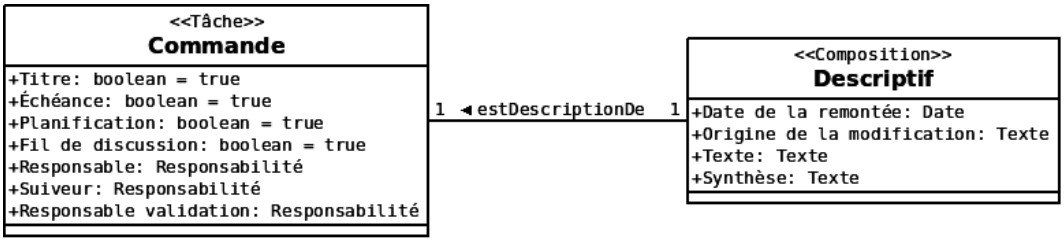


Figure 56 : modèle d'un fragment de commande

Le fichier suivant est un exemple de fragment de commande :

```
<stk:task>
  <stk:update by="BC0605AB" date="1382370761770">
    <stk:execTransition transition="initEnCours" newState="aTraiter"
 newActStage="pending"/>
    <stk:putUser account="GM1001AB" resp="e"/>
    <stk:putUser account="RedacChefSpTel" resp="v"/>
    <stk:setTitle newTitle="Santé Info Droits"/>
    <stk:setDeadline newDt="1385052626845"/>
    <stk:setDescription>
      <dic:commandeDetails>
        <dic:commandeDetailsM>
          <sp:date>2013-10-21</sp:date>
          <sp:origine>tel</sp:origine>
        </dic:commandeDetailsM>
        <sp:txt>
        <dic:taskTxt>
            <sc:para xml:space="preserve" sc:id="t2">
<sc:objectLeaf role="item" sc:id="t3" sc:refUri=
"id:6RASFlGTRHK3hdImiTGnLp"/> : il faut expliquer à quoi sert le
2ème n° de téléphone.
            </sc:para>
          </dic:taskTxt>
        </sp:txt>
      </dic:commandeDetails>
    </stk:setDescription>
  </stk:update>
  <stk:update by="GM1001AB" date="1382535749637">
    <stk:execTransition transition="aValider" newState="aValider"
newActStage="pending"/>
    <stk:addComment>
      <dic:taskTxt>
        <sc:para xml:space="preserve" sc:id="t2">précision apportée
; lien formulaire vérifié</sc:para>
      </dic:taskTxt>
    </stk:addComment>
  </stk:update>
  <stk:update by="BC0605AB" date="1382541368567">
    <stk:execTransition transition="close" newState="close"
newActStage="completed"/>
    <stk:addComment>
      <dic:taskTxt>
        <sc:para xml:space="preserve" sc:id="t2">OK</sc:para>
      </dic:taskTxt>
    </stk:addComment>
  </stk:update>
</stk:task>
```

Une fois interprété par le moteur de tâches, ce fichier XML est mis en forme comme illustré sur la figure 57.

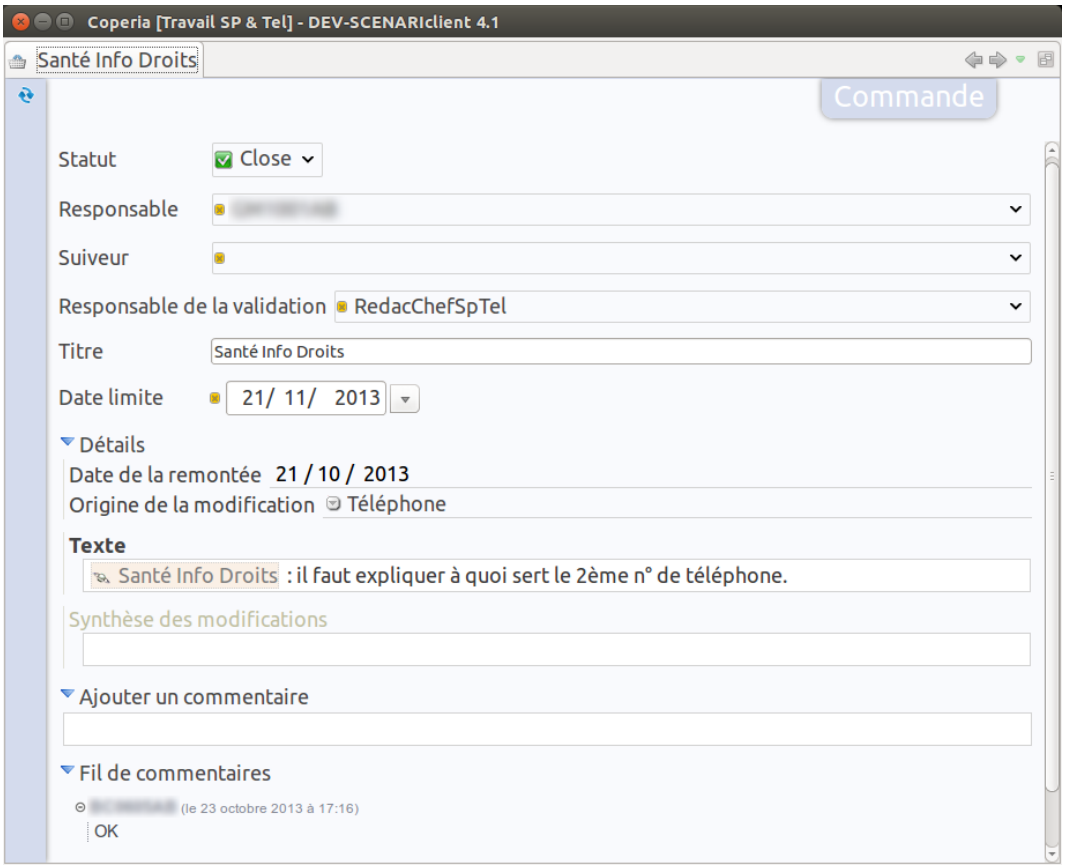


Figure 57 : capture d'écran d'une tâche commande du modèle Copéria

Le cycle de vie de la tâche de commande est défini comme illustré sur la figure 58 :

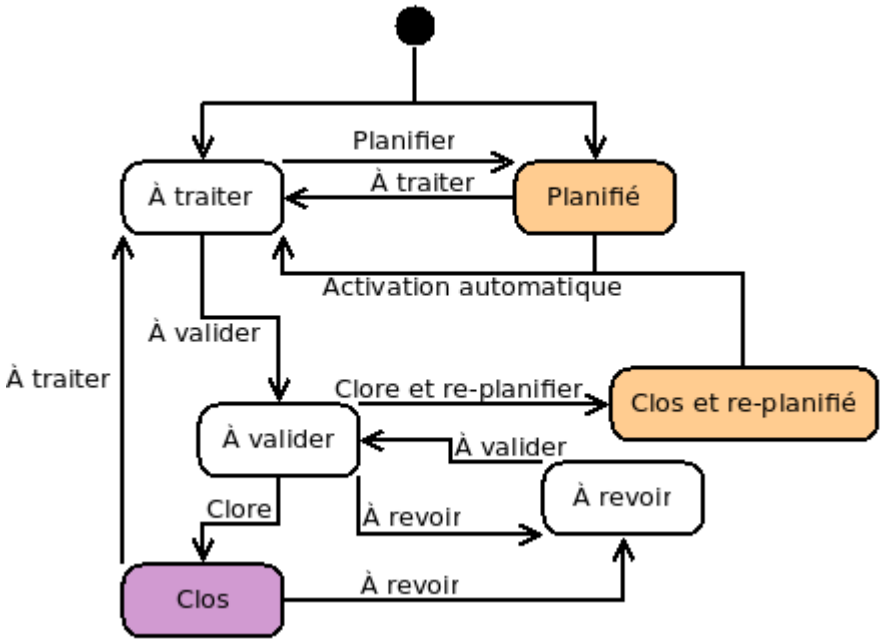


Figure 58 : cycle de vie d'un fragment de commande

	À traiter	Planifié	À valider	À revoir	Clos et re-planifié	Clos
Responsable	Exécutant	Aucun	Suiveur	Exécutant	Aucun	Aucun
Suiveur	Suiveur	Aucun	Suiveur	Suiveur	Aucun	Aucun
Responsable de la validation	Suiveur	Aucun	Exécutant	Suiveur	Aucun	Aucun

Lors de son initialisation, le fragment de commande est directement placé en état *À traiter* si l'intervention est urgente ou *Planifié* le cas échéant. Un fragment en état planifié ou clos et re-planifié n'est associé à aucun utilisateur. Un déclencheur automatique transforme le fragment en état *À traiter* le jour de la date de planification. Un fragment validé et clos peut systématiquement être ré-ouvert. L'état *Clos et Re-planifié* est utilisé pour programmer une réouverture de tâche à effectuer de façon routinière. Il est utilisé dans deux contextes :

- Certaines fiches du site sont activées uniquement à certaines périodes de l'année (comme par exemple la fiche sur les primes de fin d'année). L'état *Clos et Re-planifié* est utilisé afin de réactiver la tâche lorsqu'une fiche doit être ajoutée ou supprimée.
- Certaines fiches deviennent fausses soit par changement législatif, soit en raison d'une péremption (par exemple, la fiche sur le calcul des allocations familiales peut contenir des exemples avec des dates de naissance fictives. Ces dates doivent être remises à jour régulièrement afin que les exemples restent justes). L'état *Clos et Re-planifié* est alors utilisé pour réactiver la tâche lorsque la fiche doit être mise à jour.

Modélisation - fragment de demande de validation

Le fragment de demande de validation est initié par un rédacteur ayant pris l'initiative d'effectuer une intervention sur le graphe. Ce fragment sert à enregistrer le processus de validation de l'intervention. Deux responsabilités lui sont associées : le propriétaire (de la demande) et le responsable de la validation. Outre un intitulé, le modèle documentaire contient un champ de description et un champ de synthèse des modifications.

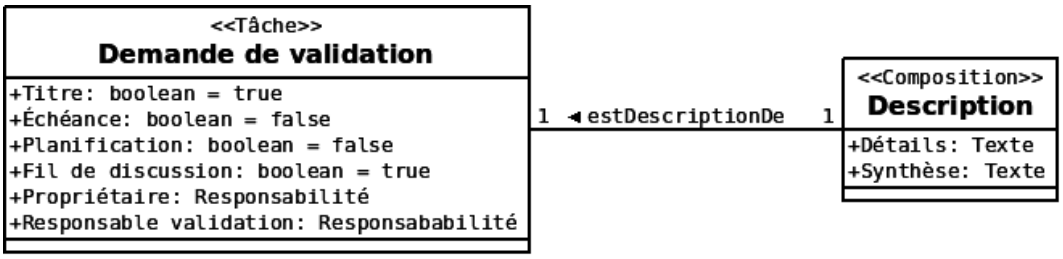


Figure 59 : modèle d'un fragment de demande de validation

Le fichier suivant est un exemple de fragment de demande de validation :

```
1 <stk:task>
2   <stk:update by="GM1001AB" date="1363699882998">
3     <stk:execTransition transition="init" newState="aValider"
  newActStage="pending"/>
4     <stk:putUser account="GM1001AB" resp="p"/>
5     <stk:putUser account="BC0605AB" resp="v"/>
6     <stk:setTitle newTitle="Remontée messagerie du 13/03"/>
7     <stk:setDescription>
8       <dic:validationDetails>
9         <sp:txt>
10          <dic:taskTxt>
11            <sc:para xml:space="preserve">
12              Item :
```

```
13              <sc:objectLeaf role="item" sc:refUri=
"id:6ZjSFlGTRHK3hdImiTGnLp"/>
14            </sc:para>
15          </dic:taskTxt>
16        </sp:txt>
17        <sp:commentaire>
18          <dic:taskTxt>
19            <sc:para xml:space="preserve" sc:id="t4">
20              ajout du certificat d'examen portant la mention
favorable
21            </sc:para>
22          </dic:taskTxt>
23        </sp:commentaire>
24      </dic:validationDetails>
25    </stk:setDescription>
26  </stk:update>
27  <stk:update by="BC0605AB" date="1363700273666">
28    <stk:execTransition transition="aRevoir" newState="aRevoir"
newActStage="pending"/>
29    <stk:addComment>
30      <dic:taskTxt>
31        <sc:para xml:space="preserve" sc:id="t3">J'ai 3 remarques :
</sc:para>
32        <sc:para xml:space="preserve">
33          1) La formulation "pour les personnes qui n'ont pas encore
reçu leur permis de conduire" n'est pas idéale, parce qu'une personne en
attente de la réception d'un duplicata suite à perte / vol peut se
sentir concernée. J'écrirais "l'original de votre permis de conduire ou,
pour si vous avez réussi l'examen pratique du permis mais n'avez pas
encore reçu votre titre définitif, l'original du certificat d'examen
portant la mention favorable,".
34        </sc:para>
35        <sc:para xml:space="preserve" sc:id="t6">
36          2) Dernière phrase : la construction est maladroite, on ne
sait pas à quoi se réfère le "il".
37        </sc:para>
38        <sc:para xml:space="preserve" sc:id="t7">3) Il manque ton
binôme.</sc:para>
39      </dic:taskTxt>
40    </stk:addComment>
41  </stk:update>
42  <stk:update by="GM1001AB" date="1363768389641">
43    <stk:execTransition transition="aValider" newState="aValider"
newActStage="pending"/>
44    <stk:addComment>
45      <dic:taskTxt>
46        <sc:para xml:space="preserve" sc:id="t2">remarques intégrées
</sc:para>
47      </dic:taskTxt>
48    </stk:addComment>
49  </stk:update>
50  <stk:update by="BC0605AB" date="1363777064848">
51    <stk:execTransition transition="close" newState="close"
newActStage="completed"/>
52    <stk:addComment>
53      <dic:taskTxt>
54        <sc:para xml:space="preserve" sc:id="t2">OK, c'est validé.
</sc:para>
55      </dic:taskTxt>
56    </stk:addComment>
57  </stk:update>
58 </stk:task>
```

Une fois interprété par le moteur de tâches, ce fichier XML est mis en forme comme illustré sur la figure 60.

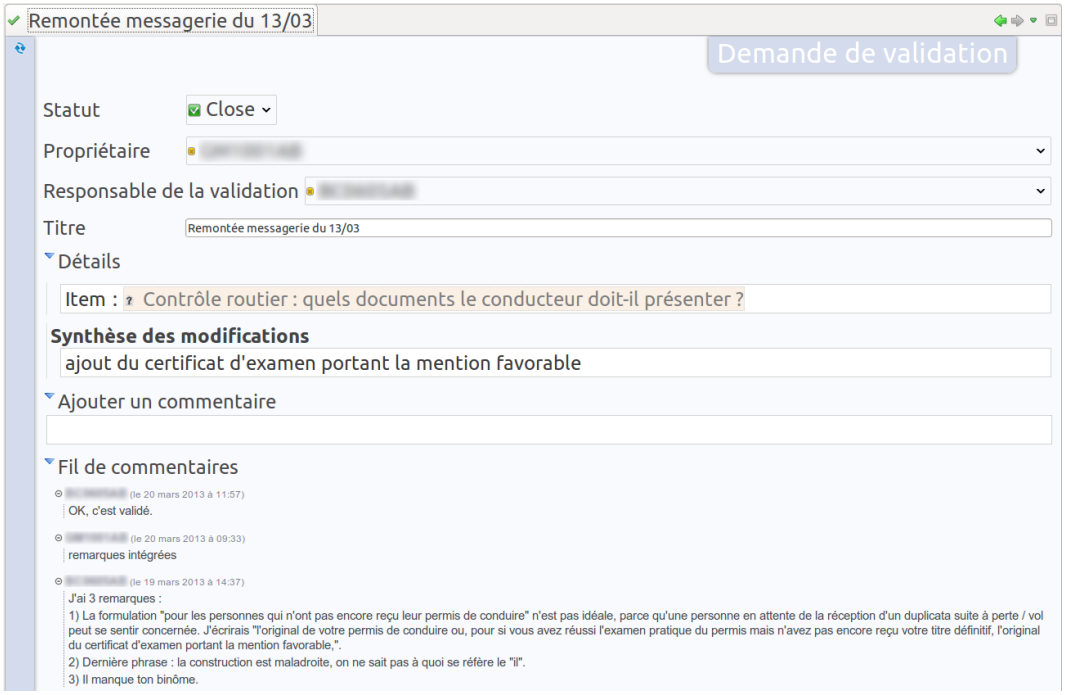


Figure 60 : capture d'écran d'une tâche de demande de validation du modèle Copéria

Le cycle de vie d'un fragment de demande de validation est défini comme illustré sur la figure 61.

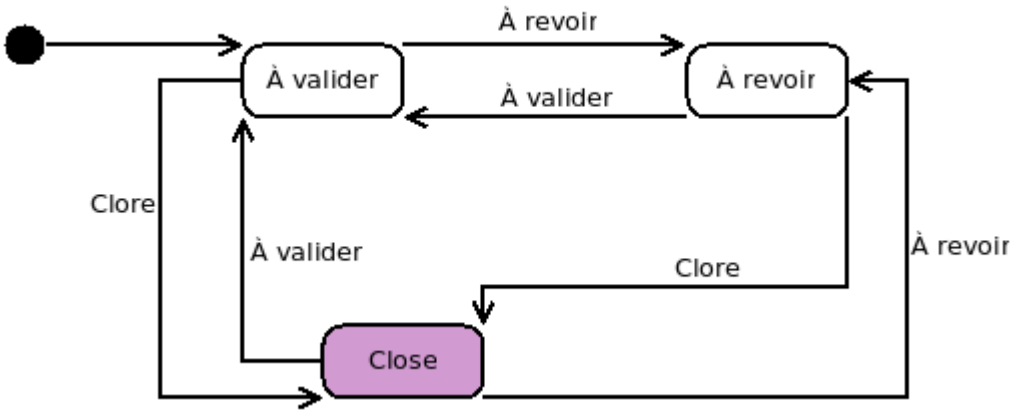


Figure 61 : cycle de vie d'un fragment de validation

	À valider	À revoir	Close
Propriétaire	Suiveur	Exécutant	Aucun
Responsable validation	Exécutant	Suiveur	Aucun

Le cycle de vie mobilisé par le fragment de demande de validation est extrêmement permissif. Il est possible de passer à chacun des états depuis chacun des autres états. On distingue bien ici l'objectif de produire un fragment d'assistance à l'organisation de l'activité plutôt qu'une gestion de l'activité par le processus.

8.3.2 Enaction Series

Contexte

Les ouvrages rédigés pour la collection Enaction Series sont enrichis avec des commentaires savants - de la glose - rédigés par de jeunes chercheurs. L'ajout de ces gloses peut susciter une réponse de l'auteur et initier ainsi une discussion entre l'auteur et ses premiers lecteurs. Les gloses viennent enrichir le processus d'écriture de l'ouvrage et peuvent justifier des évolutions dans son contenu.

Modèle

Nous avons proposé, pour ce contexte, l'usage du système de commentaires synchrones permettant ainsi à un auteur d'écrire son ouvrage dans une chaîne éditoriale et aux lecteurs de gloser sur des pages HTML proches d'une publication imprimée de l'ouvrage.

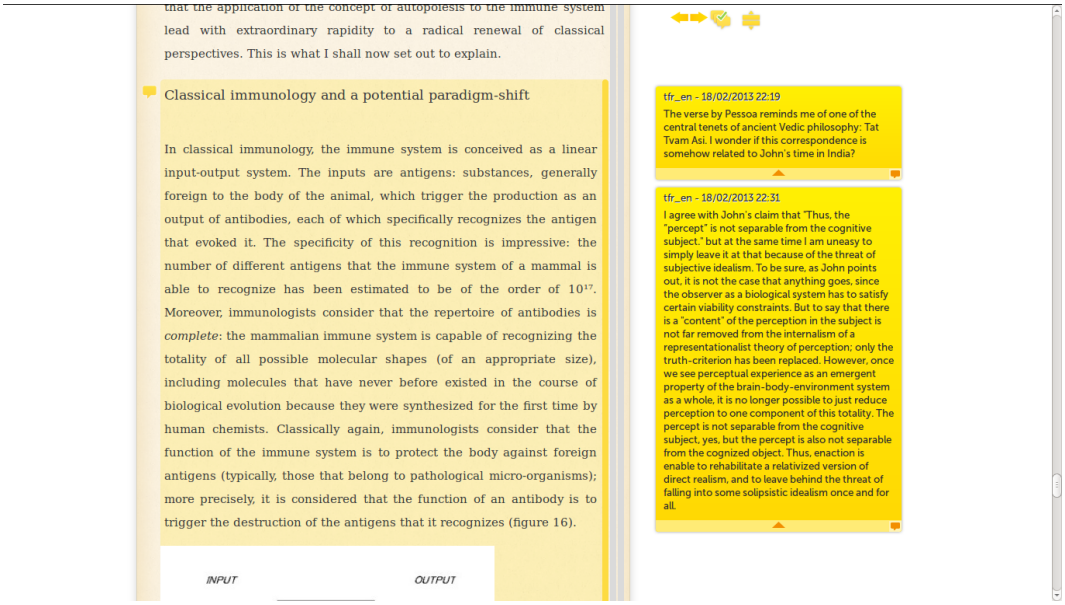


Figure 62 : enregistrement des commentaires dans une interface Web

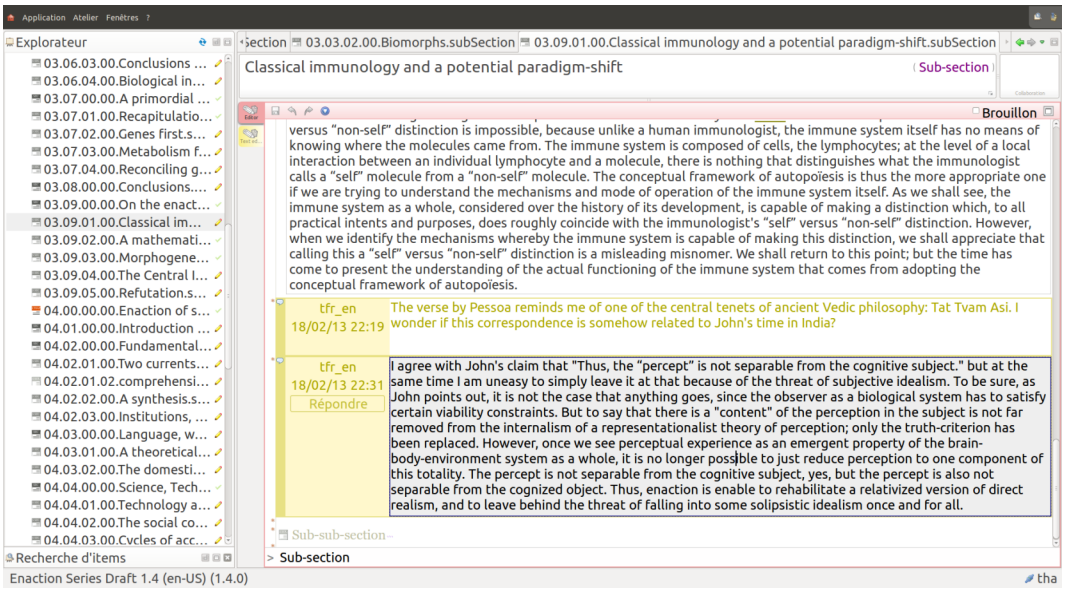


Figure 63 : commentaires insérés dans un fragment du graphe

Chapitre 9

Structurer le graphe pour en simplifier la manipulation

9.1 Projet de rééditorialisation, entre convergence et portée 112
9.1.1 Principe 112
9.1.2 Typologie des projets de rééditorialisation 113
9.2 Éléments pour la structuration des projets de rééditorialisation 118
9.2.1 Projets de rééditorialisation et système de gestion de versions 118
9.2.2 Ateliers calques 119
Multiplier les calques depuis un atelier 120
Fonctions d'atelier calque implémentées 123
Calque de travail 123
Calque de dérivation 124
Autres usages possibles 125
9.2.3 Fragment proxy 126
Multiplication de fragments proxys 127
Fonctions de proxy 128
Proxy de partage 128
Proxy de dérivation 130
9.3 Exemples 131
9.3.1 Enaction Series 132
9.3.2 DILA 132
9.4 Distribution des projets de rééditorialisation 134
9.4.1 Problème 134
9.4.2 PENS, un standard pour le déploiement de ressources pédagogiques 135
9.4.3 Proposition CID 136
Présentation 136
Implémentations 138
9.4.4 Illustrations techniques 139

Téléversement de fichier 139
Déploiement d'un fichier SCORM 140

Ce chapitre est dédié à la seconde partie de nos propositions. Nous y montrons comment exploiter la notion de projet de rééditorialisation pour simplifier le graphe perçu par les rédacteurs en le structurant dans différents ateliers. Nous présentons quatre catégories de projets de rééditorialisation qui s'agencent les uns aux autres. L'enjeu est d'isoler chacun de ces projets afin de proposer des environnements de travail simplifiés leur étant dédiés.

9.1 Projet de rééditorialisation, entre convergence et portée

9.1.1 Principe

Définition : projet de rééditorialisation

Nous parlons de projet de rééditorialisation pour désigner un projet éditorial ou auctorial nécessitant la rééditorialisation de fragments du graphe.

Rééditorialiser pour un projet éditorial consiste à réutiliser des fragments existants, avec ou sans surcharge, pour la production de nouveaux documents. Par exemple, il peut s'agir de la production d'une traduction ou d'une adaptation d'un contenu existant.

Rééditorialiser dans un projet auctorial consiste à réutiliser des fragments existants, avec ou sans surcharge, dans l'objectif d'organiser l'activité des rédacteurs. Par exemple, il peut s'agir de fragments surchargés dont la surcharge est temporaire et vise à être réintégrée dans le fragment d'origine une fois terminée. Ce procédé simplifie la maintenance de documentations sensibles où une version valide du graphe documentaire doit toujours être disponible. Les modifications se font alors sur des surcharges temporaires.

Notions de graphe et atelier

Nous appelons atelier, un espace de travail accessible à un ou plusieurs rédacteurs. Un atelier exploite un ensemble de fragments selon les modalités définies dans un modèle documentaire et un modèle d'activité. Dans un usage classique, l'atelier est l'espace d'exploitation d'un graphe documentaire complet.

Séparation des projets

Notre proposition se résume à des fonctions permettant d'isoler les projets de rééditorialisation dans de nouveaux ateliers. L'objectif est de réduire le nombre de fragments perçus par les rédacteurs. Séparer les projets revient à séparer l'atelier dans lequel un rédacteur maintient une documentation de l'atelier dans lequel un autre rédacteur réutilise cette documentation, pour en embarquer des fragments ou en faire une dérivation.

Fonction de base

Dans cette section, nous appelons *atelier premier* l'atelier dans lequel est initialement exploité un graphe documentaire (soit, avant la mise en place d'un second atelier dédié à un projet de rééditorialisation). Nous appelons *atelier second* un atelier mis en place dans un second temps pour rééditorialiser des fragments issus de l'atelier premier en vue d'un projet de rééditorialisation donné.

Lorsqu'un projet de rééditorialisation est isolé dans un atelier second, la chaîne éditoriale mutualise l'ensemble des fragments concernés. Sur l'exemple de la figure 64, le sous-graphe document issu de B0 est mutualisé entre un atelier second et l'atelier premier auquel il est

associé. Le graphe réel, c'est-à-dire les fragments stockés et exploités par la chaîne éditoriale, est montré entre les deux ateliers.

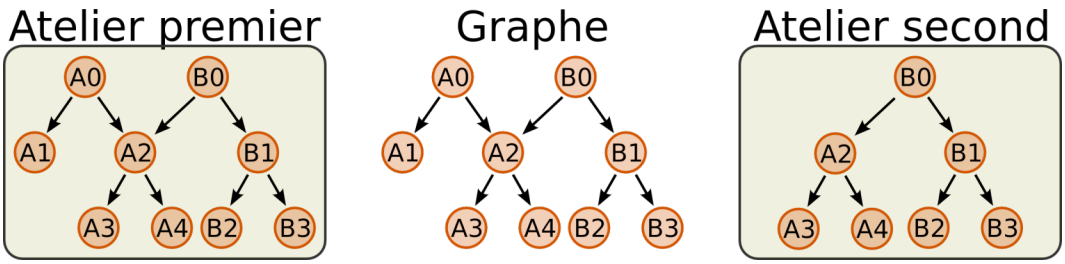


Figure 64 : fonction d'isolation d'un projet de rééditorialisation - initialisation

Lorsqu'un fragment est modifié dans l'atelier premier, la modification est directement visible depuis l'atelier second. À l'inverse, lorsqu'un fragment est modifié dans l'atelier second, la modification est considérée comme locale au projet de rééditorialisation. Le fragment est donc dupliqué et surchargé dans l'atelier second.

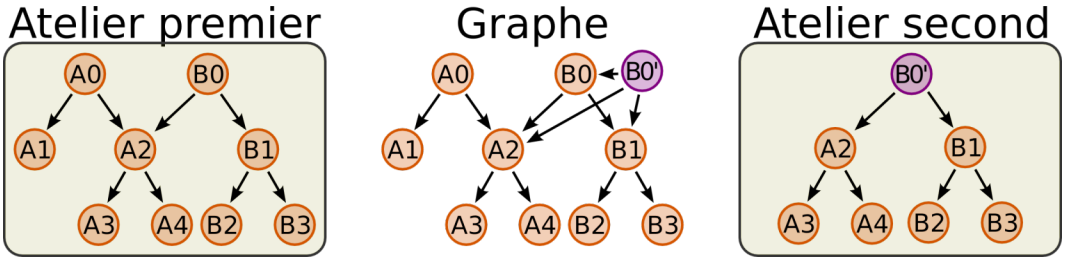


Figure 65 : fonction d'isolation d'un projet de rééditorialisation - modification

Exemple

Soit une société ayant besoin de réaliser la documentation d'un de ses produits. Cette documentation doit être multilingue car le produit est distribué dans plusieurs pays. Afin de diminuer la logistique et les frais de gestion et d'impression de cette documentation, seul un fascicule est imprimé dans plusieurs langues. La traduction d'une documentation technique est un cas typique de projet de rééditorialisation : l'ensemble des fragments contenant du texte sera surchargé tandis que l'ensemble des ressources sera utilisée en l'état. La figure 66 illustre la structuration des ateliers permettant d'instrumenter un tel besoin en séparant chacun des projets de rééditorialisation.

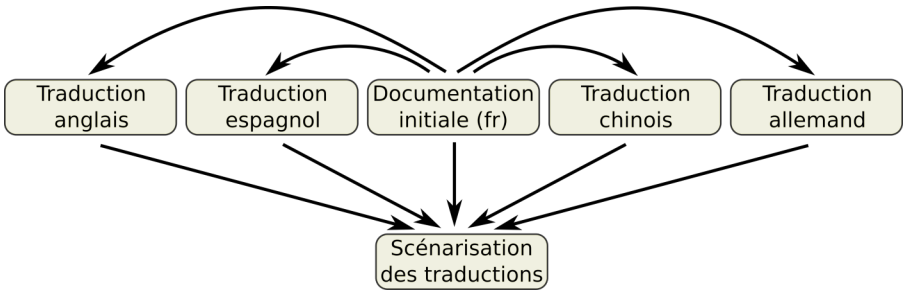


Figure 66 : ateliers d'une chaîne éditoriale

Un atelier premier est mobilisé pour écrire la documentation de référence. Le graphe documentaire est ré-exploité dans des ateliers seconds à chacune des traductions. La documentation de référence et les traductions sont alors toutes importées dans un même atelier afin de permettre la transclusion de chacun des contenus dans un fragment racine.

9.1.2 Typologie des projets de rééditorialisation

Deux critères nous permettent de différencier des catégories de projets de rééditorialisation : la convergence et la portée.

Définitions

Soit un atelier premier faisant référence par rapport à un atelier second. Nous définissons la portée d'un projet de rééditorialisation comme la partie des fragments exploités par l'atelier premier à rééditorialiser dans l'atelier second. La portée peut donc être partielle ou totale. Une **portée totale** désigne la rééditorialisation de l'ensemble des fragments de l'atelier premier dans l'atelier second. Une **portée partielle** désigne la rééditorialisation d'une partie des fragments de l'atelier premier.

Les modifications opérées dans la dérivation d'un fragment peuvent avoir vocation à produire un nouveau fragment (pour produire un contenu dérivé) ou à l'inverse, à réintégrer le fragment d'origine (la dérivation est alors souvent utilisée pour protéger le fragment de l'atelier premier de modifications non terminées). On parlera de **rééditorialisation divergente** dans le premier cas et de **rééditorialisation convergente** dans le second.

Typologie des projets de rééditorialisation

Nous distinguons ainsi quatre types de projets de rééditorialisation qui varient en fonction de la portée et de la direction :

Direction \ Portée	Totale	Partielle
Convergente	Rééditorialisation convergente totale	Rééditorialisation convergente partielle
Divergente	Rééditorialisation divergente totale	Rééditorialisation divergente partielle

Exemple : divergent total

Un projet de rééditorialisation divergent total est utilisé pour rééditorialiser l'ensemble des fragments issus d'un graphe documentaire dans le but de produire de nouveaux documents.

Soit par exemple un graphe mobilisé pour écrire la documentation technique d'un logiciel (telle que la documentation de la chaîne éditoriale OptimOffice *(http://docs.kelis.fr/models/optim/latestStable/site/co/_GuideUtilisateurFS.html)* par exemple).

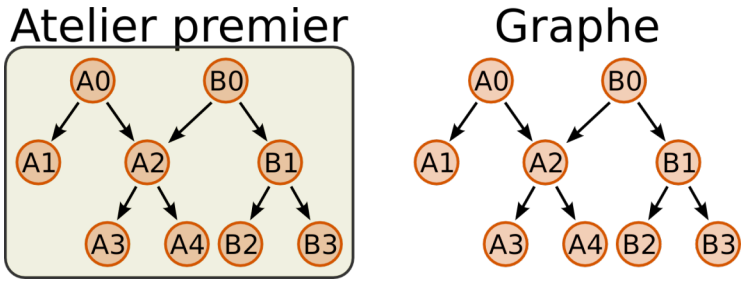


Figure 67 : atelier premier et son graphe

Afin de produire la documentation d'un logiciel dérivé par rapport au logiciel initial (comme par exemple la documentation de la chaîne éditoriale Dokiel *(http://docs.kelis.fr/models/dokiel/latestStable/site/co/_GuideUtilisateurFS.html)* par rapport à celle d'OptimOffice), un atelier second instrumentant une fonction de rééditorialisation divergente totale peut être mis en place. Les modifications sont alors rédigées dans l'atelier second sans altérer les fragments de l'atelier premier.

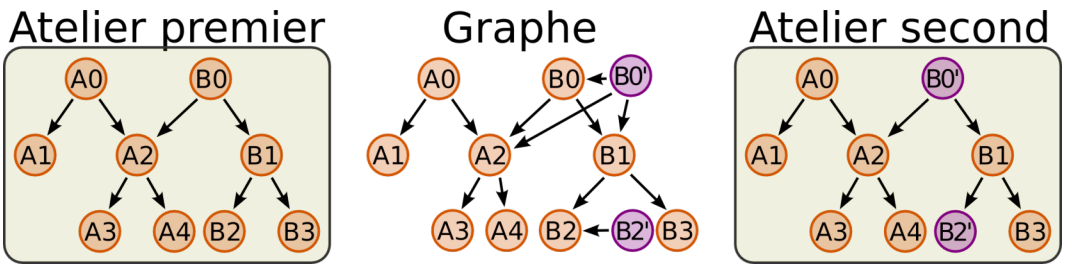


Figure 68 : ateliers premier et second et leur graphe après deux modifications dans l'atelier second

Exemple : convergent total

Un projet de rééditorialisation convergent total peut être mis en place pour protéger un graphe documentaire de modifications inopportunes. Les interventions sur le graphe sont rédigées dans un atelier dédié et les fragments sont mis à jour lorsque la modification est jugée suffisante.

Soit un graphe exploité pour la production d'une documentation sensible, comme le site service-public.fr par exemple. À l'état initial, le graphe est exploitable dans un atelier premier comme illustré sur la figure 67.

Plusieurs rédacteurs travaillent à plein temps sur la maintenance du graphe. Si le site service-public.fr contient une coquille importante à corriger sans délai, le fragment concerné est modifié rapidement et le site web est re-publié. Si d'autres rédacteurs travaillent sur des mises à jour de fiches en parallèle, des fiches altérées, encore non terminées, pourraient être publiées. Afin de protéger le graphe de ces fiches encore non terminées, un projet de rééditorialisation convergent et total peut être mis en place. Un atelier second est alors initialisé. Il s'agit ici d'un projet auctorial puisqu'il ne vise pas à produire de nouveaux documents mais à mieux outiller la production de documents existants.

Ainsi, l'ensemble des opérations de maintenance a lieu dans un atelier second. Comme illustré sur la figure 68, les fragments sont modifiés dans un atelier second sans que cela n'altère les fragments de l'atelier premier.

Lorsqu'une modification est reversée, le contenu du fragment surchargé écrase le fragment d'origine et la surcharge, devenue inutile, est supprimée. Sur la figure 69, le fragment B0' est reversé vers l'atelier premier. Son contenu modifié (matérialisé par la couleur violette) est copié dans le fragment B0. Le fragment B0' est ensuite supprimé. Ainsi, quel que soit l'instant auquel le site web est publié, il ne contient jamais de scories issues de fiches en cours de modification.

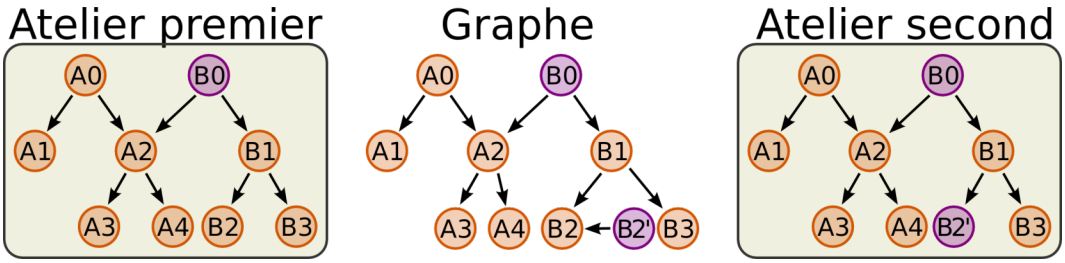


Figure 69 : ateliers premier et second et leur graphe après reversement du fragment B0'

Exemple : divergent partiel

Un projet de rééditorialisation divergent partiel est utilisé pour réutiliser et surcharger une partie d'un graphe. Soit par exemple un enseignant A et un enseignant B intervenant dans la même discipline mais à des niveaux différents.

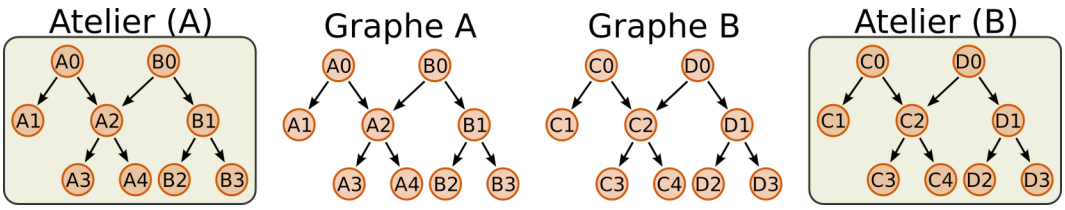


Figure 70 : deux ateliers et leur graphe

L'enseignant B peut demander à l'enseignant A de réutiliser ses contenus afin de les surcharger et de les adapter à son niveau.

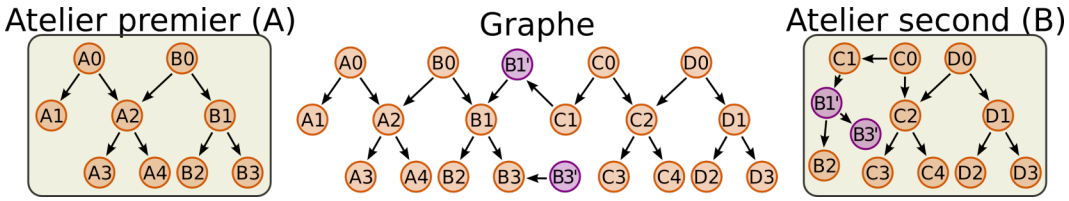


Figure 71 : deux ateliers et leur graphe après deux modifications dans l'atelier B

Exemple : convergent partiel

Un projet de rééditorialisation convergent partiel est utilisé pour permettre la réutilisation d'une partie d'un graphe dans un autre atelier et donc de lier cette portion du graphe avec d'autres fragments. Ce type de fonction peut être utilisé pour partager des fragments entre deux projets éditoriaux distincts.

Soient par exemple deux enseignants A et B exploitant chacun un graphe distinct pour la rédaction d'un cours (figure 70). Lorsque l'enseignant A veut partager des contenus avec l'enseignant B, une fonction de rééditorialisation convergente partielle peut être utilisée. Ainsi l'enseignant B peut référencer ou surcharger les contenus de l'enseignant A (figure 71).

Si une des surcharges réalisées par l'enseignant B intéresse également l'enseignant A (par exemple, pour récupérer des corrections ou des améliorations dans certains contenus), l'aspect convergent de la fonction de rééditorialisation permet à l'enseignant B de reverser ses modifications au sein de l'atelier de l'enseignant A. Comme pour l'exemple précédent, le contenu de la surcharge écrase le fragment original et la surcharge est supprimée.

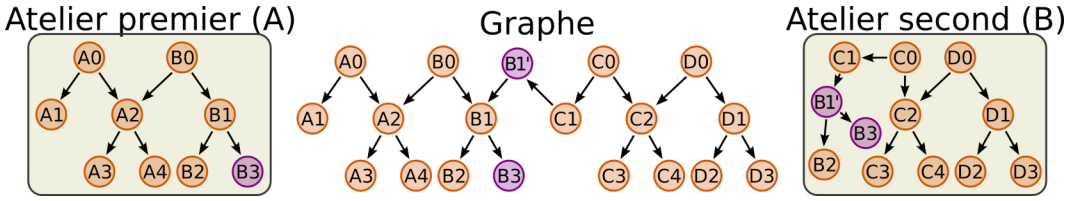


Figure 72 : deux ateliers et leur graphe après reversement du fragment B3'

Direction et gestion des identifiants

Les exemples précédents montrent une proximité entre les usages des projets de rééditorialisation convergents et divergents. La seule différence sur le plan fonctionnel semble être la possibilité de reverser des modifications pour les projets de rééditorialisation convergents. L'outillage d'un projet de rééditorialisation divergent serait alors un simple sous ensemble de celui d'un projet de rééditorialisation convergent. Outiller un projets de rééditorialisation convergent demande une contrepartie dans la gestion des identifiants du système.

Chaque fragment dispose d'un identifiant par lequel il est référencé lorsqu'il est mobilisé par un autre fragment du graphe. Lors de la surcharge d'un fragment dans un atelier dédié, le fragment d'origine est conservé et un nouveau fragment est créé. Si le nouveau fragment dispose d'un nouvel identifiant, il est alors nécessaire de mettre à jour l'ensemble des fragments le référençant. Cette mise à jour modifie le contenu des fragments en question, qui sont donc dupliqués et surchargés et ainsi de suite.

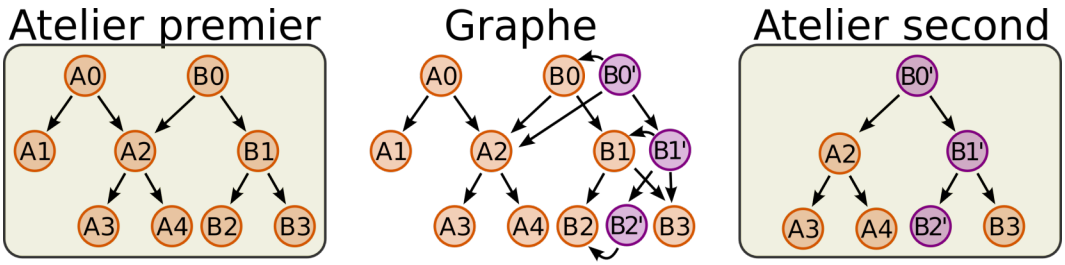


Figure 73 : gestion des identifiants

Par exemple, ce principe de changement des identifiants a été appliqué sur la figure 73. En effectuant une modification sur le fragment B2, un nouveau fragment B2' est créé. Avec ce nouvel identifiant, il est nécessaire de mettre à jour le fragment B1 dans l'atelier dédié, afin de pointer B2' à la place de B2. Cette mise à jour est une surcharge qui crée un nouveau fragment B1'. Il devient alors nécessaire de modifier le fragment B0 afin de référencer B1' dans l'atelier dédié à la place de B1. Une troisième surcharge est donc créée. Dans ce graphe fortement simplifié, une simple modification de B2 entraîne la surcharge de trois fragments.

Afin d'optimiser ce système, nous proposons de distinguer deux identifiants dans la mise en place de fonctions permettant d'isoler un projet de rééditorialisation : un identifiant privé pour le fonctionnement interne de la chaîne éditoriale et un identifiant public associé à un atelier dédié. Ainsi en effectuant une surcharge du fragment B2 dans l'atelier dédié, un nouveau fragment est créé. Son identifiant privé utilisé par la chaîne éditoriale pour manipuler le fragment est B2'. Son identifiant public associé à l'atelier dédié est B2. Ainsi, lors de la résolution des liens dans l'atelier dédié, le fragment B1 référence un fragment nommé B2 qui correspond dans ce contexte au fragment B2'. Les fragments B1 et B0 n'ont plus besoin d'être surchargé.

Ce système est efficace mais il pose un problème lorsque les mêmes contenus sont rééditorialisés dans des ateliers dédiés avant d'être mutualisés dans un même atelier comme dans l'exemple de traduction de la section précédente. Dans cet exemple, les fragments des traductions anglaise, espagnole, chinoise et allemande conservent les mêmes identifiants publics que la documentation initiale. En mutualisant les différentes documentations dans un même atelier dédié, cinq jeux de fragments seraient alors importés avec les mêmes identifiants publics, rendant inutilisable le système d'identifiant public/privé. Pour parer à ce problème, nous proposons de faire varier la gestion des identifiants en fonction de la direction de la rééditorialisation : une rééditorialisation convergente dispose d'identifiants publics et privés ; une rééditorialisation divergente met systématiquement en place de nouveaux identifiants. Avec un tel fonctionnement, les risques de conflits sont minimisés et la création de fragments surchargés pour un simple renommage des identifiants est réduite.

Note : convention de modélisation

Pour améliorer la lisibilité des représentations de graphe, nous considérons systématiquement les projets de rééditorialisation comme convergents du point de vue de la gestion des identifiants. Ainsi, contrairement à la figure 73, nous n'indiquons pas les surcharges automatiques des fragments référençant un fragment surchargé par un rédacteur.

Détection des conflits

Les conflits d'identifiants entre fragments sont identifiables selon l'architecture des ateliers mis en place. Un conflit survient lorsqu'un fragment est importé avec le même identifiant depuis deux ateliers différents. Deux ateliers partagent les mêmes identifiants pour un fragment s'ils sont reliés par une fonction convergente. La fonction convergente peut être directe (les deux ateliers sont directement reliés) ou indirecte (les deux ateliers sont reliés par une chaîne d'ateliers et de liens de rééditorialisation convergente).

Pour détecter les conflits, nous recommandons d'utiliser un graphe de contrôle sur lequel chaque atelier correspond à un sommet et chaque lien convergent à une arête. Un conflit est alors détecté par la présence de cycles. Reprenons l'exemple de la section précédente. Comme en atteste la figure 74, en maintenant uniquement des fonctions convergentes, on y décèle de nombreux cycles désignant autant de conflits d'identifiants.

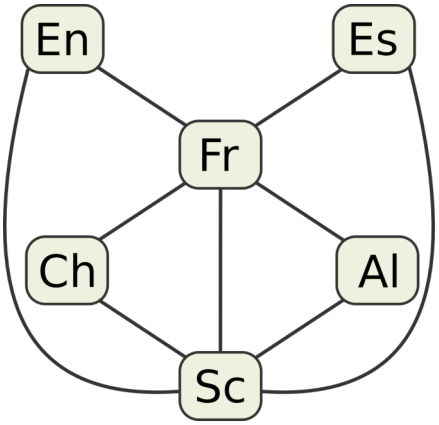


Figure 74 : graphe de contrôle

Dans cet exemple, les traductions correspondent à des réécritures du contenu. Les fragments sont modifiés et ces modifications n'ont pas vocation à être réintroduites dans la version française. Par conséquent, il n'est pas logique de maintenir des liens convergents entre la version française et les traductions. En devenant divergents, ces liens disparaissent du graphe de contrôle. Plus aucun cycle n'est visible sur la figure 75, il n'y a donc plus de conflits d'identifiants au sein des différents ateliers.

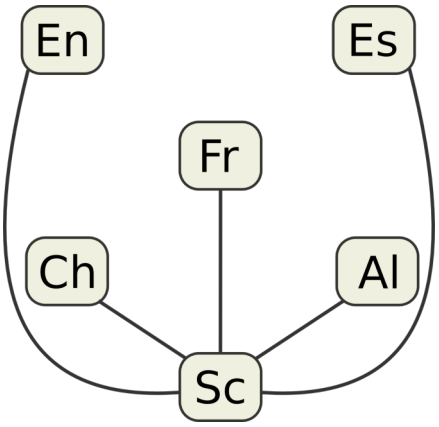


Figure 75 : graphe de contrôle corrigé

9.2 Éléments pour la structuration des projets de rééditorialisation

9.2.1 Projets de rééditorialisation et système de gestion de versions

Outiller les projets de rééditorialisation par des systèmes de gestion de versions

Les systèmes centralisés peuvent directement être utilisés pour outiller la séparation de projets convergents. Un atelier de référence sera considéré comme le serveur centralisé et un atelier dédié comme une copie locale (partielle ou totale). La mise à jour d'un contenu sur l'atelier de référence peut automatiquement être récupérée dans l'atelier dédié et une fonction de mise à jour permet de reverser les modifications de l'atelier dédié vers l'atelier de référence.

Comme décrit dans le chapitre 4 (p. 67), les systèmes centralisés ne sont pas conçus pour le développement et la maintenance de versions alternatives du dépôt principal. Cette limite est en revanche adressée par les systèmes distribués où chaque dépôt est autonome et peut récupérer ou envoyer des mises à jours depuis ou vers un autre dépôt.

Cet aspect de nos propositions est donc exploitable avec des systèmes de gestion de versions.

Limites

Cet outillage est possible mais il pose plusieurs limites fonctionnelles ou techniques.

Sur le plan fonctionnel, l'outillage de projet de rééditorialisation partiel et divergent n'est pas aisé. La gestion d'un sous-ensemble des sources est par construction, difficile à gérer par des systèmes distribués puisque leur fonctionnement repose sur une duplication de l'ensemble d'un dépôt. L'usage d'un système centralisé est également peu adapté car son fonctionnement ne permet pas une gestion efficace de versions alternatives du dépôt central. En outre, les systèmes de gestion de versions reposent sur une identification par chemin des fichiers depuis la racine d'un entrepôt. Par conséquent, un fichier doit être positionné au même endroit de l'arborescence entre un atelier initial et un atelier dédié. Cette contrainte a peu de sens dans le contexte d'exploitation des chaînes éditoriales, notamment pour la gestion des projets de rééditorialisation partiels.

Sur le plan technique, ces systèmes reposent sur la duplication de chaque fragment pour chacun des projets de rééditorialisation. Cette duplication est lourde à gérer et bien moins efficace que le système de fonction de rééditorialisation gérant automatiquement la surcharge des fragments.

En définitive, une majorité du spectre fonctionnel est adressable par les systèmes de gestion de versions. Leur conception tournée sur une problématique différente a cependant un impact sur leur fonctionnement qui rend leur usage difficile dans le contexte des chaînes éditoriales. Certains projets ne peuvent pas être correctement gérés. L'ensemble du système technique n'est pas correctement optimisé.

Propositions

Pour contourner ces limites, nous proposons de repenser la structuration des projets de rééditorialisation par l'usage de fonctions dédiées à la séparation de ces projets d'une part et au contrôle des liens et interactions entre les projets selon une perspective documentaire de l'autre.

Nous proposons ainsi le concept de l'atelier calque pour les projets de rééditorialisation d'une portée totale et le concept du fragment proxy pour les projets de rééditorialisation d'une portée partielle.

9.2.2 Ateliers calques

Principe

Un atelier calque est un atelier conçu pour dériver l'intégralité des fragments exploités dans un autre atelier appelé atelier de référence. Les ateliers calques mutualisent l'ensemble de l'arbre de gestion avec leur atelier de référence. Si un fragment est déplacé dans un atelier calque, il se retrouvera automatiquement déplacé dans l'atelier de référence.

Lorsqu'un fragment est créé dans un atelier calque, la fonction de calque crée automatiquement un fragment original dans l'atelier de référence. Le fragment calque est automatiquement considéré comme un fragment surchargé. Le fragment de l'atelier de référence est initialement masqué mais peut être visible si jamais un fragment des même nom positionné au même endroit dans l'arbre de gestion est créé dans l'atelier de référence ultérieurement. Ces fragments masqués sont appelés fragments fantômes.

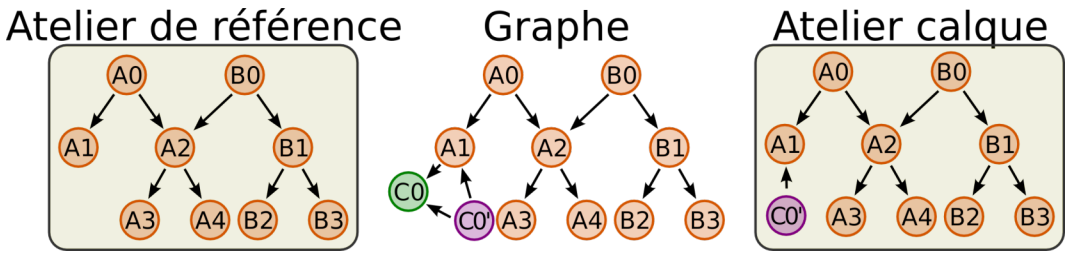


Figure 76 : fragment fantôme

Sur la figure 76, un fragment C0' a été créé dans un atelier calque. Ce fragment est donc automatiquement considéré comme une surcharge d'un fragment C0. Ce fragment n'a pas été créé dans l'atelier de référence, il a donc un statut de fantôme et n'est visible dans aucun atelier. Le jour où un rédacteur crée un fragment ayant le même nom et au même emplacement dans l'arbre de gestion, le fragment C0 sera alors réellement instancié et les fonctions de dérivation pourront fonctionner normalement (contrôle de la dérivation, traduction, envoi de la dérivée pour écraser la version d'origine).

Exploitation

La notion d'atelier calque est une fonction générique de rééditorialisation totale d'un graphe. Les fonctions de calques implémentent ce principe en suivant une direction (convergent, divergent) et en implémentant des comportements particuliers dans la gestion des liens entre les fragments originaux et les fragments surchargés dans le calque (visualisation des différences, contrôle de la validité d'une surcharge, etc.).

Multiplier les calques depuis un atelier

> **Définition**
>
> La multiplication des calques depuis un même atelier donne lieu à de nombreuses configurations possibles. Un atelier calque est nécessairement un atelier second. En fonction de la configuration, il peut également être un atelier premier. Nous proposons d'utiliser le terme **atelier maître** pour définir l'atelier principal d'exploitation d'un graphe. Soit l'atelier premier qui n'est pas un calque.

Le principe d'atelier calque est conçu pour fonctionner nativement avec plusieurs calques parallèles rattachés à un même atelier de référence.

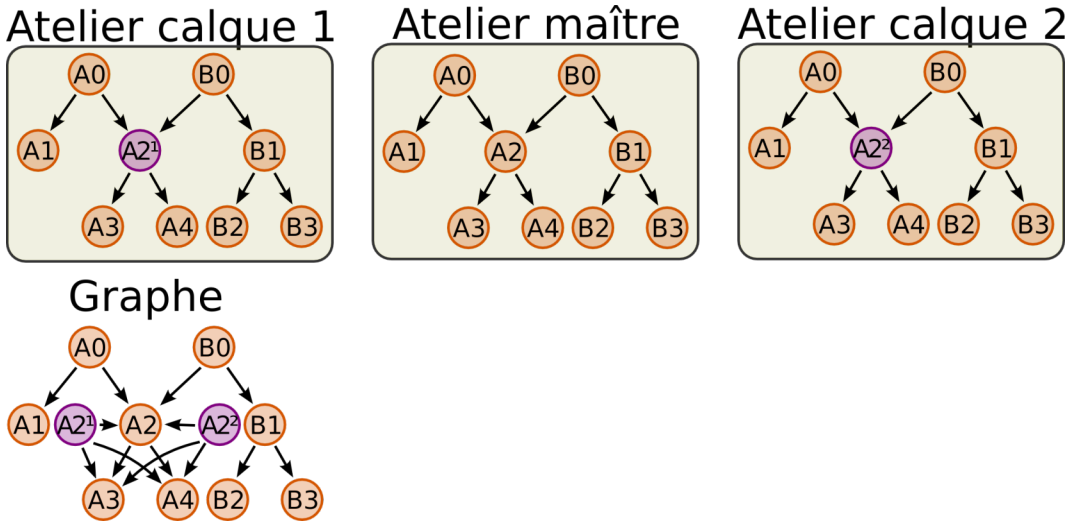


Figure 77 : deux ateliers calques parallèles

Dans un fonctionnement en parallèle, chaque calque fonctionne indépendamment des autres. Il est cependant possible d'imaginer des situations à plusieurs calques avec des interactions entre calques. Nous distinguons deux stratégies que nous nommons calques en séries et calques multi-sources.

Calques en série

La notion de calques en série est simple à percevoir, elle consiste à enchaîner les calques les uns après les autres. Un atelier dispose d'un calque, qui lui même dispose d'un calque, qui lui même dispose d'un calque et ainsi de suite.

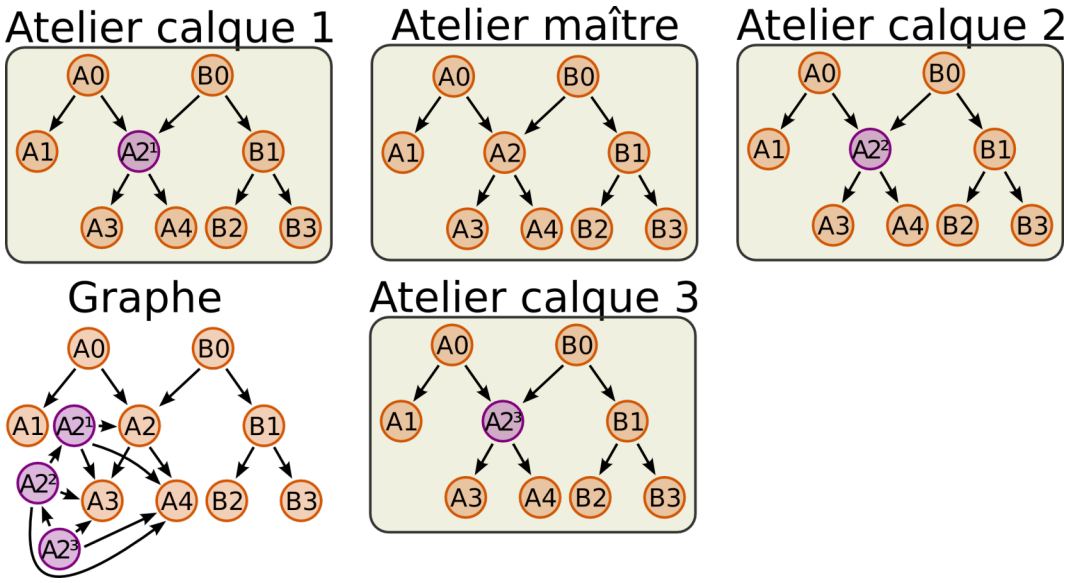


Figure 78 : calques reliés en série

La différence entre la figure 78 et la figure 77 montrant des graphes en parallèle réside dans les dérivations entre les fragments. Sur la figure 78, le fragment $A2^3$ est une dérivation de $A2^2$ qui est lui-même une dérivation de $A2^1$ qui est lui-même une dérivation de A2.

Calques multi-sources

La notion de calques multi-sources peut être vue comme des calques en série dont le chemin de résolution (l'enchaînement des calques) est résolu de façon dynamique. Ainsi, imaginons sur la figure 79 une situation initiale comprenant deux ateliers calques en parallèle.

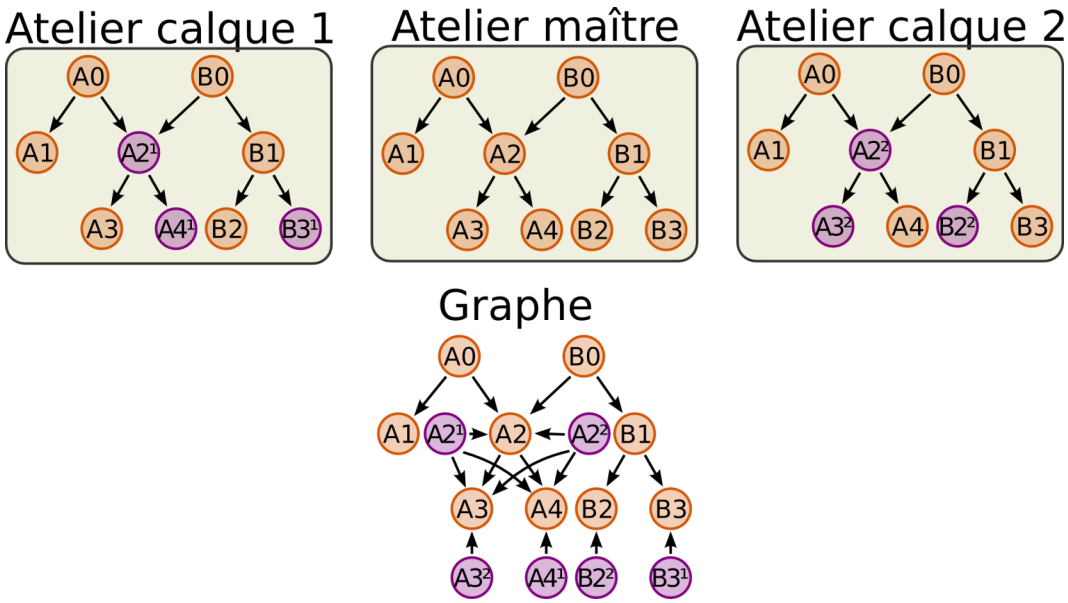


Figure 79 : deux calques en parallèle

À cet exemple, nous ajoutons deux calques multi-sources : un calque 3 dont le chemin de résolution passe par le calque 1 puis le calque 2 et un calque 4 dont le chemin de résolution passe par calque 2 puis calque 1.

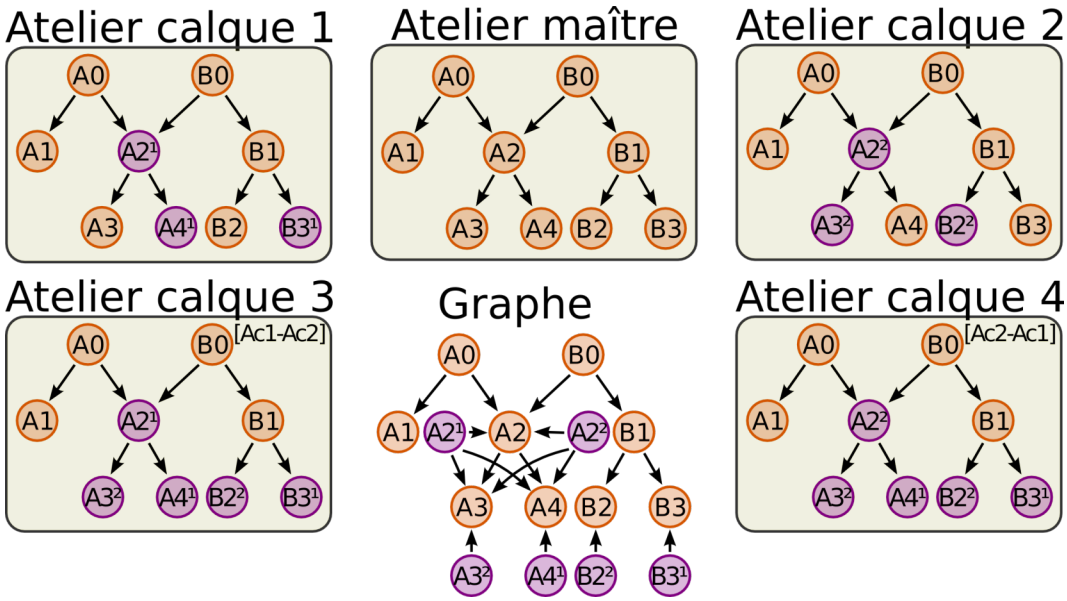


Figure 80 : ateliers calque multi-sources

Sur la figure 80, seul le fragment A2, surchargé dans les calques 1 et 2, montre réellement la différence entre le chemin de résolution de l'atelier calque 3 et l'atelier calque 4.

L'atelier calque 3 conserve en premier lieu les fragments surchargés dans l'atelier calque 1 puis, lorsque aucune surcharge n'est trouvée, l'atelier calque 3 conserve les fragments surchargés dans l'atelier calque 2. Ce comportement est similaire à des calques en série où l'atelier maître, l'atelier calque 2, l'atelier calque 1 puis l'atelier calque 3 s'enchaîneraient comme illustré sur la figure 81.

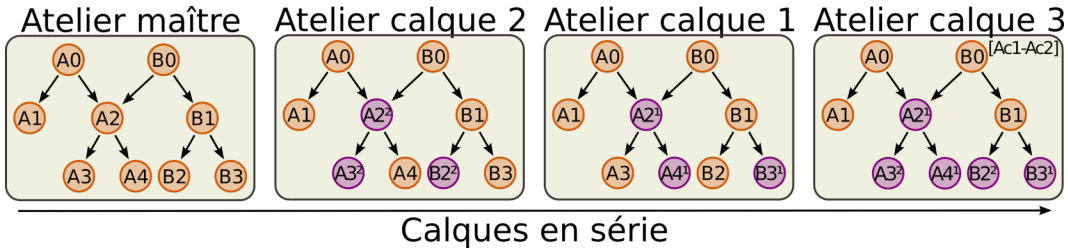


Figure 81 : équivalence, calques en série du chemin de résolution du calque 3

L'atelier calque 4 conserve en premier lieu les fragments surchargés dans l'atelier calque 2 puis, lorsque aucune surcharge n'est trouvée, l'atelier calque 4 conserve les fragments surchargés dans l'atelier calque 1. Ce comportement est similaire à des calques en série où l'atelier maître, l'atelier calque 1, l'atelier calque 2 puis l'atelier calque 4 s'enchaîneraient comme illustré sur la figure 82.

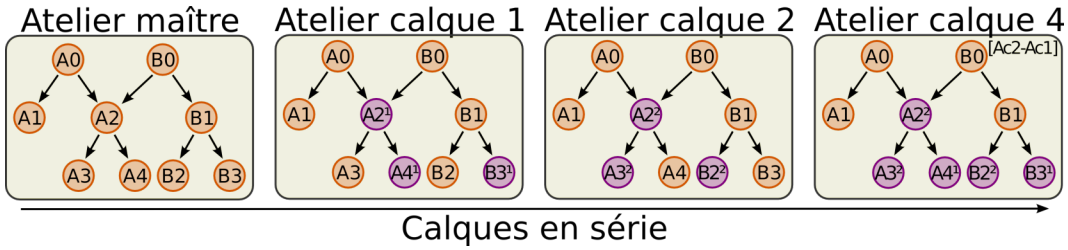


Figure 82 : équivalence, calques en série du chemin de résolution du calque 4

Fonctions d'atelier calque implémentées

Calque de travail

Fiche d'identité

- **Objet** : le calque de travail est un calque dédié à l'écriture du graphe dans un contexte protégé. Il s'agit donc d'une rééditorialisation totale convergente.
- **Direction** : convergent.
- **Multiplication des calques** : parallèle ou série.
- **Gestion des liens aux originaux** : le calque indique quels fragments ont été surchargés, supprimés ou créés dans l'atelier. Une fonction de validation permet de supprimer la surcharge et d'écraser le fragment d'origine avec le contenu du fragment surchargé et une fonction d'abandon permet de supprimer la surcharge sans écraser la source (ce qui a pour effet d'annuler les modifications locales à l'atelier calque).

La visualisation de l'état d'un fragment de l'atelier calque de travail se fait à l'aide d'un jeu d'icônes. Une icône est dédiée à chacun des 4 états possibles en fonction de la synchronisation du fragment avec la version dont il dérive dans l'atelier premier :

- fragment non surchargé ;
- fragment surchargé ;
- fragment créé dans l'atelier calque mais non existant dans l'atelier premier ;
- fragment supprimé dans l'atelier premier.

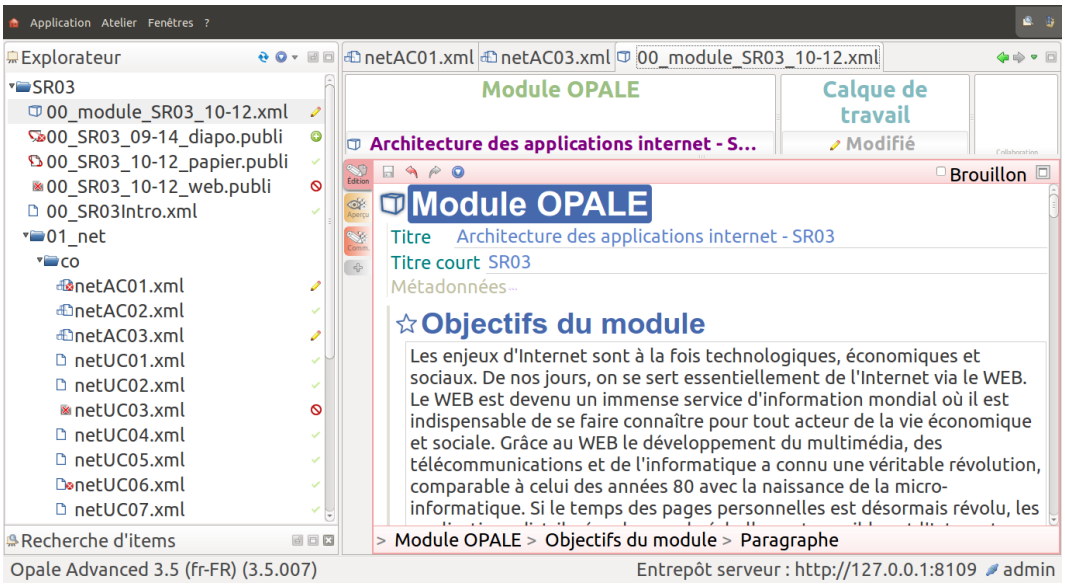


Figure 83 : copie d'écran d'un fragment édité dans un calque de travail

La figure 83 est composée de trois zones. L'arbre de gestion à gauche, le bandeau d'édition en haut et l'édition du fragment dans la zone courante. Les icônes visibles dans l'arbre de gestion permettent de connaître le statut des fragments. Le bandeau indique le titre du fragment en cours d'édition, le type d'atelier (ici, un atelier calque de travail) et le statut du fragment édité.

La surcharge d'un fragment est transparente pour le rédacteur. Seule l'icône de l'arbre de gestion ou le bandeau indique si le fragment est surchargé ou non. La simple modification et l'enregistrement d'un fragment non surchargé automatise la création d'un fragment surchargé.

Les menus contextuels de l'arbre de gestion ou du bandeau permettent de *valider* ou *abandonner les modifications* d'un fragment surchargé. Dans le premier cas, le contenu du

fragment est reversé dans l'atelier premier et la surcharge est supprimée. Dans le second cas, la surcharge est simplement supprimée.

Usages typiques

Deux usages typiques des ateliers calques de travail sont à différencier. Lorsque le maintien d'un graphe documentaire dans un état exploitable est stratégique, les modifications de maintenance peuvent être isolées dans un atelier calque de travail. Ainsi, seules des modifications déjà rédigées et terminées sont insérées dans le graphe d'origine. À l'inverse, un ou plusieurs rédacteurs peuvent chercher à isoler leur production du reste du graphe afin d'éviter que les fragments produits ne soient consultés ou réutilisés avant la fin de leur rédaction.

Calque de dérivation

Fiche d'identité

- **Objet** : le calque de dérivation est un calque divergent générique. L'enjeu est d'accompagner des modifications apportées sur l'ensemble d'un graphe.
- **Direction** : divergent.
- **Multiplication des calques** : parallèle ou multi-source.
- **Gestion des liens aux originaux** : le calque indique deux informations, la provenance du fragment et le statut de la dérivation. La provenance peut être soit locale à l'atelier dérivé (le fragment d'origine est alors un fragment fantôme), soit surchargée dans l'atelier dérivé, soit non surchargée. Le statut de la dérivation peut être soit non contrôlée (état initial), soit validée (après une validation manuelle d'un rédacteur), soit à contrôler (fragment validé modifié soit dans l'atelier dérivé, soit dans l'atelier de référence).

La visualisation de l'état d'un fragment de l'atelier dérivé se fait à l'aide d'un jeu d'icônes. Une icône est dédiée à chacun des 9 états possibles en fonction de l'origine des fragments et du statut de leur dérivation :

- non surchargé
- non surchargé, validé
- non surchargé, à contrôler
- surchargé
- surchargé, validé
- surchargé, à contrôler
- fragment local
- fragment local, validé
- fragment local, à contrôler

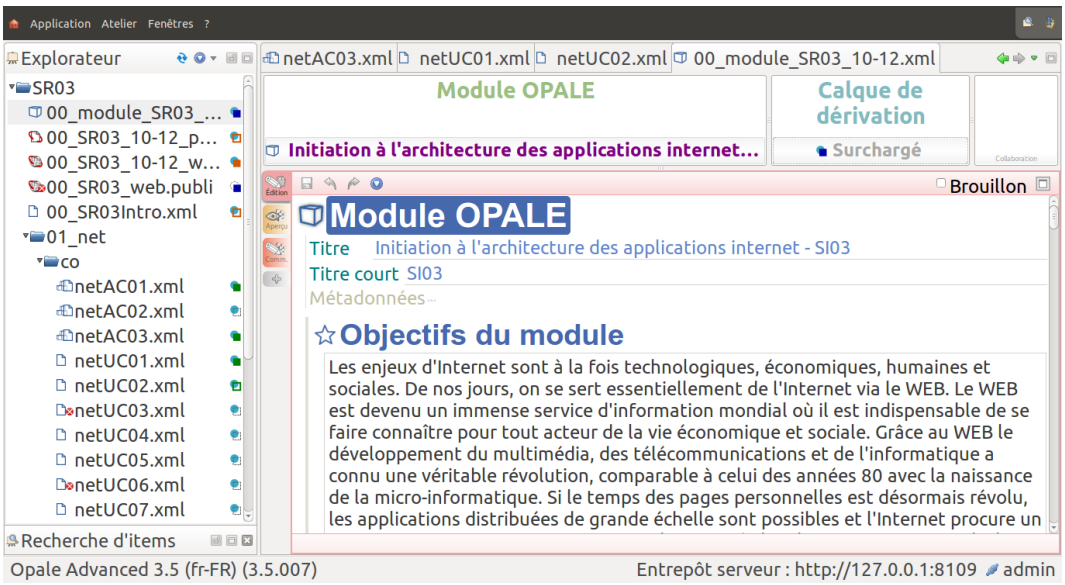


Figure 84 : copie d'écran d'un fragment édité dans un atelier dérivé

Les neufs différents états des fragments opérés dans un atelier calque sont le résultat de la combinatoire de trois types de fragments (non surchargé, surchargé et fragment local) et de trois statuts (non contrôlé, à contrôler, validé). Un fragment non surchargé est identique au fragment exploité par l'atelier premier. Un fragment surchargé est modifié par rapport à l'atelier premier et un fragment local n'existe pas dans l'atelier premier. Par défaut, tous les fragments sont non contrôlés. S'il le souhaite, par action dans un menu contextuel, un rédacteur peut valider un fragment (quel que soit son type). Dès lors, toute modification du fragment de l'atelier premier ou de sa surcharge exploitée par le calque place le fragment en état *à contrôler*. Le rédacteur doit alors valider à nouveau ce fragment par une action dans un menu contextuel.

Usages typiques

L'usage des calques de dérivation est très large. Ils sont mobilisés pour chaque dérivation opérée sur l'ensemble d'un graphe. Les deux principaux usages sont la localisation (traduction ou simple adaptation entre dialectes voisins) et la gestion de la documentation technique de familles de produits aux spécifications proches.

Autres usages possibles

Au cours de nos travaux, les deux fonctions de calques génériques ont été développées et expérimentées. Ces propositions peuvent être enrichies par des fonctions d'ateliers calques plus spécifiques afin de spécialiser le contrôle et les interactions possibles avec l'atelier premier.

Calque de traduction

Le calque de traduction constitue un cas particulier des calques divergents. L'enjeu n'est pas de produire des surcharges localisées pour adapter un graphe à un nouvel usage. Il s'agit plutôt de surcharger l'intégralité de l'ensemble des fragments.

Deux aspects des ateliers calques de dérivation classiques se montrent insuffisants pour un accompagnement approprié de la traduction. Premièrement, seule la version valide d'un fragment au sein d'un contexte d'atelier calque donné est visible. Cette vue unique est souvent insuffisante pour qualifier ou maintenir un fragment dérivé. Deuxièmement, aucune assistance au traducteur n'est apportée au sein de l'atelier. Des outils dédiés pourraient être proposés comme des dictionnaires bilingues ou des dictionnaires de synonymes. Un calque de traduction pourrait ainsi proposer de positionner côte à côte un fragment surchargé et le fragment d'origine et disposer d'outils d'accompagnement de la traduction.

Calque sous-projet

L'objectif du calque sous-projet est de proposer un calque convergent spécifique à une partie d'un arbre de gestion. L'enjeu est de restreindre la logique de l'atelier calque à une partie du graphe pour accompagner une répartition des responsabilités dans la maintenance et l'écriture.

Les calques sous-projets peuvent typiquement être utilisés lorsqu'une chaîne éditoriale est mobilisée en tant que système d'information d'une organisation. Chaque service, équipe ou projet y créé alors un atelier calque sous-projet pour y travailler et ne valide les fragments ou modifications dans l'atelier de référence qu'une fois le fragment considéré comme publique et réutilisable pour le reste de l'organisation. Un autre usage consiste dans la rédaction de documents de projets d'envergure dans lesquels les différentes équipes n'ont pas nécessairement à connaître l'ensemble du document.

9.2.3 Fragment proxy

Principe

Au sein des architectures réseau, un serveur proxy est une machine désignée comme mandataire pour assurer la connexion à un autre réseau. Toute communication depuis une machine utilisant un proxy passe donc par le serveur proxy. La notion de fragment proxy est inspirée de ce principe. Soit deux ateliers A1 et A2 qui exploitent chacun leur propre graphe documentaire (figure 85).

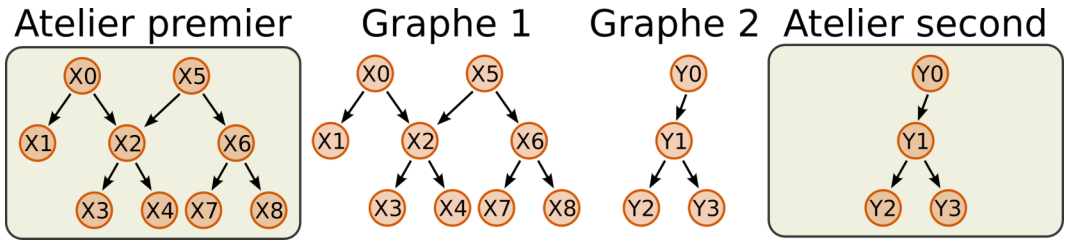


Figure 85 : deux ateliers sans proxy

Le fragment X6 du graphe exploité par A1 désigné comme proxy dans A2 donne accès à l'ensemble de son sous-réseau, soit à lui-même et aux fragments X7 et X8. Les fragments accessibles dans A2 par l'intermédiaire de X6 sont appelés des fragments proxiés.

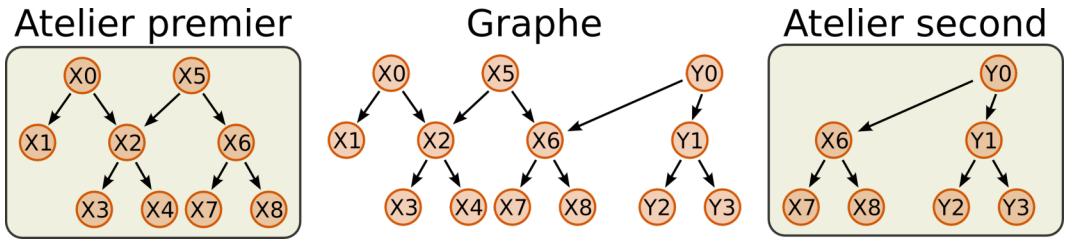


Figure 86 : deux ateliers après l'initialisation d'un fragment proxy

Les fragments proxy exploitent la même mécanique de surcharge automatique et de gestion des identifiants que les ateliers calques. Lorsqu'un fragment proxié est surchargé, une copie est automatiquement créée et liée à son fragment d'origine.

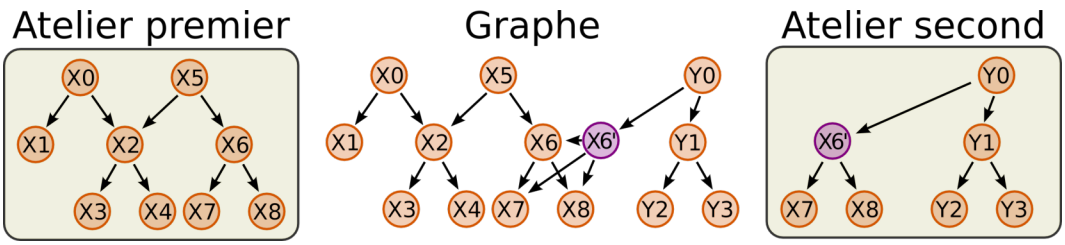


Figure 87 : deux ateliers après la surcharge d'un fragment proxié

En permettant une interconnexion entre deux graphes, les fragments proxys ne partagent pas la contrainte de mutualisation de l'arbre de gestion présente dans les ateliers calques. Afin de bien différencier les fragments issus du graphe exploité localement des fragments proxiés, les fragments proxys et leurs sous-réseau ne sont pas injectés dans l'arbre de gestion du graphe local. Un espace non répertorié dans l'arbre leur est dédié.

Multiplication de fragments proxys

La mise en place d'un fragment proxy constitue un lien entre deux fragments issus d'atelier différents. Il n'y a aucune contrainte pour multiplier les proxys entre deux ateliers.

Sur la figure 88, l'atelier second met en place trois proxys vers des fragments issus d'un même atelier premier (X1, X2 et X6).

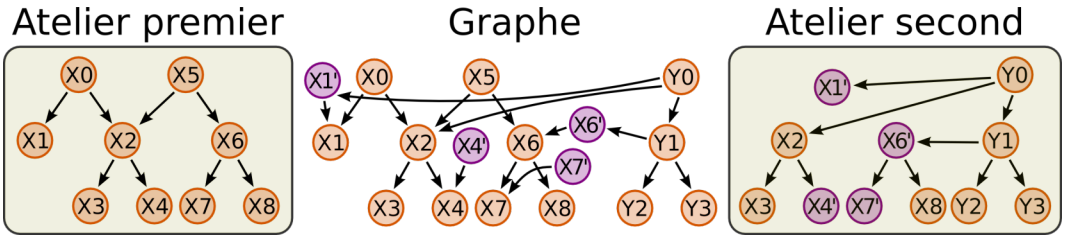


Figure 88 : trois fragments proxys entre deux ateliers

Cette multiplication peut poser problème lorsqu'un même fragment se retrouve proxié à travers deux liens proxys différents. En reprenant l'exemple illustré sur la figure 88, la question se pose lorsque les fragments X0 et X5 sont tous les deux liés comme fragment proxy. Deux comportement sont possibles pour les fragments doublement proxyés (soit X2, X3 et X4, le sous-graphe commun à X0 et X5) : la mutualisation des fragments (le sous-graphe X2, X3 et X4 est mutualisé par X0 et X5 dans l'atelier second) ou leur duplication (le sous-graphe X2, X3 et X4 est importé deux fois et les proxys X0 et X5 de l'atelier second référence chacun une version différente).

La solution la plus logique consiste à différencier le comportement en fonction de la direction du projet de rééditorialisation. Lorsqu'un projet de rééditorialisation est convergent, les fragments sont référencés dans l'atelier second avec le même identifiant que celui utilisé dans l'atelier premier. Il n'est donc pas possible de multiplier les versions d'un même fragment. En revanche lorsque le projet de rééditorialisation est divergent, un nouvel identifiant est créé pour le contexte de l'atelier second. Chaque fragment dispose ainsi d'un nouvel identifiant ce qui permet de créer plusieurs versions liées à un même fragment d'origine.

Ainsi, lorsque deux proxys convergents sont mis en place, l'atelier second mutualise les fragments selon les mêmes modalités que l'atelier premier.

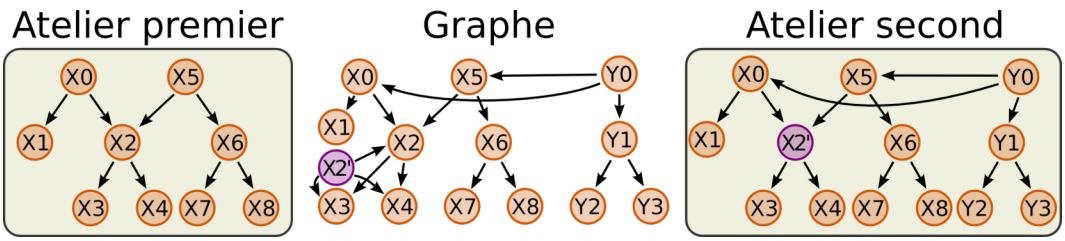


Figure 89 : deux fragments proxys convergents entre deux ateliers

Cette solution est fonctionnelle mais pose problème dans le cas où plusieurs fonctions de proxy convergentes seraient implémentées. Il est alors nécessaire de restreindre l'ensemble des fonctions de proxys convergentes entre deux ateliers à une seule et même implémentation. Le cas échéant, un même fragment issu de deux sous-graphes distincts importés par deux proxys différents exploitant deux fonctions de rééditorialisation différentes se retrouverait en conflit. Concrètement, si les deux fonctions de rééditorialisation prévoient deux méthodes différentes pour afficher l'état d'un fragment (surchargé ou identique à la source) et deux méthodes différentes pour reverser une modification locale dans le fragment original, il ne sera pas possible de déterminer quelles méthodes utiliser.

Un tel problème disparaît avec des proxys divergents car la multiplication des fragments proxys duplique les fragments proxiés plusieurs fois comme illustré sur la figure 90.

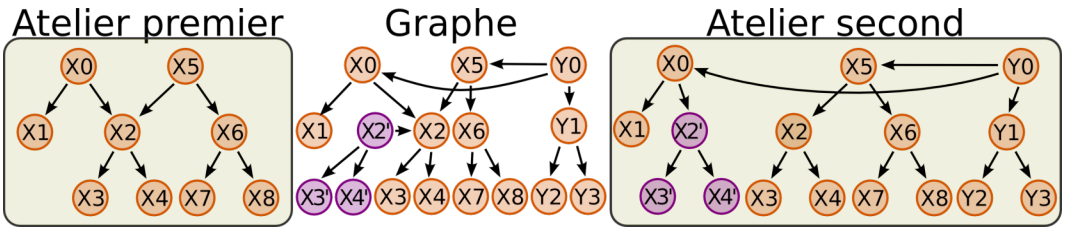


Figure 90 : deux fragments proxys divergents entre deux ateliers

Sans problèmes d'identifiants, il est possible de faire plusieurs liens proxys vers un même fragment (à partir d'une ou de plusieurs fonctions de rééditorialisation différentes, comme une traduction et une dérivation plus générique). Par exemple, la multiplication des liens peut outiller l'exemple de la documentation multilingue avec uniquement deux ateliers. Un atelier premier serait utilisé pour écrire la documentation de référence. Un atelier second ferait quatre liens proxys vers cette documentation pour permettre les quatre traductions. La scénarisation et la publication du document multilingue seraient également réalisées dans l'atelier second.

Fonctions de proxy

À ce stade d'avancement du projet de recherche initié par nos travaux de doctorat, les fonctions de proxy n'ont pas encore été développées. Nous proposons dans cette section les spécifications fonctionnelles de deux fonctions de proxy génériques : le proxy de partage, un proxy convergent pour partager des contenus entre pairs ; et le proxy de dérivation, un proxy divergent générique.

Proxy de partage

Principe

Les proxys de partage sont conçus pour mutualiser des contenus entre pairs. Par exemple, ce type de proxy peut être utilisé par des enseignants souhaitant partager des modules pédagogiques.

La fonction de proxy permet ici de faire la médiation sur des contenus théoriquement identiques entre deux projets éditoriaux différents. Malgré la volonté éditoriale de partager les contenus par transclusion, la dynamique des différents projets éditoriaux peut mettre les fragments proxiés en tension. Concrètement, imaginons deux enseignants travaillant dans la

même discipline. L'enseignant A peut proposer à l'enseignant B de réutiliser ses contenus. À l'usage, l'enseignant B souhaite modifier ces contenus afin de mieux les adapter à sa pédagogie. Dans ce contexte, il n'est pas certain que l'enseignant A souhaite récupérer les mises à jour de l'enseignant B. Le principe du proxy de partage permet d'appliquer le principe de surcharge automatique des proxys et de permettre aux deux enseignants de contrôler le versement de modification, depuis l'atelier premier vers l'atelier second ou l'inverse.

Initialisation

À l'initialisation d'un proxy de partage, les fragments proxiés depuis l'atelier premier sont classiquement rendus disponibles dans l'atelier second comme illustré sur la figure 86.

États des fragments

Sur la figure 91, les deux enseignants ont maintenu leurs contenus. Les fragments modifiés dans l'atelier premier sont marqués en vert et ceux modifiés dans l'atelier second sont marqués en violet. Il est à noter que dans le cas de proxys de partage, le système de dérivation automatique fonctionne dans les deux sens : quant un fragment est modifié dans l'atelier second et quand un fragment est modifié dans l'atelier premier. Ainsi, dans l'illustration suivante, le fragment X7 est vert car modifié dans l'atelier premier et un fragment X7', orange car identique au contenu initial, est exploité dans l'atelier second.

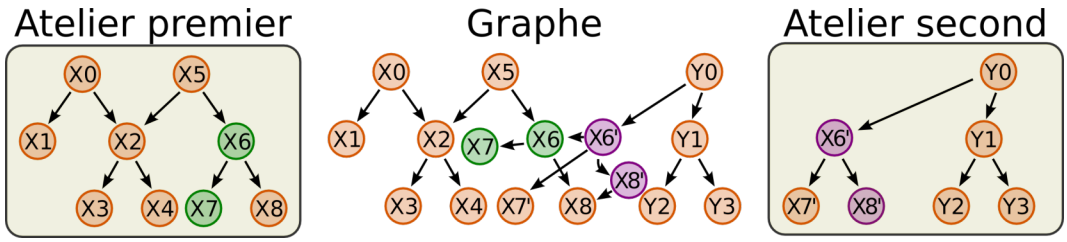


Figure 91 : deux ateliers après l'initialisation d'un fragment proxy

Le principe de surcharge automatique fonctionne dans les deux sens. Par conséquent, les deux ateliers doivent nécessairement proposer un système d'état des fragments proxiés afin de contrôler les surcharges. Cinq différents états peuvent être différenciés :

- *synchrone* : les fragments de l'atelier local (premier ou second) et distant (l'atelier complémentaire, second ou premier) sont identiques ;
- *surcharge locale* : une modification du fragment de l'atelier local a provoqué une sortie de l'état synchrone ;
- *Surcharge locale validée* : la modification locale est validée, une mise à jour du contenu est proposée dans l'atelier distant ;
- *surcharge distante* : le fragment distant a été modifié et validé, une mise à jour est disponible dans l'atelier local ;
- *en conflit* : une surcharge locale est en conflit avec une mise à jour disponible depuis l'atelier distant.

Le tableau suivant recense les états des différents fragments proxiés au sein du graphe illustré précédemment.

États	Fragments de l'atelier premier	Fragments de l'atelier second
Synchrone	Aucun	Aucun
Surcharge locale (validée ou non)	X7	X8'
Surcharge distante	X8	X7'
En conflit	X6	X6'

Mise à jour des fragments

Le versement des modifications depuis un atelier vers l'autre s'effectue en deux étapes. Lorsqu'un rédacteur fait une modification, il dispose de la possibilité de valider sa modification via les menus contextuels. Cette validation entraîne un changement d'état du fragment local (qui passe en *surcharge locale validée*) et informe le rédacteur de l'atelier distant par le changement d'état du fragment correspondant (qui passe soit de *synchrone* à *surcharge distante*, soit de *surcharge locale* validée ou non à l'état *en conflit*).

Lorsqu'un fragment est en état *surcharge distante*, le rédacteur a la possibilité de déclencher une mise à jour par le menu contextuel du fragment.

La gestion des fragments en conflit est gérée au cas par cas en comparant les différences entre les fragments locaux et distants.

Proxy de dérivation

Principe

Le proxy de dérivation est un proxy divergent générique. Il exploite les mêmes fonctions d'information et de gestion entre un fragment surchargé et sa source que le calque de dérivation (p. 124).

Les usages typiques des proxys de dérivation consistent essentiellement en la gestion de configuration d'une documentation technique (documentation de produits voisins) ou dans le partage de ressources à retravailler pour être exploitables comme deux cours sur un même sujet mais pour des niveaux différents.

Initialisation

À l'initialisation d'un proxy de partage, les fragments proxiés depuis l'atelier premier sont classiquement rendus disponibles dans l'atelier second comme illustré sur la figure 86.

Statut des fragments

La fonction de mise à jour des contenus est unidirectionnelle. Par conséquent, seuls les fragments de l'atelier second nécessitent un état particulier. L'atelier premier fonctionne quant à lui indépendamment de l'atelier second.

Comme pour le calque de dérivation, les états des fragments proxiés sont issus de la combinatoire d'un type de fragment (surchargé, non surchargé) avec un statut (non contrôlé, validé, à contrôler) :

- non surchargé
- non surchargé, validé
- non surchargé, à contrôler
- surchargé
- surchargé, validé
- surchargé, à contrôler

Comme pour le calque de dérivation, tous les fragments sont non contrôlés par défaut. S'il le souhaite, par action dans un menu contextuel, un rédacteur peut valider un fragment (quel que soit son type). Dès lors, toute modification du fragment de l'atelier premier ou dans l'atelier second place le fragment en état *à contrôler*. Le rédacteur doit alors valider à nouveau ce fragment par une action dans un menu contextuel.

Contrairement à la gestion des calques de dérivation, les fragment locaux sont exploités sans aucun lien à l'atelier premier. Par conséquent, ils ne disposent pas d'état particulier.

Mise à jour des fragments

Comme pour l'atelier calque de dérivation, les fragments non surchargés sont synchronisés avec l'atelier premier : lorsque le contenu est mis à jour dans l'atelier premier, aucun nouveau fragment n'est automatiquement créé et le contenu à jour apparaît dans l'atelier

second. Les fragments surchargés dans l'atelier second ne sont jamais mis à jour automatiquement.

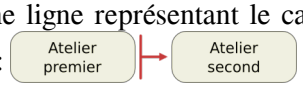
9.3 Exemples

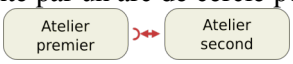
Note : convention de représentation

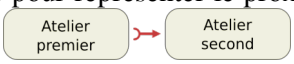
Afin de représenter des ateliers calques et fragments proxys, nous proposons les conventions suivantes :

- Un atelier sera représenté par un cadre gris :

- Un calque convergent sera représenté par une ligne matérialisant le calque et une double flèche matérialisant la convergence :

- Un calque divergent sera représenté par une ligne représentant le calque et une flèche unidirectionnelle représentant la divergence : 

- Un proxy convergent sera représenté par un arc de cercle pour représenter le proxy et une double flèche pour la divergence : 

- Un proxy divergent sera représenté par un arc de cercle pour représenter le proxy et une flèche unidirectionnelle pour représenter la divergence : 

Par exemple, la figure 92 représente cinq ateliers :

- un atelier maître contenant les versions de référence des fragments ;
- un atelier calque de travail pour la rédaction et la maintenance des fragments (à gauche) ;
- un atelier calque de dérivation (à droite) pour permettre la traduction de la documentation ;
- un atelier agrégeant les fragments issus de l'atelier maître et leur traduction par l'intermédiaire de proxys convergents (en bas)
- un atelier exploitant des proxys divergents afin de produire différentes dérivations de la documentation de référence.

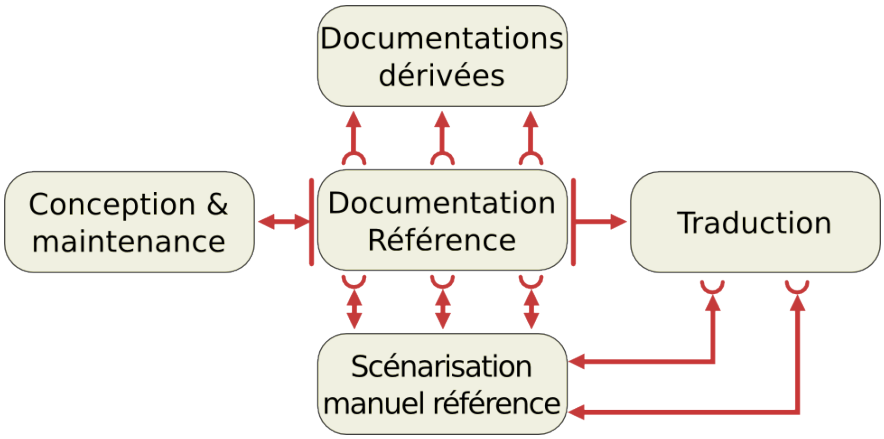


Figure 92 : exemple de représentation

Nous utiliserons une forme plus compacte pour représenter les graphes de contrôle, graphes utilisés pour le contrôle de la gestion des identifiants et présentés dans la première section de ce chapitre (p. 113). Nous supprimerons la marque proxy ou calque pour ne laisser que la double flèche de convergence.

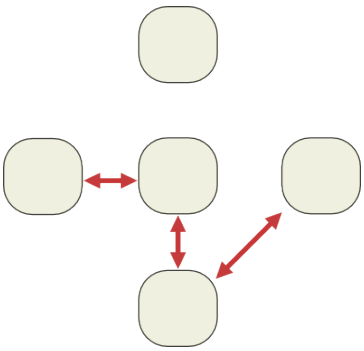


Figure 93 : graphe de contrôle

Par ailleurs, cette configuration d'ateliers n'a pas de problème de conflits puisque aucun cycle n'apparaît sur le graphe de contrôle.

9.3.1 Enaction Series

Contexte

Le processus d'édition de la collection Enaction Series se fait en deux temps. Dans un premier temps, l'ouvrage est écrit par l'auteur et retravaillé avec l'éditeur. Une fois cela terminé, une première version est alors publiée sur le site de la collection. Dans un second temps, l'ouvrage est ouvert à la glose des relecteurs et à la discussion entre l'auteur et ses relecteurs. Le contenu du livre peut alors être retravaillé par l'auteur en fonction du contenu des discussions. Une fois cette phase de discussion terminée, l'éditeur publie une version corrigée de l'ouvrage, contenant tout ou partie des commentaires.

Structuration des ateliers

La première phase d'élaboration et d'édition d'un ouvrage ne nécessite pas nécessairement une structuration des ateliers particulière. Tant que le livre n'est pas publié, il n'y a pas de besoin particulier de structuration de plusieurs ateliers. En revanche, une fois le livre publié, il est intéressant de toujours garder en l'état le graphe mobilisé pour sa publication. Ainsi, si une correction mineure est à apporter (comme par exemple, une erreur typographique à corriger), il est possible d'effectuer la correction dans le graphe et de republier une version corrigée très facilement. Par conséquent, toutes les modifications de la seconde phase de conception de l'ouvrage doivent se faire dans un contexte protégé. Nous avons outillé ce contexte avec un atelier calque de travail.

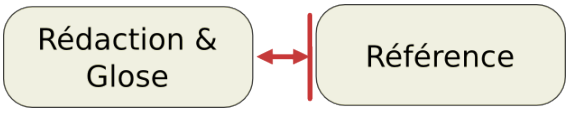


Figure 94 : organisation des ateliers de la collection Enaction Series

Lors de la première phase d'écriture et d'édition, l'auteur et l'éditeur collaborent dans l'atelier calque de travail appelé « Rédaction & Glose ». Lorsque l'édition est terminée, l'éditeur valide les contenus qui intègrent l'atelier de référence et peuvent être publiés. La phase de gloses et de réédition démarre alors dans l'atelier « Rédaction & Glose ». Une fois celle-ci terminée, les contenus seront à nouveau validés et mis à jour dans l'atelier de référence.

9.3.2 DILA

Contexte

Le graphe documentaire de la DILA est mobilisé pour publier deux documentations différentes : le site service-public.fr et la base de mémos utilisés pour répondre aux questions par téléphone.

Le graphe documentaire exploité est critique. Il doit pouvoir être publié sur le site à n'importe quel instant. Une version de référence du graphe doit donc être conservée dans un

contexte protégé des modifications en cours.

Trois équipes s'occupent de maintenir le graphe. Une équipe se charge de la maintenance des contenus dédiés à la documentation des commandes publiques, une équipe a la responsabilité de la maintenance des contenus dédiés aux particuliers et aux associations pour le site web et les mémos téléphones et la dernière équipe s'occupe de maintenir les contenus dédiés aux professionnels pour le site web et les mémos téléphones.

Les deux équipes maintenant les contenus dédiés au site web et aux mémos travaillent en forte collaboration tandis que l'équipe maintenant les contenus dédiés aux commandes publiques rédige dans un contexte plus isolé.

Structuration des ateliers

Nous avons outillé ce contexte avec trois ateliers. Un atelier de référence contenant le graphe de référence protégé de ses contextes d'édition et deux ateliers calques de travail : un atelier pour l'équipe commande publique et un atelier pour les équipes « particuliers/associations » et « professionnels ».

Figure 95 : organisation des ateliers de la DILA

Extensions possibles

Afin d'illustrer différents usages possibles de la mise en place de fonctions de structuration des ateliers, nous allons étendre cet exemple au-delà de la configuration déployée en usage industriel. Ces extensions sont inspirées d'usages plausibles au sein de la Direction de l'Information Légale et Administrative sans toutefois être réellement déployées.

La base de référence peut être mobilisée par des préfectures afin d'être réutilisée pour des publications locales. On distinguera trois usages différents : une adaptation complète de la base pour l'adapter à des spécificités locales (comme pour l'Alsace, la Lorraine ou les Territoires Ultra-Marins) ; une réutilisation partielle par une des administrations locales dépendant de la préfecture ; une adaptation partielle pour concevoir de nouveaux contenus exploitant l'information de la base.

Structuration des nouveaux ateliers

Dans ce nouveau contexte, chaque préfecture souhaitant exploiter les contenus de la base a la possibilité de faire une dérivation complète du graphe de référence. Dans un second temps, les services de l'état au sein du département ont la possibilité de récupérer des contenus à l'aide de fragments proxys afin d'être mobilisés dans de nouvelles documentations ou d'être dérivés pour de nouveaux usages.

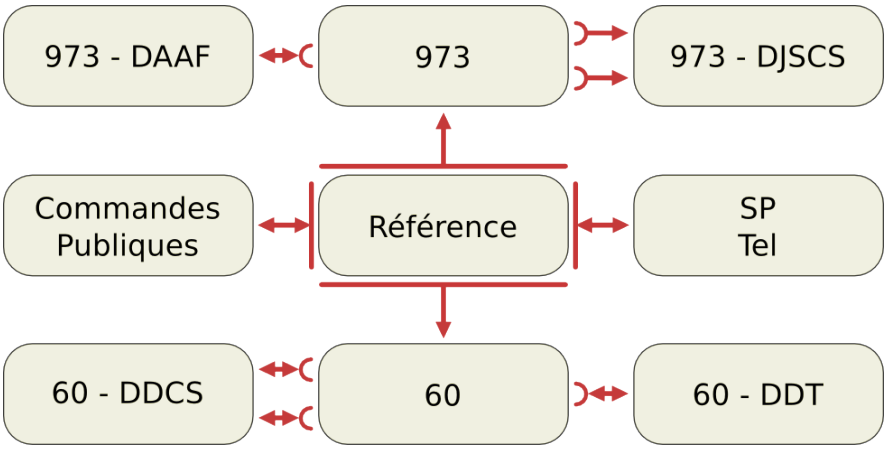


Figure 96 : exemples d'organisation avancée entre les préfectures et la DILA

La figure 96 montre une dérivation effectuée par la préfecture de l'Oise et des usages partiels de la documentation effectués par la Direction Départementale du Territoire (DDT) et par la Direction Départementale de la Cohésion Sociale (DDCS). La partie haute du schéma représente une dérivation effectuée par la préfecture de Guyane et des exploitations réalisées par la Direction de la Jeunesse, des sports et de la Cohésion Sociale (DJSCS) et par la Direction de l'Alimentation, de l'Agriculture et de la Forêt (DAAF).

9.4 Distribution des projets de rééditorialisation

L'exemple enrichi de la DILA montre un même graphe exploité par plusieurs organisations distinctes en des lieux géographiques différents. Le bon fonctionnement du système de calque ou de proxy repose sur une gestion centralisée du graphe et des différents liens entre fragments. Dans l'exemple précédent, cette gestion centralisée implique une seule et même chaîne éditoriale pour l'ensemble des préfectures et pour la DILA. Cette contrainte est peu réalisable, tant pour des raisons techniques (il paraît peu judicieux de faire travailler la préfecture de Guyane sur un serveur exploité par la DILA en métropole), qu'organisationnelle (il paraît peu probable que toutes les préfectures et la DILA mènent un projet d'équipement informatique ensemble). Pour rendre un tel exemple viable sur les plans techniques et organisationnels, il paraît judicieux de distribuer l'architecture de la chaîne éditoriale.

Deux approches coexistent pour gérer la distribution de l'architecture. Il est possible de développer un système d'adressage et de communication interne à la chaîne éditoriale Scenari ou au contraire de proposer un protocole plus générique permettant à n'importe quel système compatible de s'interfacer.

Nous distinguons deux couches dans la gestion de la distribution des projets de rééditorialisation. Une première couche est dédiée à la communication et une seconde à la logique fonctionnelle. La couche de communication détermine les modalités par lesquelles deux chaînes éditoriales vont s'adresser des messages. La logique fonctionnelle détermine l'heuristique des messages permettant aux deux systèmes de partager des projets de rééditorialisation.

Dans cette section, nous formulons une proposition de protocole, nommé *Content Interactive Delivery* (CID), pour encadrer la couche de la communication (Arribe, Crozat & Spinelli, 2014). Cette proposition constitue un préalable nécessaire mais non suffisant à la conception de chaînes éditoriales distribuées. L'enjeu de ce protocole est d'encadrer les transactions documentaires entre deux systèmes. Cette section se structure en quatre parties :

- dans une première partie, nous exposons la problématique du développement d'un protocole pour l'encadrement des transactions documentaires entre systèmes ;
- dans une seconde partie, nous exposons le seul standard au spectre fonctionnel proche, Package Exchange Notification Service (PENS) ;
- dans une troisième partie, nous présentons le protocole CID ;
- nous illustrons cette proposition par deux exemples techniques dans une quatrième et dernière partie.

Les développements du protocole CID ont été initiés dans le cadre du projet investissement d'avenir SUP E-educ. Plus d'informations sont fournies dans l'annexe dédiée à notre proposition (p. 231).

9.4.1 Problème

Trois grandes familles de systèmes participent au cycle de vie des documents :

- les systèmes de production comme les LCMS (*Learning Content Management System*), les CCMS (*Component Content Management System*) ou les chaînes éditoriales dont l'enjeu principal est d'optimiser la production des documents
- les systèmes d'exploitation pédagogiques comme les LMS (*Learning Management System*), les CMS (*Content Management System*) ou les MOOC (*Massive Open Online Course*) dont l'enjeu est de faciliter l' exploitation des documents ;
- les systèmes de gestion et d'archivage comme les logiciels de gestion électronique de

documents (GED) ou les systèmes de gestion de bibliothèque dont l'enjeu est d'accompagner le cycle de vie des documents.

Pour assurer leur fonction, ces systèmes proposent des processus fonctionnels dédiés à leur métier (comme par exemple des processus de publication, de déploiement, d'archivage ou de mise à jour). Pour être opérationnels, les systèmes déployés dans un même environnement doivent communiquer entre eux des informations (comme par exemple des métadonnées) ou directement s'échanger des documents. Ces transactions sont assurées, soit manuellement par l'usager, soit par un module de connexion développé spécifiquement pour la liaison entre deux systèmes. Dans le premier cas, les transactions manuelles sont laborieuses et sources d'erreurs. Dans le second, les contraintes techniques relatives à chaque système imposent le développement et la maintenance d'un connecteur par paire de systèmes, voire d'une version de connecteur par version de paire de systèmes.

La solution idéale serait un protocole automatisant l'ensemble du processus. Cela reviendrait à disposer d'un bouton qui, une fois cliqué, assurerait l'ensemble de la transaction, quels que soient le système source, le type de communication et le système cible. Une telle automatisation est impossible à mettre en œuvre. La représentation des documents inhérente à chaque système, les métadonnées associées ainsi que les contraintes techniques relatives au métier ne permettent pas la mise en place d'un tel protocole universel.

La médiation entre deux systèmes ne peut se faire sans l'usager. C'est sa compréhension de chacun des systèmes qui lui permet d'ajuster les processus fonctionnels utilisés. Il est le seul à savoir ce que signifie l'arrivée d'un document dans son contexte. Par exemple, lui seul sait si le transfert d'un document depuis une chaîne éditoriale vers une LMS sert à alimenter un nouveau cours (qui vient de s'ouvrir en formation continue) ou à mettre à jour un ancien cours (en formation initiale, déjà démarré).

Il y a une forte tension entre un besoin d'encadrement et d'automatisation, un contexte fortement hétérogène, et une nécessité de médiation des transactions par les usagers. Notre contribution s'intéresse à cette tension. Reformulée différemment, elle cherche à fluidifier les échanges entre systèmes documentaires hétérogènes tout en proposant une médiation optimisée de l'usager.

9.4.2 PENS, un standard pour le déploiement de ressources pédagogiques

PENS (AICC 2006) est un standard développé par l'Aviation Industry Computer-based training Committee (AICC), l'organisme américain de normalisation des technologies d'apprentissage à distance pour le domaine de l'aviation. Son objet est d'encadrer le déploiement d'un fichier SCORM (ADL) sur une plate-forme de formation à distance.

PENS constitue l'initiative la plus proche du problème exposé dans la section précédente. Son étude nous permet de déterminer si son usage peut être étendu à des contextes plus génériques que le déploiement de paquet SCORM.

Principe

PENS standardise un principe d'échanges de notifications qui permet à un logiciel de production de documents (le client) de déployer un paquet SCORM sur une plate-forme LMS (le serveur). Le client envoie une première notification au serveur contenant une URL vers une ressource produite (accessible via les protocoles HTTP, HTTPS, FTP ou FTPS), une description du contenu, les identifiants permettant de récupérer la ressource et une URL permettant au serveur d'envoyer des notifications d'avancement. Le serveur collecte le contenu et envoie une notification de collecte réussie si l'URL de notification de retour à été spécifiée. Le serveur peut alors déployer le contenu et à nouveau envoyer une nouvelle notification.

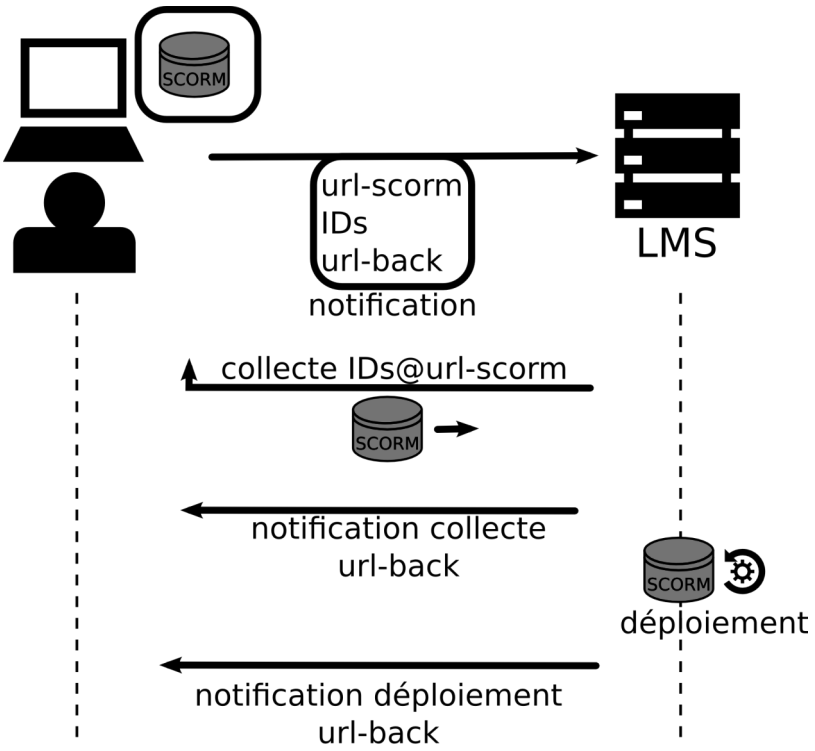


Figure 97 : exécution d'un déploiement conforme au standard PENS

Limites

Le standard PENS pose l'hypothèse d'une possible automatisation du déploiement d'un contenu SCORM à partir de sa description et des identifiants de l'usager. Nous réfutons cette hypothèse ; quand bien même son application est limitée à un spectre fonctionnel réduit, la médiation de l'usager nous semble souvent nécessaire.

Tout en considérant le spectre fonctionnel restreint du standard, les modalités techniques de PENS semblent peu judicieuses. Elles nécessitent un client et un serveur adressables sur le réseau, soit deux logiciels fonctionnant sur une machine disposant d'une IP publique. L'usage d'une plate-forme LMS implique son adressabilité sur le réseau. En revanche, il est courant que des logiciels de production soient des logiciels personnels pouvant être utilisés sur un sous-réseau invisible de la plate-forme LMS (par exemple, derrière un routeur depuis une connexion internet personnelle), ou ne soient pas conçus pour écouter les requêtes issues du réseau (comme par exemple un logiciel de bureau de type outil-auteur de contenus pédagogiques).

À l'instar de débats ayant déjà eu lieu sur les standards d'encapsulation des contenus pour en permettre l'interopérabilité (Crozat, Delestre, Qyeyrut, Baillon, Vanoirbeek Velut & Hennequin, 2006), nous soutenons que la diversité des contextes ne permet pas l'adoption d'un standard définissant un procédé de déploiement universel. Le standard a pour rôle d'accompagner la diversité des contextes et non d'imposer leur uniformisation. Un standard pour le déploiement de ressources pédagogiques doit proposer un cadre, certes universel, mais également spécialisable pour être adapté à chaque contexte.

9.4.3 Proposition CID

Présentation

L'objet du protocole CID est d'encadrer les transactions documentaires entre trois acteurs : un logiciel client à la source de la transaction ; un logiciel serveur, cible de la transaction ; un usager, à l'origine de la transaction.

Fonctionnement

Un serveur propose des processus fonctionnels. Le protocole CID permet la spécification

des processus fonctionnels dédiés à des transactions documentaires dans un manifeste. Ce fichier descriptif doit être librement téléchargeable sur un serveur HTTP.

Un client est un logiciel permettant d'instancier des transactions documentaires avec un serveur. Le client doit récupérer le manifeste via une simple requête HTTP non authentifiée. Un manifeste peut décrire plusieurs transactions. En interaction ou non avec l'usager (lien numéro 3 de la figure 98), le client doit sélectionner la transaction à effectuer et exécuter les étapes qui y sont spécifiées. Ces étapes peuvent être :

- de simples requêtes entre le client et le serveur (lien numéro 1 de la figure 98). Elles transportent alors un fichier ou des métadonnées ;
- une interaction entre l'usager et le serveur à travers une interface web envoyée par le serveur et affichée par le client (lien numéro 2 de la figure 98).

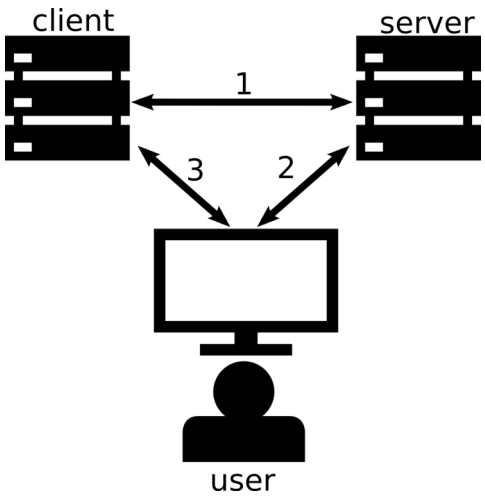


Figure 98 : CID : un protocole entre trois acteurs

Le manifeste

Le manifeste est composé de trois parties principales : une liste de processus fonctionnels, les schémas d'authentification et les spécifications de la couche transport à utiliser.

Chaque processus est composé d'une spécification des métadonnées qu'il utilise (attendues pour l'exécution ou renvoyées en résultat d'une étape) puis de l'enchaînement des étapes attendues. Les étapes peuvent être de simples échanges de métadonnées, l'envoi de fichiers vers le serveur ou des interactions directes entre l'usager et le serveur.

La partie dédiée au transport permet de lier chacune des étapes avec des modalités de transport. CID propose exclusivement l'usage du protocole HTTP pour la couche transport. La description du transport spécifie les requêtes à utiliser (GET, PUT, POST), la façon de stocker les méta-données (en paramètre de l'URL, dans l'entête ou dans le corps de la requête), la nécessité d'inclure des propriétés de session ou encore l'usage de cookies HTTP.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <cid:manifest xmlns:cid="http://www.cid-protocol.org/schema/v1/core"
  >
3   <cid:process>
4     <cid:label xml:lang="en">SCORM deployment</cid:label>
5     <cid:meta name="Content-type">
6       <cid:value>application/xx-scorm;v1.2</cid:value>
7       <cid:value>application/xx-scorm;v2004</cid:value>
8     </cid:meta>
9     <cid:upload url="http://lms.com/upload" needMetas="Content-type"
   required="true"/>
10    <cid:interact url="http://lms.com/interact" required="true"/>
11  </cid:process>
12  <cid:authentications><cid:basicHttp/></cid:authentications>
13  <cid:transports>
14    <cid:webTransport needCookies="true">
```

```
15        <cid:webInteract>
16          <cid:request method="GET"/>
17        </cid:webInteract>
18        <cid:webUpload>
19          <cid:request method="PUT" properties="queryString"/>
20        </cid:webUpload>
21      </cid:webTransport>
22      </cid:transports>
23  </cid:manifest>
```

L'exemple précédent définit un processus de déploiement de fichiers SCORM qui se déroule en deux étapes : l'envoi du fichier (*cid:upload*) et une interaction entre l'usager et le serveur (*cid:interact*). Lors de l'envoi, le client doit spécifier le type de fichier envoyé (*needMetas="Content-type"*). Le serveur n'accepte que deux types de fichiers : les archives SCORM version 1.2 et version 2004 (*application/xx-scorm;v1.2*, *application/xx-scorm;v2004*). Le fichier doit être envoyé au serveur dans une requête HTTP PUT (*cid:webUpload method="PUT"*), la métadonnée y est stockée dans l'URL sous la forme d'une *query string*. Le serveur spécifie également la nécessité de support des cookies par le client afin de compléter la transaction (*needCookies="true"*).

Pour qu'un client déploie une ressource SCORM en exploitant ce manifeste, il doit :

1. récupérer et interpréter le manifeste par une requête HTTP GET (par exemple, vers l'URL http://www.lms.com/manifest) ;
2. envoyer une requête HTTP PUT contenant le fichier (à l'URL http://www.lms.com/upload?Content-type=application/xx-scorm;v1.2) ;
3. envoyer une requête HTTP GET (vers l'URL http://www.lms.com/interact) et afficher le résultat dans une frame web.

Étape Exchange

L'étape *exchange* spécifie un simple échange de métadonnées entre le client et le serveur. Le manifeste décrit les métadonnées attendues, l'URL à utiliser pour la requête et les métadonnées retournées. Par exemple, cette étape peut-être utilisée pour tester l'authentification et les autorisations d'un utilisateur avant l'envoi d'un fichier.

Étape Upload

L'étape *upload* permet de spécifier l'envoi d'un fichier par le client à destination du serveur. Ses spécifications sont du même ordre qu'une étape *exchange*.

Étape Interact

L'étape *interact* permet de spécifier une interaction directe entre le serveur et l'usager. Le serveur envoie une page web que le client doit afficher dans une frame web. L'interaction se termine par un événement Javascript personnalisé lancé au sein de la frame (*cid-end-interaction*). Le client peut alors récupérer des métadonnées issues de l'interaction dans le corps de l'événement puis fermer la page d'interaction. Cette étape est la principale originalité de CID. Elle permet au serveur d'ajuster la transaction en médiation directe avec l'usager sans développements *ad-hoc* côté client.

Implémentations

Deux parties clientes et trois parties serveur ont été implémentées dans le cadre du projet SUP E-educ. Plus de détails sont fournis dans l'annexe sur notre contribution (p. 231) à propos des clients/serveurs génériques et du client Scenari.

Client web générique

Afin de faciliter les démonstrations et les intégrations du protocole CID, nous avons développé un client générique HTML. L'objectif de ce client est d'être utilisé avec n'importe quelle partie serveur ou d'être directement intégré au sein d'une plate-forme souhaitant disposer

d'un client.

Serveurs PHP simples

Deux serveurs PHP aux spectres fonctionnels très simples ont été développés. Le premier permet de téléverser un fichier et l'expose sur un dossier mis en ligne par le serveur web. Le second permet également de téléverser un fichier mais propose en plus une interface permettant de choisir la localisation et le nom du fichier une fois téléversé. Ces deux serveurs acceptent une authentification selon le standard Basic HTTP. Le premier est conçu avec un unique identifiant tandis que le second peut en gérer plusieurs.

Client Scenari

La cœur logiciel du client Scenari embarque un client CID. Ce client doit être appelé dans un modèle documentaire pour être exploitable dans une chaîne éditoriale. Ce client générique gère n'importe quelle configuration de serveur et peut être appelé pour effectuer diverses opérations au sein d'une chaîne éditoriale Scenari comme par exemple l'envoi d'un document produit avec Scenari ou la gestion de ressources distantes dans un DAM.

Serveur Nuxéo

Une partie serveur du protocole CID a été développée dans le logiciel Nuxéo. Le protocole est utilisé pour téléverser des documents. Des éléments spécifiques ont été ajoutés pour permettre la reconnaissance de documents-dossiers (Crozat, 2012) issus d'une chaîne éditoriale et les représenter au sein de la structure document complexe de Nuxéo.

Serveur Moodle

Une partie serveur du protocole CID a été développée au sein de la plate-forme Moodle. Son spectre fonctionnel est similaire à celui de standard PENS : elle permet le téléversement de ressources pédagogiques enregistrées au format SCORM.

9.4.4 Illustrations techniques

Dans cette section, nous proposons deux exemples simples permettant de bien comprendre le fonctionnement du protocole. Plus d'exemples peuvent être trouvés sur le site dédié au protocole *(http://www.cid-protocol.org)*.

Téléversement de fichier

Contexte

Le téléversement de fichier constitue l'exemple le plus simple. Il est constitué d'un serveur CID utilisé pour téléverser et entreposer des fichiers. C'est ce manifeste qui est utilisé par le plus simple des serveurs CID PHP.

Manifeste

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <cid:manifest xmlns:cid="http://www.cid-protocol.org/schema/v1/core"
  >
3   <cid:process is="http://schema.org/SendAction">
4     <cid:label xml:lang="en">Files deposit</cid:label>
5     <cid:label xml:lang="fr">Dépot de fichiers</cid:label>
6     <cid:doc xml:lang="en">A simple remote repository.</cid:doc>
7       <cid:meta name="File-name" cardinality="?" is=
  "http://purl.org/dc/elements/1.1/title">
8         <cid:label xml:lang="fr">Nom du fichier</cid:label>
9         <cid:label xml:lang="en">File name</cid:label>
10        <cid:doc xml:lang="en">
11          The file-name will be used in the public url of the file
12          Example: http://.../1d57fe87zr45sdz796m/monImage.png
13        </cid:doc>
```

```
      </cid:meta>
      <cid:meta name="Public-url" is="http://schema.org/URL"/>
      <cid:upload url="http://example.com/upload.php" useMetas="File-name" returnMetas="Public-url" required="true"/>
  </cid:process>

  <cid:authentications>
    <cid:basicHttp/>
  </cid:authentications>

  <cid:transports>
    <cid:webTransport>
      <cid:webUpload>
        <request method="PUT"  properties="header queryString"/>
        <request method="POST" properties="header queryString"/>
        <request method="POST;multipart/form-data"  properties="header queryString post"/>
      </cid:webUpload>
    </cid:webTransport>
  </cid:transports>
</cid:manifest>
```

Exécution du processus

Étape 1. Le serveur doit exposer le manifeste affiché ci-dessus sur une URL accessible à l'aide d'une simple requête HTTP GET non authentifiée.

Étape 2. Le client doit télécharger et interpréter le manifeste.

Étape 3. Le client exécute les étape définies par le manifeste. Dans le cas présent, il s'agit d'une simple requête HTTP de téléversement. Par exemple, le client peut envoyer le document dans le corps d'une requête HTTP PUT contenant la métadonnée *File-name* dans l'entête de la requête. Le corps de la réponse du serveur contiendra un objet JSON contenant la métadonnée *Public-url*.

Déploiement d'un fichier SCORM

Contexte

Dans ce contexte d'usage, le serveur CID est une plate-forme d'exploitation de documents pédagogiques qui accepte les fichiers SCORM. Le protocole est utilisé pour déployer un fichier SCORM depuis un logiciel de production documentaire tel que la chaîne éditoriale Opale. Ce contexte d'usage correspond au spectre fonctionnel du protocole PENS.

Manifeste

```
<?xml version="1.0" encoding="UTF-8"?>
<cid:manifest xmlns:cid="http://www.cid-protocol.org/schema/v1/core" >

  <cid:process is="http://schema.org/SendAction">
    <cid:label xml:lang="fr">SCORM deployment</cid:label>
    <cid:restriction name="Zip-file-names-encoding" value="UTF-8">
      <cid:label xml:lang="en">Encoding of the files-name</cid:label>
    </cid:restriction>
    <cid:meta name="Content-type" is="http://purl.org/dc/elements/1.1/type">
      <cid:value>application/xx-scorm;v1.2</cid:value>
      <cid:value>application/xx-scorm;v2004</cid:value>
    </cid:meta>
    <cid:upload url="http://example.com/upload?response=noCookies" needMetas="Content-type" required="true"/>
```

```
14     <cid:interact url=
   "http://example.com/interact?response=noCookies" required="true"/>
15   </cid:process>
16
17   <cid:authentications>
18     <cid:basicHttp/>
19   </cid:authentications>
20
21   <cid:transports>
22     <cid:webTransport>
23       <cid:webInteract>
24         <request method="GET" properties="header queryString"/>
25         <request method="POST;application/x-www-form-urlencoded"
   properties="queryString header post"/>
26         <request method="POST;multipart/form-data" properties=
   "header queryString post"/>
27       </cid:webInteract>
28       <cid:webUpload>
29         <request method="PUT"  properties="header queryString"/>
30         <request method="POST" properties="header queryString"/>
31         <request method="POST;multipart/form-data"  properties=
   "header queryString post"/>
32       </cid:webUpload>
33     </cid:webTransport>
34   </cid:transports>
35 </cid:manifest>
```

Exécution du processus

Étape 1. Le serveur doit exposer le manifeste affiché ci-dessus sur une URL accessible à l'aide d'une simple requête HTTP GET non authentifiée.

Étape 2. Le client doit télécharger et interpréter le manifeste.

Étape 3. Le client exécute les étape définies par le manifeste.

- La première étape consiste à envoyer le fichier SCORM selon une des modalités de transport définies par le manifeste. Selon la description des restrictions imposées par le serveur CID, les noms de fichiers de l'archive SCORM doivent nécessairement être encodés en UTF-8. Dans cet exemple, le client peut envoyer l'archive dans un formulaire au sein d'une requête HTTP POST. Le protocole CID impose d'utiliser la clé *cidContent* pour stocker le fichier téléversé dans le formulaire. La métadonnée demandée (le type de fichier SCORM) peut également être transmise dans le formulaire. Le nom de la méta-donnée est alors utilisé comme clé.
- La seconde étape du déploiement consiste en une interaction entre le serveur et l'usager. Cette étape permet à l'usager de directement déployer son archive dans l'activité d'apprentissage voulue. Cette interaction est une page web envoyée par le serveur et affichée dans une *frameweb* par le client.

Chapitre 10

Vers une industrialisation de la rééditorialisation documentaire

10.1 Méthodologie 144
10.1.1 Structurer les projets de rééditorialisation 144
10.1.2 Documentariser l'activité 146
10.2 Exemples 147
10.2.1 Chaîne éditoriale pour Quick 147
Structure des ateliers 147
Documentarisation de l'activité 150
Dossier d'homologation équipement 150
Dossier d'homologation produit 151
Documentation technique de référence 152
Rédaction du référentiel 153
Vue d'ensemble 154
10.2.2 Chaînes éditoriales pour l'enseignement supérieur 154
Structure des ateliers 155
Documentarisation de l'activité 159
Modèle enseignant 159
Modèle TICE 160
Vue d'ensemble 162

Dans le chapitre 8 (p. 87), nous avons montré comment prendre en compte le discours pragmatique dans la modélisation des fragments et comment utiliser les éléments constitutifs de ce discours pour construire des fonctions d'assistance à l'action. En enregistrant le discours de l'action, cette évolution des fragments documentaires permet de documentariser l'activité sur le graphe et simplifie ainsi le contrôle de sa cohérence. Plus que les éléments présentés, ce chapitre propose une approche pour assister les rédacteurs dans la gestion de leur graphe. Les propositions, expérimentations et exemples qui accompagnent cette approche ne visent pas un

objectif de généricité ou d'universalité dans la modélisation et l'exploitation du discours pragmatique. De nombreuses nouvelles modélisations propres à des contextes métiers particuliers et de nouvelles exploitations sont à expérimenter afin de continuer à affiner la documentarisation de l'activité et ainsi, l'assistance au contrôle du graphe.

Dans le chapitre 9 (p. 111), nous avons montré comment exploiter la notion de projet de rééditorialisation pour simplifier le graphe perçu par les rédacteurs en le structurant dans différents ateliers. Nous avons présenté quatre catégories de projets de rééditorialisation qui s'agencent les uns aux autres. L'enjeu est d'isoler chacun de ces projets afin de proposer des environnements de travail simplifiés leur étant dédiés. Chacun de ces projets peut être implémenté de plusieurs façons en fonction de la rééditorialisation à accompagner. Au même titre que le chapitre précédent, ce chapitre pose avant tout une approche basée sur une structuration des ateliers en fonction des projets de rééditorialisation. Les fonctions particulières qui lient ces projets entre eux ne sont que des exemples d'implémentations. De nouvelles fonctions propres à des contextes d'usage métiers particuliers devront encore être expérimentées pour tendre vers l'exhaustivité des usages.

Dans ce chapitre, nous montrons comment agencer ces deux approches. Il s'agit avant tout d'une méthodologie pour la conception des chaînes éditoriales. Elle se structure en deux étapes : une première étape d'analyse du contexte d'usage d'une chaîne éditoriale afin de mettre en exergue les différents projets de rééditorialisation et la structure des ateliers qui en découle ; une seconde étape d'analyse de l'activité au sein de chacun des projets et de modélisation du discours pragmatique afin de documentariser l'activité. Nous soutenons l'hypothèse qu'une fois agencées ainsi, la simplification des graphes perçus et la documentarisation de l'activité outillent les rédacteurs pour maintenir la cohérence du graphe, quelle qu'en soit la complexité.

10.1 Méthodologie

10.1.1 Structurer les projets de rééditorialisation

La structuration des projets de rééditorialisation se mène en deux temps.

1. En premier lieu, il convient de structurer le graphe afin de répondre aux enjeux éditoriaux du contexte de production documentaire. Cela revient à concevoir une structure proposant un atelier par projet éditorial.
2. Ensuite, cette structure est complétée par les ateliers nécessaires à la prise en compte des projets de rééditorialisation auctoriaux. Cela revient à mettre en place des ateliers seconds permettant la distribution de production documentaire et la protection des projets éditoriaux qui le nécessitent.

Analyse et structuration éditoriale

L'analyse éditoriale détermine la liste des projets éditoriaux. Son enjeu est de recenser les documents et catégories de documents qui seront produits par une chaîne éditoriale et les liens de rééditorialisation que ces documents tissent entre eux. Par exemple, il est utile de déterminer quelle catégorie de documents exploite les fragments produits par une autre ou quelle catégorie de documents dérive d'une autre. À partir de cette analyse, la méthodologie de structuration des ateliers s'effectue en quatre étapes ordonnées.

1. La première étape consiste à identifier les **projets éditoriaux principaux** et à mettre en place un atelier pour chacun d'entre eux. Un projet éditorial est considéré comme principal lorsque la documentation qu'il produit n'est pas intégralement dérivée d'un autre projet. Au contraire, ses fragments peuvent être mobilisés pour être dérivés dans d'autres projets éditoriaux. Lors de l'initialisation, ces ateliers sont indépendants les uns des autres (aucune fonction de calque ou de proxy ne les lie).

   Si les projets éditoriaux principaux entretiennent des liens de rééditorialisation (c'est-à-dire, si une partie des fragments sont partagés ou dérivés depuis un projet éditorial vers un autre), il convient alors de mettre en place les fragments proxys nécessaires à ces rééditorialisations.
2. La seconde étape consiste à outiller les projets éditoriaux qui rééditorialisent l'intégralité

des fragments d'un des projets principaux. Nous proposons d'appeler ces projets des **projets éditoriaux dérivés**. Ces projets sont outillés par des ateliers calques divergents. Nous considérons un atelier calque nécessaire si et seulement si l'ensemble des contenus et l'organisation de la gestion des fragments doivent être partagés entre deux projets éditoriaux. Le cas échéant, il est plus judicieux de partager une portion des fragments à l'aide de proxys comme expliqué dans l'étape 3.

3. La troisième étape de la structure éditoriale consiste à outiller les **projets éditoriaux satellites** : les projets non centraux dans l'organisation de la documentation mais qui nécessitent malgré tout la rééditorialisation d'une partie du graphe. Ces projets sont outillés dans des ateliers initialement isolés de la structure existante avant d'être liés à d'autres ateliers par l'intermédiaire de fragments proxys.
4. La quatrième et dernière étape consiste à outiller les projets éditoriaux composés principalement de l'agrégation d'autres projets : logiquement appelés des **projets éditoriaux d'agrégation**. Lorsque l'agrégation nécessite la dérivation de certains fragments ou la rédaction de nouveaux fragments originaux (comme par exemple une introduction ou une mise en contexte), il convient de l'outiller par un atelier dédié exploitant des fragments proxy. Lorsque le projet éditorial se compose exclusivement de l'agrégation de documents produits par la chaîne éditoriale, des stratégies externes peuvent être mises en place (comme un outil d'agrégation de documents PDF ou une plate-forme web d'agrégation et d'exposition de documents). Ces solutions sortent alors du spectre des chaînes éditoriales (et n'apparaîtront donc pas dans la suite de ce mémoire).

Analyse et structure auctoriale

L'analyse auctoriale détermine la liste des contraintes liées à l'édition du graphe. Ces contraintes sont de deux ordres. Elles peuvent être liées à la protection du graphe (protection d'une portion du graphe contre des modifications en cours de rédaction ou protection des modifications en cours de rédaction contre un usage dans le reste du graphe). Elles peuvent également être liées aux dynamiques d'organisation dans la rédaction. Notre proposition consiste à mettre en place des modèles d'activités au plus proche des usages de chaque équipe de rédacteurs. Il convient donc d'isoler un atelier par rédacteur ou équipe de rédacteurs organisant sa production de façon particulière.

Suite à cette analyse, la structuration des ateliers se mène en deux temps.

1. Il convient d'abord de mettre en place les **projets auctoriaux de protection**, soit les ateliers calques de travail ou les ateliers seconds liés par des fragments proxys.
2. Les **projets auctoriaux d'équipe**, soit les projets dédiés à isoler le contexte de production d'une équipe afin de disposer d'un modèle d'activité dédié. Ces projets peuvent également être outillés par des ateliers calques de travail ou des ateliers seconds liés par des fragments proxys.

Notion de projet

La notion de projet (*éditorial*, *auctorial* ou *de rééditorialisation*) est structurante dans notre approche d'organisation du graphe. Elle est mobilisée dans le chapitre 9 (p. 113) afin de définir une typologie des projets de rééditorialisation. Elle est à nouveau utilisée au sein d'une approche fonctionnelle dans cette section, afin de proposer une méthodologie de structuration du graphe.

Le tableau suivant montre le croisement des deux approches : une case verte y signifie que la catégorie technique de projet de rééditorialisation peut être utilisée pour accompagner le projet éditorial ou auctorial correspondant.

		Approche fonctionnelle					
		Projet éditorial				Projet auctorial	
		Projet principal	Projet dérivé	Projet éditorial satellite	Projet d'agrégation	Projet de protection	Projet d'équipe
Approche technique Types de projets de rééditorialisation (p. 113)	Projet convergent partiel exemple : fragment proxy de partage (p. 128)	✓	✗	✗	✓	✓	✓
	Projet convergent total exemple : atelier calque de travail (p. 123)	✗	✗	✗	✗	✓	✓
	Projet divergent partiel exemple : fragment proxy de dérivation (p. 130)	✓	✗	✓	✓	✗	✗
	Projet divergent total exemple : atelier calque de dérivation (p. 124)	✗	✓	✗	✗	✗	✗

10.1.2 Documentariser l'activité

La documentarisation de l'activité s'appuie sur une modélisation du discours pragmatique au plus proche des usages d'un rédacteur ou d'une équipe de rédacteurs. Le discours pragmatique est modélisé au sein de modèles d'activités, compléments au modèle documentaire. Pour exploiter sa section du graphe, chaque atelier dispose de son propre modèle d'activité. Il est donc possible d'ajuster la modélisation de chacun des modèles aux discours pragmatiques utilisés par les rédacteurs qui y travaillent afin de proposer des structures au plus proche des habitudes de travail des rédacteurs.

En fonction de la structure préalablement posée, certains ateliers seront utilisés par des rédacteurs pour y rédiger des fragments tandis que d'autres ne feront que stocker une partie du graphe. Ce sera majoritairement le cas des ateliers d'agrégation ou des ateliers maîtres reliés à un atelier calque de travail. Ces structures indiquent que la rédaction est déportée au sein d'un autre atelier. Seuls les ateliers dans lesquels les fragments sont réellement rédigés et maintenus

nécessitent un modèle d'activité.

L'analyse d'un contexte doit permettre de répondre à deux questions : comment l'activité est-elle formalisée par les rédacteurs et quels sont les besoins de documentarisation.

La formalisation de l'activité dépend de l'organisation des rédacteurs. Il s'agit de définir comment les rédacteurs se coordonnent entre eux pour intervenir sur le graphe. Par exemple, les rédacteurs peuvent utiliser des documents régissant l'organisation ou simplement se répartir certaines responsabilités ou encore mobiliser un cycle de vie sur les fragments. Cette formalisation doit aboutir à un modèle mobilisant des structures documentaires classiques accompagnées des différents éléments présentés dans le chapitre 08 (p. 87).

Pour le besoin de documentarisation, il s'agit de déterminer quel est l'usage de l'information générée par l'usage du modèle de discours pragmatique. L'information peut simplement être exploitée pour assister les rédacteurs de l'atelier. À l'inverse, elle peut être diffusée à travers l'ensemble des ateliers afin d'assister l'ensemble de la chaîne. Cette différence dans la portée de l'exploitation a un impact important sur le modèle. En effet, les champs spécifiques au support de l'action que nous avons mis en exergue et qui permettent l'enregistrement de tâches ou d'informations pragmatiques adossées aux fragments documentaires classiques ont été conçus pour être exploités dans un et un seul atelier. Lorsque des fragments sont partagés via un calque ou un proxy, ces informations restent stockées dans l'atelier d'origine sans être diffusées dans le reste de la chaîne éditoriale. Il convient donc de combiner cette approche avec des structures documentaires plus classiques afin de faire circuler l'information au sein des différents ateliers.

10.2 Exemples

Les deux exemples mobilisés dans cette section permettent de bien comprendre la finalité de nos propositions : une documentarisation locale de l'activité propre à un atelier couplée à une structuration globale du graphe au sein de différents ateliers.

L'exemple de chaîne éditoriale pour Quick est inspiré des travaux de re-conception de la chaîne, menés actuellement par Kelis. Le second exemple est inspiré de nos travaux au sein du projet SUP E-educ.

10.2.1 Chaîne éditoriale pour Quick

Structure des ateliers

Projets éditoriaux

La chaîne éditoriale utilisée par la société Quick est principalement utilisée pour la production de deux documentations différentes. Le département innovation produit des dossiers d'homologation dédiés à de nouveaux équipements ou à de nouvelles recettes. Le département I2M produit le référentiel en réutilisant et en surchargeant les fragments des dossiers d'homologation ayant été validés.

Deux versions du référentiel sont éditées : une pour la Belgique et une seconde pour la France. Ces deux versions ne partagent pas nécessairement la même structure (certaines procédures sont en usage dans un seul pays). En outre, la version belge doit être éditée en deux langues (néerlandais et français).

Projets principaux

Nous proposons de mettre en place deux projets principaux. Le premier est dédié aux dossiers d'homologation. Il contient l'ensemble des dossiers d'homologation ayant été rédigés par la société. Le second est le projet de reprise et d'enrichissement des fiches issues des dossiers d'homologation. Ce projet vise à rassembler et enrichir dans un même atelier les fiches qui seront ensuite mobilisées dans le référentiel français ou belge.

Deux ateliers doivent donc être mis en place. Lorsqu'un nouveau dossier d'homologation est validé et que ses procédures sont mises en exploitation dans les magasins, un fragment proxy est initié afin de mettre à disposition les procédures dans l'atelier de *référence technique*. Ces

proxys sont convergents car toutes modifications locales à l'atelier *référence technique* (composées principalement d'enrichissement des procédures par des images et de reformulations pédagogiques) ont vocation à être reversées dans la base des dossiers d'homologation.

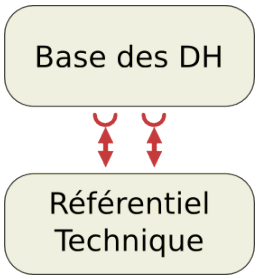


Figure 99 : projets principaux - Quick

Projets de dérivation

La documentation technique de référence est dérivée pour chacun des projets de publication. On distingue donc un premier atelier calque de dérivation pour la version française du référentiel, un second pour la version belge francophone et un troisième pour la version belge néerlandophone.

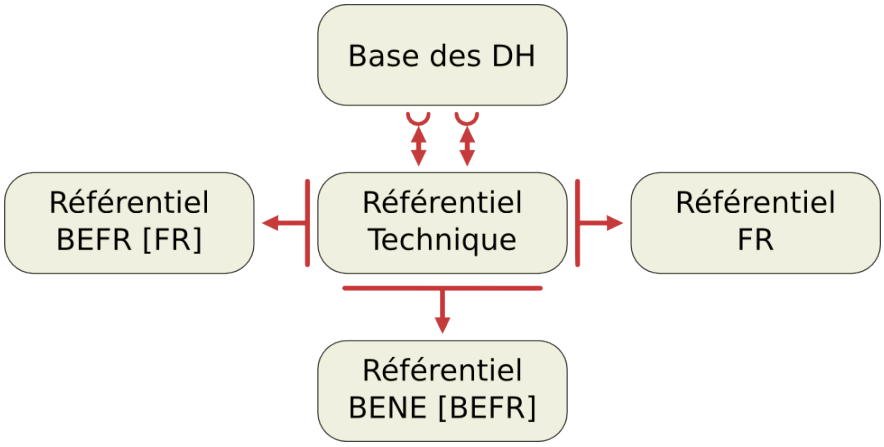


Figure 100 : ajout des projets de dérivation - Quick

À noter que nous avons indiqué sur la figure 100 les chemins de dérivation sur les ateliers dédiés au référentiel belge. La version néerlandaise hérite donc de l'ensemble des modifications de la version belge francophone qui hérite elle même de la version française.

Projet d'agrégation

La version bilingue belge du référentiel est constituée par agrégation des deux versions monolingues belges.

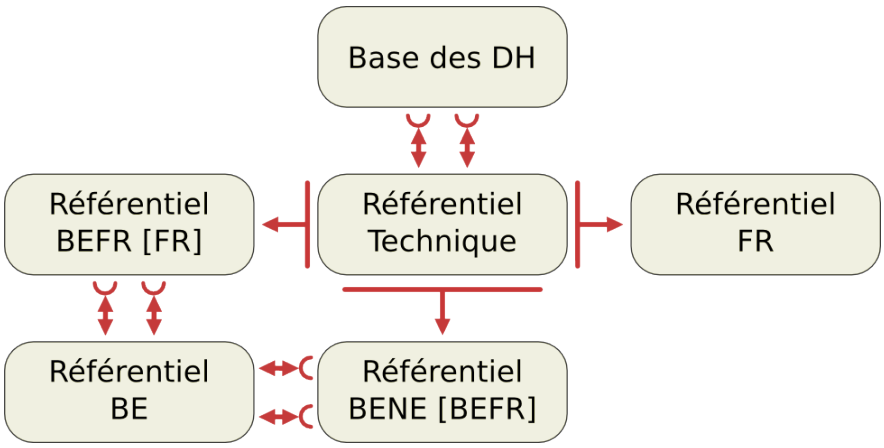


Figure 101 : ajout du projet d'agrégation - Quick

Organisation auctoriale

Les différentes documentations produites par Quick n'ont pas à être publiables en tout instant. Par conséquent, il n'est pas nécessaire de mettre en place un atelier calque afin de protéger les fragments prêts à être publiés des fragments en cours de rédaction, aucun atelier calque ou fragment proxy convergent n'est nécessaire.

Néanmoins, deux équipes travaillent sur la base des dossiers d'homologation. Chacune de ces équipes a ses propres modalités d'organisation et de validation. Afin de proposer des modèles d'activité dédiés, nous proposons de séparer l'activité de chacune des équipes.

Projets auctoriaux satellites

Deux ateliers sont donc mis en place pour la rédaction des dossiers d'homologation. Les équipes ne travaillant pas sur les mêmes contenus, il n'est pas nécessaire de mettre en place des ateliers calques de travail. L'usage d'un proxy convergent par dossier en cours de rédaction suffit à outiller correctement la production documentaire.

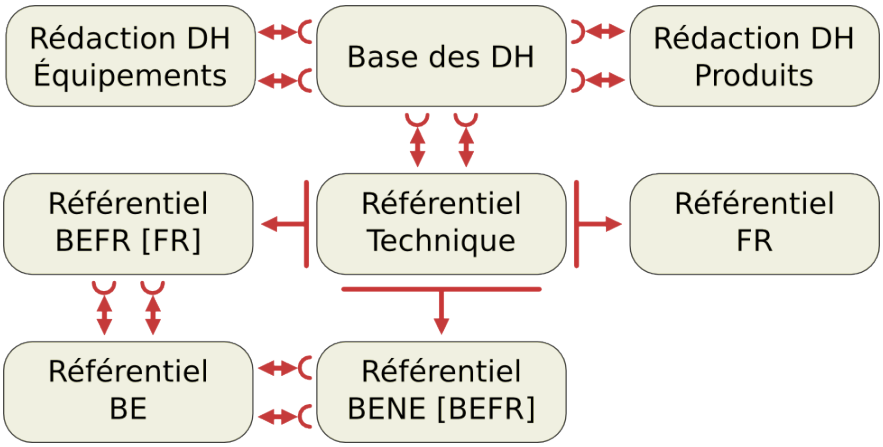


Figure 102 : ajout des projets auctoriaux - Quick

Graphe de contrôle

Le graphe de contrôle de la structure ne montre aucun conflit d'identifiant.

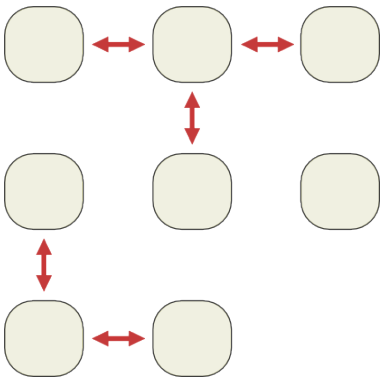


Figure 103 : Graphe de contrôle - Quick

Documentarisation de l'activité

Dans cette structure, deux ateliers n'hébergent aucune activité de rédaction : l'atelier principal dédié aux dossiers d'homologation et l'atelier permettant l'agrégation des deux versions belges du référentiel.

Les dossiers d'homologation sont exclusivement édités dans les deux ateliers prévus à cet effet (produit et équipement) ou dans l'atelier contenant toutes les fiches de référence technique. Les modifications opérées sur le référentiel belge sont quant à elles issues des ateliers belges francophones et néerlandophone.

Dans cette section, nous proposons ainsi quatre modèles d'activité dédiés respectivement :

- aux ateliers de rédaction des dossiers d'homologation (un pour l'équipe rédigeant les dossiers *équipement* et un autre pour l'équipe *produit*),
- à l'atelier de référence technique,
- aux ateliers de rédaction du référentiel.

Dossier d'homologation équipement

Le processus de rédaction des dossiers d'homologation dédiés aux équipements fait intervenir plusieurs relectures et plusieurs validations formelles (validation générale, technique, qualité, méthode et organisation, sécurité ou comité de décision). Ces validations doivent apparaître sur les documents publiés, à l'issue du processus éditorial.

Nous proposons un modèle d'activité contenant deux éléments : un système de relecture et validation s'appuyant sur les commentaires d'une part et la documentarisation des validations de l'autre.

Relecture des dossiers

La relecture d'un dossier d'homologation est effectuée par un expert métier. Elle doit s'effectuer sur le document tel qu'il est publié en fin de processus et non sur un ensemble de fragments.

Pour outiller ce besoin, nous proposons une publication dynamique de relecture embarquant le système de commentaires en ligne. Ainsi, les experts ont la possibilité d'ajouter un commentaire général de validation du document et des commentaires localisés lorsque nécessaire.

Documentarisation des validations

Les validations des dossiers d'homologation d'équipement doivent apparaître sur la page de garde des documents finaux. Il est donc nécessaire d'inscrire ces validations dans le modèle documentaire.

Une fois modélisées et générées, ces structures peuvent être éditées au sein des fragments *dossier d'homologation*.

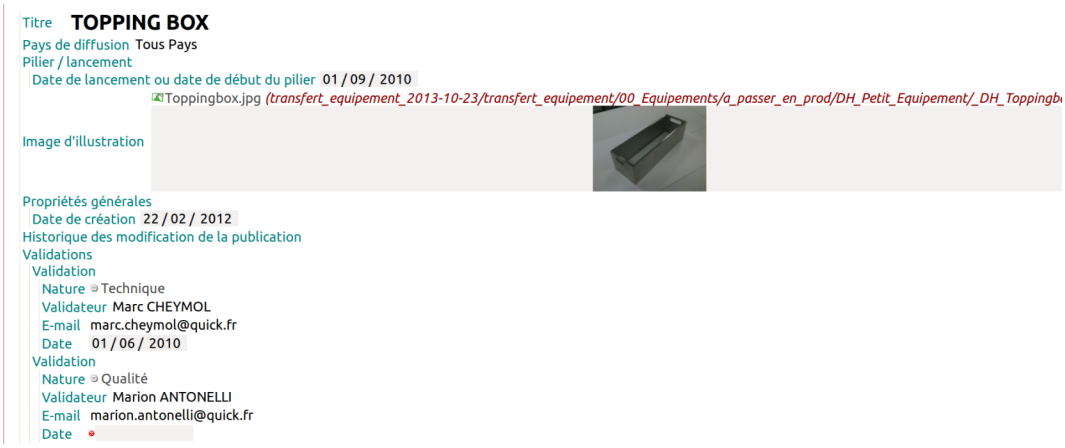


Figure 104 : édition des validations d'un dossier d'homologation

Une fois publiées, les validations apparaissent sur la page de garde du document.

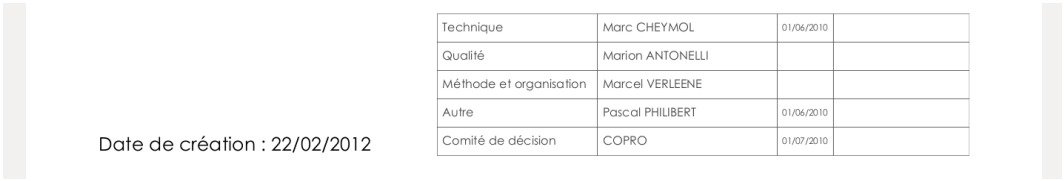


Figure 105 : publication des validations d'un dossier d'homologation

Enrichissement des fragments pragmatiques

La validation de fragments constitue un exemple typique de possible enrichissement du principe de documentarisation de l'activité. Dans le modèle tel qu'il est proposé ici, ces validations sont inscrites dans le document par des inscriptions d'ordre purement documentaire (extension du modèle de document et de ses algorithmes de publication).

Étendre le principe de documentarisation de l'activité à la validation de fragments revient à enrichir le principe de responsabilité ou de cycle de vie d'un fragment par un nouveau champ de validation. Une fonction de validation peut être instanciée et mise à disposition au sein des publications web dynamiques. Ainsi, en relisant et commentant un document, un expert pourrait aisément valider un dossier (par exemple, en utilisant bouton disponible dans la publication dynamique) et inscrire cette validation dans le document.

Dossier d'homologation produit

L'équipe de rédaction des dossiers d'homologation dédiés aux nouveaux produits utilise un système de validation plus simple ou un unique expert est en charge de la validation. En outre, il est plus fréquent que plusieurs rédacteurs collaborent à la rédaction d'un dossier. Il devient alors nécessaire de bien identifier quel est le rédacteur principal, responsable du dossier.

Nous proposons un modèle d'activité légèrement différent du premier :

- la publication de relecture y est conservée ;
- le système de validation est simplifié et s'appuie sur la mécanique des cycles de vie ;
- une responsabilité est modélisée et associée aux dossiers d'homologation.

Relecture et validation

La relecture d'un dossier d'homologation est effectuée par un seul expert métier. Elle doit également s'effectuer sur le document tel qu'il est publié en fin de processus et non sur un ensemble de fragments.

Pour outiller ce besoin, nous proposons de mutualiser la même publication dynamique de relecture que celle proposée pour les dossiers d'homologation équipements.

Responsabilité éditoriale

Afin d'identifier clairement le rédacteur en charge de la rédaction d'un dossier, chaque dossier dispose d'une responsabilité intitulée « responsable éditorial ». Cette responsabilité est fixée à la création d'un nouveau dossier et peut être attribuée à n'importe quel rédacteur de l'équipe.

Cycle de vie des dossiers

Les dossiers d'homologation dédiés aux produits sont validés par un unique expert. Le cycle de vie d'un tel document est ainsi relativement simple à modéliser. Nous proposons d'attacher le cycle de vie suivant à chacun des dossiers comme illustré sur la figure 106.

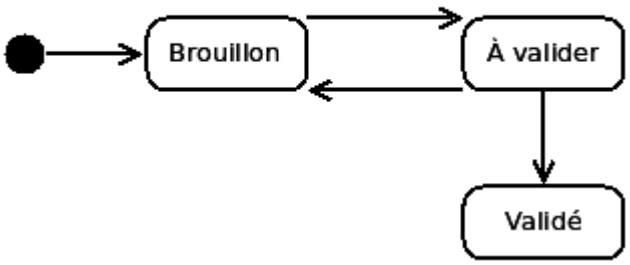


Figure 106 : cycle de vie d'un dossier d'homologation "produit"

Documentation technique de référence

La documentation technique de référence est constituée de l'ensemble des fiches issues des dossiers d'homologation. Dans cet atelier, la seule activité éditoriale consiste en l'enrichissement des nouvelles fiches et la maintenance des fiches existantes.

Nous proposons un modèle d'activité contenant un modèle de tâche « demande d'enrichissement média ». Cette tâche est mobilisée pour accompagner les principaux rédacteurs en charge de la maintenance du référentiel dans leurs demandes d'enrichissement des fiches auprès d'autres services (directement en restaurant, auprès des services de rédaction du référentiel belge, etc.).

Tâche de demande d'enrichissement média

Les tâches de demande d'enrichissement média ont pour objet l'accompagnement de la demande, de l'édition et de la validation des enrichissements opérés dans les fiches issues des dossiers d'homologation. Dans la plupart des cas, un enrichissement média consiste en l'ajout d'une ou plusieurs photos pour illustrer une fiche. Une tâche de demande d'enrichissement média est ainsi composée d'un titre, d'une échéance (les médias sont à ajouter avant la prochaine publication du référentiel), d'une description de la demande, d'un responsable éditorial en charge de la validation et d'un responsable média.

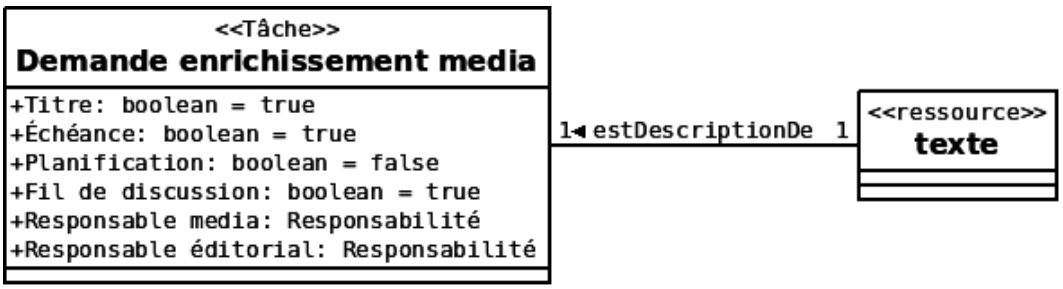


Figure 107 : modèle d'une tâche de demande d'enrichissement média

Le cycle de vie d'une demande d'enrichissement média contient uniquement trois états (*à traiter*, *à valider* et *validé*) comme illustré sur la figure 108.

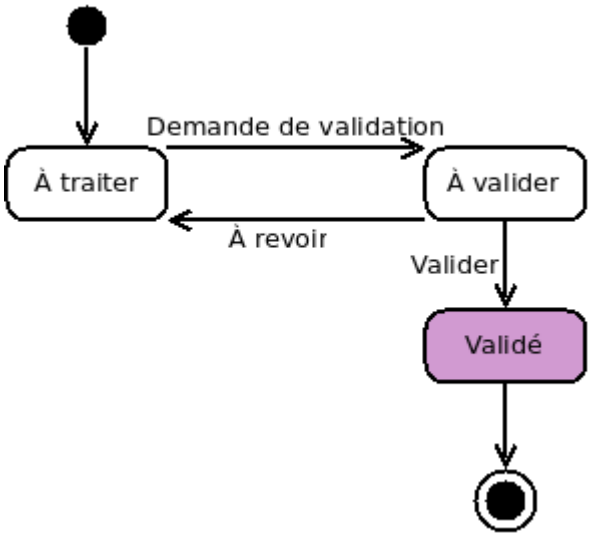


Figure 108 : cycle de vie d'une tâche d'enrichissement média

Table de correspondance entre états et responsabilités :

	À traiter	À valider	Validé
Responsable éditorial	Suiveur	Exécutant	Aucun
Responsable média	Exécutant	Suiveur	Aucun

Rédaction du référentiel

La rédaction du référentiel français s'opère principalement en réalisant des mises à jour des fiches référencées. Les fiches sont ajoutées ou supprimées du référentiel en fonction des machines en usage et des produits en vente dans les restaurants.

Nous proposons un modèle de tâche permettant de recenser les interventions sur le référentiel.

Fiche d'intervention

Un tâche « fiche d'intervention » a pour objet la documentarisation d'une action de mise à jour du référentiel. La tâche est personnelle, instanciée par un rédacteur et directement considérée comme terminée.

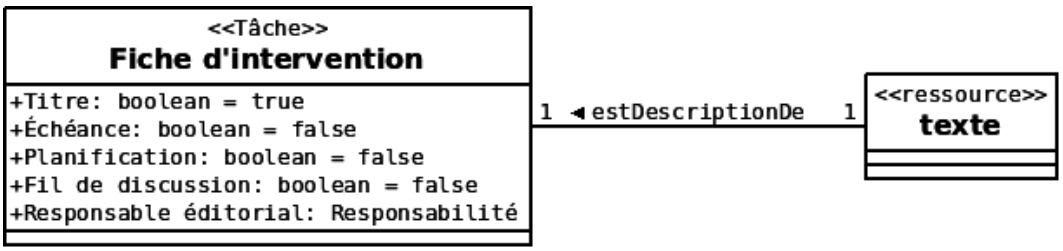


Figure 109 : modèle de fiche d'intervention

Le cycle de vie d'une telle tâche ne contient qu'un unique état : *clos*.

Instanciation d'une fiche d'intervention

L'objectif des fragments pragmatiques *fiche d'intervention* consiste à documentariser l'activité sur le fragment racine du référentiel. Afin de fluidifier l'instanciation de telles tâches, il peut être judicieux d'automatiser l'ouverture de l'interface de création d'une tâche dès lors que le référentiel est édité.

Documentarisation des interventions

Le principal intérêt de ces fiches d'interventions réside dans la documentarisation de l'activité. Ainsi, lors de la publication de chaque nouvelle version du référentiel, il devient aisé de regrouper l'ensemble des interventions entre la précédente version et l'actuelle afin de rédiger une note de version attirant le lecteur vers les principales nouveautés du référentiel.

Vue d'ensemble

Sur la figure 110, les cercles rouges servent à spécifier le modèle opéré par un atelier :

- M désigne le modèle d'enrichissement média ;
- R désigne le modèle de documentarisation des interventions sur le référentiel ;
- P désigne le modèle de validation des DH *produits* ;
- E désigne le modèle de validation des DH *équipements* ;

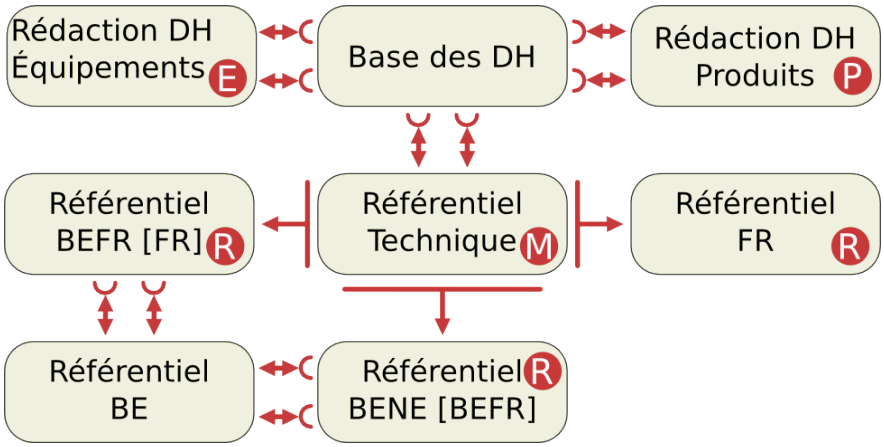


Figure 110 : vue d'ensemble de la chaîne éditoriale

L'objectif d'une telle structure est d'industrialiser la production documentaire rééditorialisée. Les actions manuelles d'import des dossiers d'homologation sont ainsi remplacées par des fragments proxys. La mise en place d'un fragment implique que le dossier d'homologation recense des fiches procédures en cours d'usage dans les restaurants.

Afin de simplifier la rédaction des dossiers d'homologation, deux ateliers seconds sont mis en place. Seuls les dossiers en cours de rédaction sont ainsi importés. Le nombre de fragments visibles est limité au maximum. Dans un même objectif de simplification, l'enrichissement des fiches est séparé de leur agencement au sein du référentiel.

Nos hypothèses de recherches consistent à s'appuyer sur une telle structure pour accompagner l'industrialisation de la rééditorialisation documentaire tout en améliorant le contrôle et la qualité des documents. D'un point de vue plus concret, nous soutenons qu'une telle chaîne éditoriale permet de pallier aux défauts de l'outil constatés au sein du chapitre 2 (p. 50) et permet ainsi une meilleure maîtrise du graphe documentaire, une limitation du nombre de fragments copiés et une amélioration globale de la production documentaire.

10.2.2 Chaînes éditoriales pour l'enseignement supérieur

Cet exemple est tiré de nos travaux au sein du projet SUP E-educ. L'enjeu est de concevoir une chaîne éditoriale permettant la production et la maintenance de l'ensemble des supports pédagogiques produits par une université. Nous intégrerons ici la production des contenus par les enseignants, leur mutualisation au niveau des différentes formations, leurs traductions pour les étudiants internationaux et leur rescénarisation pour des formations dérivées (comme la formation à distance ou l'apprentissage).

Le modèle documentaire mobilisé par cet exemple est Opale. Chaque atelier de la structure opérera donc le modèle Opale et les modèles d'activité proposés sont des compléments

du modèle Opale.

Structure des ateliers

Projets éditoriaux

Chaque enseignant responsable d'une unité d'enseignement est également responsable de la production de la documentation de référence associée (du cours). Pour mener à bien les enseignements, la chaîne éditoriale peut également être mobilisée pour rédiger des exercices et les sujets d'examens. En fonction de leur collaboration, il est possible que certains enseignants soient amenés à mutualiser des contenus.

La cellule TICE a la responsabilité de traduire l'ensemble des contenus dans les langues les plus parlées par les étudiants internationaux. Pour cet exemple, nous posons l'hypothèse que la cellule TICE traduit les contenus en anglais et en chinois. La cellule TICE peut également être amenée à effectuer des corrections de style, de syntaxe ou d'orthographe au sein des contenus rédigés par les enseignants.

Le responsable pédagogique de chaque formation a la responsabilité d'agréger les contenus issus des différentes unités d'enseignements qui la composent afin de produire une documentation de référence par semestre. Cette documentation contient tous les cours dans une langue. Il existe un document par langue.

L'université peut disposer également de formations dérivées. Il s'agit de formation à distance, de formation continue ou de formation en apprentissage. Ces formations peuvent concerner une unique unité d'enseignement (dans le cas de formation à distance très spécialisée par exemple) ou un parcours plus complet. Les unités d'enseignement de ces formations ont également un responsable qui s'occupe aussi de la rédaction des cours. La seule particularité réside dans le fait que les cours sont systématiquement des dérivations de cours existants.

Structuration des ateliers

Dans ce contexte, la production des contenus s'appuie sur une multitude de projets éditoriaux indépendants agrégés les uns avec les autres pour produire des documents plus complets. Deux structures d'ateliers radicalement différentes peuvent être utilisées pour un tel contexte. Il est possible de s'affranchir d'atelier central en mobilisant une structure distribuée : chaque enseignant dispose de son propre atelier dont les contenus sont importés dans des projets éditoriaux d'agrégation pour les formations. À l'inverse, il est possible de mettre en place un atelier central géré par la cellule TICE à partir duquel l'ensemble des enseignants pourrait mutualiser leurs contenus par le biais de fragments proxys.

Dans la mesure où la cellule TICE a la responsabilité de traduire l'ensemble des contenus, il paraît plus judicieux de poser un atelier principal afin d'utiliser un unique atelier calque par traduction.

Projet principal

Nous considérons donc un seul et unique projet principal, le projet central de la cellule TICE.

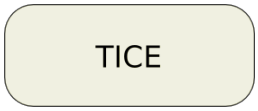


Figure 111 : atelier principal

L'ensemble des contenus en français de l'université sont présents dans cet atelier.

Projets de dérivation

Chacune des traduction du projet éditorial principal constitue un projet de dérivation. Nous proposons donc deux ateliers de calques divergents.

Figure 112 : ajout des ateliers calques divergents

projets satellites

Chaque unité d'enseignement représente un projet éditorial satellite dans lequel une partie des contenus du projet principal est rédigée. Les contenus de référence sont issus de fragments proxys liés à l'atelier principal.

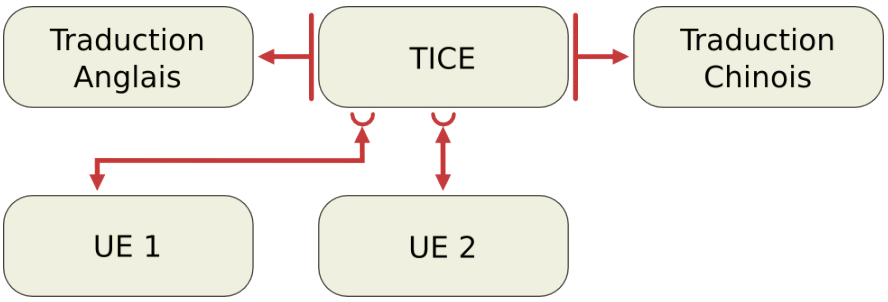


Figure 113 : ajout des ateliers satellites

La figure 113 est une simplification ne montrant que deux ateliers dédiés à des unités d'enseignement. Une université proposant 500 unités d'enseignement aura alors 500 ateliers satellites reliés à l'atelier TICE principal.

Les ateliers satellites peuvent également être exploités pour produire des contenus qui ne sont pas partagés avec l'atelier TICE. Ce sera par exemple le cas de sujets d'examens ou d'exercices.

Projets d'agrégation

La documentation de référence d'une formation est obtenue en agrégeant l'ensemble des cours qui la composent. Cette agrégation est faite à partir des cours réceptionnés et éventuellement corrigés par la cellule TICE.

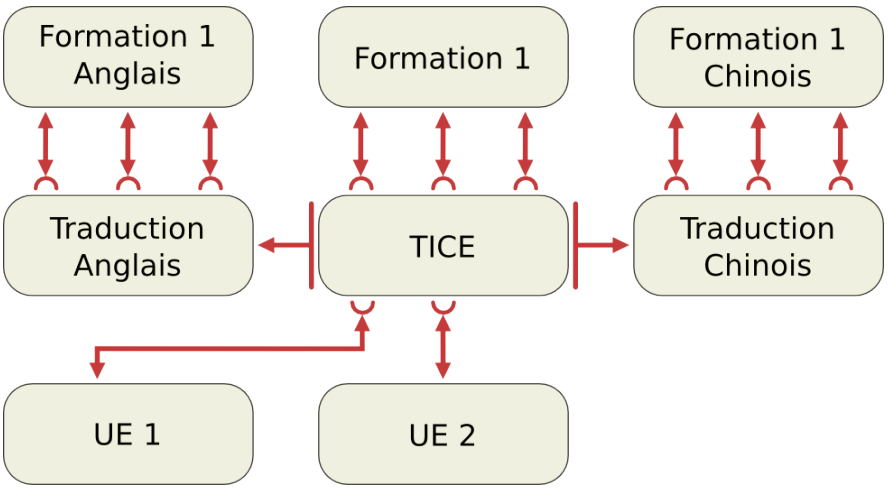


Figure 114 : ajout des ateliers d'agrégation

La figure 114 ne montre que des agrégations monolingues. Des agrégations bilingues ou trilingues peuvent également être effectuées.

Formations dérivées

Les formations dérivées s'appuient sur la dérivation des contenus d'une ou plusieurs unités d'enseignement. Les unités d'enseignement dérivées sont des projets satellites comme les

autres avec la particularité d'importer des contenus depuis d'autres projets satellites par l'intermédiaire d'un ou plusieurs fragments proxys.

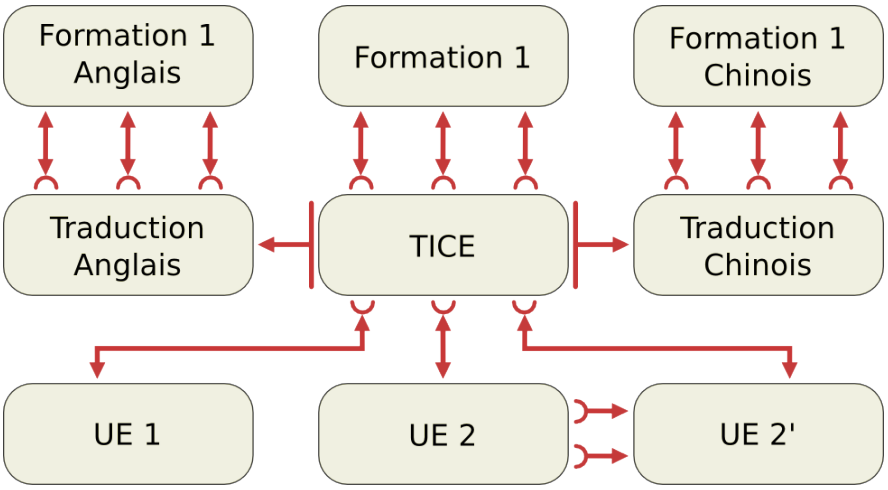


Figure 115 : ajout d'un atelier satellite dérivant les contenus d'un autre

Les documentations de référence des formations dérivées sont constituées de la même façon : par agrégation des contenus depuis l'atelier TICE.

Partage de contenus entre enseignants

Deux enseignants peuvent avoir besoin de partager des contenus entre eux. Ce partage peut être préalable à une surcharge des fragments dans un des projets. Auquel cas, il suffit simplement de reproduire la même structure que pour une unité d'enseignement dérivée. Lorsque aucune surcharge n'est nécessaire, il est judicieux de remplacer les proxys divergents par leur pendant convergent.

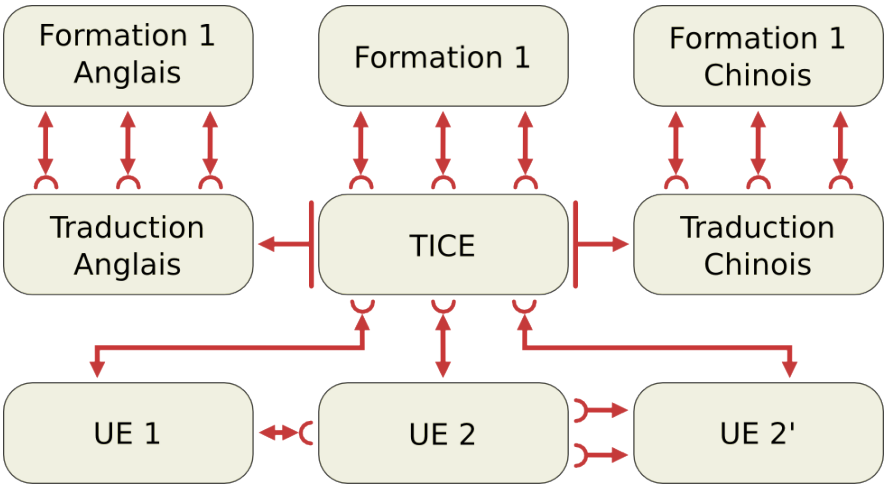


Figure 116 : ajout d'un atelier satellite exploitant les contenus d'un autre

Néanmoins, une telle structure laisse apparaître un conflit dans la gestion des identifiants :

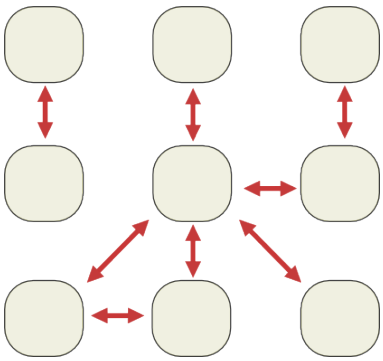


Figure 117 : graphe de contrôle

On constate en effet un cycle entre les trois ateliers en bas à gauche du graphe de contrôle. Certes, les liens convergents représentent des fragments proxys dédiés à des cours différents. Néanmoins, il suffit d'un seul fragment partagé entre UE1 et UE2 parmi l'ensemble du sous-graphe issu des fragments proxys pour que le conflit soit déclenché. Par exemple, si chaque cours contient un logo ou une description de l'université, alors cette structure peut être conflictuelle. Pour remédier à ce conflit, nous préconisons de systématiquement passer par l'atelier TICE pour la mise en place de partages entre plusieurs unités d'enseignement.

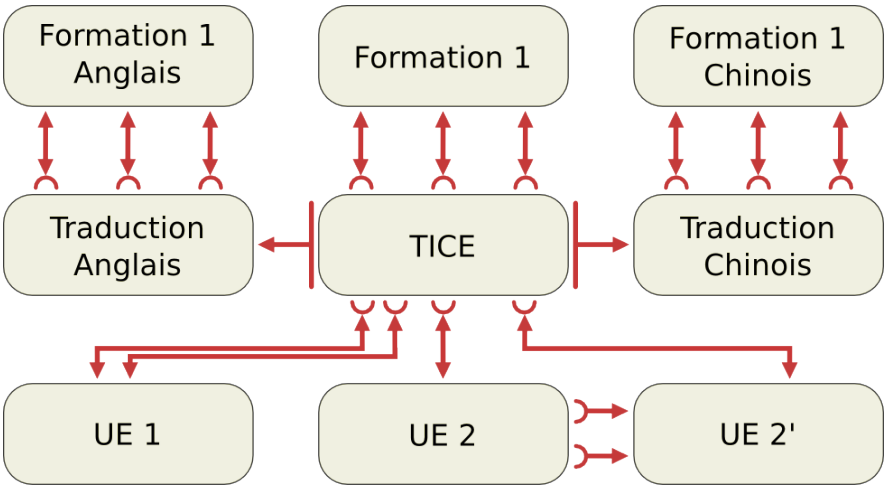


Figure 118 : changement de configuration

Cette structure ne laisse plus apparaître aucun conflit.

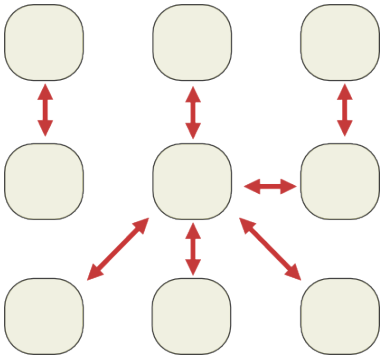


Figure 119 : graphe de contrôle

Organisation auctoriale

Les documentations de référence des unités d'enseignement ou des formations sont publiées par la cellule TICE au début de chaque semestre. Entre temps, il n'est pas nécessaire de garder le graphe exploité par un atelier continuellement en état d'être re-publié. Par conséquent, aucun atelier calque convergent n'est nécessaire.

En fonction de leur organisation, les enseignants pourront mettre en place des ateliers calques ou des fragments proxys convergents pour protéger la visibilité de leurs fragments en cours de rédaction des autres rédacteurs travaillant avec eux dans l'atelier dédié à une unité d'enseignement. Ce contexte est le seul pour lequel de nouveaux ateliers pourraient être utilisés.

Documentarisation de l'activité

Nous distinguons deux types de projets de rééditorialisation supportant une réelle activité auctoriale. Le premier concerne l'ensemble des projets dédiés aux unités d'enseignement. Le second concerne l'ensemble des projets gérés par la cellule TICE.

Les modalités d'organisation de l'activité de ces deux projets sont fondamentalement différentes. Nous proposons donc un modèle d'activité pour chacun.

Modèle enseignant

L'enjeu du modèle enseignant est d'assister les enseignants dans la documentarisation de l'activité réalisée sur leur graphe. Les ateliers dédiés aux unités d'enseignement sont gérés par l'enseignant responsable de l'UE. Il peut ainsi choisir de travailler seul ou en équipe. Afin d'assister la rédaction, nous proposons deux aspects : l'enrichissement des fragments standards du modèle Opale afin de marquer l'avancement de la rédaction et l'ajout d'un fragment de type tâche permettant d'enregistrer des notes personnelles.

Enrichissement des fragments standards du modèle Opale

Nous proposons deux éléments pour l'enrichissement des fragments standards du modèle Opale : la définition d'un responsable sur chaque fragment afin d'aider les enseignants à déterminer quel collègue a pris la responsabilité de le rédiger et de valider son contenu et la définition d'un cycle de vie sur chacun des fragments. L'objectif du cycle de vie n'est pas d'obstruer l'écriture des fragments en imposant un processus dans l'écriture mais plutôt de donner des méthodes permettant de répertorier les fragments en fonction de leur état : en cours de rédaction, terminé, validé.

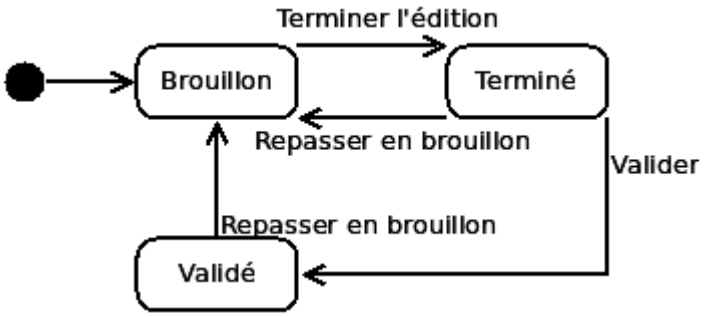


Figure 120 : cycle de vie des fragments du modèle enseignant

Fragment de note personnelle

La « note personnelle » mobilisée dans le modèle enseignant a des caractéristiques similaires à celle modélisée dans le contexte de la DILA. Il s'agit de proposer un principe de tâches simple : la constitution de note personnelle, de date de planification de ces notes et de rappel. Le contexte d'écriture étant relativement peu formel, il est inutile d'ajouter des étapes de validation par un rédacteur tiers, ou tout autre mécanisme faisant intervenir d'autres rédacteurs. Le modèle documentaire de la note contient une description textuelle et une synthèse des opérations à réaliser sur le graphe.

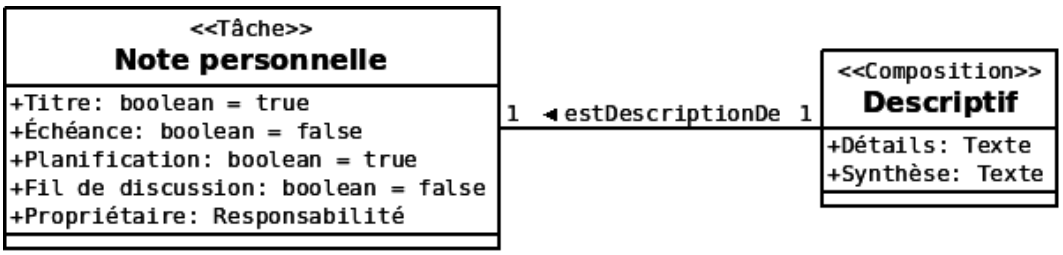


Figure 121 : modèle d'une tâche note personnelle

Cycle de vie associé aux notes personnelles :

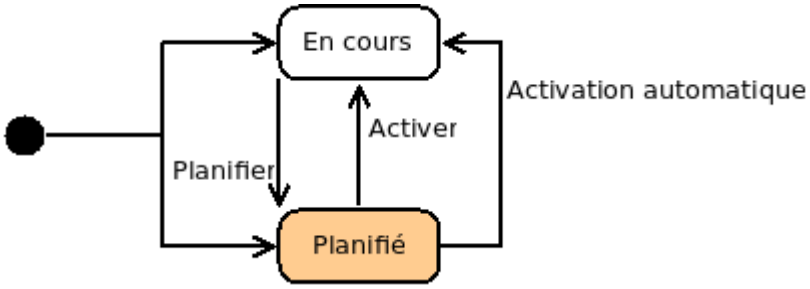


Figure 122 : cycle de vie d'une note personnelle

Table de correspondances entre états et responsabilités :

	En cours	Planifiée
Propriétaire	Exécutant	Aucun

Modèle TICE

L'objectif du modèle pour les cellules TICE est de fournir des fragments pour encadrer une organisation plus formelle de l'activité. L'enjeu est de permettre à des ingénieurs pédagogiques d'organiser et de superviser l'action réalisée par des relecteurs ou des traducteurs sur des graphes volumineux.

Ce contexte est plus formel que le précédent. Nous y mobilisons trois tâches inspirées de la modélisation de la DILA. Il s'agit d'une tâche de validation, une tâche de mission et une tâche de note personnelle.

Fragment mission

Le fragment de mission est initié par un ingénieur pédagogique. Il dispose de deux champs *responsabilité* (*responsable* pour le rédacteur, *référent* pour l'ingénieur pédagogique), d'un titre, et d'une date d'échéance. Le modèle documentaire associé à une mission contient le descriptif des interventions à effectuer et un champ *remarques*. Le fil de discussion est ouvert et permet un échange entre le rédacteur et l'ingénieur pédagogique.

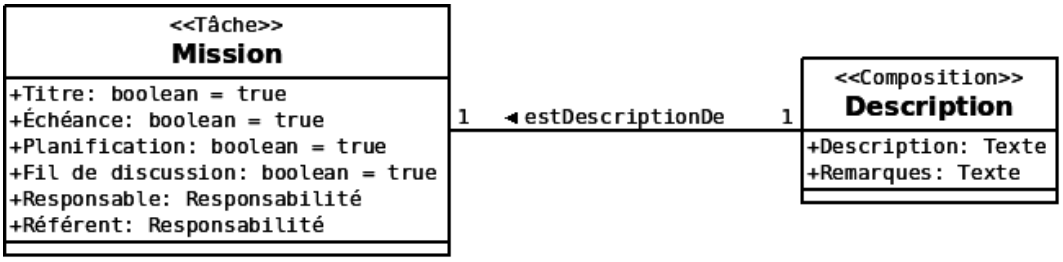


Figure 123 : modèle d'une mission

Le cycle de vie d'une tâche de mission est défini comme suit :

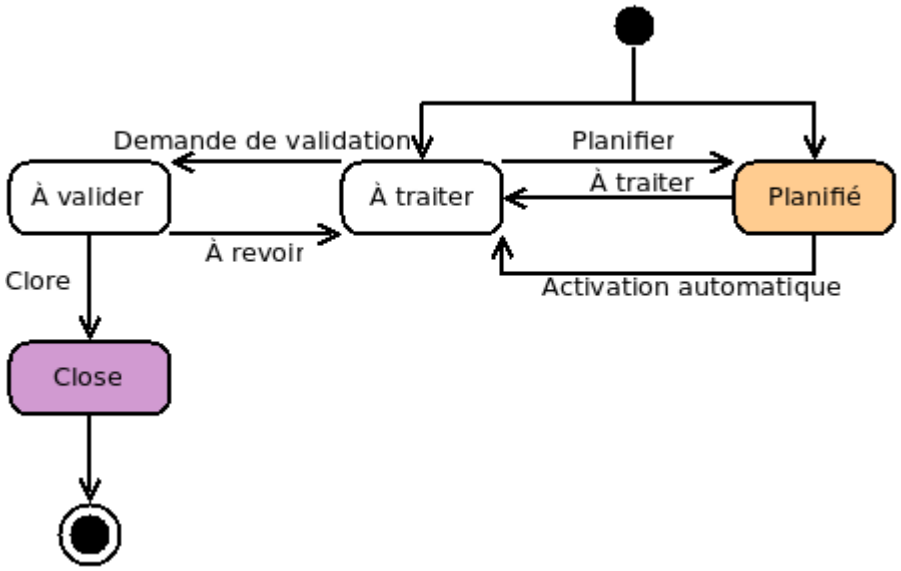


Figure 124 : cycle de vie d'une mission

	À traiter	Planifiée	À valider	Planifiée
Responsable	Exécutant	Aucun	Suiveur	Aucun
Suiveur	Suiveur	Aucun	Suiveur	Aucun
Responsable de la validation	Suiveur	Aucun	Exécutant	Aucun

Lors de son initialisation, le fragment mission est soit directement placé en état *À traiter* si l'intervention est urgente ou *Planifié* le cas échéant. Un fragment en état planifié n'est associé à aucun utilisateur. Un déclencheur automatique transforme le fragment en état *À traiter* le jour de la date de planification.

Fragment validation

Le fragment de validation est initié par un rédacteur ayant pris l'initiative d'effectuer une intervention sur le graphe. Ce fragment sert à enregistrer le processus de validation de son intervention par un ingénieur pédagogique. Il n'est donc pas nécessaire d'utiliser une date d'échéance ou de planification. Le modèle documentaire associé contient uniquement le motif et le résumé des modifications.

Figure 125 : modèle d'une validation

Le cycle de vie d'un fragment de demande de validation est strictement identique à la demande de validation modélisée pour la DILA :

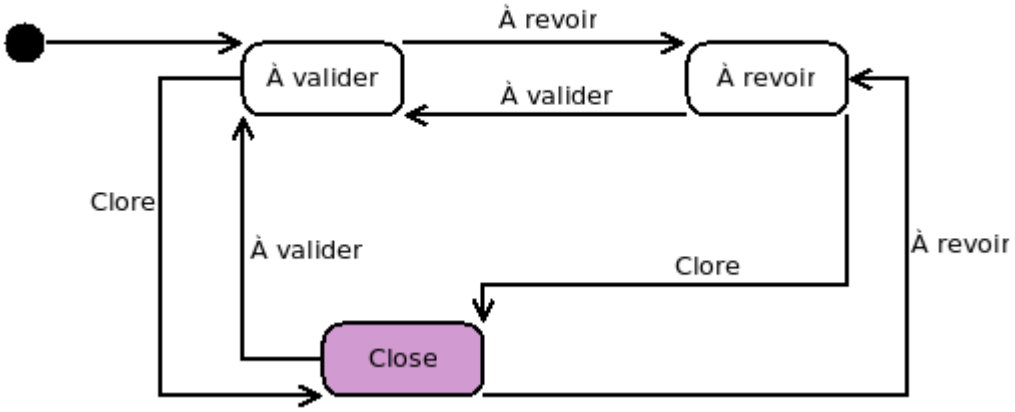


Figure 126 : cycle de vie d'un fragment de validation

	À valider	À revoir	Close
Propriétaire	Suiveur	Exécutant	Aucun
Responsable validation	Exécutant	Suiveur	Aucun

Fragment note personnelle

Le fragment de note personnelle est identique à celui exploité dans le modèle enseignant.

Vue d'ensemble

Sur la figure 127, les cercles rouges servent à spécifier le modèle opéré par un atelier :

- O désigne le modèle Opale standard
- E désigne la version enseignant du modèle Opale
- T désigne la version TICE du modèle Opale

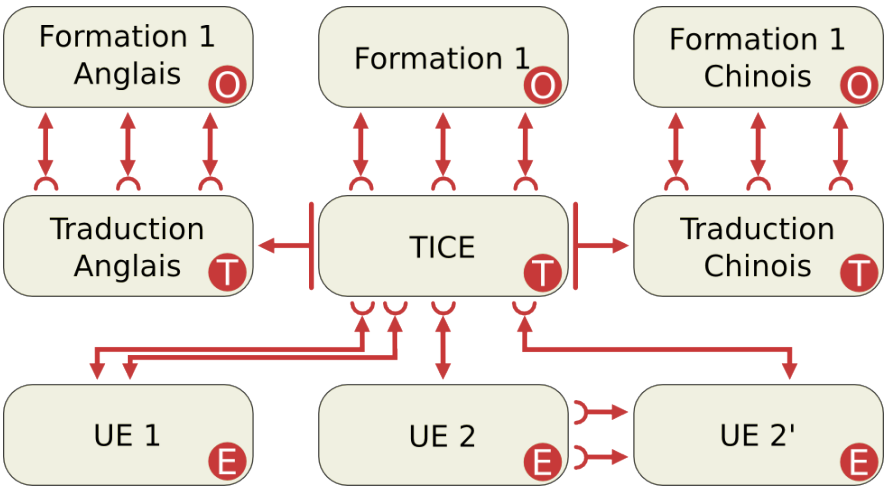


Figure 127 : vue d'ensemble de la chaîne éditoriale

À partir du graphe documentaire Opale montré dans le chapitre 2 (p. 46), nous estimons que le nombre moyen de fragments mobilisés pour la rédaction d'un cours universitaire est autour de mille. Ainsi, chaque enseignant travaille dans un atelier lui montrant un graphe de taille maîtrisée sans percevoir sa contribution au graphe complet de plusieurs centaines de milliers de fragments. En outre, les enseignants ont la possibilité d'enregistrer leur activité au

sein de leur portion de graphe afin d'assister l'organisation de leur rédaction.

Au sein de la cellule TICE, le graphe est séparé en trois parties. L'ensemble des fragments rédigés par les enseignants sont disponibles dans l'atelier de la cellule TICE. Ces fragments sont intégralement dérivés dans chacun des ateliers de traduction. Cette séparation permet à nouveau de maîtriser la taille globale du graphe. L'organisation de la cellule étant plus structurée et le graphe à maintenir plus volumineux, les éléments permettant de documentariser l'activité sont plus structurants que dans le modèle enseignant.

En combinant structure globale et documentarisation locale, nous avons modélisé une chaîne éditoriale exploitable pour l'ensemble d'une université et permettant la rééditorialisation des fragments à très grande échelle en exploitant des graphes de plusieurs centaines de milliers voire plusieurs millions de fragments.

Partie IV
Positionnement épistémologique et évaluation

La spécificité de notre objet de recherche et de notre approche réinterroge son évaluation. Nos travaux s'inscrivent dans une recherche technologique dont l'objet d'étude est la rééditorialisation documentaire et les chaînes éditoriales. Notre objectif est, comme le rappelle Crozat (2012), d'exhumer de nouvelles formes d'écriture documentaire. Notre approche vise à concevoir, outiller et accompagner ces nouvelles formes. En tenant compte de cette posture, l'évaluation de notre recherche doit d'abord être pensée selon l'évolution des usages, tant d'un point de vue théorique : est-ce que les usages que nous avons voulu ouvrir sont désormais possibles ; que pratique : est-ce que la pratique de ces usages est mobilisée dans des contextes de production documentaire.

Afin d'évaluer nos travaux, nous avons souhaité étudier les modalités de validité d'un programme de recherche comme le nôtre. Il s'agit de mettre en exergue la méthode de recherche et les critères de scientificité afin d'en retenir des modalités d'évaluation. Nous menons cette étude en trois chapitres. Dans un premier, nous retraçons brièvement les contours de l'épistémologie et plus particulièrement, les propositions de Theureau (2006 ; 2009) pour une épistémologie des programmes de recherche empiriques ou technologiques. Dans un second chapitre, nous adaptons les propositions de Theureau à notre programme de recherche et nous positionnons nos travaux au sein de ce programme. Cette analyse et ce positionnement nous permettent de mener une évaluation de notre recherche dans un troisième et dernier chapitre.

Chapitre 11

Épistémologie et recherche technologique

11.1 Épistémologie : critères de scientificité et programmes de recherche 167
11.2 Épistémologie normative interne des programmes de recherche empirique 169
11.3 Épistémologie normative interne des programmes de recherche technologique 171

11.1 Épistémologie : critères de scientificité et programmes de recherche

Fondations épistémologiques

Les travaux philosophiques dédiés à la caractérisation de la recherche scientifique se sont principalement intéressés aux sciences physiques depuis l'époque de Galilée jusqu'à la recherche contemporaine. Les principaux auteurs que nous étudions ici, soit Popper, Kuhn, Lakatos, Koyré et Theureau, s'entendent sur deux caractéristiques dans l'analyse de la méthode scientifique : l'universalité des lois et le principe de réfutation.

Ils considèrent tous que les lois scientifiques sont universelles. En 1934, Popper (1973, p. 483) montre ainsi que « les lois universelles transcendent l'expérience, ne fut-ce que parce qu'elles sont universelles et transcendent donc n'importe quel nombre fini de leurs illustrations observables ».

En considérant les lois scientifiques comme universelles, il devient difficile de les démontrer. En revanche, elles peuvent être réfutées lorsqu'elles sont fausses ou échapper à la réfutation ce qui n'écarte pas leur validité. Popper explique ainsi que « c'est naturellement à cause de cette transcendance que les lois ou théories scientifiques ne sont pas vérifiables et que la possibilité d'être soumises à des tests ou d'être réfutées est la seule chose qui les distingue, en général, des théories métaphysiques » (ibid., p. 483). Popper explique également que pour faciliter les tests, l'énonciation des lois doit être la plus claire et limpide possible : « le système théorique doit être formulé avec assez de clarté et de précision pour qu'on puisse facilement y reconnaître ce que chaque nouvelle hypothèse y représente » (ibid., p. 69).

Popper et la concorde des savoirs scientifiques

Popper ajoute principalement deux éléments à ces caractéristiques. Premièrement, il distingue les sciences des non-sciences parmi lesquelles il regroupe la religion, la philosophie des grecs, les doctrines économiques ou politiques ou encore la psychanalyse. Deuxièmement, il pose une hypothèse de concorde parmi les disciplines scientifiques : il reconnaît certes un morcellement des connaissances entre disciplines mais soutient qu'il y a un consensus sur les

connaissances et les lois valides au sein d'une discipline. Le consensus évolue au fur et à mesure que de nouvelles réfutations sont réalisées.

Popper s'inscrit ainsi dans une vision de construction universelle des connaissances.

Kuhn et les paradigmes scientifiques

En 1962, Kuhn (1972) complexifie le principe de concorde scientifique en lui superposant un niveau supérieur, celui du « paradigme scientifique ». Kuhn borne les périodes de concorde par des périodes de crise pendant lesquelles plusieurs référentiels théoriques « luttent à mort » jusqu'à la non-réfutation d'un unique paradigme, ce qui ouvre ainsi une nouvelle période de concorde.

L'ensemble des lois valides dans un paradigme tel que défini par Kuhn ne le sont plus dans le paradigme suivant ou concurrent. Par exemple, l'ensemble des lois physiques mobilisées par les grecs ne sont plus valides dans le paradigme galiléen. Les lois de Galilée et de Newton sont elles-mêmes caduques dans un paradigme plus récents tel que celui de la physique quantique.

Lakatos et les programmes de recherche

En 1965, Lakatos (1994) s'oppose à Popper et Kuhn sur le principe de concorde. Pour lui, la recherche s'organise en différents « programmes de recherche ». Un programme peut avoir une portée plus ou moins générale et plus ou moins transversale. Lakatos spécifie la méthodologie d'un programme de recherche dont nous proposons un résumé en deux propriétés principales : disposer d'un noyau théorique protégé de toute réfutation par une ceinture d'hypothèses auxiliaires et susciter un travail de recherche visant à compléter le noyau par de nouvelles hypothèses explicatives ou prédictives.

Lakatos considère que différents programmes de recherche sont systématiquement en partie concurrents, en partie semblables et en partie complémentaires entre eux. Il soutient que c'est la confrontation entre les différents programmes de recherche qui est un des principaux moteurs de l'histoire des sciences.

Koyré et les critères de scientificité

Lakatos a inscrit ses travaux dans l'analyse de la recherche en sciences physiques et en mathématiques. Les critères de scientificité lui semblaient implicites au sein de ces communautés. Il n'a donc nullement éprouvé le besoin de les formuler. Afin de généraliser la notion de programme de recherche, Koyré (1973) en a proposé un enrichissement en 1966 en définissant des critères de scientificité que le noyau et la ceinture d'hypothèses doivent respecter. Koyré formalise ainsi trois critères : la littéralisation de l'empirique, la production de propositions réfutables et la relation organique avec la technique.

Theureau (2009) explique ces critères comme suit.

- « La littéralisation de l'empirique, c'est "que l'on use de symboles qu'on peut et doit prendre à la lettre, sans égard à ce qu'éventuellement ils désignent" et "que des conséquences empiriques puissent suivre du maniement aveugle et réglé de quelques lettres" ou symboles » (ibid., p. 439).
- « Une proposition réfutable est "une proposition telle que l'on puisse construire *a priori* une conjonction finie de propositions qui y contredisent et trancher empiriquement cette contradiction" » (ibid., p. 439).
- « Le critère de relation organique avec la technique s'oppose directement à l'idée d'une "science pure". [...] Il conduit à penser l'expérimentation scientifique en continuité avec la technique, au lieu de la ramener à la contemplation » (ibid., p. 439).

Theureau, le cours d'action et l'épistémologie normative interne aux programmes de recherche

Theureau (2004 ; 2006) a mené un programme de recherche nommé « cours d'action » comportant à la fois des aspects empiriques en sciences humaines et sociales et des aspects technologiques.

Dans un ouvrage de réflexion sur le programme de recherche « cours d'action » (2009), Theureau propose une généralisation de la méthode de recherche mobilisée dans le programme afin de compléter les concepts de Koyré et Lakatos. Il propose ainsi une matrice entre différentes approches épistémologiques et différentes catégories de programmes de recherche.

Au niveau des approches, Theureau distingue « **épistémologie interne** (à une démarche scientifique donnée) et **épistémologie externe** (portant sur une démarche scientifique donnée en référence à d'autres démarches scientifiques, voire à la démarche scientifique en général) ; **épistémologie descriptive** (épistémologie spontanée d'un chercheur ou d'un groupe de chercheurs, que l'on peut éventuellement dégager par l'analyse de son (leur) activité) et / ou l'histoire de ses (leurs) recherches) et **épistémologie normative** (**interne** : idéal épistémologique de ce même chercheur ou ensemble de chercheurs, ou **externe** : idéal épistémologique appliqué de l'extérieur à une recherche donnée) » (ibid., p. 564).

Au niveau des programmes, Theureau propose de structurer quatre catégories différentes : des programmes philosophiques dont l'objet d'étude est « la circonscription des questions possibles », des programmes empiriques portant sur des « hypothèses empiriques », des programmes technologiques portant sur des « hypothèses de création technologique, organisationnelle et culturelle » et des programmes de recherches mathématiques portant sur des « hypothèses de création virtuelle » (ibid., p. 454). Les deux prochaines sections de ce chapitre sont dédiées aux propositions de Theureau pour une épistémologie normative (donc une méthode idéale) interne (menée par des scientifiques sur leurs propres démarches) des programmes de recherches empiriques et technologiques.

11.2 Épistémologie normative interne des programmes de recherche empirique

L'épistémologie interne normative des programmes de recherche de Theaureau (figure 128) ne poursuit pas les mêmes objectifs que les propositions de Lakatos et de Koyré. Là ou ses prédécesseurs proposent des éléments pour analyser ce qui relève de la science ou de la non-science afin d'assister l'historien des sciences ou l'épistémologue s'intéressant à une nouvelle discipline, Theureau propose des éléments pour assister un chercheur ou groupe de chercheurs à analyser sa propre pratique. Les différentes propositions de Lakatos et Koyré se retrouvent donc disséquées dans une méthode en neuf « mouvements ». Six de ces mouvements sont appelés fondamentaux et forment le noyau théorique et heuristique. Les trois mouvements complémentaires (2.1', 3.1' et 3.3) délimitent ce que Theureau appelle une zone de « fertilité ». Ils correspondent à la ceinture d'hypothèses auxiliaires de Lakatos et Koyré que Theureau nomme lui la « capacité de croissance ».

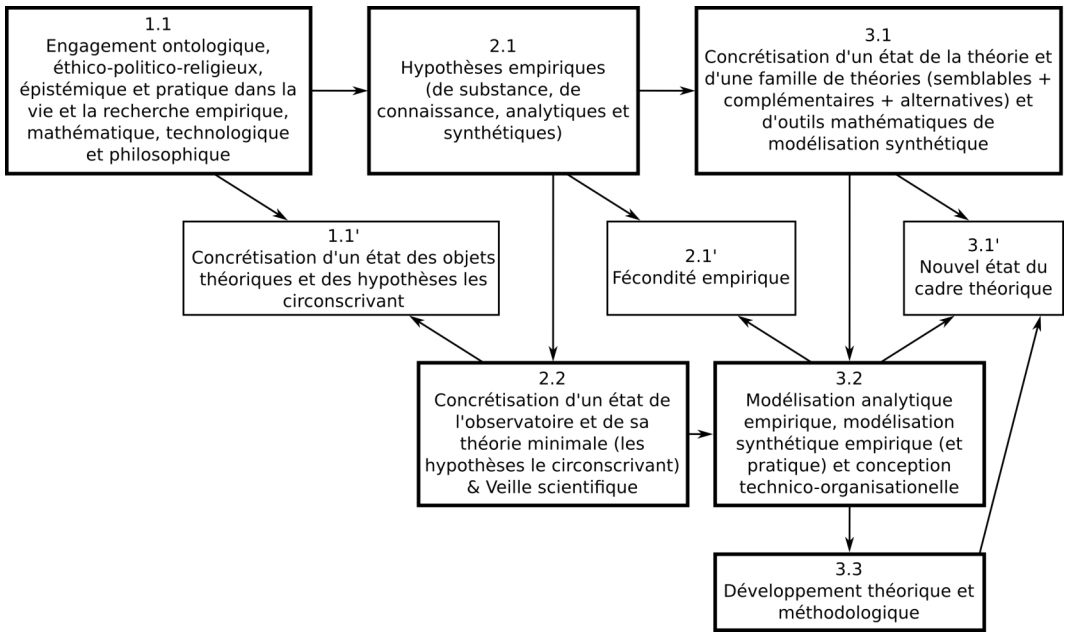


Figure 128 : programme de recherche empirique général (Theureau, 2009, p. 462)

Dans cette section, nous revenons sur les différents mouvements proposés par Theureau.

L'engagement du chercheur (1.1)

Theureau pose comme fondement à tout programme de recherche l'engagement des chercheurs qui y contribuent. Cet engagement est à la fois ontologique soit à propos d'une certaine « croyance concernant la "nature des choses" » (Theureau, 2009, p. 461), éthico-politico-religieux soit propre à « un certain mode de relation [...] auquel participe un principe d'espérance » (ibid., p. 461), épistémologique soit lié à « un certain "régime de vérité" » (ibid., p. 461) et pratique soit dans l'attente de la « réalisation de certains effets pratiques » (ibid., p. 461).

Cet engagement est propre à chaque chercheur et souvent partiellement en phase entre les différents chercheurs d'un même programme de recherche. Theureau souligne en outre son aspect « indéterminable avec précision et exhaustivité pour chacun » (ibid., p. 463).

Hypothèses empiriques (2.1)

Ces hypothèses constituent le cœur du noyau théorique de Lakatos. Theureau insiste sur le caractère nécessairement « non-trivial » d'une partie significative de ces hypothèses (ibid., p. 464). C'est ce caractère qui permet une « première partie du second critère de scientificité de Koyré, celui des "propositions réfutables" » (ibid., p. 464).

Theureau parle « d'hypothèses de substance » pour désigner les hypothèses permettant de définir les objets théoriques et « d'hypothèses de connaissance » pour désigner celles permettant de construire une méthodologie et des critères de réfutation (ibid., p. 464). Ce sont ces hypothèses qui permettent de délimiter le programme de recherche.

Objets théoriques (1.1')

Le mouvement 1.1' consiste en une transformation de 1.1 par 2.1. Il s'agit de l'engagement d'un chercheur ou d'un groupe de chercheurs raisonné à travers une série d'hypothèses de façon à spécifier des objets d'étude et des objets théoriques. Theureau souligne l'importance de la formalisation d'objets théoriques puisqu'ils participent à la constitution du premier critère de scientificité de Koyré à savoir la littéralisation de l'empirique.

Instance de réfutation (2.2)

L'instance de réfutation constitue la seconde partie du critère de réfutation de Koyré. Il s'agit de déterminer, dans un cadre conforme à celui posé par les hypothèses de connaissance, et pour un objet théorique conforme à celui déterminé par un chercheur en relation avec les hypothèses de substance, une méthodologie permettant de réfuter les hypothèses propres au programme de recherche.

Theureau insiste sur la nécessité d'isoler l'instance de réfutation du cadre théorique. « C'est grâce à cette indépendance que l'instance de réfutation-décision peut occasionner des surprises, c'est-à-dire produire des données qui remettent en cause tout ou partie des théories de départ du chercheur » (ibid., p. 465).

Cadre théorique (3.1)

Ce mouvement définit l'ensemble du cadre théorique dans lequel viennent s'inscrire les hypothèses du mouvement 2.1. L'enchaînement des mouvements de 1.1 à 3.1 illustre le caractère *interne* de la démarche de Theureau qui démarre son analyse au chercheur et à son engagement dans ses recherches avant de l'étendre au cœur du programme de recherche (2.1 et 1.1') puis de positionner ce cœur dans un cadre théorique dont certains éléments sont souvent communs avec d'autres programmes. Une analyse descriptive et externe d'un programme de recherche pourrait avoir la méthodologie inverse, soit introduire le cadre théorique, spécifier les hypothèses de recherche et l'objet de recherche avant d'en établir les relations avec l'engagement du ou des chercheurs.

Dans ce mouvement, Theureau insiste sur la structuration des théories avec une théorie centrale et des théories complémentaires (ibid., p. 466).

Modélisation (3.2) et fécondité

Le mouvement de modélisation consiste en une observation et un traitement des données issues de l'expérience, soit issues de l'instance de réfutation 2.2. Ces données peuvent être traitées selon deux démarches complémentaires. Elles peuvent être traitées à travers une « modélisation analytique », soit une modélisation permettant d'interpréter et de comprendre les données de l'expérience. De façon complémentaire, elles peuvent être mobilisées à travers une « modélisation synthétique », soit une simulation permettant à la fois de vérifier ou de faire évoluer les hypothèses empiriques 2.1, afin de comparer les résultats des modèles synthétiques par rapport aux données issues de l'expérience.

Fécondité (2.1', 3.3, 3.1')

Depuis la modélisation, trois mouvements complémentaires s'enchaînent. Ces mouvements sont les résultats du programme de recherche. Ils représentent les objectifs des chercheurs investis dans le programme.

La modélisation rapportée aux hypothèses de départ permet d'entrer dans un mouvement que Theureau nomme la fécondité empirique (2.1'). Il s'agit donc des hypothèses de départ revisitées par l'analyse et la synthèse du résultat de l'expérience. Cet état de fécondité empirique peut être le support de nouvelles recherches, c'est-à-dire l'origine d'une redéfinition de l'objet théorique de recherche 1.1', de l'instance de réfutation 2.2 et de nouvelles modélisations 3.2.

En complément, la modélisation de ce que Theureau appelle les « surprises de l'expérience », c'est-à-dire les données issues de l'expérience non conformes aux résultats attendus, permet indirectement de participer à la concrétisation de développements du cadre théorique. C'est ainsi que se constitue le mouvement 3.3 qui est un objectif indirect à tout programme de recherche (contrairement aux résultats 2.1' de la modélisation 3.2 en lien avec les hypothèses 2.1 qui constituent l'objectif principal du programme de recherche).

Le dernier mouvement consiste en la transformation du cadre théorique 3.1 par les avancées 3.3 issues de la modélisation 3.2. Ce nouveau cadre théorique est numéroté 3.1'.

11.3 Épistémologie normative interne des programmes de recherche technologique

Pour Theureau (2009, p. 478), un programme de recherche technologique s'appuie d'une part sur plusieurs programmes de recherches empiriques et d'autre part a pour objet la technique dans sa relation organique avec la science (soit la définition de la technologie selon Koyré (1973) et Theureau). Theureau fait ainsi évoluer le schéma de l'épistémologie normative interne afin de prendre en compte ces deux caractéristiques (figure 129).

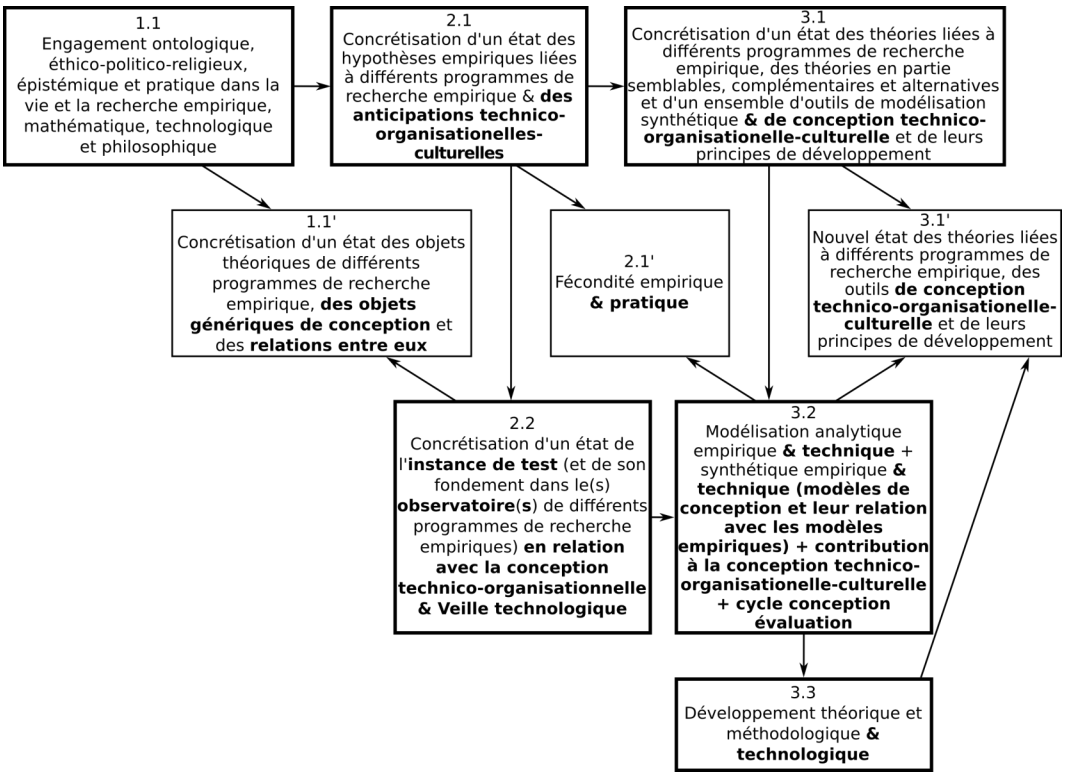


Figure 129 : programme de recherche technologique général (Theureau, 2009, p. 480)

Le texte en gras met en exergue les principaux changements entre le schéma de la section précédente et cette proposition. Parmi les principales modifications, nous relevons : une modification dans les objets théoriques (1.1') et une modification diffusée sur une majorité des mouvements en lien avec un facteur « technico-organisationel-culturel ».

Au sein du mouvement 1.1' Theureau propose d'enrichir les objets théoriques étudiés avec des objets génériques de conception. Il s'agit de formaliser les différents objets théoriques au cœur des programmes de recherche empirique mobilisés ainsi que de faire émerger leur relation avec l'objet technologique au cœur du programme de recherche technologique.

Nous interprétons le facteur technico-organisationnel-culturel de Theureau comme un caractère ayant trait à la technique tout en étant raisonné vis-à-vis de son impact sur les organisations humaines ou plus largement vis-à-vis de ses impacts culturels. La technique n'est pas appréhendée en soi pour améliorer une performance, elle est mobilisée dans un objectif d'assistance aux organisations.

Cette interprétation impacte plusieurs mouvements. Ainsi, le cadre théorique prend en compte : la conception d'objets techniques pour répondre à des problèmes organisationnels ou culturels ; les principes de développement de ces techniques et leurs modalités d'évaluation. Les hypothèses prévoient des anticipations, à la fois sur la technique et donc *in fine*, sur les développements liés à son usage. L'instance de test met en œuvre les modalités techniques anticipées ; les procédures d'observations sont mises en perspective des usages de l'objet technique. Le mouvement de modélisation intègre la dimension technique aux modèles analytiques et synthétiques et prévoit en outre une contribution technique comme partie intégrante du programme de recherche.

Chapitre 12

Positionnement de notre recherche

12.1 Épistémologie descriptive 173
12.2 Positionnement : notre contribution au programme 177

Dans ce chapitre, nous exploitons les éléments épistémologiques introduits dans le chapitre précédent en poursuivant l'objectif de mener une évaluation de notre recherche. Ce chapitre se segmente en deux sections : une première où nous menons une analyse épistémologique descriptive du programme de recherche technologique « rééditorialisation documentaire » et une seconde où nous positionnons nos travaux au sein de ce programme de recherche.

12.1 Épistémologie descriptive

Introduction

Les travaux de recherche dans lesquels s'inscrivent notre contribution ont été fondés par les projets PolyTex (Bachimont & Charlet, 1998) et Scenari (Crozat, 2002 ; Crozat, 2007). Aujourd'hui menés par l'Unité Ingénierie des Contenus et des Savoirs, ces travaux ont conduit au développement des chaînes éditoriales numériques XML (Crozat & Gebers, 2009). La notion de programme de recherche telle qu'exprimée par Theureau et plus particulièrement son épistémologie normative interne des programmes de recherche technologique nous semble être un cadre adéquat pour en mener une analyse.

Dans un premier temps, nous rappellerons l'objet central des recherches qui s'inscrivent dans notre programme de recherche. Dans un second, nous proposons de faire évoluer le schéma de l'épistémologie normative de Theureau afin de le mettre en adéquation avec l'idéal poursuivi par les chercheurs investis dans notre programme. Ces deux éléments préalables nous permettent dans un troisième temps de mener une épistémologie descriptive soit d'expliquer les éléments concrets constitutifs de chacun des mouvements.

Objet

Comme expliqué dans les hypothèses de notre première partie, notre recherche vise à exhumer de nouvelles formes d'écriture propres au support numérique. Elle vise plus particulièrement à accompagner un ensemble de techniques regroupées derrière l'intitulé : « rééditorialisation documentaire ».

Nous proposons de considérer la rééditorialisation comme objet théorique central du

programme. Les chaînes éditoriales numériques ne sont alors que des objets technologiques qui permettent l'expérimentation de la rééditorialisation et la production de documents rééditorialisés.

Épistémologie normative interne

L'épistémologie normative (soit la méthode de recherche idéale) interne (établie par les chercheurs investis dans cette recherche) de notre programme de recherche diffère légèrement des propositions de Theureau.

Theureau conserve le même enchaînement de mouvements entre un programme empirique et un programme technologique. L'instance de test et les modalités d'observation dégagent des données (2.2) traitées selon une modélisation analytique et synthétique permettant de formaliser une contribution « technico-organisationnelle-culturelle » (3.2) ouvrant ainsi à des développements théoriques, méthodologiques et technologiques (3.3).

Le programme « rééditorialisation documentaire » cherche à anticiper les évolutions des formes d'écriture induites par le support numérique. L'expérimentation dans un contexte de laboratoire ne peut être le seul élément mobilisé pour valider de nouvelles formes d'écriture. En suivant le principe de tendance tel que formalisé par Leroi-Gourhan (1971) et Stiegler (1994), une nouvelle forme d'écriture correctement outillée sera naturellement mobilisée par certains contextes d'usages car ses apports seront significatifs. Ces formes d'écriture ne peuvent être valides qu'à la condition d'être réellement mobilisées, ces mobilisations font donc partie intégrante des étapes nécessaires afin de valider ou réfuter nos hypothèses. **L'expérimentation de laboratoire permet de valider la possibilité de mobiliser la nouvelle forme d'écriture étudiée. Sa mobilisation par de réels contextes d'usages fait partie de l'instance de réfutation décision.**

Nous proposons d'enrichir le schéma de Theureau par un nouveau mouvement dédié numéroté 2.3 qui correspond aux usages dans des situations réelles du système développé. Ce mouvement se situe entre la concrétisation de l'instance de test (qui correspond au développement d'un prototype et de ses modalités d'expérimentation afin de valider, en laboratoire, que l'outillage de situations réelles peut être initié) et les modélisations analytiques et synthétiques (qui correspondent à la formalisation des propositions).

Nous proposons également de faire évoluer les liens afin d'instaurer un cycle entre les mouvements 2.2, 2.3 et 3.2. Le développement d'une instance peut être réellement exploratoire. Les premiers usages font alors remonter des données donnant lieu à des modélisations partielles qui demandent une modification de l'instance de test et une réinterrogation vis à vis des usages afin d'être complétée. Ce n'est qu'une fois la modélisation complète que l'état de fertilité est atteint et que des développements sont proposés (3.3, 3.1') où les hypothèses de départ sont réinterrogées (2.1').

Nous proposons enfin une modification du terme désignant l'objet étudié. Theureau utilise le concept d'instance de test en faisant référence à un procédé expérimental de laboratoire jetable à l'issu des expérimentations. L'aspect technologique de l'objet étudié, la continuité du cycle des mouvements 2.2, 2.3 et 3.2 et le développement technologique comme issue du processus nous incitent à adopter un vocabulaire moins connoté. Nous parlerons donc de versions de la technologie développée et par abus de langage, simplement de versions.

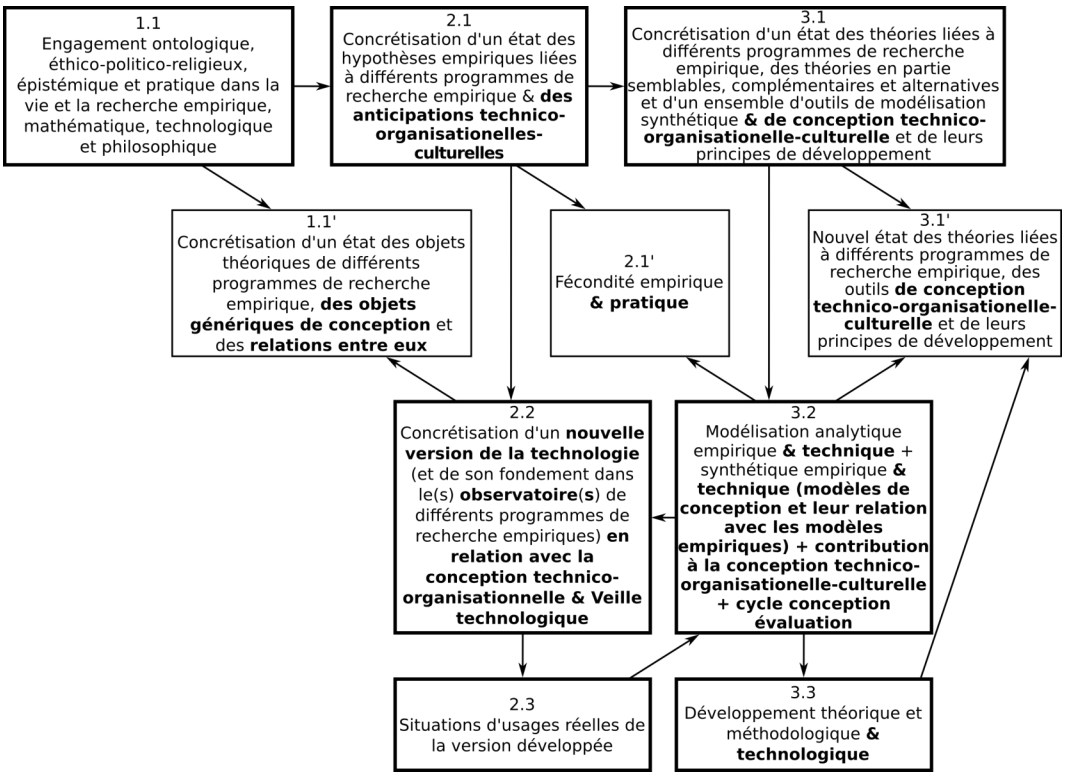


Figure 130 : idéal épistémologique du programme de recherche "rééditorialisation documentaire"

Cadre théorique (3.1)

La théorie centrale mobilisée par le programme de recherche « rééditorialisation documentaire » est à la fois philosophique et scientifique, centrée sur le support numérique et tournée vers sa compréhension et son exploitation. Il s'agit essentiellement des travaux de Bachimont (2010) introduits dans la section 1.2 (p. 17). L'objectif du programme à proprement parler est résumé par Crozat comme un objectif « d'inventer - pour les exhumer - des formes documentaires qui ouvrent un nouveau champ du possible, pour tenter d'en anticiper les incidences cognitives et sociétales en genèse » (Crozat, 2012, p. 183).

Cette théorie centrale se retrouve en lien avec plusieurs autres programmes de recherche empirique tels que l'histoire des techniques (Leroi-Gourhan, 1971) et plus particulièrement des supports d'inscription des connaissances (Barbier, 2012 ; Goody, 1979), une philosophie des techniques (Stiegler, 1994) et du numérique et une théorie du document (Pédauque, 2003 ; Pédauque, 2005 ; Pédauque, 2006). Cette théorie est également supportée par le programme de recherche en ingénierie documentaire qui a vu l'élaboration des documents structurés et des logiciels permettant leurs manipulations (André et *al.*, 1989). Toutes ces théories annexes qui fondent la théorie centrale sont introduites dans la section 1.1 (p. 17).

Hypothèses (2.1)

Dans ce mouvement, nous réduisons l'ensemble du cadre théorique à une série d'hypothèses de connaissance et de substance ainsi qu'aux anticipations « technico-organisationnelles-culturelles » attendues.

Les hypothèses de connaissances sont tirées du cadre théorique. Elles posent le cadre des recherches effectuées au sein du programme.

- Les supports d'inscription des connaissances impactent la rationalité avec laquelle ces connaissances sont élaborées, donc impactent les connaissances elles-mêmes.
- Le support d'inscription graphique implique une rationalité propre appelée la raison graphique dont les structures logiques de représentation des connaissances sont typiquement la liste, le tableau et la formule.
- Le support d'inscription numérique implique une rationalité propre appelée la raison computationnelle dont les structures logiques de représentation des connaissances sont typiquement le programme, le réseau et la couche.

- L'évolution des techniques suit des tendances indépendantes des contingences propres aux différentes civilisations humaines.
- La tendance du numérique est constituée d'un mouvement de fragmentation et recombinaison combiné à un mouvement de désémantisation et de réinterprétation.

Les hypothèses de substance permettent la définition de l'objet théorique et des objets génériques de conception. Il s'agit donc de circonscrire l'objet principal des recherches.

- La rééditorialisation est une technique de production de documents s'inscrivant dans la tendance du numérique.
- Les techniques de rédaction structurée et la rééditorialisation sont plus efficaces pour la rédaction et la maintenance de fonds documentaires importants.
- La rédaction de documents structurés est plus efficace avec des modèles de documents dédiés à leur contexte d'usages.

Les anticipations « technico-organisationnelles-culturelles » attendues représentent à la fois la technologie développée et ses impacts attendus. Nous listons trois entrées :

- Production de chaînes éditoriales numériques XML instrumentant la rééditorialisation.
- Production d'outils de conception de chaînes éditoriales XML.
- Diffusion de l'usage des chaînes éditoriales numériques XML pour la production de contenus.

Objets de recherche (1.1')

L'objet théorique de recherche est la rééditorialisation documentaire. Les objets génériques de conception à travers lesquels cet objet théorique est étudié sont les chaînes éditoriales numériques XML.

Cycles d'expérimentation/usages/modélisation (2.2, 2.3, 3.2)

À l'étude depuis plus d'une dizaine d'années, le programme de recherche « rééditorialisation documentaire » s'est structuré autour de nombreux projets de recherche. Chaque projet reprend une méthode similaire au schéma présenté en introduction de cette section. Les projets disposent ainsi de leurs propres cycles d'expérimentation et aboutissent sur d'éventuelles reconsidérations du cadre théorique qui sert de base au projet suivant.

Prenons pour l'exemple le projet Epicure financé par le Conseil Régional de Picardie en 2005 qui s'est intéressé à la problématique de coûts de développement pour la spécialisation des chaînes éditoriales. L'enjeu était de valider l'hypothèse d'une meilleure efficacité des chaînes éditoriales spécifiques, trop souvent écartées en raison de leurs coûts de développement. La version de la technologie développée avait pour objet de faciliter la conception de nouvelles chaînes éditoriales. Les résultats attendus étaient une augmentation du développement des chaînes éditoriales spécifiques et donc, une meilleure diffusion de ce type de solutions (par hypothèse, solutions optimales pour produire des documents rééditorialisés).

Sur le plan des observables, nous soutenons que la validation technologique d'un tel projet s'opère selon deux niveaux d'observations.

- Le premier est technique et se déroule systématiquement à l'issue du mouvement 2.2. Son objectif est de valider que les nouveaux usages ouverts par la version développée sont bien possibles et que les usages en situations réelles peut se poursuivre. La démonstration technologique, voire l'expérience de pensée, permettent cette évaluation à priori. Dans le cas présent, une fois le projet terminé, la démonstration, soit la comparaison des coûts de développement d'une chaîne éditoriale spécifique avec ou sans la version de la technologie développée permettent cette première évaluation.
- L'évaluation selon les usages est plus difficile à mener car elle s'inscrit dans un temps plus long que celui de l'expérience de laboratoire. Dans l'exemple du projet Epicure, une mesure des usages peut survenir plusieurs années à l'issue du projet en observant l'évolution du nombre de chaînes éditoriales et plus particulièrement la mobilisation de la technologie développée pendant le projet. Cette étude peut être menée aujourd'hui et montre un accroissement de l'ordre de dix pour un du nombre de chaînes éditoriales développées sur une période et des moyens équivalents. L'ensemble de ces outils a été

conçu en mobilisant la technologie développée par le projet.

Les situations d'usages permettant de tester la version développée en situation réelle ont étés menées en partenariat avec l'Institut de Recherche et Coordination Acoustique/Musique (IRCAM) et l'Institut National de l'Audiovisuel (INA).

Les modélisations des données issues de l'expérience et des usages ont permis d'établir un méta-modèle de chaînes éditoriales qui abstrait la complexité technique de la spécialisation en spécifiant un langage de modélisation de plus haut niveau. La contribution technique a donné naissance au logiciel de modélisation SCENARIbuilder.

Fécondité empirique et pratique (2.1')

En proposant un dispositif technique facilitant la spécialisation de chaînes éditoriales spécifiques, le projet Épicure a facilité le développement de nouvelles chaînes éditoriales et a effectivement simplifié l'usage de la rééditorialisation. Les observations sur l'augmentation de cet usage ne réfutent aucune des hypothèses de départ qui restent donc encore valides. En revanche, ce développement met en exergue de nouveaux verrous dans la diffusion de l'usage de la rééditorialisation. Par exemple, la diffusion de nombreuses chaînes éditoriales aux modèles documentaires spécifiques a mis en exergue le besoin de transformer automatiquement des fragments d'un modèle à l'autre afin de faciliter les échanges entre rédacteurs. Cette prolifération de chaînes éditoriales est également à l'origine d'une augmentation du nombre de rédacteurs et du nombre de projets éditoriaux outillés par de tels outils. C'est cette situation initiale qui est à l'origine de nos travaux de doctorat sur la gestion de graphes documentaires complexes.

Développements (3.3)

Les principaux développement liés au projet Épicure sont un méta-modèle de chaînes éditoriales et une technologie de modélisation et génération des chaînes éditoriales.

Évolution du cadre théorique (3.1')

Élément inattendu, les développements du projet Épicure ont été menés parallèlement au programme de recherche en ingénierie des modèles présenté en section 1.5 (p. 32). Les propositions de ce dernier dépassent et généralisent les contributions du projet Épicure (3.3) et amènent à considérer le programme « ingénierie des modèles » comme un nouveau programme de recherche empirique mobilisé dans le cadre théorique du projet « rééditorialisation documentaire ».

12.2 Positionnement : notre contribution au programme

Notre recherche s'inscrit dans le programme de recherche technologique « rééditorialisation documentaire ». Elle constitue les fondations d'un projet de recherche qui la dépasse et dont l'enjeu est de lever un verrou observé sur les usages de la rééditorialisation, notamment lorsque la volumétrie des fragments et des liens de rééditorialisation devient importante.

Dans cette section, nous revenons sur les particularités des différents mouvements du projet de recherche par rapport au programme dans lequel il s'inscrit.

Compléments au cadre théorique (3.1)

Nous l'avons vu dans la section précédente, le programme de recherche empirique sur l'ingénierie des modèles fait maintenant partie intégrante du cadre théorique. En outre nous avons été amenés à reconsidérer le cadre théorique afin de lui adjoindre trois nouveaux programmes de recherche.

Nous proposons de reconsidérer l'objet logique manipulé par les chaînes éditoriales. Au lieu de penser un fonds documentaire rééditorialisé en un ensemble de fragments de documents, nous proposons la structure du graphe. Cette structure permet de prendre la mesure du nombre et de la diversité des liens de rééditorialisation utilisés dans un fonds. Ce faisant, cette structure logique met en exergue certaines difficultés dans la maintenance d'un fonds rééditorialisé. Dès

lors et comme montré dans le chapitre 2 (p. 27), le programme de recherche empirique sur la théorie des graphes devient une des théories complémentaires mobilisées.

Les deux autres théories complémentaires font suite à notre recherche de solutions mises en œuvre au sein d'autres programmes pour assister des utilisateurs de systèmes complexes à gérer la granularité et la diversité des objets techniques qu'ils manipulent. Il s'agit du programme de recherche technologique dans l'assistance au développement informatique présenté dans le chapitre 4 (p. 65) et du programme de recherche empirique du travail coopératif assisté par ordinateur présenté dans le chapitre 5 (p. 71).

Hypothèse (2.1)

Dans le cadre de notre projet de recherche, une hypothèse de substance vient compléter les hypothèses déjà présentées dans le programme :

- un fonds de documents rééditorialisés peut être représenté sous la forme d'un graphe orienté où chaque fragment est représenté par un sommet et chaque fonction de rééditorialisation (transclusion, dérivation, déclinaison) est représentée par un arc.

Cette hypothèse est accompagnée d'une nouvelle anticipation « technico-organisationnelle-culturelle » liée aux effets attendus suite aux résultats de notre recherche. Cette anticipation peut être formulée selon différents termes :

- d'un point de vue théorique, nous anticipons une industrialisation des usages de la rééditorialisation dans plus de contextes,
- cela revient à une meilleure gestion de la complexité du graphe documentaire par les rédacteurs d'une chaîne éditoriale,
- cela revient également à faire de meilleurs usages de la rééditorialisation.

Objets d'étude (1.1')

L'objet d'étude théorique est toujours la rééditorialisation documentaire. Les objets génériques de conception sont toujours les chaînes éditoriales. Nous proposons en outre d'y ajouter les graphes documentaires comme objet de représentation et d'analyse d'un fonds documentaire rééditorialisé.

Version de la technologie développée et procédures observables (2.2)

Dans la continuité du programme de recherche « rééditorialisation documentaire », la version de la technologie développée s'appuie, comme expliqué dans nos propositions, sur la suite Scenari. Nous avons donc, en partenariat avec l'unité ICS et la société Kelis, participé à l'implémentation des propositions. L'état d'avancement des développements peut être consulté dans l'annexe 2 (p. 219).

Comme indiqué dans la section précédente, deux niveaux d'évaluation sont nécessaires pour valider les développements technologiques d'un tel projet. Les propositions peuvent être évaluées sur le plan technique et par leurs mobilisations au sein de réels contextes d'usages.

- Sur le plan technique, l'enjeu est de valider que les usages recherchés (une industrialisation de la rééditorialisation) sont bien ouverts par nos propositions et donc bien possibles dans la version de la technologie développée. Cette validation est possible par l'expérience de pensée sur le plan théorique et par une démonstration de la version développée sur le plan technologique.
- L'évaluation par l'évolution des usages se mène en deux temps. Dans un premier temps, il convient de recenser les mobilisations de la technologie développée au sein des différents contextes d'usages. Dans un second, l'enjeu sera d'analyser l'évolution de la rééditorialisation avec et sans la technologie proposée. Cette analyse peut se mener par une comparaison des graphes documentaires avant et après la mise en production de la technologie développée.

Situations d'usages (2.3)

Plusieurs situations d'usages ont été mises en place afin de tester les propositions et de les affiner au fur et à mesure des retours. Nous avons présenté dans ce mémoire quatre

situations différentes, de contextes protégés et exploratoires à des situations industrielles pour les développements les plus mûrs. Il s'agit du contexte Enaction Series ou des usages liés aux universités dans la première catégorie et des usages de la société Quick et de la DILA pour la seconde.

La maturité des situations d'usages est à mettre en perspective avec le spectre des développements planifiés dans notre recherche afin d'illustrer ce que nous appelons des cycles courts entre développement d'une nouvelle version, usages et modélisation. Les premiers éléments mis au point sont déjà utilisés en situation industrielle tandis que le développement des derniers éléments, soit des fonctions proxys, n'est que planifié.

Modélisation et contribution (3.2)

La modélisation correspond à la forme de nos propositions telles que nous les soumettons dans ce mémoire. Il s'agit de mettre en place des stratégies de distribution du graphe afin de n'en donner qu'une section à manipuler accompagnées d'une stratégie d'assistance aux rédacteurs dans la gestion de leur activité.

La contribution à la conception reprend les éléments développés dans le cadre de nos recherches, soit par exemple le système de tâches, le système de commentaires, les fonctions de calques et les fonctions de proxys. Au sein de ces éléments, notre contribution se compose principalement du système de commentaires, de développements préalables à la mise en place des proxys et du protocole CID pour la gestion de la distribution des chaînes éditoriales.

Développements attendus (3.3)

Les développements attendus se structurent sur trois niveaux :

- un niveau théorique de compréhension de la rééditorialisation et de la façon de la mettre en œuvre à grande échelle,
- un niveau méthodologique afin d'accompagner les concepteurs de chaînes éditoriales à proposer des outils permettant la gestion d'une rééditorialisation à grande échelle
- un niveau technologique avec l'enrichissement de la suite Scenari pour permettre la mise en œuvre de cette méthode.

Chapitre 13

Évaluation

13.1 Évaluation technique 182
13.1.1 Fragments pragmatiques 182
13.1.2 Structuration du graphe 183
13.1.3 Agencement d'une documentarisation locale avec une structure globale 184
13.2 Évaluation des usages 186
13.2.1 Enaction Series 186
13.2.2 DILA 187
13.2.3 Quick 188
13.2.4 Pédagogie UTC 188
13.3 Synthèse 189

État d'avancement du projet de recherche

Le projet de recherche dans lequel s'inscrivent nos travaux de doctorat est toujours en cours.

En se basant sur l'épistémologie normative exposée dans la section précédente (p. 173), les premiers mouvements sont achevés (les mouvements initiaux 1.1, 2.1 et 3.1 et le mouvement de concrétisation de l'objet théorique 1.1').

Dans la continuité, le projet de recherche est actuellement dans les cycles de développements (2.2), usages (2.3) et modélisations (3.2). Les propositions, les exemples tirés des usages et la contribution technique présentés dans ce mémoire sont le reflet de l'état d'avancement de ces cycles.

Il est trop tôt, à ce stade, pour mener une évaluation finale et pertinente du projet de recherche. Une évaluation de la mobilisation de la technologie développée à l'issue du projet et une analyse de l'impact de cette mobilisation quant à notre problématique de recherche permettront une évaluation pertinente de ces travaux.

Nous proposons dans ce chapitre quelques pistes de réflexion afin de mener une évaluation intermédiaire du projet. La pertinence des propositions quant à notre problématique et sur un plan plus générique, quant à notre environnement de recherche sera ainsi estimée dans une première section. Un état des lieux des contextes d'usages clos, en cours de production ou en cours de développement sera présenté dans une seconde section. La troisième et dernière section

de ce chapitre sera dédiée à la synthèse de cette évaluation.

13.1 Évaluation technique

Sur le plan technique, nous cherchons à déterminer si les usages que nous cherchons à outiller, soit une meilleure gestion de la production documentaire rééditorialisée à l'échelle industrielle, sont bien ouverts par nos propositions.

Nous proposons ainsi quatre critères pour mener cette estimation :

- le soutien de la rééditorialisation,
- la spécialisation de la technologie à ses contextes d'usages,
- l'aspect évolutif de ces spécialisations,
- le passage à l'échelle de la technologie.

Le premier critère est lié au cœur même du programme de recherche. Il vise à s'assurer que l'objectif final de tout projet de recherche reste cohérent avec le programme. Le critère de spécialisation permet d'évaluer le positionnement des technologies développées dans le paradigme de meilleure adaptation des logiciels spécialisés à un contexte d'usages spécifique par rapport aux applications génériques. Le critère d'évolution des spécialisations permet d'inscrire la technologie dans les cycles de conception/usages/modélisation propres au programme. Le dernier critère vise à valider que la technologie proposée répond bien à notre problématique de recherche.

Dans cette section, nous menons une brève analyse de nos trois chapitres de propositions vis à vis de ces critères.

13.1.1 Fragments pragmatiques

Rappel des propositions

La proposition exposée dans le chapitre 8 (p. 87) consiste à modéliser des fragments permettant d'enregistrer l'activité sur le graphe et à concevoir simultanément des fonctions d'assistance à la gestion du graphe qui mobilisent ces fragments comme structure d'enregistrement. Ainsi, le simple usage de ces fonctions produit une documentation de l'activité sur le graphe documentaire.

Cette approche a été expérimentée selon trois axes avec le moteur de tâches (p. 91), le système de commentaires (p. 97) et la gestion des responsabilités et cycles de vie des fragments documentaires (p. 101).

Critère 1, soutien de la rééditorialisation

L'usage de fragments pragmatiques a pour objet l'assistance des rédacteurs dans le contrôle du graphe. En permettant la gestion de tâches, l'insertion de fils de discussion, la désignation de responsabilités ou l'usage de cycles de vie, les fragments pragmatiques proposent un champ fonctionnel enrichi pour mieux maîtriser les interventions sur le graphe. En documentarisant ces interventions, les fragments pragmatiques permettent de mieux suivre l'évolution des fragments et leurs mobilisations par leurs différents contextes de publication.

Par construction, les fragments pragmatiques sont un renfort à une bonne compréhension des différents usages des fragments et donc à une rééditorialisation efficace.

Critère 2, spécialisation de la technologie à ses contextes d'usages

La modélisation de fragments pragmatiques s'opère dans un modèle d'activité. Ces modèles sont des compléments aux modèles documentaires et se conçoivent et s'exploitent selon les mêmes modalités : des primitives dédiées sont disponibles dans le logiciel SCENARIbuilder ; lors de la compilation d'un modèle, ces primitives sont transformées en code source de spécialisation des chaînes éditoriales et compressées dans une archive avec le reste du modèle.

Par conséquent, la capacité de spécialisation est du même ordre que celle des fragments documentaires classiques, abordée dans la section dédiée à la modélisation des

chaînes éditoriales (p. 32).

L'approche est valide quant à ce critère. Elle doit cependant être enrichie en fonction des futurs besoins rencontrés afin de s'adapter à de plus nombreux contextes de production et ainsi, augmenter les possibilités de spécialisation.

Critère 3, aspect évolutif de ces spécialisations

L'évolutivité d'une spécialisation suit les mêmes modalités pour les modèles d'activités que pour les modèles documentaires. En conservant le modèle conçu dans SCENARIbuilder, les développeurs gardent la possibilité de générer à nouveau l'archive contenant le code de spécialisation de la chaîne éditoriale. Il suffit alors d'une simple modification du modèle afin de générer une nouvelle archive adaptée à l'évolution du contexte de production.

Comme pour le second critère, l'aspect évolutif des spécialisations est du même ordre que l'aspect évolutif des modèles documentaires classiques.

Critère 4, passage à l'échelle de la technologie

Le passage à l'échelle du système de fragments pragmatiques n'est pas avéré. L'usage de modèles d'activité ajoute :

- de l'information aux fragments classiques,
- des fragments,
- des liens entre fragments.

L'usage des fragments pragmatiques donne une assistance aux rédacteurs mais participe dans le même temps à l'augmentation de la taille et de la densité du graphe. Cette approche est donc une des réponses pour assister les rédacteurs à gérer la complexité du graphe mais ne peut en constituer la totalité.

Synthèse

	Soutien de la rééditorialisation	Spécialisation de la technologie à ses contextes d'usages	Aspect évolutif de ces spécialisations	Passage à l'échelle de la technologie
Fragments pragmatiques	Par construction	Héritée de la modélisation des chaînes éditoriales ; À enrichir	Hérité de la modélisation des chaînes éditoriales	Non avéré

13.1.2 Structuration du graphe

Rappel des propositions

La proposition exposée dans le chapitre 9 (p. 111) consiste à structurer le graphe en plusieurs ateliers. Chaque atelier devient ainsi dédié à un projet de rééditorialisation (éditorial ou auctorial). Cette structuration s'opère notamment à travers les ateliers calques (p. 119) et les fragments proxys (p. 126).

La structuration du graphe permet :

- de simplifier le graphe perçu par les rédacteurs au sein d'un atelier ;
- d'automatiser la dérivation de fragments ;
- d'assister les rédacteurs dans le contrôle de ces dérivations.

Critère 1, soutien de la rééditorialisation

Lorsqu'un atelier second est mis en place, la relation entre l'atelier premier et l'atelier second s'opère autour d'une fonction de rééditorialisation. Par exemple, il peut s'agir d'un calque

de travail qui permet de contrôler les modifications réalisées dans l'atelier second et automatise la synchronisation du contenu d'un fragment de l'atelier second avec celui de l'atelier premier.

Par construction, les fonctions de structurations du graphe outillent une fonction de rééditorialisation.

Critère 2, spécialisation de la technologie à ses contextes d'usages

La structure d'un atelier est un paramétrage des chaînes éditoriales. En cours de production et à condition de disposer des droits pour ce type d'opération, un rédacteur peut créer un nouvel atelier calque ou des fragments proxys.

L'objectif de la structure est de coller au plus proche de l'organisation de la production documentaire. À raison d'un atelier par projet éditorial ou auctorial il est possible d'outiller n'importe quel contexte de production.

Par construction, cette proposition est fortement spécialisable.

Critère 3. aspect évolutif de ces spécialisations.

La structure des ateliers n'est pas un élément à redévelopper pour chaque nouveau contexte de production mais bien une simple configuration de la chaîne éditoriale. Par conséquent, il est toujours possible, en cours de production documentaire, de supprimer des ateliers existants ou de créer de nouveaux ateliers seconds liés par des calques ou proxys afin d'affiner la structure.

Par construction, la spécialisation de la structure des ateliers est évolutive.

Critère 4, passage à l'échelle de la technologie

L'architecture technique de la chaîne éditoriale supporte aisément la montée en charge du nombre d'ateliers. SCENARIchain peut supporter plusieurs milliers d'ateliers sans que cela n'altère ses performances. **Par construction, sur le plan technique, cette solution supporte très bien le passage à l'échelle.**

Les modalités techniques du passage à l'échelle ne forme qu'une partie des enjeux à estimer par ce critère. La seconde est la possibilité, pour un rédacteur, de gérer des graphes plus importants. En réduisant le nombre de fragments perçus par le système du calque ou du proxy, **cette solution nous semble, par hypothèse, supporter le passage à l'échelle sur le plan fonctionnel**.

Synthèse

	Soutien de la rééditorialisation	Spécialisation de la technologie à ses contextes d'usages	Aspect évolutif de ces spécialisations	Passage à l'échelle de la technologie
Structuration du graphe	Par construction	Par construction	Par construction	Par construction au niveau technique ; Par hypothèse au niveau fonctionnel

13.1.3 Agencement d'une documentarisation locale avec une structure globale

Rappel des propositions

La proposition finale telle qu'elle est résumée dans le chapitre 10 (p. 143), consiste en un agencement d'une documentarisation de l'activité locale à un projet de rééditorialisation avec une structuration globale des projets de rééditorialisation dans des ateliers distincts.

Dans cette section, nous posons des hypothèses sur l'estimation de la pertinence de nos propositions agencées ensemble vis à vis des quatre critères d'évaluation de la technologie développée.

Critère 1, soutien de la rééditorialisation

L'agencement de deux propositions nous semble avoir un impact neutre quant au support de la rééditorialisation. Nos deux propositions sont conçues pour assister différents aspects de la rééditorialisation.

Critère 2 et 3, spécialisation et évolution des spécialisations

Sur les deux critères liés à la spécialisation de la technologie et aux possibilités d'évolution de cette spécialisation, l'agencement des deux parties de nos propositions nous semble plus efficace. Le champ des spécialisations possibles est plus large et l'évolution d'une configuration à une autre reste aisée.

Critère 4, passage à l'échelle de la technologie

Lorsqu'un projet de rééditorialisation devient trop complexe pour être maintenu dans un seul et même atelier, il est possible de le structurer en sous-projets, isolés dans des ateliers par des ateliers calques ou fragments proxys. Ainsi, quelle que soit la complexité amenée par la documentarisation de l'activité, il est toujours possible de réduire à nouveau afin de simplifier la gestion et la maintenance du graphe documentaire. Nous soutenons ainsi que le passage à l'échelle de la structuration du graphe compense le surplus de complexité pouvant être introduit par la documentarisation de l'activité.

Synthèse

Le tableau suivant fait la synthèse du positionnement de nos propositions quant à leurs quatre critères d'évaluation.

	Documentarisation de l'activité	Structuration du graphe	Agencement des deux approches
Soutien de la rééditorialisation	Par construction	Par construction	Agencement neutre
Spécialisation de la technologie à ses contextes d'usages	Héritée de la modélisation des chaînes éditoriales ; À enrichir	Par construction	Meilleures capacités de spécialisations
Aspect évolutif de ces spécialisations	Hérité de la modélisation des chaînes éditoriales	Par construction	Meilleures capacités d'évolutions
Passage à l'échelle de la technologie	Non avéré	Par construction au niveau technique ; Par hypothèse au niveau fonctionnel	Passage à l'échelle garanti par la structuration du graphe

À défaut d'un recul suffisant sur les usages de la technologie développée, le tableau précédent synthétise nos hypothèses quant à la pertinence de nos propositions vis-à-vis des quatre critères d'évaluation retenus. La mention *par construction* indique que le respect de ce critère est pris en compte au cœur de la conception de l'objet technique et de la modélisation de nos propositions. Le terme *héritée de* indique que le respect de ce critère s'appuie directement sur un mécanisme déjà exploité par les chaînes éditoriales.

La dernière colonne est la plus difficile à évaluer. Nos propositions s'établissent sur deux champs fonctionnels différents (le modèle et l'articulation interne à un atelier pour la documentarisation de l'activité, l'articulation externe entre un atelier et le reste de la structure pour la seconde partie de nos propositions). À ce stade du projet, la complémentarité de ces deux parties n'est qu'hypothétique. Les hypothèses de complémentarité sont indiquées dans le

tableau et seul un recul plus important sur les usages nous permettra de les valider.

13.2 Évaluation des usages

Dans cette section, nous dressons un panorama des contextes d'usages accompagnés, des implémentations expérimentées, des retours issus de la production et des évolutions que nous souhaitons apporter dans les prochaines versions de la technologie développée.

13.2.1 Enaction Series

Rappel du contexte

Le contexte de production documentaire de la revue scientifique Enaction Series a été présenté au sein de la première section du chapitre 2 (p. 40).

La revue Enaction Series, portée par le laboratoire Costech *(www.utc.fr/costech)* de l'UTC, publie en ligne des ouvrages scientifiques dédiés au paradigme des sciences cognitives de l'énaction. L'objectif est de valoriser les travaux de chercheurs expérimentés tout en proposant un échange savant, appelé de la glose dans ce contexte, entre auteurs et jeunes chercheurs.

Outillage du projet

Comme illustré dans le chapitre 9 (p. 132), l'outillage du projet Enaction Series s'appuie sur une structure de deux ateliers : un atelier premier contenant les versions de référence des ouvrages scientifiques. Le contenu de cet atelier est utilisé pour publier la version web des ouvrages. L'atelier second instrumente une fonction de calque de travail (p. 123). Il est utilisé comme atelier de rédaction et de glose des contenus. Lorsqu'un nouvel ouvrage est rédigé, l'auteur transmet son écrit à l'éditeur qui s'assure sa médiation au sein de l'atelier second. Une fois l'auteur et l'éditeur en accord sur le contenu, une première version de l'ouvrage est validée et reversée depuis l'atelier second vers l'atelier premier afin d'être publiée sur le web.

L'étape de glose commence alors par l'intermédiaire de l'outil d'édition des commentaires en ligne (p. 97). Cette publication dynamique a été présentée au sein du chapitre 8 (p. 109). La glose est ainsi initiée par les jeunes chercheurs. S'ensuit une discussion entre les glossateurs et l'auteur avant d'éventuelles mises à jour du texte suite aux échanges ou à la publication de la glose.

Retours de l'expérimentation

Ce contexte de production est à la limite de ce que notre recherche vise à outiller. Comme le montre le graphe documentaire mobilisé pour la publication des ouvrages (p. 40), peu de fragments sont mobilisés, les ouvrages ne rééditorialisent pas de fragments entre eux et la densité globale du graphe reste faible.

Dans ce contexte, la technologie proposée n'a pas réellement correspondu aux attentes des éditeurs d'Enaction Series. Le système de commentaires a été considéré trop compliqué à prendre en main pour des rédacteurs peu coutumiers des technologies de l'édition documentaire. Le système de calque a été considéré comme une source de complexité plutôt que comme une assistance à la maintenance du graphe.

Ce contexte d'usages est intéressant car il montre une limite de nos propositions. Dans un cadre artisanal, avec un public de rédacteurs occasionnels sur un graphe de faible complexité, les solutions d'industrialisation que nous avons mises en place se sont avérées trop lourdes à exploiter pour apporter une réelle assistance.

Évolutions envisagées

À l'issue de ces trois années, ce contexte de production va sortir des terrains d'usages mobilisés par notre recherche. Un outillage plus simple, composé d'un unique atelier de production sans fragments pragmatiques va être mis en place afin de simplifier et pérenniser le projet éditorial.

13.2.2 DILA

Rappel du contexte

Le contexte de production de la Direction de l'Information Légale et Administrative est présenté au sein de la troisième section du chapitre 2 (p. 47).

La DILA utilise une chaîne éditoriale pour rédiger les contenus du site service-public.fr et la documentation de référence utilisée pour assister les opérateurs du service d'information administrative par téléphone (le 3939). Trois équipes s'occupent à plein temps de la rédaction et de la maintenance des contenus : les deux premières travaillent ensemble pour la majeure partie du graphe tandis que la dernière travaille uniquement sur les fragments de documentation du procédé de réponse aux appels d'offre.

En cas d'erreur identifiée sur le site service-public.fr, la mise à jour doit être effectuée rapidement, sans attendre que les autres travaux de maintenance en cours soient finalisés.

Outillage du projet

Comme illustré au sein du chapitre 9 (p. 132), la production documentaire de la DILA a été outillée par un atelier premier contenant la version de référence du graphe et deux ateliers seconds, chacun opéré par des ateliers calque de travail (p. 123). L'ensemble de la maintenance s'effectue dans les ateliers seconds. L'atelier premier contient les fragments de référence, toujours prêts à être publiés.

L'atelier second dédié aux équipes de maintenance des fiches téléphoniques et de la majeure partie du site web exploite un modèle d'activité présenté au sein du chapitre 8 (p. 103). Ce modèle définit les modalités d'exploitation de tâches de commande et de tâches de validation.

Retours sur les usages

La structure des ateliers mise en place est vitale pour le bon déroulement du projet éditorial. Sans un moyen efficace de garder une version de référence pouvant être publiée à chaque instant et sans maîtrise des mises à jour de cette version de référence, la production documentaire nécessaire à la maintenance du site service-public.fr ne peut pas se faire sereinement.

Le modèle d'activité développé est très fortement mobilisé par les rédacteurs. Plusieurs milliers de tâches ont été instanciées depuis la mise en production en décembre 2012. À titre d'exemple, le seul mois d'avril 2013 a vu la création de 339 nouvelles tâches.

Afin d'évaluer les opérations de maintenance, les rédacteurs en chef constituent des rapports d'activité recensant les différentes mises à jour effectuées par les rédacteurs. Ce type de document est à l'origine de notre proposition de documentarisation puisqu'il s'agit de produire des documents décrivant les actions réalisées par les rédacteurs sur le graphe. En raison de contraintes de coûts et de temps de développement liées à la mise en production de ce projet industriel, ces rapports d'activité s'appuient sur des structures documentaires constitutives des fragments de tâche. Lorsqu'une mise à jour est validée par un rédacteur en chef, la structure dédiée est complétée avec le nom du rédacteur à l'origine de la mise à jour et une indication de l'importance de celle-ci. Cette instrumentation peut être améliorée : le besoin de documentarisation de l'activité est clairement identifié mais son instrumentation s'appuie sur une saisie des rédacteurs au lieu d'une documentarisation automatique de leur activité. Le principe de documentarisation de l'activité, tel qu'il est exposé dans ce mémoire doit permettre d'enregistrer les actions de mises à jour directement au sein de fragments pragmatiques sans l'intermédiaire d'une saisie dédiée.

Évolution envisagée

La principale évolution envisagée concerne la documentarisation de l'activité des rédacteurs. Un nouveau type de fragment pragmatique pourrait être développé. Son objet serait d'enregistrer les actions liées à la validation des fragments depuis un atelier calque vers son atelier premier. D'un point de vue plus générique, ce développement pourrait ouvrir un nouveau champ de modélisation pour la documentarisation de l'activité en proposant d'enregistrer dans le

graphe toute action de contrôle des contenus entre un atelier premier et un atelier second (validation depuis un atelier calque, validation d'une dérivation, etc.). Cette évolution reste néanmoins contrainte par le contexte industriel de la production et devra attendre les circonstances nécessaires (mise à jour de la suite logicielle de la DILA, demande explicite du client, etc.) pour que les développements soit réalisés.

13.2.3 Quick

Rappel du contexte

Le contexte de production documentaire de la société Quick a été présenté au sein du chapitre 2 (p. 50).

Le département innovation rédige des dossiers d'homologation afin de faire évoluer les procédures en cours d'usages au sein des restaurants. Lorsqu'un dossier d'homologation est validé par l'expert métier de la société, le département I2M (Institut du Management et des Métiers), en charge de la maintenance de la documentation technique de référence récupère les nouvelles procédures et les intègre à la documentation existante.

Outillage existant

La mise en production du contexte industriel Quick a été menée avant nos travaux de recherche. À ce jour, aucun élément constitutif de nos propositions n'a été mis en place.

La chaîne éditoriale utilisée par Quick est constituée de deux ateliers (non reliés entre eux). Le premier est dédié à la rédaction des dossiers d'homologation et le second à la maintenance du référentiel. Lorsqu'un dossier d'homologation est validé par l'expert métier, les rédacteurs du référentiel se connectent à l'atelier du département innovation et copient une version du dossier au sein de l'atelier dédié au référentiel.

Retours sur les usages

L'outillage actuel pose problème. Comme exposé dans l'analyse du graphe mobilisé pour la production du référentiel menée dans le chapitre 2 (p. 52), le graphe mobilisé par les rédacteurs du référentiel ne peut pas être maîtrisé en l'état. De trop nombreuses copies réalisées vers ou depuis des espaces différents de l'arbre de gestion et selon des techniques de copies différentes empêchent la bonne exploitation du graphe.

Nous interprétons les difficultés rencontrées par les rédacteurs du référentiel comme une matérialisation concrète de la problématique à laquelle notre recherche s'intéresse. En n'assistant pas suffisamment les rédacteurs dans le contrôle de la maintenance du graphe, l'outil devient trop complexe à utiliser et les rédacteurs effectuent des copies pour limiter les interdépendances entre fragments. En réalisant ces copies, le graphe gagne en taille et en complexité et devient de plus en plus difficile à maintenir.

Évolutions envisagées

Dans ce mémoire, nous suggérons au sein du chapitre 10 (p. 147) une mise en œuvre de nos propositions afin de mieux outiller le contexte de production Quick.

Nous proposons une structure de 8 ateliers dédiés aux différents projets de rééditorialisation. Ces ateliers sont reliés entre eux par des fragments proxys et des ateliers calques de rédaction. Chaque atelier hébergeant une activité auctoriale dispose en outre de son propre modèle d'activité.

Nous soutenons qu'une telle chaîne éditoriale permet de pallier aux difficultés observées dans la production documentaire des dossiers d'homologation et des différentes versions du référentiel.

Au même titre que dans le contexte d'usages de la DILA, cette re-conception reste néanmoins contrainte par le contexte industriel de la production et devra attendre les circonstances nécessaires (mise à jour de la suite logicielle de Quick, demande explicite du client, etc.) pour que les développements soient réalisés.

13.2.4 Pédagogie UTC

Présentation des contextes

Au sein de l'université de technologie de Compiègne, deux contextes de production nécessitant certaines de nos propositions sont en train d'être initiés.

Le premier est porté par la Cellule d'Appui Pédagogique (CAP), adossée à l'unité ICS, qui a pour objet l'assistance technologique et méthodologique des enseignants de l'université qui en feraient la demande. Parmi les infrastructures techniques mises en place par la CAP, une chaîne éditoriale pédagogique opérant le modèle Opale est proposée aux enseignants. Une réflexion est en cours sur la structuration des ateliers adéquate afin de permettre aux enseignants de partager des contenus tout en disposant d'un espace de rédaction protégé. Les deux structures envisagées s'appuient sur un système de fragments proxys (p. 126) pour l'une et sur des ateliers calques de travail (p. 123) pour l'autre.

Le second projet est porté par l'unité ICS. Il consiste en un projet de recherche appelé Opale-Line s'intéressant à la fois aux innovations dans les supports pédagogiques, aux formes d'écriture multimédia et à la rééditorialisation de volumes de contenus importants. Le volet dédié au support pédagogique s'intéresse à l'usage de cours en ligne enrichis par un enregistrement sonore permettant l'écoute du cours plutôt que la lecture. Le volet sur l'écriture multimédia vise à proposer un environnement auteur pour le découpage d'une bande son et sa liaison avec les différents fragments qui composent un cours. Le volet qui nous intéresse est celui de la rééditorialisation. L'enjeu du processus de production est de désolidariser l'écriture des cours de leurs enregistrements sonores. Un traitement de masse des enregistrements sonores pourrait ainsi être pris en charge par la cellule TICE d'une université au même titre que la traduction des contenus pédagogiques. Cette rééditorialisation de masse pourrait par exemple se mettre en place dans un atelier calque de dérivation.

Ces deux contextes sont en cours de développement et leur mise en production prochaine permettra d'enrichir le cycle de développement, usages et modélisation de notre projet de recherche.

13.3 Synthèse

Une validité technique relative

La méthode de recherche et plus particulièrement la mise en place des cycles de développement, d'usages et de modélisation nous permet de diriger nos développements et de construire notre modélisation par rapport aux retours des contextes d'usages dès le commencement du projet. Cette méthode apporte très tôt de premières certitudes quant à la validité technique de nos propositions.

Ainsi, à ce stade, **nous pouvons valider sur le plan technique les éléments implémentés et en production dans des contextes industriels** soit :

- pour les fragments pragmatiques, le système de tâches, la gestion des fils de discussion et la gestion des responsabilités et cycles de vie sur les fragments ;
- pour la structuration des ateliers, les calques de travail et les calques de dérivation.

Cette validité technique ne peut être que **relative** car le projet n'est pas encore terminé. Certains développements ne sont pas encore réalisés à l'image des fragments proxys. D'autres méritent d'être enrichis comme les fragments pragmatiques où de nouvelles possibilités de modélisation devront être explorées et les ateliers calques où de nouvelles modalités de dérivation et de contrôle doivent être implémentées. Enfin les futurs usages accompagnés engendreront nécessairement de nouveaux besoins particuliers pour l'accompagnement de la rééditorialisation à échelle industrielle. Ces évolutions ne peuvent pas encore être anticipées à ce jour.

Une validation technologique encore impossible

La validation technologique, soit la validation du projet de recherche par rapport à sa problématique et ses objectifs, **ne peut pas encore être menée**. Nous soutenons que les usages que nous cherchons à exhumer - la production de documents rééditorialisés à l'échelle

industrielle - ne peut pas être mesuré après trois ans de projet. Il convient au préalable de terminer le projet de recherche et d'en mesurer ensuite les impacts sur l'évolution des usages. Cette évolution sera possible dans une perspective de plusieurs années.

Conclusion

Les travaux présentés dans ce mémoire s'intéressent à la problématique de l'industrialisation de la production de documents rééditorialisés. L'enjeu de cette industrialisation est d'accompagner l'activité des rédacteurs afin de permettre la constitution et l'exploitation de graphes de fragments rééditorialisés, quelles qu'en soient l'échelle et la complexité.

Propositions

Afin de répondre à cet enjeu, nous avons formalisé dans ce mémoire nos contributions théoriques, méthodologiques et technologiques développées au cours de ces trois années de recherche.

Sur le plan théorique, nous proposons deux approches complémentaires pour accompagner cette industrialisation. La première est bâtie autour d'une formalisation du discours pragmatique au sein de fragments modélisés pour cet usage et ré-exploités afin de proposer des fonctions d'assistance de l'activité. Cette production de fragments participe à la documentarisation de l'activité. La seconde consiste à projeter le graphe dans différents ateliers outillant ainsi des contextes de rééditorialisation dans des espaces simplifiés.

Sur le plan technologique, nous avons développé certaines de nos propositions au sein de l'équipe de développement de la société Kelis. Nous avons ainsi directement conçu et développé le système de gestion de commentaires. Nous avons effectué des développements préparatoires à l'implémentation des fragments proxys. Nous proposons en outre un protocole pour accompagner les transactions documentaires afin de permettre une distribution de nos propositions de structuration des ateliers.

Sur le plan méthodologique, nous proposons une méthode afin d'assister les concepteurs de chaînes éditoriales dans le développement de logiciels supports à une rééditorialisation d'ampleur industrielle. Cette méthode se structure en deux étapes. Une première étape de structuration des ateliers permet la mise en place d'une architecture support aux différents projets éditoriaux et auctoriaux d'une organisation. Une seconde de modélisation du discours pragmatique exploité dans chacun des ateliers permet d'assister chaque équipe de rédacteurs et participe à la documentarisation de l'activité sur le graphe.

Évaluation

L'aspect appliqué et technologique de notre recherche nous a amené à étudier les modalités de la validation de nos travaux. Nous avons ainsi fait émerger une proposition pour une épistémologie normative interne du programme de recherche technologique « rééditorialisation documentaire » directement adaptée des propositions de Theureau (2009) et une épistémologie descriptive de ce même programme.

Cette réflexion préalable nous a permis de situer l'avancement du projet de recherche englobant notre thèse de doctorat et d'en mener une évaluation intermédiaire cherchant à estimer la pertinence de nos propositions sur un plan technique et recensant les divers contextes d'usages ouverts.

Sur le plan technique, nous proposons quatre critères en relation avec nos hypothèses de recherche. Pour toutes nos propositions, il s'agit d'évaluer :

- le soutien des propositions à la rééditorialisation en lien avec l'objet du programme de recherche dans lequel nos travaux s'inscrivent ;
- les spécialisations possibles en lien avec l'hypothèse de meilleure efficacité des logiciels conçus spécifiquement pour un contexte d'usages ;
- l'aspect évolutif de la spécialisation en lien avec la méthodologie de recherche qui s'appuie sur des cycles courts entre développement d'une instance de test, usages et modélisation ;
- le passage à l'échelle pour supporter la rééditorialisation à l'échelle industrielle.

Une évaluation technique partielle peut être menée sur les éléments développés et mobilisés dans des contextes de production documentaire réels. Il est ainsi possible de valider l'ouverture de nouveaux usages suite aux éléments développés au cours de ces trois années de recherche. Cette validation technique n'est pas terminée et devra se poursuivre au fur et à mesures des prochains développements.

L'évaluation technologique de notre recherche, soit la validation que la technologie développée permet de réduire la tension entre rééditorialisation à grande échelle et cohérence des documents produits, pourra être menée une fois les contextes d'usages plus nombreux et plus avancés dans la phase de production. Cette évaluation permettra de clore le projet de recherche dans lequel s'inscrivent nos travaux.

Perspectives

Les deux approches que nous proposons donnent le contexte global dans lequel les prochains développements devront se poursuivre. Les deux fonctions de fragments proxys génériques doivent ainsi être développées, puis les éléments proposés dans chacune des deux approches devront être enrichis par de nouveaux modèles de fragments pragmatiques d'un côté et de nouvelles fonctions de proxys et de calques de l'autre.

Les cycles de développement, d'usages et de modélisation pourront ainsi se conclure sur une évaluation et donc une validation technologique qui s'appuiera autant sur l'analyse technique de l'outil et de l'ensemble des possibles ouverts que sur une analyse des usages et de leurs évolutions suite à la mobilisation de la technologie développée.

Bibliographie

- Advanced Distributed Learning (ADL). *Sharable Content Object Reference Model (SCORM).* http://www.adlnet.org/scorm
- Aerospace and Defence Industries of Europe (ASD). *International specification for technical publications. S1000D.* S1000D-I9005-01000-00. 2008.
- AICC. *Guidelines for Package Exchange Notification Services.* DOCUMENT No. CMI010. 2006. http://aicc.org/docs/tech/cmi010v1a.pdf
- André J, Futura R, Quint V. *Structured Documents.* Cambridge University Press, 1989. ISBN : 0-521-36554-6
- Arribe T, Crozat S, Bachimont B, Spinelli S. « Chaînes éditoriales numériques: allier efficacité et variabilité grâce à des primitives documentaires. ». *Actes du Colloque international sur le document électronique, CiDE. 15,«Métiers de l'information, des bibliothèques et des archives à l'ère de la différentiation numérique»,, Tunis, Tunisie, 2012.*
- Arribe T, Crozat S, Spinelli S. « «Content Interactive Delivery». Proposition de protocole pour l'encadrement et la fluidification des transactions entre systèmes documentaires. ». *Actes du Colloque JOCAIR 2014, «Enseigner sans enseignants ? Tendances et problèmes des arts et métiers numériques de la formation», Paris, 2014 (poster).*
- Arribe T, Crozat S, Spinelli S. « CID : un protocole de transaction pour les documents pédagogiques ». *Actes du colloque TICE, Béziers, 2014.*
- Austin J L, Lane G (traducteur). *Quand dire, c'est faire. Points* : Éditions du Seuil, 1991. ISBN : 978-2-02-012569-7
- Austin J L, Urmson J O, Sbisà M (eds). *How to do Things with Words.* Oxford : Clarendon Press, 1962. ISBN : 0-674-41152-8
- Austin J L, Urmson J O, Warnock G J (eds). *Philosophical Papers.* Oxford : Clarendon Press, 1979. ISBN : 0-19-824627-7
- Austin J L, Warnock G J (ed). *Sense and Sensibilia.* Oxford : Clarendon Press, 1962. ISBN : 0-19-824579-3
- Bachimont B, Cailleau I, Crozat S, Majada M, Spinelli S. « Outils auteurs : approche industrielle versus approche artisanale ». *Actes du colloque ARIADNE'2002, Lyon, 2002.* http://edutice.archives-ouvertes.fr/docs/00/02/70/06/PDF/bachimont.pdf
- Bachimont B, Charlet J. « PolyTeX : un environnement pour l'édition structurée de polycopiés électroniques multisupports ». *Actes de la conférence « EuroTeX'98 », Saint Malo, France, 1998.*
- Bachimont B, Crozat S. « Instrumentation numérique des documents : pour une séparation fonds/forme ». *Information - Interaction - Intelligence*, 2004. Vol. 4 N°1. http://archivesic.ccsd.cnrs.fr/sic_00001017.html
- Bachimont B. *Ingénierie des connaissances et des contenus : le numérique entre ontologies et documents.* Paris : *Hermès* : Lavoisier, 2007. Science informatique et SHS. ISBN : 978-2-7462-1369-2
- Bachimont B. *Le sens de la technique : le numérique et le calcul. Encres Marines* : Les belles lettres, 2010. À présent. ISBN : 978-2-35088-035-8
- Barbier F. *Histoire du livre en occident. 3ième édition* : Armand Colin, 2012. ISBN : 978-2-200-28154-0

- Barthes R. *La chambre claire : Note sur la photographie, Cahiers du Cinéma.* Paris : Gallimard Seuil, 1980. ISBN : 2-07-020541-X
- Bastian M, Heymann S, Jacomy M. « Gephi: An Open Source Software for Exploring and Manipulating Networks ». *Proceedings of the Third International ICWSM Conference, San Jose, California, 2009.* The AAAI Press. p361-362. https://www.aaai.org/ocs/index.php/ICWSM/09/paper/view/154
- Bézivin J. « On the unification power of models ». *Software & Systems Modeling*, 2005. Volume 4, Issue 2, p 171-188. DOI : 10.1007/s10270-005-0079-0
- Bitbol M. *La Pratique des Possibles. Une Lecture Pragmatiste et Modale de la Mécanique Quantique.* Enaction Series, 2013. http://enactionseries.com/library/bookmb
- Brambilla M, Cabot J, Wimmer M. *Model-Driven Software Engineering in Practice.* Morgan & Claypool Publishers, 2012. ISBN : 9781608458820
- Bretto A, Faisant A, Hennecart F. *Éléments de théorie des graphes. Collection IRIS* : Springer, 2012. ISBN : 978-2-8178-0280-0
- Briet S. *Qu'est-ce que la documentation.*. Éditions documentaires et industrielles, 1951. http://martinetl.free.fr/suzannebriet/questcequeladocumentation/
- Bush V. « As we may think ». *The atlantic monthly*, 1945. 101-108. http://www.theatlantic.com/magazine/archive/1945/07/as-we-may-think/303881
- Button G, Harper R. « Taking the organisation into account ». *In : Technology in Working Order: Studies of Work, Interaction and Technology.* Button G, 1993. p 98-107.
- Button G. « What's wrong with speech-act theory ». *Computer Supported Cooperative Work (CSCW)*, 1994. Volume 3, Issue 1, p 39-42. DOI : 10.1007/BF01305842
- Cassirer E. *Substance et fonction. Éléments pour une théorie du concept.* Les Éditions de Minuit, 1977. ISBN : 978-2-7073--0186-4
- Crozat S, Bachimont B, Cailleau I, Bouchardon S, Gaillard L. « Éléments pour une théorie opérationnelle de l'écriture numérique ». *Document numérique*, 2011. vol. 14/3, p 9-33. DOI : 10.3166/dn.14.3.9-33
- Crozat S, Delestre N, Qyeyrut J, Baillon F, Vanoirbeek C, Velut P, Hennequin X. « Standardisation des formats documentaires pour les chaînes éditoriales d'UNIT : un schéma pivot ». *Actes du colloque TICE, Toulouse, 2006.* http://hal.archives-ouvertes.fr/hal-00442860/
- Crozat S. « Chaînes éditoriales et rééditorialisation de contenus numériques ». *In :* Calderan L. *Le document numérique à l'heure du web de données.* ADBS éditions, 2012. p179-220. ISBN : 978-2-84365-142-7 ; http://hal.inria.fr/hal-00740268
- Crozat S. *Éléments pour la conception industrialisée des supports pédagogiques numériques.* Thèse de doctorat, Université de technologie de Compiègne, 2002.
- Crozat S. *Scenari, la chaîne éditoriale libre.* Eyrolles, 2007. ISBN : 978-2-212-12150-6
- De Alwis J, Sillito J. « Why are software projects moving from centralized to decentralized version control systems? ». *Workshop on Cooperative and Human Aspects on Software Engineering, 2009. CHASE '09. ICSE, Vancouver, BC, 2009.* IEEE. p36-39. DOI : 10.1109/CHASE.2009.5071408
- De Michelis G, Grasso M. A. « Situating conversations within the language/action perspective: the Milan conversation model ». *Proceeding CSCW '94 Proceedings of the 1994 ACM conference on Computer supported cooperative work, Chapel Hill, North Carolina, USA, 1994.* p 89-100. DOI : 10.1145/192844.192880
- Dourish P. « Process descriptions as organisational accounting devices: the dual use of workflow technologies ». *Proceedings of the 2001 International ACM SIGGROUP*

Conference on Supporting Group Work, Boulder, Colorado, USA., 2001. ACM. p 52-60. DOI : 10.1145/500286.500297

- Dumas M, Van Der Aalst W. M, Ter Hofstede A. H. *Process-Aware Information Systems: Bridging People and Software Through Process Technology.* Wiley, 2005. ISBN : 978-0-471-66306-5
- Dunn M. « Single-source publishing with XML ». *IT Professional*, 2003. Volume:5 , Issue: 1, p51-54. DOI : 10.1109/MITP.2003.1176491
- Ellis C A, Gibbs S J. « Concurrency control in groupware systems ». *Proceedings of the 1989 ACM SIGMOD international conference on Management of data, 1989.* p399-407. DOI : 10.1145/67544.66963
- Fussell S, Lutters W. *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing.* Baltimore, MD, USA : ACM, 2014. ISBN : 978-1-4503-2540-0
- Gebers E, Crozat S. « Chaînes éditoriales Scenari et unité ICS. Solutions pour la production documentaire numérique ». *Distances et savoirs*, 2009. Vol. 7, Num 3, p421-442. DOI : 10.3166/ds.7.421-442
- Goody J. *La raison graphique: la domestication de la pensée sauvage.*. Les Éditions de Minuit, 1979. ISBN : 978-2-7073-0240-3
- Hopkins B, Wilson R J. « The Truth about Konigsberg. ». *The Genius of Euler. The MAA Tercentenary Euler Celebration., 2007.*
- Jacomy M, Venturini T, Heymann S, Bastian M. « ForceAtlas2, a Continuous Graph Layout Algorithm for Handy Network Visualization Designed for the Gephi Software ». *PLOS ONE*, 2014. DOI : 10.1371/journal.pone.0098679
- Kabbedijk J, Jansen S. « Steering Insight: An Exploration of the Ruby Software Ecosystem ». *In : Regnell B, Van De Weerd I, De Troyer O. Software Business. Second International Conference, ICSOB 2011, Brussels, Belgium, June 8-10, 2011. Proceedings, Bruxelles, Belgique, 2011.* Springer. p 44-55. DOI : 10.1007/978-3-642-21544-5_5
- Koyré A. *Études d'histoire de la pensée scientifique. TEL* : Gallimard, 1973. ISBN : 9782070703357
- Kruskal J B. « On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem ». *Proceedings of the American Mathematical Society*, 1956. Vol. 7, No. 1, p48-50. DOI : 10.1090/S0002-9939-1956-0078686-7
- Kuhn T S. *La structure des révolutions scientifiques.* Paris : Flammarion, 1972. ISBN : 2080811150
- Lakatos I. *Histoire et méthodologie des sciences. Programmes de recherche et reconstruction rationnelle. Bibliothèque d'histoire des sciences* : Presses Universitaires de France, 1994. ISBN : 2-13-045599-9
- Leroi-Gourhan A. *L'homme et la matière. Sciences d'aujourd'hui* : Albin Michel, 1971. ISBN : 978-2-226-06213-0
- Levie F. *L'Homme qui voulait classer le monde, Paul Otlet et le Mundaneum.* Les impressions nouvelles, 2006. ISBN : 2-87449-022-9
- Lin S. « Computer solutions of the traveling salesman problem ». *The Bell System Technical Journal*, 1965. Volume 44, Issue 10, p2245 - 2269. DOI : 10.1002/j.1538-7305.1965.tb04146.x
- Marichal R. « Le livre des prieurs de Sorbonne (1431-1485) ». *In :* Monfrin J. *Les lectures de Guillaume Fichet et de Jean Heynlin d'après le registre de prêts de la Bibliothèque de la Sorbonne.* Bibliothèque d'Humanisme et Renaissance, 1987. p7-23.

- Nelson T H. *Literary Machines.* Mindful Press, 1981. ISBN : 0-89347-062-7
- OASIS. *Darwin Information Typing Architecture (DITA).* OASIS Standard. 2010. http://docs.oasis-open.org/dita/v1.2/os/spec/DITA1.2-spec.html
- OASIS. *Open Document Format for Office.* OASIS Standard. 2005. https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=office
- OASIS. *RELAX NG Specification.* OASIS Committee Specification. 2001. http://relaxng.org/spec-20011203.html
- OASIS. *The DocBook Schema.* OASIS Standard. 2009. http://docs.oasis-open.org/docbook/specs/docbook-5.0-spec-os.html
- Object Management Group (OMG). *OMG Unified Modeling Language - Infrastructure.* OMG Document number : formal/2011-08-05. 2011. http://www.omg.org/spec/UML/2.4.1/
- Object Management Group (OMG). *OMG Unified Modeling Language - Superstructure.* OMG Document number : formal/2011-08-06. 2011. http://www.omg.org/spec/UML/2.4.1/
- Otlet P. *Traité de documentation : le livre sur le livre, théorie et pratique.* Bruxelles : Editiones Mundaneum, 1934. http://search.ugent.be/meercat/x/view/rug01/000990276
- Pédauque R T. *Document : forme, signe et médium, les re-formulations du numérique.* 2003. http://archivesic.ccsd.cnrs.fr/sic_00000511/
- Pédauque R T. *Document et modernités.* 2006. http://archivesic.ccsd.cnrs.fr/sic_00001741
- Pédauque R T. *Le texte en jeu. Permanence et transformations du document.* 2005. http://archivesic.ccsd.cnrs.fr/sic_00001401/
- Popper K R. *La logique de la découverte scientifique.* Payot, 1973. ISBN : 2-228-11391-3
- Rothenberg J. « The nature of modeling ». *In :* Widman L E, Loparo K A, Nielsen N R. *Artificial intelligence, simulation & modeling.* John Wiley & Sons, Inc., 1989. p75-92. ISBN : 0-471-60599-9
- Sachs H, Stiebitz M, Wilson Rs J. « An historical note: Euler's Königsberg letters ». *Journal of Graph Theory*, 1988. Volume 12, Issue 1, p133-139. DOI : 10.1002/jgt.3190120114
- Salaün J M. « La redocumentarisation, un défi pour les sciences de l'information ». *Études de communication*, 2007. Num. 30, p13-23. DOI : 10.4000/edc.428
- Scandura J M. « AuthorIT: Breakthrough in authoring adaptive and configurable tutoring systems? ». *Technology, Instruction, Cognition and Learning*, 2005. Vol. 2, p185-230.
- Searle J R. *Speech acts: An essay in the philosophy of language.* Cambridge university press, 1969. ISBN : 0-521-09626-X
- Stewart J. *Questioning Life and Cognition: Some Foundational Issues in the Paradigm of Enaction.* Enaction Series, 2013. http://enactionseries.com/library/bookjs
- Stiegler B. *La technique et le temps. Tome 1 : la faute d'Épiméthée.* Paris : Galilée, 1994. ISBN : 2-7186-0440-9
- Suchman L. « Do Categories Have Politics? The language/action perspective reconsidered ». *In : De Michelis G, Simone C, Schmidt K. Proceedings of the Third European Conference on Computer-Supported Cooperative Work, Milan, Italy, 1993.* Springer. p1-14. DOI : 10.1007/978-94-011-2094-4_1
- Suchman L. « Office procedure as practical action: models of work and system design ». *ACM Transactions on Information Systems (TOIS)*, 1983. Volume 1 Issue 4, p 320-328. DOI : 10.1145/357442.357445

- Theureau J. *Le cours d'action : méthode développée. Collection Travail & activité humaine* : Octares Éditions, 2006. ISBN : 2-915346-35-6
- Theureau J. *Le cours d'action : méthode élémentaire. Collection Travail & activité humaine* : Octares Éditions, 2004. ISBN : 978-2-915346-05-4
- Theureau J. *Le cours d'action : méthode réfléchie. Collection Travail & activité humaine* : Octares Éditions, 2009. ISBN : 978-2-915346-64-0
- W3C. *Extensible Markup Language.* W3C Recommendation. 2008. http://www.w3.org/TR/REC-xml/
- W3C. *HTML5.* W3C Candidate Recommendation 31 July 2014. 2014. http://www.w3.org/TR/2014/CR-html5-20140731/
- W3C. *XML Path Language (XPath).* W3C Recommendation. 1999. http://www.w3.org/TR/xpath/
- W3C. *XML Schema Requirements.* W3C Note. 1999. http://www.w3.org/TR/NOTE-xml-schema-req
- Wang D, Mah H, Lassen S. *Google Wave Operational Transformation.* Google Inc., 2010. Google White Paper
- Winograd T, Flores F. *Understanding Computer and Cognition.* Ablex Publishing Corporation, 1986. ISBN : 0-89391-050-3
- Winograd T. « A language/action perspective on the design of cooperative work ». *Proceeding CSCW '86 Proceedings of the 1986 ACM conference on Computer-supported cooperative work, Austin, Texas, USA, 1986.* p203-220. DOI : 10.1145/637069.637096
- Ying A T T, Wright J L, Abrams S. « Source code that talks: an exploration of Eclipse task comments and their implication to repository mining ». *Proceedings of the 2005 international workshop on Mining software repositories, Saint Louis, Missouri, USA, 2005.* p1-5. DOI : 10.1145/1083142.1083152
- Zacklad M. « Processus de documentarisation dans les Documents pour l'Action (DopA) : statut des annotations et technologies de la coopération associées ». *Actes du colloque "Le numérique : Impact sur le cycle de vie du document pour une analyse interdisciplinaire"., Montréal, Québec., 2005.* Éditions de l'ENSSIB.. http://archivesic.ccsd.cnrs.fr/sic_00001072/
- Zacklad M. « Réseaux et communautés d'imaginaire documédiatisées ». *In :* Skare R, Lund W. L, Varheim A. *A Document (Re)turn.* Peter Lang, 2007. p279-297. ISBN : 978-3-631-56294-9
- Zanetti M S, Schweitzer F. « A Network Perspective on Software Modularity ». *ARCS Workshops, Muenchen, 2012.* p175-186. ISBN : 978-3-88579-294-9
- Zhu H, Hall P A V, May J H R. « Software unit test coverage and adequacy ». *ACM Computing Surveys (CSUR)*, 1997. Volume 29 Issue 4, p366-427. DOI : 10.1145/267580.267590

Annexes

Annexe 1

Glossaire

Concepts théoriques 202
La documentarisation 202
La raison graphique 203
La redocumentarisation 203
La raison computationnelle 204
La rééditorialisation 205
Discours pragmatique 205
Concepts techniques génériques 206
Chaîne éditoriale numérique 206
Ingénierie dirigée par les modèles 207
Graphe 209
Fragment de document 210
Graphe documentaire 210
Sous-graphe document 211
Atelier 212
Modèle documentaire 212
Classe de fragment 213
Concepts techniques proposés 213
Atelier calque 213
Atelier maître 214
Fragment proxy 214
Modèle d'activité 215
Fragment de tâche 216
Responsabilité 216
Cycle de vie 217
Commentaires 217
Atelier calque de travail 217

Atelier calque de dérivation 218
Fragment proxy de partage 218
Fragment proxy de dérivation 218

Ce glossaire reprend et définit les principaux concepts utilisés dans ce mémoire. Pour chaque concept, nous proposons :

- une définition ;
- un ensemble de relations montrant l'articulation avec les autres concepts ;
- les références bibliographiques liées.

Ce glossaire est segmenté en trois parties. La première est dédiée aux concepts théoriques mobilisés par ce mémoire, la seconde reprend les concepts techniques mobilisés et pré-existants à notre recherche, la dernière détaille les concepts techniques proposés dans ce mémoire.

Concepts théoriques

La documentarisation

Termes liés

Mobilise : La raison graphique (p. 203)
Précède : La redocumentarisation (p. 203)

Définition

La documentarisation est un terme mobilisé par le collectif Pédauque pour désigner l'accélération de la production documentaire survenue lors des différentes révolutions industrielles. Le mouvement de documentarisation est accompagné par l'émergence du métier de documentaliste et plus généralement, des techniques et méthodes de classification et d'indexation des documents au sein de centres de documentation.

Zacklad propose un sens plus orienté sur l'exploitation des documents. Il s'agit de « doter ces supports d'attributs spécifiques permettant de faciliter (i) leur gestion parmi d'autres supports, (ii) leur manipulation physique, condition d'une navigation sémantique à l'intérieur du contenu sémiotique et enfin, (iii) l'orientation des récepteurs » (Zacklad, 2005, p. 11)

Exploitation

Nous mobilisons le terme documentarisation pour désigner la production d'une documentation liée à une activité. Dans ce mémoire, l'expression « documentarisation de l'activité » désigne donc la production d'une documentation liée à l'activité des rédacteurs sur un graphe de fragments.

Références

- Pédauque R T. *Document : forme, signe et médium, les re-formulations du numérique.* 2003. http://archivesic.ccsd.cnrs.fr/sic_00000511/
- Pédauque R T. *Le texte en jeu. Permanence et transformations du document.* 2005. http://archivesic.ccsd.cnrs.fr/sic_00001401/
- Pédauque R T. *Document et modernités.* 2006. http://archivesic.ccsd.cnrs.fr/sic_00001741
- Levie F. *L'Homme qui voulait classer le monde, Paul Otlet et le Mundaneum.* Les impressions nouvelles, 2006. ISBN : 2-87449-022-9

- Otlet P. *Traité de documentation : le livre sur le livre, théorie et pratique.* Bruxelles : Editiones Mundaneum, 1934. http://search.ugent.be/meercat/x/view/rug01/000990276
- Briet S. *Qu'est-ce que la documentation.*. Éditions documentaires et industrielles, 1951. http://martinetl.free.fr/suzannebriet/questcequeladocumentation/
- Bush V. « As we may think ». *The atlantic monthly*, 1945. 101-108. http://www.theatlantic.com/magazine/archive/1945/07/as-we-may-think/303881
- Zacklad M. « Processus de documentarisation dans les Documents pour l'Action (DopA) : statut des annotations et technologies de la coopération associées ». *Actes du colloque "Le numérique : Impact sur le cycle de vie du document pour une analyse interdisciplinaire"., Montréal, Québec., 2005.* Éditions de l'ENSSIB.. http://archivesic.ccsd.cnrs.fr/sic_00001072/

La raison graphique

Terme lié

Précède : La raison computationnelle (p. 204)

Définition

La raison graphique est l'expression utilisée par Goody pour désigner une rationalité propre aux civilisations mobilisant un support graphique pour la conception et la transmission des connaissances.

Goody fait émerger trois structures logiques propres à la représentation des connaissances dans une rationalité graphique : la liste, le tableau et la formule.

L'hypothèse défendue par Goody est que la construction des sociétés modernes, autant dans l'organisation administrative que dans la construction du savoir, repose sur une rationalité propre au support graphique.

Référence

- Goody J. *La raison graphique: la domestication de la pensée sauvage.*. Les Éditions de Minuit, 1979. ISBN : 978-2-7073-0240-3

La redocumentarisation

Termes liés

Mobilise : La raison computationnelle (p. 204)

Suit : La documentarisation (p. 202)

Implique : La rééditorialisation (p. 205)

Définitions

Le terme redocumentarisation est mobilisé par trois auteurs différents pour désigner des réalités différentes :

- Le collectif Pédauque désigne de manière très générique le nouvel ordre documentaire issu de l'émergence des documents numériques. Le terme désigne « à la fois un retour sur une documentarisation ancienne et une révolution documentaire » (Pedauque, 2006)
- Zacklad désigne une nouvelle action de documentarisation en prenant en compte les possibles exploitations numériques. Il s'agit donc de faciliter l'accès au document, autant en interne entre un document et ses parties (annotation, extraction et ré-agencement), qu'en externe pour faciliter l'accès au document (gestion des collections, archives).
- Salaün désigne l'action de numériser les documents issus de la documentarisation pour permettre leur exploitation dans ce nouvel ordre documentaire. Il s'agit de l'exploitation

externe de la définition de Zacklad.

Exploitation

Au sein de ce mémoire, nous mobilisons la définition plus large et consensuelle du collectif Pédauque. Le terme de redocumentarisation souligne donc à la fois une accélération de la production documentaire et un nouvel ordre documentaire où les modalités de création, d'appropriation et d'accès sont remises en jeu.

Références

- Pédauque R T. *Document et modernités.* 2006. http://archivesic.ccsd.cnrs.fr/sic_00001741
- Zacklad M. « Processus de documentarisation dans les Documents pour l'Action (DopA) : statut des annotations et technologies de la coopération associées ». *Actes du colloque "Le numérique : Impact sur le cycle de vie du document pour une analyse interdisciplinaire"., Montréal, Québec., 2005.* Éditions de l'ENSSIB.. http://archivesic.ccsd.cnrs.fr/sic_00001072/
- Salaün J M. « La redocumentarisation, un défi pour les sciences de l'information ». *Études de communication*, 2007. Num. 30, p13-23. DOI : 10.4000/edc.428

La raison computationnelle

Terme lié

Suit : La raison graphique (p. 203)

Définition

La raison computationnelle est une théorie proposée par Bachimont qui s'appuie sur les travaux ethnographiques de Goody pour proposer de façon exploratoire les contours d'une rationalité propre au support numérique pour la conception et la transmission des connaissances.

Bachimont fait évoluer les propositions de Goody en trois nouvelles structures logiques mobilisées pour la représentation des connaissances dans une rationalité computationnelle :

- le programme succède à la liste ;
- le réseau succède au tableau ;
- la couche succède à la formule.

En outre, Bachimont propose une quatrième structure complémentaire pour la raison graphique : le schéma ; et son pendant numérique : la maquette numérique.

Extension

Les figures du schéma et de la maquette numérique nous semblent être des cas particuliers qui s'inscrivent dans les figures, plus larges, du modèle et de la méthode d'ingénierie dirigée par les modèles.

Dans son usage classique, un modèle est une représentation simplifiée d'une réalité. Lors de la conception d'artefacts, différents modèles sont mobilisés pour planifier certains aspects d'un objet technique.

L'ingénierie dirigée par les modèles s'appuie sur un méta-modèle qui définit l'ensemble des possibles pour une modélisation donnée. Un programme de modélisation permet de manipuler les modèles en garantissant leur conformité au méta-modèle.

Mobilisation

Notre recherche s'inscrit dans la théorie de Bachimont puisqu'il s'agit de s'appuyer sur ces structures logiques et sur les propriétés intrinsèques du numérique pour explorer les possibilités de conception des écrits sur un support numérique.

Références

- Bachimont B. *Ingénierie des connaissances et des contenus : le numérique entre ontologies et documents.* Paris : *Hermès* : Lavoisier, 2007. Science informatique et SHS. ISBN : 978-2-7462-1369-2
- Bachimont B. *Le sens de la technique : le numérique et le calcul. Encres Marines* : Les belles lettres, 2010. À présent. ISBN : 978-2-35088-035-8

La rééditorialisation

Termes liés

Mobilise : La raison computationnelle (p. 204)

S'inscrit dans : La redocumentarisation (p. 203)

Est instrumenté par : Chaîne éditoriale numérique (p. 206)

Définition

« La construction de ce mot tente une première synthèse entre les concepts d'édition au sens de publication d'une œuvre, d'éditorialisation au sens d'expression d'un point de vue propre, de réédition au sens de nouvelle proposition de lecture. Elle tente une seconde synthèse entre les fonctions d'éditeur, celui qui met en forme et diffuse, et d'auteur , celui qui écrit, fonctions qui tendent à se mêler dans le contexte du numérique. La rééditorialisation est donc la publication d'une œuvre originale dans son point de vue, sa forme, sa scénarisation, à partir de contenus qui ne le sont pas tous »(Crozat, 2012, p. 189).

Fonctions de rééditorialisation

La rééditorialisation s'opère par le biais de fonctions d'écriture permettant le découpage d'un document en fragments et le ré-assemblage des fragments entre eux.

Le ré-assemblage des fragments peut se faire en conservant chacun des fragments dans sa forme d'origine. La fonction d'assemblage s'appelle ici une **transclusion**.

À l'inverse, le ré-assemblage peut se faire en modifiant les fragments combinés. Cette modification est appelée **surcharge**. Elle peut être réalisée par une réécriture de certains passages d'un fragment, on parlera ici de **dérivation**. Elle peut également être réalisée en programmant des modifications automatiques (comme par l'injection de variable ou la sélection de fragments à profondeur variable), on parlera alors de **déclinaison**.

Références

- Crozat S. « Chaînes éditoriales et rééditorialisation de contenus numériques ». *In :* Calderan L. *Le document numérique à l'heure du web de données.* ADBS éditions, 2012. p179-220. ISBN : 978-2-84365-142-7 ; http://hal.inria.fr/hal-00740268
- Nelson T H. *Literary Machines.* Mindful Press, 1981. ISBN : 0-89347-062-7

Discours pragmatique

Terme lié

Est modélisé par : Modèle d'activité (p. 215)

Définition

La pragmatique est une discipline d'analyse de l'action réalisée par le langage. Initialement développée par Austin et Searle, la discipline a largement inspiré le domaine du Travail Coopératif Assisté par Ordinateur dans la conception de systèmes informatiques d'assistance à l'activité coopérative. Cette discipline a également été mobilisée par Zacklad pour

formaliser le concept de Document pour l'Action.

Dans ce mémoire, nous mobilisons la notion de « discours pragmatique » pour désigner de façon générique l'ensemble du discours ayant attrait à l'organisation de l'action d'un ou de plusieurs rédacteurs.

Références

- Austin J L, Warnock G J (ed). *Sense and Sensibilia.* Oxford : Clarendon Press, 1962. ISBN : 0-19-824579-3
- Austin J L, Urmson J O, Sbisà M (eds). *How to do Things with Words.* Oxford : Clarendon Press, 1962. ISBN : 0-674-41152-8
- Austin J L, Urmson J O, Warnock G J (eds). *Philosophical Papers.* Oxford : Clarendon Press, 1979. ISBN : 0-19-824627-7
- Austin J L, Lane G (traducteur). *Quand dire, c'est faire. Points* : Éditions du Seuil, 1991. ISBN : 978-2-02-012569-7
- Searle J R. *Speech acts: An essay in the philosophy of language.* Cambridge university press, 1969. ISBN : 0-521-09626-X
- Winograd T, Flores F. *Understanding Computer and Cognition.* Ablex Publishing Corporation, 1986. ISBN : 0-89391-050-3
- Winograd T. « A language/action perspective on the design of cooperative work ». *Proceeding CSCW '86 Proceedings of the 1986 ACM conference on Computer-supported cooperative work, Austin, Texas, USA, 1986.* p203-220. DOI : 10.1145/637069.637096
- Zacklad M. « Processus de documentarisation dans les Documents pour l'Action (DopA) : statut des annotations et technologies de la coopération associées ». *Actes du colloque "Le numérique : Impact sur le cycle de vie du document pour une analyse interdisciplinaire"., Montréal, Québec., 2005.* Éditions de l'ENSSIB.. http://archivesic.ccsd.cnrs.fr/sic_00001072/
- Zacklad M. « Réseaux et communautés d'imaginaire documédiatisées ». *In :* Skare R, Lund W. L, Varheim A. *A Document (Re)turn.* Peter Lang, 2007. p279-297. ISBN : 978-3-631-56294-9

Concepts techniques génériques

Chaîne éditoriale numérique

Termes liés

Instrumente : La rééditorialisation (p. 205)

Mobilise : L'ingénierie dirigée par les modèles (p. 207)

Permet l'édition de : Graphe documentaire (p. 210)

Exploite : Modèle documentaire (p. 212)

Exploite : Modèle d'activité (p. 215)

Définition

Dans son acceptation la plus générique, une chaîne éditoriale numérique est un logiciel mobilisé pour couvrir l'ensemble des métiers d'une chaîne éditoriale classique (auteur, éditeur, imprimeur) sur un support numérique. Une chaîne éditoriale numérique permet donc d'assister des rédacteurs dans la production et la publication de leurs contenus.

Dans ce mémoire, le terme « chaîne éditoriale » désigne les chaînes éditoriales numériques XML. Il s'agit de chaînes éditoriales numériques qui enregistrent les documents écrits par les rédacteurs dans un format conforme au standard XML et qui permettent ensuite la **génération** (ou **publication**) des fichiers XML en fichiers au format standard comme PDF,

HTML ou Open Document.

Le schéma XML permettant l'édition et le contrôle des documents ainsi que les algorithmes de génération sont appelés **modèle documentaire**.

Référence

- Crozat S. *Scenari, la chaîne éditoriale libre.* Eyrolles, 2007. ISBN : 978-2-212-12150-6
- Crozat S. « Chaînes éditoriales et rééditorialisation de contenus numériques ». *In :* Calderan L. *Le document numérique à l'heure du web de données.* ADBS éditions, 2012. p179-220. ISBN : 978-2-84365-142-7 ; http://hal.inria.fr/hal-00740268

Ingénierie dirigée par les modèles

Terme lié

Est mobilisé par : Chaîne éditoriale numérique (p. 206)

Définition

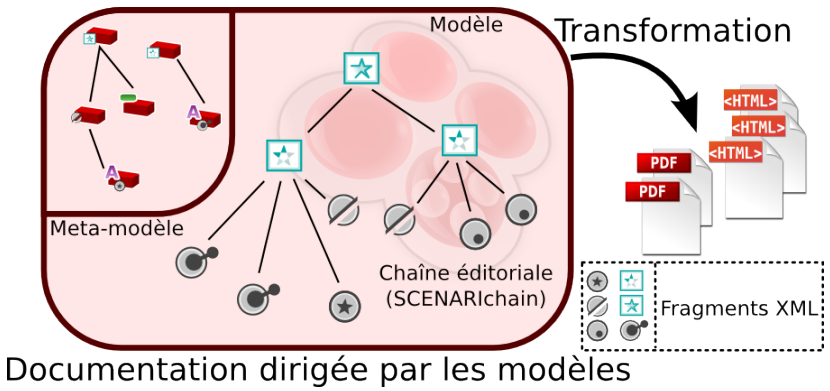
L'ingénierie dirigée par les modèles est une discipline qui étudie l'exploitation des modèles pour la conception d'artefacts. Elle prône un renversement de l'exploitation des modèles d'une simple définition pour clarifier un aspect de la conception à une réelle ressource opérationnelle pour la production de l'artefact. Les modèles sont alors utilisés comme données d'entrée d'un algorithme de transformation qui produit ainsi tout ou partie de l'artefact désiré.

Une méthode de conception exploitant l'ingénierie dirigée par les modèles s'appuie sur l'usage d'un méta-modèle : une définition formelle et interprétable par un logiciel de modélisation d'un langage de modélisation. La validité des modèles ainsi conçus peut être contrôlée auprès du méta-modèle. Un algorithme de transformation peut ensuite être conçu à partir du méta-modèle afin de transformer n'importe quel modèle en artefact final.

Par son principe de modélisation et de transformation, l'ingénierie dirigée par les modèles permet de concevoir des artefacts en abstrayant les éléments simplifiés par le modèle.

Exploitation par les chaînes éditoriales

Le principe des chaînes éditoriales numériques XML repose sur une approche d'ingénierie dirigée par les modèles. Le modèle documentaire fournit le langage d'expression et de contrôle des fragments. La chaîne éditoriale mobilisée pour l'écriture est le logiciel de modélisation. L'action d'écrire peut alors être rapportée à la modélisation et les fragments de documents sont des morceaux de modèles de documents. L'algorithme de publication qui transforme des fragments de documents écrits dans un langage XML en documents au format standard est l'algorithme de génération.



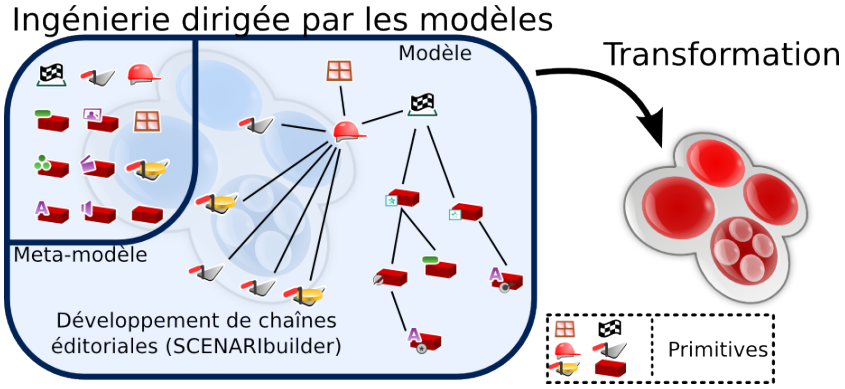
Ingénierie dirigée par les modèles pour la production de documents

Cette méthode d'ingénierie dirigée par les modèles permet d'abstraire l'environnement technologique standard d'expression des documents, rendant ainsi possible une approche

multi-formats où plusieurs algorithmes de transformation permettront de générer les mêmes fragments de documents dans plusieurs formats standards différents.

Exploitation pour la modélisation des chaînes éditoriales

La suite Scenari exploite une seconde fois ce principe de modélisation et de génération pour la conception des modèles documentaires. Le logiciel de modélisation de chaînes éditoriales SCENARIbuilder intègre un méta-modèle de chaînes éditoriales qui définit les modalités d'expression d'un modèle documentaire, soit des classes de fragments et leurs transformations. SCENARIbuilder permet l'édition des primitives qui constituent le modèle et la transformation de ces primitives en une archive de code source appelée WorkspacePack. Cette archive contient l'ensemble du code source à associer à un atelier pour permettre l'exploitation d'un graphe.

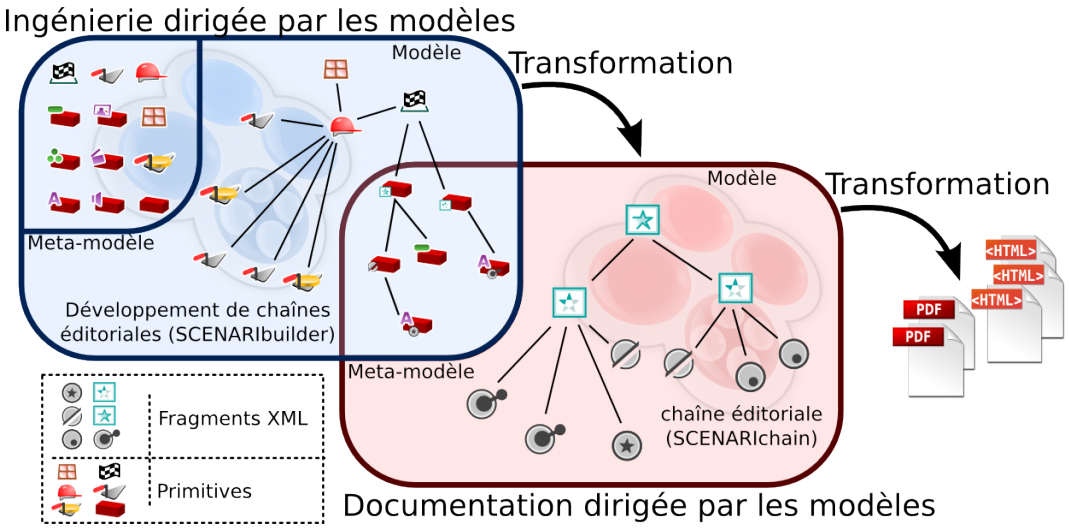


Ingénierie dirigée par les modèles pour la conception de chaînes éditoriales

Cette méthode d'ingénierie dirigée par les modèles permet d'abstraire la complexité du code source de spécialisation à produire. Au lieu d'écrire du code source à la redondance forte (la définition des classes de fragments est centrale et réutilisée dans de nombreux aspects du code produit), ce code est intégralement généré à partir de la définition des primitives.

Vue d'ensemble

La suite Scenari duplique le même principe de modélisation et génération pour la conception des chaînes éditoriales et la production des documents. Sur le plan conceptuel, les deux méthodes s'imbriquent l'une dans l'autre puisque les primitives mobilisées dans la conception des chaînes éditoriales représentent le méta-modèle de documents exploités à l'écriture des fragments XML.



Ingénierie dirigée par les modèles dans les chaînes éditoriales Scenari

Sur le plan technologique, cette mutualisation dans l'approche permet également de mutualiser l'architecture du logiciel de modélisation de chaînes éditoriales (SCENARIbuilder) et du logiciel d'écriture des documents (SCENARIchain). Dit autrement, le logiciel de modélisation de chaînes éditoriales peut également être vu comme une chaîne éditoriale de modèles documentaires. Son propre modèle documentaire est écrit à la main par les développeurs de la suite Scenari, les primitives mobilisées sont des fragments XML agencés les uns avec les autres et édités et contrôlés par le même éditeur que celui utilisé dans SCENARIchain. La transformation des fragments XML est faite dans le même environnement technologique et le code produit est une archive de code exprimé dans des fichiers XML. Ce format de sortie s'inscrit dans un contexte technologique similaire à des standards d'écriture des documents comme OpenDocument (OASIS 2005) ou SCORM (ADL).

Références

- Bézivin J. « On the unification power of models ». *Software & Systems Modeling*, 2005. Volume 4, Issue 2, p 171-188. DOI : 10.1007/s10270-005-0079-0
- Brambilla M, Cabot J, Wimmer M. *Model-Driven Software Engineering in Practice.* Morgan & Claypool Publishers, 2012. ISBN : 9781608458820
- OASIS. *Open Document Format for Office.* OASIS Standard. 2005. https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=office
- Advanced Distributed Learning (ADL). *Sharable Content Object Reference Model (SCORM).* http://www.adlnet.org/scorm

Graphe

Terme lié

Concept parent de : Graphe documentaire (p. 210)

Définitions

Graphe. « Un graphe est un triplet Γ = (V ; E, N) où V est l'ensemble des sommets du graphe ; il sera commode d'utiliser la notion V(Γ) pour désigner l'ensemble des sommets du graphe ; N est un ensemble qui sert à étiqueter les arêtes (par exemple N = {1, 2, ..., *p*}, N = {bleu, rouge, vert, ..., violet}, N = ℕ...) ; E ⊂ P_2(V) × N est l'ensemble des arêtes ; notation E = E(Γ) » (Bretto et *al.*, 2012, p. 1).

Degré. « Le degré d'un sommet x ∈ V est le nombre d'arêtes incidentes à x » (ibid., p. 6).

Chaîne. « Soit un graphe Γ = (V ; E, N) et x, y ∈ V ; une chaîne C de x à y est une succession finie d'arêtes » (ibid., p. 7). La longueur de la chaîne k correspond à la somme des arrêtes reliant x à y. « Une chaîne est simple si elle ne contient pas deux fois une même arrête. elle est élémentaire si elle ne contient pas deux fois un même sommet » (ibid., p. 8).

Cycle. « Un cycle est une chaîne fermée simple » (ibid., p. 8).

Graphe connexe. « Un graphe est dit connexe si pour toute paire de sommets x, y ∈ V, il existe une chaîne entre x et y : on dit alors que les sommets x et y sont connectés » (ibid., p. 8).

Sous-graphes. « Un graphe Γ' = (V' ; E', N') est appelé sous-graphe de Γ si V' c V et E' ⊂ E » (ibid., p. 9).

Graphe orienté. « Un graphe orienté ou digraphe $\vec{\Gamma}$ (ou simplement Γ) est un triplet $\vec{\Gamma}$ = (V ;$\vec{E}$, N) défini de la manière suivante : V est l'ensemble des sommets ; notation V = V($\vec{\Gamma}$) ; $\vec{E}$ ⊂ V × V × N est l'ensemble des arcs ; notation $\vec{E}$ = $\vec{E}$(Γ) ; N est un ensemble servant à étiqueter les arcs. Un arc a ∈ $\vec{E}$ sera noté ((x, y), n) : l'arc va de x vers y » (ibid., p. 14).

Chemin. « Un chemin C de x à y est une suite finie [d'arcs] ; l'entier k est la longueur du chemin. » « Un chemin est simple s'il ne contient pas deux fois le même arc ; il est élémentaire s'il ne contient pas deux fois le même sommet » (ibid., p. 15).

Circuit. « Un circuit est un chemin simple dont les extrémités coïncident » (ibid., p. 15).

Références

- Bretto A, Faisant A, Hennecart F. *Éléments de théorie des graphes. Collection IRIS* : Springer, 2012. ISBN : 978-2-8178-0280-0

Fragment de document

Termes liés

Fait partie de : Graphe documentaire (p. 210)

Instance de : Classe de fragment (p. 213)

Est exploité par : Chaîne éditoriale numérique (p. 206)

Est édité dans : Atelier (p. 212)

Définition

Un fragment de document ou fragment documentaire est une partie d'un document. Par exemple, une section d'un document bureautique ou une page HTML peuvent être considérés comme des fragments.

Dans ce mémoire, ce terme désigne les items exploités par les chaînes éditoriales numériques XML. Un fragment est donc l'instanciation d'une classe de fragment. Il s'agit d'un document XML bien formé et valide par rapport au schéma défini par la classe.

Graphe documentaire

Termes liés

Est un : Graphe (p. 209)

Est édité par : Chaîne éditoriale numérique (p. 206)

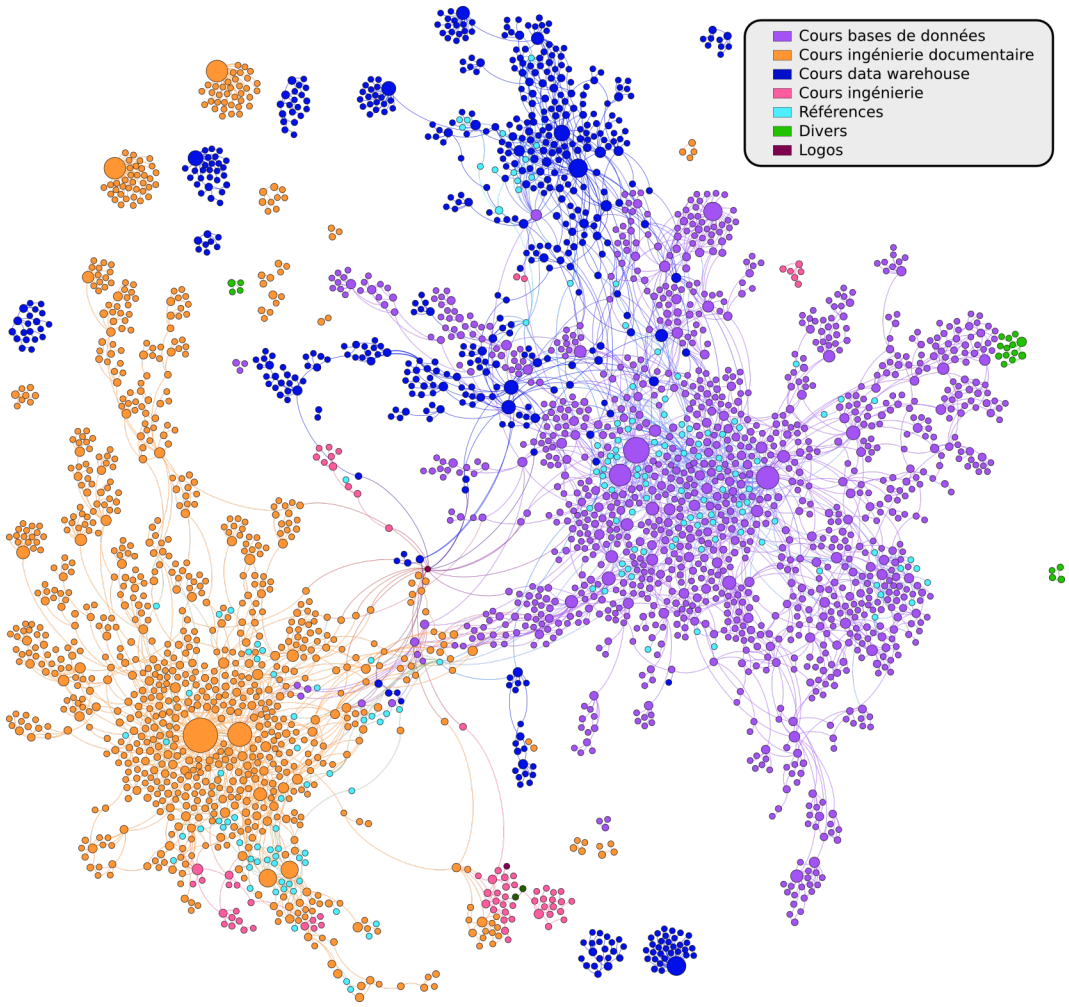
Définition

Nous appelons « graphe documentaire » un graphe orienté représentant des documents rééditorialisés. Noté $\Gamma = (V ; E, N)$, un graphe documentaire est composé :

- de l'ensemble des sommets V représentant chacun un fragment de document ;
- de l'ensemble des arcs E représentant les liens entre fragments ;
- de l'ensemble des étiquettes N représentant les fonctions de rééditorialisation à l'origine du lien (transclusion, référence, dérivation, déclinaison).

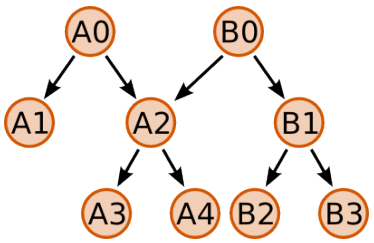
Représentation des graphes documentaires

Dans ce mémoire, nous mobilisons deux catégories de représentation des graphes. Une première issue de l'analyse d'un réel fonds documentaire rééditorialisé. Le fond est analysé et mis en forme avec des algorithmes de spatialisation des sommets en fonction des liens comme dans l'exemple suivant :



Graphe documentaire - Opale

Une seconde représentation est mobilisée afin d'illustrer des graphes documentaires très largement simplifiés. Ces schémas ne correspondent pas à des usages réels et sont conçus dans un logiciel de création graphique.



Rééditorialisation par transclusion

Sous-graphe document

Terme lié

Fait partie de : Graphe documentaire (p. 210)

Définition

Afin de caractériser la notion de document dans un graphe documentaire, nous proposons les définitions suivantes :

Sommet racine. Soit un graphe documentaire D = (V ; E, N). Un sommet issu de V est

appelé racine lorsque le modèle documentaire associe un algorithme de génération de document au fragment qu'il représente. Nous noterons l'ensemble des sommets racines Vr. Notons que par définition, Vr ∈ V.

Sous-graphe document. Soit un graphe documentaire D = (V ; E, N) et un sommet racine Vr_1 tel que Vr_1 ∈ Vr, un sous-graphe documentaire est noté D'd (V'd ; E'd, N) avec

- V'd : l'ensemble des sommets de V tel que quel que soit le sommet x ∈ V'd, il existe un chemin entre Vr_1 et x.
- E'd : l'ensemble des arcs de E dont les extrémités appartiennent à l'ensemble V'd.

Atelier

Termes liés

Fait partie de : Chaîne éditoriale numérique (p. 206)

Est associé à : Modèle documentaire (p. 212)

Est associé à : Modèle d'activité (p. 215)

Définition

Un atelier est un espace de travail mis en place au sein d'une chaîne éditoriale. Dans l'usage classique des chaînes éditoriales, un atelier est l'espace dans lequel est édité un graphe documentaire. L'édition et le contrôle des fragments se fait à l'aide du modèle documentaire.

Dans ce mémoire, nous proposons de revisiter la notion d'atelier pour y exploiter non pas l'intégralité d'un graphe documentaire mais seulement une portion de graphe. Cette portion est délimitée par l'usage d'un calque ou d'un ou plusieurs fragments proxys. Nous proposons également d'enrichir le modèle documentaire par un modèle d'activité définissant des structures d'enregistrement du discours pragmatique.

Atelier, graphe et perception des rédacteurs

Un atelier permet l'exploitation d'un graphe ou d'une portion de graphe. Le graphe est une représentation d'un ensemble de fragments liés les uns aux autres par des fonctions de rééditorialisation. Ce n'est cependant pas la représentation la plus facile à appréhender par un rédacteur. Afin de remédier à cette difficulté, un atelier définit un arbre de gestion. Il s'agit d'un arbre d'espaces simulant un système de fichiers classique. Les fragments du graphe sont ainsi projetés dans les espaces afin de permettre une classification propre à l'auteur.

Modèle documentaire

Termes liés

Est exploité par : Chaîne éditoriale numérique (p. 206)

Est associé à : Atelier (p. 212)

Définition

Nous appelons modèle documentaire l'ensemble des ressources informatiques mobilisées par une chaîne éditoriale numérique XML afin de définir les structures XML exploitables, leurs modalités d'édition et leurs possibilités de transformation en documents encodés dans des formats standards tels que PDF, HTML ou OpenDocument.

Un modèle documentaire est donc classiquement composé :

- de schémas XML ;
- d'informations complétant le schéma en spécifiant comment éditer ou contrôler certaines structures documentaires ;

- d'algorithmes de transformation associés à certaines classes de fragments et permettant la transformation et la publication du graphe.

Référence

- Arribe T, Crozat S, Bachimont B, Spinelli S. « Chaînes éditoriales numériques: allier efficacité et variabilité grâce à des primitives documentaires. ». *Actes du Colloque international sur le document électronique, CiDE. 15,«Métiers de l'information, des bibliothèques et des archives à l'ère de la différentiation numérique»,, Tunis, Tunisie, 2012.*

Classe de fragment

Termes liés

Est défini par : Modèle documentaire (p. 212)

Classe de : Fragment de document (p. 210)

Définition

Une classe de fragment est un modèle de structure documentaire mobilisé par la chaîne éditoriale pour instancier des fragments. Un modèle documentaire définit plusieurs classes de fragments.

Par exemple, le modèle documentaire Opale définit plusieurs classes de fragments dont : un module, une activité d'apprentissage, une activité d'auto-évaluation ou encore un grain de contenu.

Concepts techniques proposés

Atelier calque

Termes liés

Est un : Atelier (p. 212)

Fait partie de : Chaîne éditoriale numérique (p. 206)

Est associé à : Modèle documentaire (p. 212)

Est associé à : Modèle d'activité (p. 215)

Définition

Un atelier calque est un atelier permettant d'effectuer une dérivation du graphe ou de la portion de graphe exploité par un autre atelier appelé atelier premier.

Comme tout atelier, un atelier calque est associé à un modèle documentaire qui peut être complété par un modèle d'activité afin d'exploiter sa portion de graphe.

La définition formelle d'un atelier calque comprend donc :

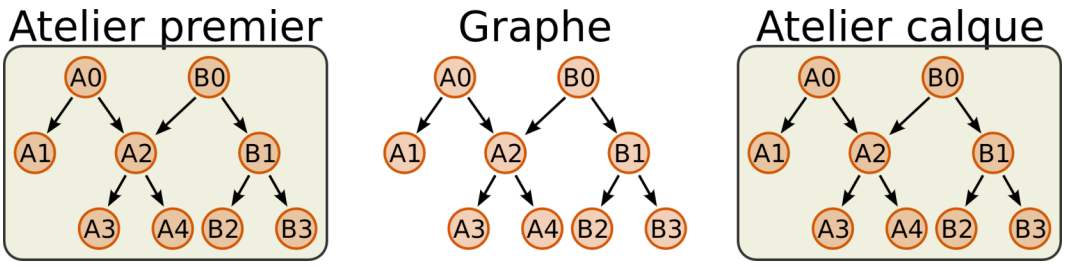
- un atelier premier ;
- un modèle documentaire éventuellement complété par un modèle d'activité ;
- une fonction de dérivation.

La fonction de dérivation définit comment exploiter le lien entre un fragment original de l'atelier maître et un fragment dérivé dans l'atelier calque.

Graphe et atelier calque - initialisation

À l'initialisation d'un atelier calque, l'ensemble des fragments visibles dans l'atelier sont issus de l'atelier premier. Aucun fragment n'est dupliqué pour la mise en place de ce nouvel

espace de travail.



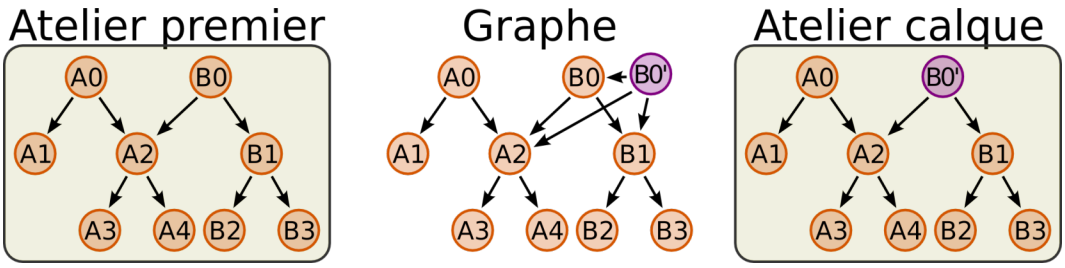
Initialisation d'un atelier calque

Graphe et atelier calque - écriture

Lorsqu'un rédacteur édite un fragment au sein de l'atelier calque :

- le graphe est enrichi avec un fragment dérivé, lié par un lien de dérivation à son fragment d'origine ;
- l'atelier premier n'évolue pas ;
- l'atelier calque substitue le fragment d'origine par sa dérivation.

Par exemple, le fragment B0 est surchargé dans l'atelier calque du schéma suivant :



Dérivation d'un fragment dans un atelier calque

Atelier maître

Termes liés

est un : Atelier (p. 212)

Fait partie de : Chaîne éditoriale numérique (p. 206)

Est associé à : Modèle documentaire (p. 212)

Est associé à : Modèle d'activité (p. 215)

Définition

Un atelier maître n'est pas un atelier calque. La portion de graphe exploitée par un atelier maître n'est pas intégralement issue d'un autre atelier et l'édition d'un fragment ne crée pas de surcharge particulière. Un atelier calque dépend nécessairement d'un et un seul atelier maître (directement ou à travers un ou plusieurs ateliers calques positionnés les uns derrière les autres en série).

Fragment proxy

Terme lié

Est un : Fragment de document (p. 210)

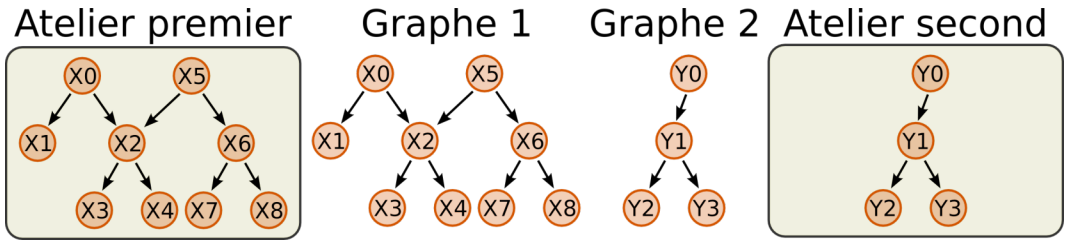
Définition

Un fragment proxy est en premier lieu un fragment classique exploité dans un atelier calque ou un atelier maître.

Le qualificatif proxy désigne la mobilisation du fragment comme passerelle entre son atelier d'origine (l'atelier premier) et un atelier second. La mise en place de la fonction de proxy rend le fragment et l'ensemble de son sous-graphe disponibles pour être référencés dans l' atelier second. Les fragments proxys disposent du même principe de dérivation que les ateliers calques : une modification dans l'atelier premier modifie effectivement le fragment documentaire. Une modification dans l'atelier second créé un fragment dérivé lié à l'original.

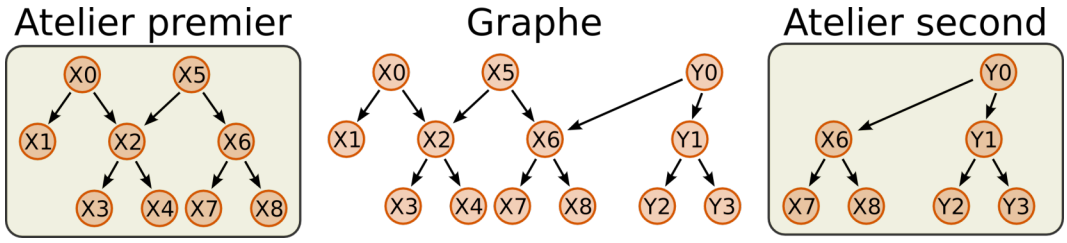
Mise en place d'un fragment proxy

Avant l'initialisation d'un premier proxy, les deux ateliers ne sont pas liés l'un à l'autre.



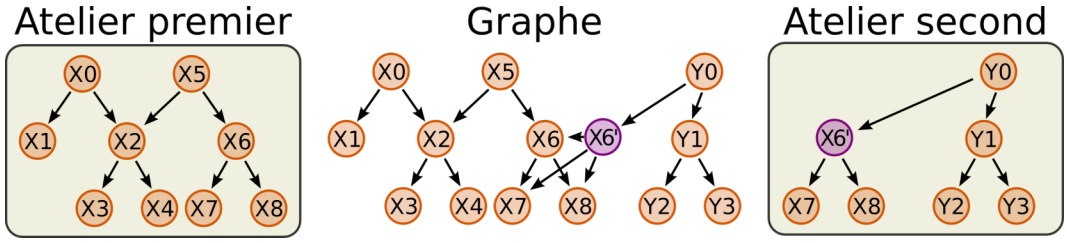
Deux ateliers sans proxy

Lors de l'initialisation du proxy, le fragment proxié et son sous-graphe se retrouvent visibles dans l'atelier second.



Deux ateliers après l'initialisation d'un fragment proxy

Lorsqu'un fragment est édité dans l'atelier second, il se retrouve aussitôt dupliqué, surchargé et lié à son fragment d'origine.



Deux ateliers après la surcharge d'un fragment proxié

Modèle d'activité

Termes liés

Est exploité par : Chaîne éditoriale numérique (p. 206)

Est associé à : Atelier (p. 212)

Complète : Modèle documentaire (p. 212)

Définition

Un modèle d'activité est une extension du modèle documentaire. Il complète les classes de fragments existantes et propose de nouvelles classes dédiées à l'enregistrement de l'activité des rédacteurs sur le graphe documentaire.

Le modèle d'activité peut contenir :

- des fragments documentaires classiques dédiés à l'enregistrement de l'activité ;
- des fragments de tâches assistant les rédacteurs dans l'organisation de l'action ;
- des informations complémentaires aux fragments classiques du modèle documentaire et assistant les rédacteurs dans l'organisation de l'activité (cycle de vie ou responsabilité) ;
- des générations dynamiques permettant de commenter les fragments depuis un navigateur web.

Fragment de tâche

Termes liés

Est un : Fragment de document (p. 210)

Est défini par : Modèle d'activité (p. 215)

Peut contenir : Responsabilité (p. 216)

Peut contenir : Cycle de vie (p. 217)

Définition

Les fragments de tâches sont des fragments documentaires particuliers : ils sont conçus pour assister les rédacteurs dans l'organisation de l'activité sur le graphe tout en produisant la documentation de cette activité.

Lors de la modélisation d'une classe de fragments de tâche, certaines structures (titre, responsabilité, cycle de vie, planification, échéance, fil de discussion) sont mobilisées par le gestionnaire de tâches pour stocker les différentes informations constitutives des tâches. D'autres structures sont d'ordre purement documentaire et participent à la documentarisation des tâches.

En exploitant le moteur de tâches, les rédacteurs produiront ainsi automatiquement des fragments enregistrant l'activité sur le graphe.

Responsabilité

Termes liés

Est défini par : Modèle d'activité (p. 215)

Peut compléter : Fragment de document (p. 210)

Peut compléter : Fragment de tâche (p. 216)

Définition

Une responsabilité est une structure documentaire faisant partie d'un fragment dont le contenu est contrôlé par la chaîne éditoriale et dont l'objet est d'assister les rédacteurs dans l'organisation de l'activité sur le graphe. Un cycle de vie définit un ensemble d'états et de transitions entre ces états.

Une responsabilité définit un rôle d'un rédacteur sur un fragment. Le contenu d'une structure de type responsabilité est contrôlé par la chaîne éditoriale et est nécessairement le login d'un utilisateur déclaré dans le logiciel.

Des responsabilités peuvent être modélisées au sein des tâches ou au sein des fragments documentaires classiques. Dans le premier cas, la responsabilité est exploitée pour associer une

tâche à un usager. Les responsabilités constituent également des critères du moteur de recherche.

Cycle de vie

Termes liés

Est défini par : Modèle d'activité (p. 215)

Peut compléter : Fragment de document (p. 210)

Peut compléter : Fragment de tâche (p. 216)

Définition

Un cycle de vie est une structure documentaire faisant partie d'un fragment dont le contenu est contrôlé par la chaîne éditoriale et dont l'objet est d'assister les rédacteurs dans l'organisation de l'activité sur le graphe. Un cycle de vie définit un ensemble d'états et de transitions entre ces états.

Outre la gestion des transitions, les chaînes éditoriales exploitent les états des cycles de vie afin de constituer des critères pour le moteur de recherche.

Commentaires

Termes liés

Fait partie de : Fragment de document (p. 210)

Fait partie de : Fragment de tâche (p. 216)

Définition

Les commentaires sont des fils de discussion insérés dans n'importe quel fragment à n'importe quel niveau en exploitant le principe des commentaires XML. Du code XML permettant l'expression d'une discussion est injecté dans les commentaires XML d'un fragment.

Ces commentaires permettent aux rédacteurs de former une documentation du graphe au sein même des fragments.

Un système d'édition des commentaires dans un navigateur web permet aux rédacteurs ou à d'autres utilisateurs de visualiser ou éditer les commentaires des fragments.

Atelier calque de travail

Terme lié

Est un : Atelier calque (p. 213)

Définition

Un atelier calque de travail est un atelier calque dont la fonction de dérivation consiste à permettre la surcharge de fragments pour reverser les modifications dans les fragments d'origine une fois celles-ci validées.

Il s'agit d'un projet de rééditorialisation convergent car les modifications apportées dans l'atelier second ont vocation à être réintégrées dans les fragments d'origine.

Atelier calque de dérivation

Terme lié

Est un : Atelier calque (p. 213)

Définition

Un atelier calque de dérivation est un atelier calque dont la fonction de dérivation est conçue pour maintenir des surcharges permanentes sur les fragments du graphe d'origine.

il s'agit d'un projet de rééditorialisation divergent car les modifications apportées dans l'atelier second n'ont pas vocation à être réintégrées dans les fragments d'origine.

Fragment proxy de partage

Terme lié

Est un : Fragment proxy (p. 214)

Définition

Un fragment proxy de partage est un fragment proxy dont les surcharges (du fragment ou de son sous-graphe) ont vocation à être reversées dans les fragments d'origine.

Fragment proxy de dérivation

Terme lié

Est un : Fragment proxy (p. 214)

Définition

Un fragment proxy de dérivation est un fragment proxy dont les surcharges (du fragment ou de son sous-graphe) n'ont pas vocation à être reversées dans les fragments d'origine.

Annexe 2

Système développé

Synthèse des propositions 219
Documentarisation de l'activité 222
Moteur de tâches 222
Système de commentaires 224
Enrichissement des fragments 224
Structuration des projets de rééditorialisation 225
Atelier calque de travail 225
Atelier calque de dérivation 226
CID - protocole pour l'encadrement des transactions documentaires 227
Prochains développements 228

Cette annexe présente nos propositions techniques développées au sein de la suite logicielle Scenari. Elle se compose de quatre parties :

- la première fait une synthèse des propositions théoriques, techniques, de la partie technique développée et de notre contribution au sein de ces développements ;
- la seconde présente l'instrumentation au sein de Scenari des propositions techniques pour documentariser l'activité ;
- la troisième présente l'instrumentation au sein de Scenari des propositions techniques pour structurer le graphe ;
- la dernière fait un état des lieux des prochains développements à mettre en œuvre afin de poursuivre le projet de recherche.

Cette annexe a pour principal objet de donner à voir le système manipulé. Par conséquent, les deux sections dédiées à montrer les propositions techniques développées sont principalement constituées de copies d'écrans. Afin de fluidifier la lecture, nous avons réduit ces images à la taille de vignettes. Nous encourageons le lecteur à consulter cette annexe en ligne à l'adresse suivante *(http://ics.utc.fr/~tha/)* pour une meilleure visualisation.

Synthèse des propositions

Proposition théorique

Nous formulons deux propositions théoriques complémentaires articulées autour de la notion d'atelier.

La première, interne à l'atelier, consiste à documentariser l'activité au sein de fragments pragmatiques. Dans un premier temps, il convient de permettre la modélisation de discours pragmatiques. Dans un second, il s'agit de développer des fonctions d'assistance à l'organisation des activités sur le graphe qui s'appuient sur les fragments pragmatiques ainsi modélisés. Il devient alors aisé de modéliser des fragments pragmatiques qui documentarisent l'activité au fur et à mesure que les fonctions d'assistance à l'organisation de l'activité sont utilisées.

La seconde approche, externe aux ateliers, consiste à structurer le graphe au sein de plusieurs ateliers. Chaque atelier matérialise ainsi un projet de rééditorialisation, lié à un ou plusieurs autres selon quatre grandes catégories de projets. Ces catégories sont issues de la combinatoire entre la caractéristique de convergence (convergent ou divergent) et la portée (partielle ou totale). L'objectif de cette structuration est à la fois de diminuer le nombre de fragments perçus par les rédacteurs et d'automatiser le contrôle de l'articulation des différents projets de rééditorialisation entre eux.

Propositions techniques - documentarisation de l'activité

Dans ce mémoire, nous proposons trois axes d'expérimentations des fragments pragmatiques.

Le premier explore la figure de la tâche. Il permet la modélisation de tâches et leur exploitation dans un moteur de tâches. Le simple usage des fonctions du moteur de tâches (usage du cycle de vie, association d'un usager, planification) documentarise l'activité dans le fragment de tâche.

Le second est dédié aux commentaires. Il permet l'ouverture de fils de commentaires génériques à n'importe quel niveau d'un fragment. Il propose en outre un système d'édition de ces commentaires, soit au sein de l'éditeur de la chaîne éditoriale, soit directement dans les documents transformés en documents HTML, au sein d'un navigateur.

Le troisième propose d'étendre les fragments existants en ouvrant des champs pragmatiques adossés aux fragments. Il est ainsi possible de modéliser des responsabilités ou des cycles de vie propres aux fragments.

Ces propositions techniques ne sont que trois exemples qui s'inscrivent dans notre approche de documentarisation de l'activité. De nouvelles possibilités d'assistance des rédacteurs et de modélisation sont à expérimenter en fonction des contextes d'usage à outiller.

Propositions techniques - structuration du graphe

Dans ce mémoire, nous présentons une proposition technique par catégorie de projet de rééditorialisation.

- La fonction d'atelier calque de travail permet d'outiller les projets de rééditorialisation convergents et à la portée totale. L'objet d'un atelier calque de travail est d'isoler un environnement de rédaction des fragments sur l'ensemble du graphe. Un fragment modifié localement peut ainsi être validé et resynchronisé avec sa version de référence.
- La fonction d'atelier calque de dérivation permet d'outiller les projets de rééditorialisation divergents et à la portée totale. L'objet de tels calques est de rééditorialiser l'ensemble des fragments d'un atelier afin de proposer des versions alternatives des documents originaux.
- La fonction de fragment proxy de partage permet d'outiller les projets de rééditorialisation convergents et à la portée partielle. L'objet de telles fonctions est de permettre le partage d'un fragment et de l'ensemble de son sous-graphe entre deux ateliers différents.
- La fonction de fragment proxy de dérivation permet d'outiller les projets de rééditorialisation divergents et à la portée partielle. L'objet de telles fonctions est de

permettre, au sein d'un atelier, la dérivation d'un fragment et de l'ensemble de son sous-graphe issus d'un autre atelier.

En l'état, toutes ces propositions ne peuvent fonctionner qu'à la condition d'être opérées dans une même chaîne éditoriale. Afin de permettre une distribution des projets de rééditorialisation, nous exposons également nos travaux sur le protocole *Content Interactive Delivery* (CID) dont l'enjeu est d'encadrer les transactions documentaires entre différents systèmes. L'objet de ce protocole dépasse cependant le spectre fonctionnel des chaînes éditoriales pour permettre tout type d'échange de métadonnées ou documents entre différents systèmes.

Propositions techniques développées

À l'issue de nos trois années de recherche :

- l'ensemble des propositions techniques pour la documentarisation de l'activité sont développées ;
- les deux fonctions de calques exposées sont développées ;
- des travaux préalables au développement des fragments proxys ont été menés mais les deux fonctions de proxys ne sont pas encore développées ;
- plusieurs premières versions du protocole CID ont été spécifiées et développées au sein de Scenari et de plusieurs autres systèmes ; l'expérimentation, le développement et les spécifications du protocole doivent cependant être poursuivies afin de gagner en robustesse et en généricité.

Notre contribution

Au sein des différentes propositions techniques développées, nous avons assuré :

- la conception et le développement du système de commentaires ;
- une partie des développements préalables au développement des fragments proxys
- la conception et le développement du client CID de Scenari et des clients et serveurs CID génériques proposés sur le site du protocole *(http://www.cid-protocol.org)*.

Synthèse

Propositions théoriques	Propositions techniques	Propositions techniques développées	Contribution technique
Documentariser l'activité	Moteur de tâches	✓	✗
	Système de commentaires	✓	✓
	Extension des fragments (responsabilité et cycle de vie)	✓	✗
Structurer le graphe	Calque de travail	✓	✗
	Calque de dérivation	✓	✗
	Fragment proxy de partage	~	~
	Fragment proxy de dérivation	~	~
	CID, protocole pour l'encadrement des transactions documentaires	✓	✓

Documentarisation de l'activité

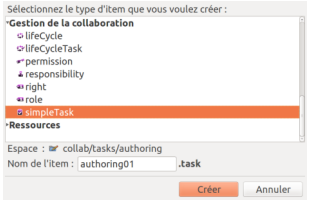
Moteur de tâches

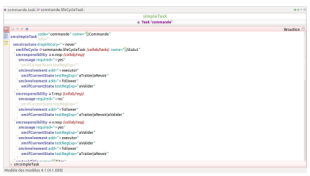
Modélisation d'une tâche

Un tâche est modélisée dans une primitive appelée *simpleTask*. La primitive définit :

- le code du fragment (utilisé en interne par la chaîne éditoriale) ;
- un lien vers une primitive de définition des cycles de vie d'une tâche (p. 222) ;
- la définition des responsabilités (p. 223) et de l'association entre une responsabilité et un état du cycle de vie (exécutant, suiveur, aucun) ;
- l'intitulé des champs de titre, de planification et d'échéance ;
- les liens vers les modèles documentaires de la description de la tâche et des commentaires.

La seconde partie (derrière la balise *sm:authoring*) concerne des éléments de paramétrage et de mise en forme de l'interface homme machine des commentaires.

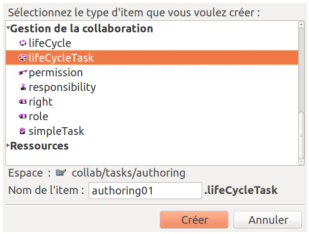




Modélisation du cycle de vie d'une tâche

Le cycle de vie d'une tâche est modélisé dans une primitive appelée *lifeCycleTask*. La primitive définit :

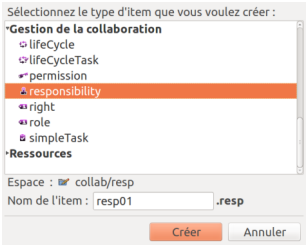
- les différents états et leurs statuts associés pour la chaîne éditoriale (tâche en cours, à venir, close) ;
- les transitions entre les tâches (une transition est composée d'un code interne au système, d'un état d'arrivée, d'un nom et d'un certain nombre d'états à partir desquels cette transition peut être exécutée.



Modélisation d'une responsabilité

Une tâche est modélisée dans une primitive appelée *responsability*. La primitive définit :

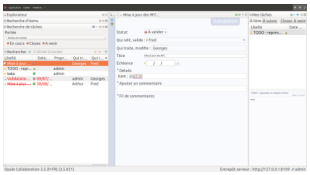
- un code (utilisé en interne par la chaîne éditoriale) ;
- un nom ;
- une cardinalité (combien de personnes ou de groupes est-il possible d'inscrire dans cette responsabilité) ;
- une définition des utilisateurs autorisés à prendre une responsabilité.

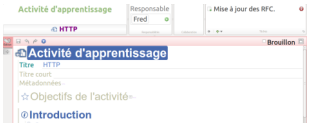


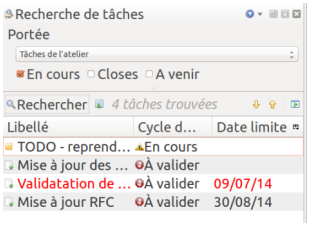
Exploitation des tâches

Une fois la tâche modélisée et générée et l'archive de code source produite associée à un atelier, la chaîne éditoriale adapte son environnement pour permettre la manipulation des tâches.

- À gauche, un panneau de recherche de tâches peut être instancié ;
- À droite, un panneau de gestion des tâches personnelles de l'utilisateur connecté peut être instancié ;
- Une tâche ouverte s'édite dans le panneau central ;
- Lorsqu'un fragment classique est édité, le bandeau de gestion (en haut du panneau central) affiche les tâches associées au fragment en cours d'édition.







Système de commentaires

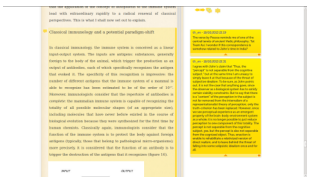
Exploitation des commentaires

Le système de commentaires ne nécessite aucune modélisation particulière pour être activé. En revanche, la fonction de commentaire sur les publications dynamiques doit être intégrée au générateur dynamique afin d'être utilisable. Nous ne montrerons pas de copie d'écran de la modélisation ici puisque à ce jour, aucune primitive dédiée n'a été modélisée. L'intégration du système de commentaires s'opère par l'ajout de code Javascript et CSS.

Au sein d'un environnement auteur de Scenari, l'ajout d'un commentaire se fait par le menu contextuel en cliquant sur un élément d'un fragment.

Une fois un commentaire ajouté, le bouton permet de supprimer un commentaire du fil de discussion et le bouton permet de répondre au fil de discussion. Le bouton permet quant à lui de clore un fil de commentaires.

Au sein des publications dynamiques, la disposition des commentaires et les différents boutons de commandes peuvent varier en fonction des publications.



Enrichissement des fragments

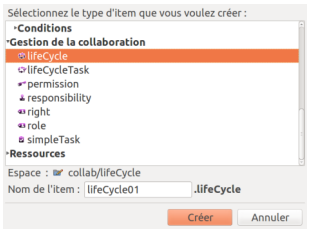
Modélisation d'une responsabilité

La modélisation d'une responsabilité sur un fragment s'opère de façon identique aux responsabilités sur les tâches (p. 223).

Modélisation d'un cycle de vie

Un cycle de vie est modélisé dans une primitive appelée *lifeCycle*. La primitive définit :

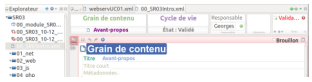
- les différents états ;
- les variations opérées sur les droits d'édition des fragments en fonction des états ;
- les transitions entre états ;
- des restrictions sur les transitions en fonction du type de fragment.



Exploitation des responsabilités et cycle de vie des fragments

Une fois ces éléments modélisés et générés et l'archive de code source produite associée à un atelier, la chaîne éditoriale adapte son environnement.

- Le bandeau situé en haut du panneau central contient une partie dédiée à la gestion du cycle de vie et une autre aux responsabilités.
- L'ajout ou la suppression d'une responsabilité se fait en éditant le champs dédié dans le panneau.
- Le passage d'un fragment d'un état à un autre se fait en utilisant le menu contextuel à partir de l'arbre de gestion du panneau de gauche ou à partir du bandeau dédié au cycle de vie.



Structuration des projets de rééditorialisation

Atelier calque de travail

Gestion des ateliers calques

Les chaînes éditoriales Scenari sont structurées en entrepôts et en ateliers. Un entrepôt est un serveur pouvant héberger plusieurs ateliers.

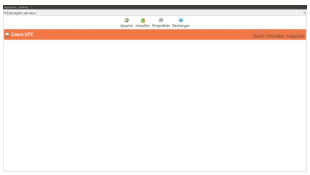
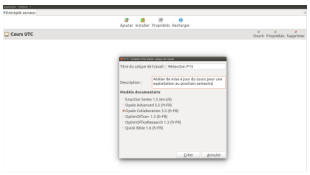
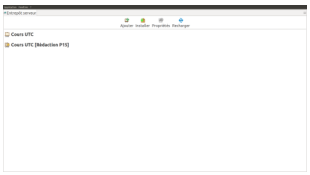
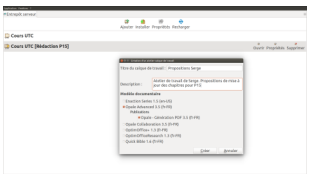
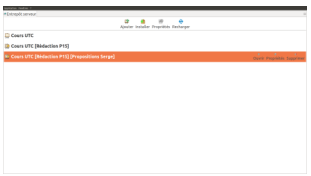
À l'ouverture d'une chaîne éditoriale (sur un poste de travail local ou un entrepôt distant), un écran d'accueil liste les ateliers disponibles.

La création d'un atelier calque se fait dans le menu contextuel après un clic droit sur un atelier existant.

Une boîte de dialogue s'ouvre alors pour paramétrer l'atelier calque en demandant :

- le choix d'un titre ;
- une description facultative ;
- le choix du modèle documentaire.

Les modèles disponibles sont issus de SCENARIbuilder. Ils sont constitués à la fois du modèle documentaire et du modèle d'activité. Par exemple, les deux modèles Opale de la liste de la copie d'écran contiennent le même modèle documentaire mais un modèle d'activité différent.

Exploitation d'un calque de travail

À l'initialisation, un calque de travail affiche l'ensemble des fragments disponibles dans l'atelier premier.

- Ces fragments sont affichés comme synchrones (icône dans l'arbre du panneau de gauche et dans le bandeau du haut du panneau central).
- Un fragment modifié localement est signalé par l'icône .
- Un fragment créé localement est signalé par l'icône .
- Un fragment supprimé localement est signalé par l'icône .

À partir du menu contextuel, il est possible :

- de valider et reverser un fragment surchargé (le contenu du fragment est alors copié dans la version de référence et la surcharge locale est supprimée) ;
- d'annuler une surcharge (la surcharge locale est simplement supprimée) ;
- de valider la suppression d'un fragment (le fragment disparaît alors définitivement de l'arbre à gauche et sa version de référence est également supprimée) ;
- d'annuler la suppression d'un fragment (le fragment est alors remis en état synchrone).

Atelier calque de dérivation

Gestion des ateliers calques

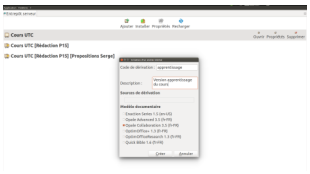
La création d'un atelier calque de dérivation se fait dans le menu contextuel après un clic droit sur un atelier existant.

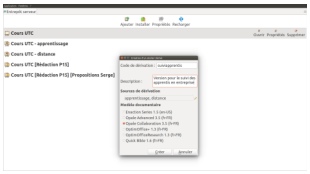
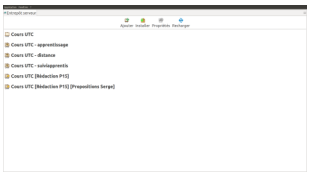
Une boîte de dialogue s'ouvre alors pour paramétrer l'atelier calque en demandant :

- un code de dérivation ;
- une description facultative ;
- un chemin de dérivation facultatif (utilisé dans le troisième atelier calque de dérivation

instancié, dans cet atelier, les contenus viennent en premier lieu de la dérivation *apprentissage*, puis de celle nommée *distance*, puis de l'atelier maître) ;

- le choix du modèle documentaire.



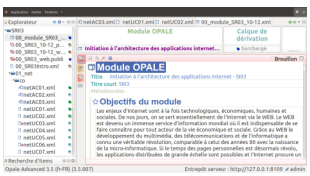
Exploitation d'un calque de dérivation

La visualisation de l'état d'un fragment de l'atelier dérivé se fait à l'aide d'un jeu d'icônes. Une icône est dédiée à chacun des 9 états possibles en fonction de l'origine des fragments et du statut de leur dérivation :

- non surchargé
- non surchargé, validé
- non surchargé, à contrôler
- surchargé
- surchargé, validé
- surchargé, à contrôler
- fragment local
- fragment local, validé
- fragment local, à contrôler

Sans action particulière du rédacteur, les fragments sont non surchargés ( ), surchargés après une modification locale ( ) ou locaux au calque ( ). Quelles que soient les modifications opérées dans l'atelier local ou dans l'atelier premier, seuls ces trois états sont visibles.

S'il le souhaite, un rédacteur peut marquer la dérivation comme achevée (, ou). Dès lors, toute modification du fragment local ou de sa version de référence dans l'atelier premier transforme automatiquement l'état du fragment en *à contrôler* (, ou). Le rédacteur doit à nouveau marquer la dérivation du fragment comme achevée afin de lui faire retrouver l' état *validé*.



CID - protocole pour l'encadrement des transactions documentaires

Notre contribution technique liée au protocole CID est embarquée dans la chaîne éditoriale et ne donne pas lieu à de nouvelles fonctionnalités liées à la structuration des ateliers pour le moment.

Plus d'informations sont disponibles sur le protocole CID au sein de l'annexe dédiée à notre contribution (p. 231).

Prochains développements

Dans cette section, nous menons un rapide état des lieux des prochains développements envisagés afin de poursuivre l'implémentation de nos propositions.

Primitives de commentaires

À ce stade de nos développements, l'insertion du système de commentaires dans une publication dynamique se fait par l'ajout de code Javascript.

Une part plus importante du code source Javascript pourrait être mutualisée entre les différents modèles. Nous préconisons de créer une primitive d'insertion des commentaires contenant le code source mutualisé et des options pour les paramètres spécifiques à une publication.

Le développement de cette primitive devrait permettre de simplifier grandement l'intégration et la maintenance du système de commentaires au sein des modèles dans lesquels il est utilisé.

Enrichir les fragments pragmatiques

Nos propositions techniques pour la documentarisation de l'activité sont des exemples d'instanciation des fragments pragmatiques. Nous soutenons que de nouveaux contextes d'usage amèneront de nouveaux besoins de modélisation et d'outillage afin de poursuivre le soutien de l'action des rédacteurs.

Par exemple, au cours de ce mémoire (p. 187), nous avons suggéré la création d'un nouveau type de fragment pragmatique au sein du contexte de la DILA afin de documentariser l'action de validation depuis un atelier calque de travail vers un atelier premier.

Nous avons également suggéré d'étendre les champs pragmatiques attachés aux fragments documentaires classiques afin d'outiller les validations opérées dans le contexte de production de la société Quick (p. 150).

Fonctions de proxy

Les fonctions de fragments proxys ne sont pas encore développées. La plupart des développements préalables ont été menés. Il convient maintenant d'entamer réellement ce chantier et de développer les deux fonctions de proxy de base : proxy de partage et proxy de dérivation.

Enrichir les fonctions de calques et proxys

Les fonctions de calques et de proxys proposées permettent d'utiliser les quatre catégories de projets de rééditorialisation décrits dans ce mémoire (convergent partiel, convergent total, divergent partiel et divergent total).

Une fois le mécanisme de création et de gestion des calques et proxys développé, l'enrichissement de ces éléments par de nouvelles fonctions permettra de mieux accompagner la diversité des projets de rééditorialisation.

Distribuer les projets de rééditorialisation

Dernière étape planifiée au sein de notre projet de recherche, afin de densifier les usages de la production rééditorialisée, nous proposons de distribuer un même graphe au sein de plusieurs instances de chaînes éditoriales différentes, distantes les unes des autres. Cette distribution des projets de rééditorialisation permettrait d'accompagner plus de situations d'écriture. Elle permettrait par exemple :

- d'outiller correctement des projets largement distribués comme la production documentaire de la DILA et la rééditorialisation opérée par les collectivités territoriales ;

- d'accompagner des usages plus ponctuels en permettant par exemple à des enseignants ne travaillant pas dans les mêmes établissements de travailler ensemble.

Annexe 3

Contribution technique

Système de commentaires 231
Architecture 231
Système de liens profonds 233
Système d'édition distribué et synchrone 235
Transformées opérationnelles 235
Contraintes 236
Fonctionnement 238
Développements 241
Items flottants 241
Content Interactive Delivery 242
Manifeste 242
Client Scenari 245
Client et serveur de démonstration 246
Client générique 246
SingleCIDServer : un serveur de fichiers simple 246
SimpleCIDServer : un serveur de fichiers 249

Cette annexe détaille les éléments techniques constituant notre contribution. Elle se compose de trois parties, une première dédiée aux systèmes de commentaires, une seconde dédiée aux items flottants, préalable aux fragments proxy et une dernière dédiée au protocole CID.

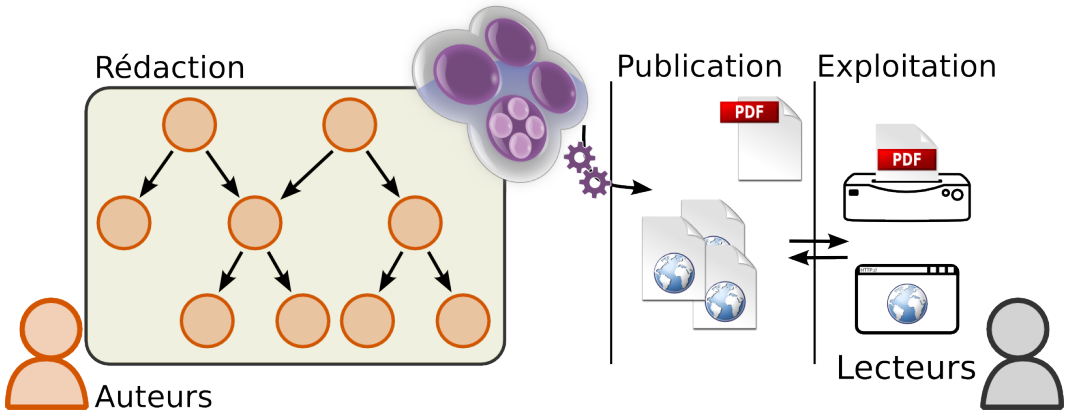
Système de commentaires

Architecture

Publications au sein d'une chaîne éditoriale

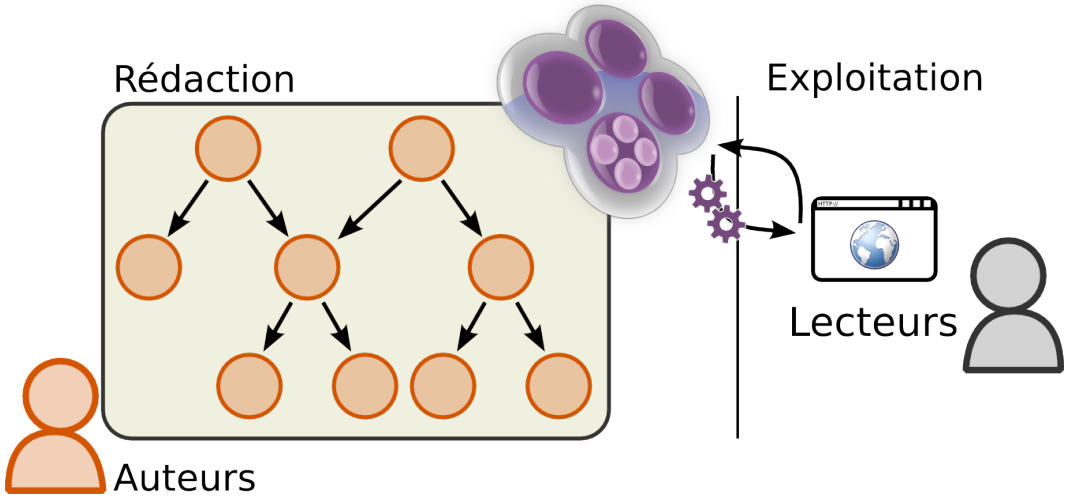
Nous distinguons deux modes de publication dans les chaînes éditoriales : les

publications statiques et les publications dynamiques. Dans une publication statique, un fragment racine du graphe documentaire est sélectionné par un usager et transformé par la chaîne éditoriale. Cette transformation produit des documents aux formats standards qui peuvent ensuite être utilisés indépendamment de la chaîne éditoriale.



Publication statique

Dans une publication dynamique, un fragment racine du graphe documentaire est sélectionné par un rédacteur et le lien HTML permettant sa transformation est copié et transmis aux autres usagers souhaitant afficher le document. Lorsque le lien est copié dans un navigateur web, une requête est envoyée à la chaîne éditoriale qui transforme à la volée les fragments demandés en une page HTML.



Publication dynamique

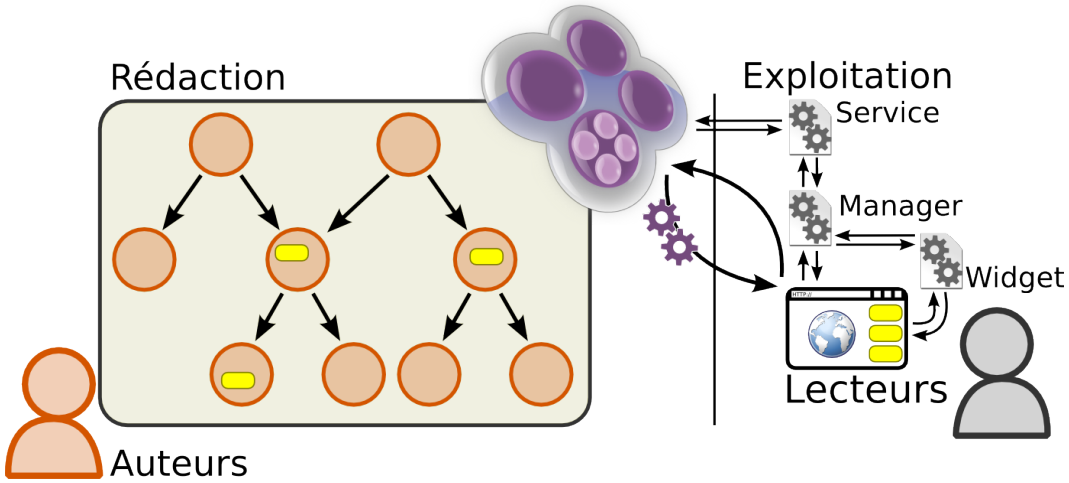
Ces deux modes de publication peuvent être comparés à l'affichage des résultats d'une requête au sein des systèmes de gestion de bases de données. Une publication dynamique peut être assimilée à une vue (affichage des résultats dynamique, recalculé à chaque requête) tandis qu'une publication statique peut être assimilée à une vue matérialisée (copie statique des résultats d'une requête).

Architecture du système de commentaires

Le système de commentaires s'appuie sur le principe de publication dynamique. L'usage d'un générateur dynamique permet de lever la contrainte liée à la version du document publié utilisé pour éditer les commentaires. En mobilisant une page HTML produite à la volée, seule les dernières mises à jour sont mobilisées.

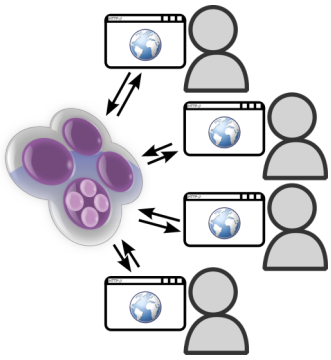
Les pages HTML envoyées aux navigateurs contiennent des scripts qui permettent d'adresser un service de la chaîne éditoriale dédié à la gestion des commentaires. Trois fichiers de script sont utilisés :

- un fichier appelé *service* qui assure la communication avec la chaîne éditoriale ;
- un fichier appelé *manager* qui communique avec le service, analyse la page HTML et positionne les commentaires ;
- Un fichier appelé *widget* qui fait partie du modèle documentaire et qui a pour but de manipuler le code HTML des documents pour ajouter les commentaires et les fonctions d'édition associées de façon adéquate en fonction des documents publiés.



Manager et Service du client du module de commentaires

La chaîne éditoriale écoute les requêtes HTTP dédiées aux commentaires. Le service dédié permet de lister les commentaires existants dans un ou plusieurs fragments ou d'éditer des commentaires : ajouter un nouveau fil de discussion, supprimer un fil ou un commentaire de la discussion, éditer un commentaire d'un fil de discussion, clore ou ré-ouvrir un fil de discussion et enfin répondre à une discussion. L'architecture du système permet à plusieurs utilisateurs de visualiser simultanément une publication dynamique et donc d'éditer simultanément des commentaires.



Consultation et édition des commentaires simultanée

Système de liens profonds

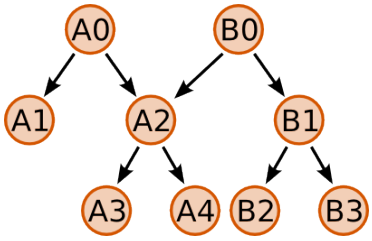
Afin de déterminer la localisation précise d'un commentaire au sein d'un fragment, nous utilisons un principe de lien profond. Le mot *lien* fait référence à un système d'adressage d'un fragment à un autre. Le mot *profond* désigne la possibilité d'aller désigner un élément à l'intérieur d'un fragment.

Un lien profond se structure en deux parties. La première partie définit une adresse vers un fragment résolvable par une chaîne éditoriale. Pour les chaînes éditoriales Scenari, il s'agit soit d'un chemin sur le système de fichiers, soit d'un identifiant de la base de données (en fonction du module de stockage des fragments utilisés). La seconde est la désignation d'un élément XML au sein du fragment. La technologie classique pour la sélection d'un élément XML est XPATH. Cependant, cette technologie nécessite une connaissance du schéma XML

puisque la sélection s'opère par le nom des éléments. Nous avons adopté un système plus neutre vis à vis de la structure du fichier XML. Un premier signe « / » désigne l'élément racine. Un nombre désigne ensuite le numéro de l'élément fils sélectionné (« /0 » pour le premier fils, « /3 » pour le quatrième). Un nouveau caractère « / » et un nouveau nombre permettent de sélectionner un élément petit fils et ainsi de suite. Nous utilisons le caractère « @ » pour séparer les deux parties du lien profond.

Exemple

Soit le graphe documentaire suivant. Les étiquettes des sommets correspondent aux identifiants des fragments en base de données.



Graphe documentaire

Posons l'hypothèse que le fragment A2 ait le contenu suivant :

```
1 <sc:item xmlns:sc="http://www.utc.fr/ics/scenari/v3/core">
2   <of:section xmlns:of="scpf.org:office" xmlns:sp=
  "http://www.utc.fr/ics/scenari/v3/primitive">
3     <of:sectionM>
4       <sp:title>La rééditorialisation documentaire, du document au
  fragment</sp:title>
5     </of:sectionM>
6     <sp:sec sc:refUri="id:A3"/>
7     <sp:sec sc:refUri="id:A4"/>
8   </of:section>
9 </sc:item>
```

La sélection du second élément *sp:sec* se fait donc comme suit par le lien profond « A2@/0/2 ». Le zéro sélectionne le premier fils de la racine, soit l'élément *of:section* et le chiffre deux sélectionne le troisième fils de cet élément, soit le second *sp:sec*.

Association d'un commentaire à un élément

Nous avons choisi d'insérer les commentaires en tant que premiers éléments fils de l'élément auquel ils sont associés. Par exemple, dans l'exemple de code suivant, un commentaire est associé au second élément *sp:sec* :

```
1 <sc:item xmlns:sc="http://www.utc.fr/ics/scenari/v3/core">
2   <of:section xmlns:of="scpf.org:office" xmlns:sp=
  "http://www.utc.fr/ics/scenari/v3/primitive">
3     <of:sectionM>
4       <sp:title>La rééditorialisation documentaire, du document au
  fragment</sp:title>
5     </of:sectionM>
6     <sp:sec sc:refUri="id:A3"/>
7     <sp:sec sc:refUri="id:A4">
8     <!--<comment xmlns="scenari.eu:comment:1.0" type="thread">
9       <comment creationTime="1405687267908">
10      La notion d'ingénierie documentaire n'est pas assez présente
  dans cette section.
11      À reprendre.
12      </comment>
13      <comment creationTime="1405687276904">
14      Ok, c'est noté.
```

```
15        </comment>
16      </comment>-->
17      </sp:sec>
18    </of:section>
19 </sc:item>
```

Le lien profond permettant de sélectionner le fil de commentaire est « A2@/0/2/0 » (car l'élément de commentaire XML est le premier fils de « A2@/0/2 »). Afin de désigner un commentaire au sein d'une discussion, nous proposons d'utiliser une troisième partie au sein des liens profonds. Cette troisième partie spécifie uniquement le numéro du commentaire dans la liste. Par exemple, le lien profond permettant de sélectionner le commentaire contenu dans le fil de commentaires est « A2@0/2/0@0 ». Le premier commentaire de réponse à la discussion sera désigné par « A2@0/2/0@1 ».

Insertion des liens profonds dans les documents HTML

Lorsqu'un élément issu d'un fragment est transformé en un élément HTML, l'algorithme de transformation ajoute automatiquement un attribut *data-origin* à l'élément HTML. Cet attribut suit les recommandations du standards HTML5 pour l'insertion de données dans des pages HTML (insertion dans un attribut dont l'intitulé débute par « data- ») (W3C, 2014, section 3.2.5.9). Ainsi, au fur et à mesure de la génération du fichier HTML, plusieurs éléments sont enrichis avec des liens profonds vers des éléments XML contenus dans des fragments de la chaîne éditoriale.

Détection des liens profonds et édition des commentaires

La partie cliente intégrée à une publication web détecte l'ensemble des éléments contenant un lien profond. Ces éléments sont considérés comme pouvant contenir des commentaires. L'ensemble des ces liens sont transmis au service dédié de la chaîne éditoriale qui, pour chacun des liens, liste l'ensemble des commentaires associés. La liste complète des commentaires à afficher est ainsi envoyée par le serveur et les commentaires sont dynamiquement attribués aux éléments HTML la page publiée.

Le système de lien profond permet ensuite de désigner de façon rigoureuse les éléments classiques commentables, les fils de commentaires et les commentaires au sein de ces fils afin de proposer des fonctions d'édition, d'ajout ou de suppression.

Système d'édition distribué et synchrone

Transformées opérationnelles

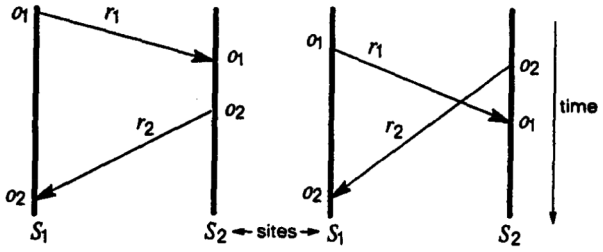
La notion de transformée opérationnelle désigne une technique mobilisée dans le domaine des travaux coopératifs assistés par ordinateur pour développer des systèmes collaboratifs, distribués et supportant l'édition synchrone. Elle a été initialement définie par Ellis et Gibbs (1989), implémentée et testée au sein du logiciel GROVE.

Les systèmes d'édition synchrones s'appuient sur une architecture distribuée. Un système G, noté <S, O> y est défini par un certain nombre de sites S et un ensemble d'opérateurs O. Un site correspond à un programme utilisé par un utilisateur relié aux autres à travers un réseau de communication. Les opérateurs O définissent l'ensemble des actions réalisables dans l'éditeur (par exemple pour un éditeur de texte : insertion, suppression ou substitution d'un caractère).

Une opération correspond à un opérateur et ses paramètres. Par exemple, l'insertion d'un caractère est une opération mobilisant l'opérateur insertion, la lettre insérée et la position du caractère (comme par exemple : ligne 22, colonne 12). Un site peut soit : générer des opérations suite aux actions de l'utilisateur et envoyer ces opérations aux autres sites ; réceptionner une opération d'un autre site et l'exécuter localement. Chaque site dispose ainsi d'une file d'opérations ordonnancées en fonction de la date de leur génération.

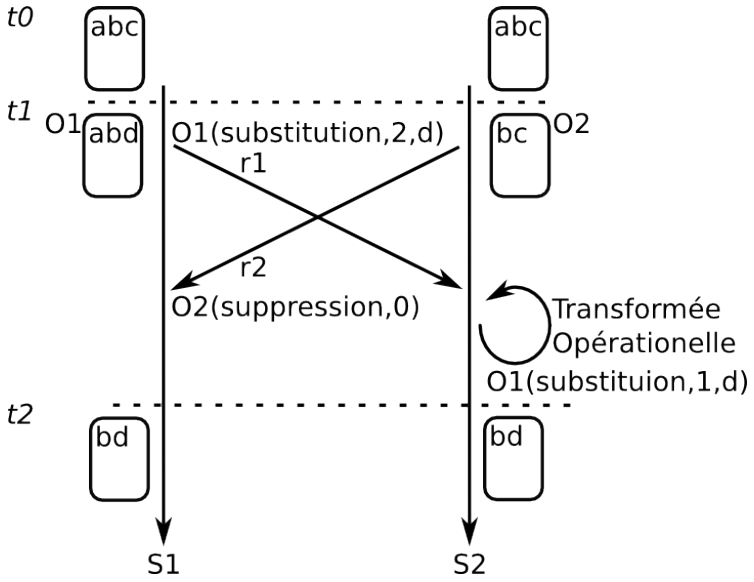
Ellis et Gibbs définissent un système au repos lorsque toutes les opérations générées ont été exécutées sur tous les sites (« A groupware session is quiescent iff all generated operations have been executed at all sites, that is, there are no requests in transit or waiting to be executed by a site process. » (ibid., p. 402)). Une opération est non conflictuelle si elle est réalisée sur un

site ayant exécuté toutes les opérations générées. À l'inverse, une opération sera conflictuelle si certaines opérations générées sur un site distant n'ont pas encore été exécutées sur le site local au moment de leur génération. La figure suivante illustre une suite d'opérations non conflictuelles sur deux sites (gauche) et la même suite conflictuelle (à droite).



Situation non conflictuelle (gauche) et conflictuelle (droite) (Ellis & Gibbs, 1989)

La notion de transformée opérationnelle désigne la technique de transformation des opérations afin de permettre la résolution des situations conflictuelles. Par exemple, soit un système composé de deux sites permettant l'édition synchrone de texte. À t0, les deux sites affichent la suite de caractères « abc ». Au même moment t1, S1 génère une opération O1 de substitution de la colonne 2 par la lettre d (donc « abc » devient « abd » en numérotant la première colonne 0) et S2 génère une opération O2 de suppression du caractère situé à la colonne 0 (donc « abc » devient « bc »). À la réception de la requête r1, le système S2 effectuera une transformation de l'opération O1 afin de prendre en compte l'opération O2 déjà générée et exécutée. Dans le cas présent, le numéro de la colonne (2) doit prendre en compte la suppression de la première colonne et être transformé en 1. L'exécution de l'opération O1 avant l'opération O2 n'a en revanche aucune incidence sur le site S1 puisque la colonne 0 désigne toujours la même ressource (la lettre a). Une fois la transformation effectuée par S2 et le système revenu dans une situation de repos, les deux sites affichent la chaîne de caractères « bd ».



Exemple de transformée opérationnelle

Le système de transformées opérationnelles a été considérablement amendé et retravaillé depuis la proposition originale de Ellis et Gibbs jusqu'à connaître des usages industriels poussés au sein des logiciels Google Documents ou Google Wave (Wang, 2010).

Contraintes

Par rapport à des systèmes de transformées opérationnelles classiques, l'édition de commentaires dans le graphe à travers ses transformations en documents HTML pose plusieurs contraintes.

Système centralisé

Les systèmes de transformées sont fondés sur un usage distribué. L'objet du système de commentaires distribué est de visualiser et éditer des commentaires dans le graphe. Seule la chaîne éditoriale a accès en lecture et écriture sur le graphe. Il est donc nécessaire de s'appuyer sur une architecture centralisée : la chaîne éditoriale au centre, chaque site correspondant à une copie locale d'une partie des commentaires du graphe. En cas de conflit, c'est systématiquement la version centralisée qui primera. Chaque site se retrouve donc en communication exclusive avec le serveur. Aucune opération générée depuis un site ne peut être directement transmise aux autres.

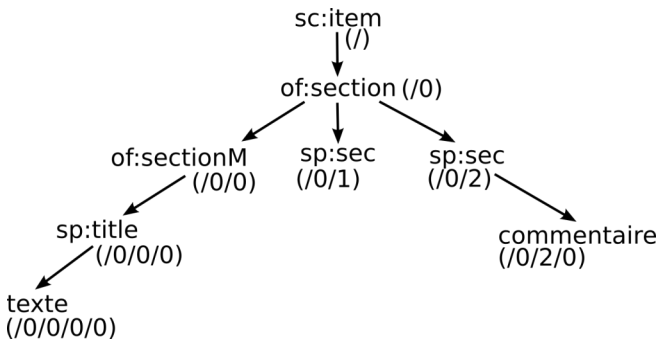
Structures éditées hétérogènes

Deux structures de données différentes sont éditées sur un site. En ajoutant ou en supprimant un fil de commentaires, des éléments XML sont ajoutés ou supprimés des fichiers matérialisant les fragments, modifiant ainsi les chemins profonds. En ajoutant ou supprimant un commentaire dans un fil de discussion, des éléments XML sont ajoutés ou supprimés du fil, modifiant ainsi les numéros des commentaires dans leurs fils respectifs.

Soit par exemple le fichier XML suivant :

```
1 <sc:item xmlns:sc="http://www.utc.fr/ics/scenari/v3/core">
2   <of:section xmlns:of="scpf.org:office" xmlns:sp=
  "http://www.utc.fr/ics/scenari/v3/primitive">
3     <of:sectionM>
4       <sp:title>La rééditorialisation documentaire, du document au
  fragment</sp:title>
5     </of:sectionM>
6     <sp:sec sc:refUri="id:A3"/>
7     <sp:sec sc:refUri="id:A4">
8     <!--<comment xmlns="scenari.eu:comment:1.0" type="thread">
9       <comment creationTime="1405687267908">
10      La notion d'ingénierie documentaire n'est pas assez présente
  dans cette section.
11      À reprendre.
12      </comment>
13      <comment creationTime="1405687276904">
14      Ok, c'est noté.
15      </comment>
16    </comment>-->
17    </sp:sec>
18  </of:section>
19 </sc:item>
```

Ce fichier peut être représenté sous la forme de l'arbre suivant :



Fragment XML représenté sous la forme d'un arbre

L'ajout d'un nouveau fil de discussion en premier fils de l'élément *of:section* décale l'ensemble des chemins profonds.

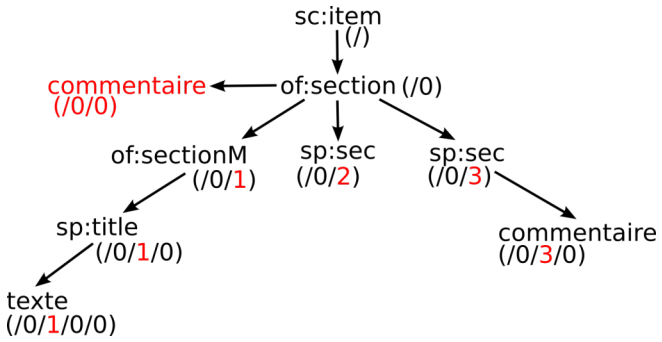


Illustration de la modification des liens profonds suite à l'ajout d'un commentaire

Éditeur XML de la chaîne éditoriale

Le service de commentaires effectuant le traitement des requêtes de chacun des sites permet la définition d'opérateurs génériques et de lister de façon fine les opérations d'ajout, de suppression ou de modification de commentaires effectuées sur un fragment. En revanche, les fragments restent éditables dans l'éditeur XML de la chaîne éditoriale. L'architecture de cet éditeur ne mémorise pas les opérations effectuées sur le fragment. La seule information valable correspond à une modification du fragment source. Cette modification peut concerner un caractère sur un élément XML aussi bien que l'ensemble du fichier.

Hétérogénéité des données stockées par chaque site

Enfin, dernière contrainte à prendre en compte, chaque site ne partage pas les mêmes données. Un site correspond à la transformation à la volée d'une partie du graphe en une page HTML. Plusieurs sites pourront avoir des portions de graphe se recouvrant intégralement, partiellement, ou pas du tout.

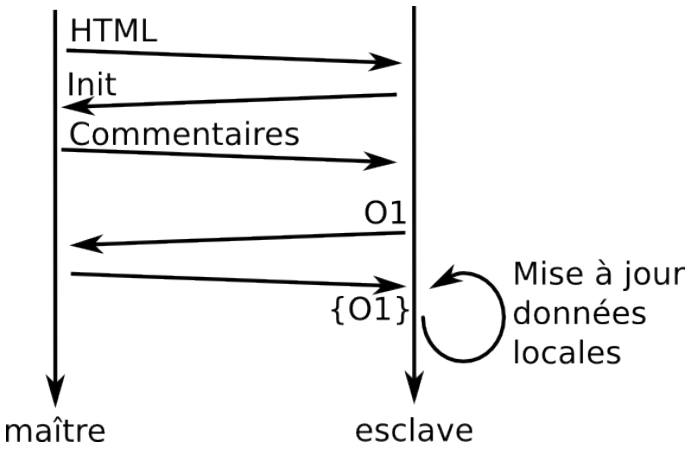
Fonctionnement

Site maître - sites esclaves

Pour que le système puisse fonctionner, nous définissons un site maître et des sites esclaves. Le site maître représente la chaîne éditoriale. C'est lui qui tient à jour la liste des opérations en mémoire afin de synchroniser les sites esclaves. Nous définissons le système de commentaires au repos lorsqu'aucune opération n'est stockée en mémoire.

Un site esclave représente un document HTML permettant l'édition de commentaires. Contrairement au site maître, un site esclave est donc éphémère et dure le temps d'une session d'un utilisateur sur un navigateur web. À son initialisation, le site esclave analyse ses éléments HTML et fournit une liste de liens profonds au site maître qui analyse le graphe et fournit une liste de commentaires datée (par date, nous entendons une mesure précise de l'instant à la milliseconde près selon la technique en usage dans les systèmes informatiques, soit le nombre de millisecondes depuis le 1er janvier 1970 à 00h00:00 UTC. Cette mesure du temps est également appelée temps UNIX).

Seuls les sites esclaves peuvent générer des opérations. Lorsqu'une opération est générée, elle est envoyée au site maître sans être exécutée localement. Le site maître applique si nécessaire une transformée opérationnelle avant d'exécuter l'opération. À ce stade, le modèle de données du site esclave n'est pas modifié (les liens profonds du code HTML, la liste des commentaires et de leurs liens profonds respectifs). Une fois l'opération exécutée, le site maître envoie une réponse au site esclave contenant l'ensemble des modifications à apporter à son modèle de données. Ces modifications sont réordonnancées par le site maître dans l'ordre de leur exécution. Il n'y a donc plus de transformées à exécuter.



Site maître - site esclave

Dans cette illustration, l'initialisation d'un esclave se fait en trois temps : la transformation d'une partie du graphe en une page HTML, la demande d'initialisation de l'esclave et enfin la liste des commentaires en cours. Lorsqu'une opération O1 est générée par le site esclave, elle est envoyée au site maître sans être exécutée localement. Le site maître la réceptionne, l'exécute et renvoie la liste de toutes les opérations effectuées depuis la dernière mise à jour du site esclave. Cet exemple est trivial puisqu'un seul site esclave est mobilisé. La liste retournée n'est composée que de O1 et aucune transformée opérationnelle n'est nécessaire.

Opérateurs

Lorsqu'un esclave génère une opération, il mobilise un langage d'opérateurs variés permettant de manipuler les commentaires : création d'un fil de discussion, réponse à un commentaire, fermeture ou ré-ouverture d'un fil ou encore suppression d'un commentaire.

Le site maître exécute ces commandes et traduit ces actions dans un langage d'opérateurs plus simple, optimisé pour maintenir la validité des liens profonds. Chacun de ces opérateurs prend le lien profond et les détails du fil de discussion en paramètre. Trois opérateurs sont utilisés :

- la création d'un nouveau fil de discussion est traduit par un opérateur d'ajout, noté « + » ;
- la modification d'un commentaire, la réponse, la clôture ou réouverture d'un fil de discussion sont exprimées par un opérateur de modification, noté « ~ » ;
- la suppression d'un fil de commentaires se traduit par un opérateur de suppression, noté « - ».

Lors de la réception d'une liste d'opérations, un site esclave exécute les opérations les unes après les autres. Une opération d'ajout ou de suppression nécessite la mise à jour de l'ensemble des liens profonds. Une opération de modification implique uniquement un remplacement de la structure de données du fil de commentaires et de son affichage.

Mise à jour des liens profonds

Le site esclave maintient une liste de liens profonds composée du lien de chaque fil de discussion et de chaque élément HTML disposant d'un attribut « data-origin ». Cette liste doit être à jour pour associer un fil de discussion à son premier élément XML parent au sein du fragment de document d'origine.

Soit un opérateur « + »ou « - » associé au lien profond « L =id@/X/.../Y/Z ». On notera L:id la valeur de l'identifiant et L:chemin le tableau des nombres [X, ..., Y, Z]. L:chemin.taille désigne la taille du tableau et L:chemin[i] la valeur du ième nombre.

Pour chacun des liens profonds Ln, l'algorithme de mise à jour procède comme suit :

```
1 SI (Ln:id != L:id) ALORS STOP FINSI
2 SI (Ln:chemin.taille < L:chemin.taille) ALORS STOP FINSI
3 POUR (i = 0 ; i < L:chemin.taille - 1 ; i++) FAIRE
4   SI (L:chemin[i] != Ln:chemin[i]) ALORS
```

```
    STOP
  FINSI
FINPOUR
SI (L:chemin[L:chemin.taille-1] < Ln:chemin[L:chemin.taille-1])
ALORS
  SI (opérateur == +) ALORS
    Ln:chemin[L:chemin.taille-1]++
  FINSI
  SI (opérateur == -) ALORS
    Ln:chemin[L:chemin.taille-1]--
  FINSI
FINSI

```

Stockage des listes d'opérations

Au repos, le site maître ne contient pas de liste d'opérations. Lorsqu'une opération est reçue, une liste associée à un fragment est constituée. Toute nouvelle opération sur ce fragment est ajouté à cette liste.

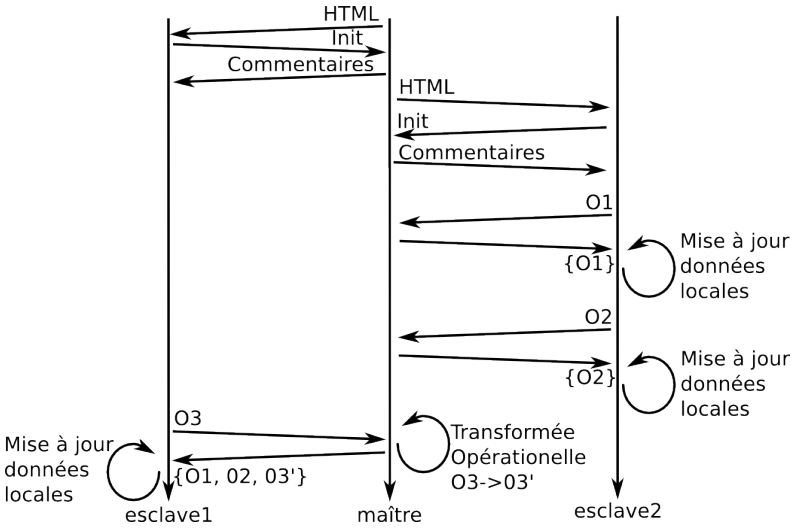
Il existe deux méthodes de suppression d'une liste :

- lorsqu'un fragment est modifié dans l'éditeur XML par un rédacteur, aucune indication sur la nature de la modification n'est conservée, la liste d'opérations est alors supprimée ;
- lorsque la chaîne éditoriale demande la libération d'espace mémoire, une fonction supprime les listes existantes en fonction de la priorité de la demande de libération de mémoire et de l'âge du dernier ajout d'opération à la liste.

Si des sites esclaves associés à une liste sont en usage lors de la suppression, leur usage devient impossible car les transformées opérationnelles ne peuvent plus être réalisées. Il est nécessaire de recommencer l'édition en réinitialisant un nouveau site.

Résolution des conflits

La situation typique de gestion de conflits s'opère comme illustrée sur le schéma suivant :



Gestion des conflits

Deux sites esclaves s'initialisent l'un après l'autre. Le second effectue plusieurs opérations avant que le premier ne valide sa première opération. Lorsque cette opération est réceptionnée par le site maître, elle est transformée si nécessaire puis envoyée avec les autres modifications apportées depuis la dernière réponse du site maître.

Les conflits au sein d'un fil de discussion ne sont pas gérés actuellement. Pour modifier un fil de discussion, un site esclave doit nécessairement avoir la dernière version à jour avant

d'effectuer l'envoi.

Développements

La partie serveur des développements est intégrée au cœur Java de Scenari. Elle peut être consultée au sein du dépôt de code source *(http://scenari-platform.org/svn/dev-core/branches/4.2.x/Jav_Wsp/src/eu/scenari/wsp/service/comment/)*.

Le service et le manager sont à mutualiser au sein de l'ensemble des publications web dynamiques. Ils sont placés sur un entrepôt de code source spécifique *(http://scenari-platform.org/svn/modelet/branches/sc41/scCommentMgr/model/sources/)* dont l'enjeu est de proposer des librairies Javascript prêtes à être embarquées au sein des modèles Scenari.

Les fichiers complémentaires permettant de finaliser l'intégration du système de commentaires sont disponibles au sein des sources du modèle de chaînes éditoriales libre Opale *(http://scenari-platform.org/svn/opale/branches/3.5-sc41/model/sources/academic/gen/quadra/comments/web/)*.

Items flottants

> Le terme *item* est l'appellation générique utilisée par Scenari pour désigner les fragments. Au sein des chaînes éditoriales Scenari, un arbre de gestion est systématiquement utilisé afin de permettre aux rédacteurs de ranger les items d'un atelier comme ils l'entendent. L'objectif d'un proxy est de permettre à des rédacteurs travaillant dans un atelier de rééditorialiser des items issus d'un autre atelier. Afin de clairement marquer la différence entre les items locaux à un atelier et les items importés par un mécanisme de proxy, nous suggérons d'introduire de nouveaux mécanismes d'indexation et d'accès aux items.
>
> Un *item flottant* est un item qui n'est pas rattaché à l'arbre de gestion. Son accès se fait à travers le moteur de recherche des items.

Le développement des items flottants constitue un préalable nécessaire au développement des proxys. L'enjeu est d'adapter la représentation des items au sein du cœur Java et de mettre à jour l'interface homme machine afin de permettre la recherche et le référencement d'items non situés dans l'arbre de gestion.

Développements

La partie serveur des développements est intégrée au cœur Java de Scenari. Il s'agit d'adapter la base de données de type graphe utilisée par Scenari. Un atelier y est représenté par un enregistrement (auquel sont associés les propriétés de l'atelier comme l'intitulé et le modèle documentaire). Un atelier est lié à un ou plusieurs enregistrements de type *item* ou *espace* par l'intermédiaire de liens représentant l'arbre de gestion. Les enregistrements de type *espace* sont eux mêmes liés à un ou plusieurs enregistrements de type *item* ou *espace*. La structure de l'arbre de gestion est donc strictement reproduite au sein de la base de données.

Les développements au sein du cœur Java de Scenari ont consisté à autoriser le référencement d'une liste d'enregistrements de type *item* directement depuis l'enregistrement de type *atelier* en utilisant une nouvelle typologie de lien (afin d'éviter de faire apparaître ces fragments dans l'arbre à gauche). Il devient ainsi possible de référencer un item issu d'un autre atelier et de débuter les développements de la fonction de proxy.

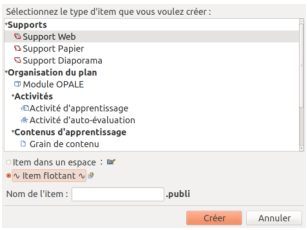
La partie cliente des développements concerne trois aspects de l'interface.

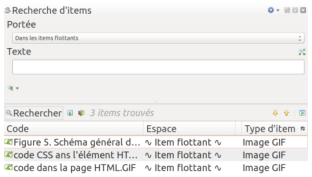
- Une action particulière à déclencher dans le menu contextuel d'un item afin de transformer un item classique en item flottant et inversement.
- La mise à jour du panneau de création des fragments classiques afin de permettre la création d'items flottants.
- La mise à jour du moteur de recherche afin de permettre de chercher des items flottants.

Les sources de ces développements sont disponibles au sein des sources de l'entrepôt

Scenari *(http://scenari-platform.org/svn/dev-core/branches/4.2.x/Xul_Wsp/content/)*.

Exploitation





Content Interactive Delivery

Content Interactive Delivery (CID) est une proposition de protocole dont l'enjeu est d'encadrer les transactions de documents et/ou de métadonnées entre deux systèmes. L'enjeu est de proposer une surcouche d'information et de décision au dessus des modalités d'échanges (requêtes HTTP, FTP, etc. correspondant à la couche applicative du modèle ISO) afin de permettre à deux systèmes de choisir les modalités d'une interaction.

Le principe du protocole consiste à imposer à un serveur de décrire ses processus fonctionnels et les modalités de transport acceptées au sein d'un manifeste librement accessible par une requête HTTP. Un client peut donc récupérer le manifeste, l'interpréter et sélectionner un processus fonctionnel (téléversement, mise à jour, sélection d'un fichier distant, etc.) et les modalités de transport l'accompagnant (par exemple, une requête HTTP POST avec des métadonnées stockées dans un formulaire).

La définition d'un processus fonctionnel se fait en définissant des métadonnées et des étapes (facultatives ou non) à enchaîner. Ces étapes peuvent être un simple échange de données (*exchange*), le téléversement d'un fichier (*upload*) ou une interaction entre le serveur et l'usager du client (*interact*).

La couche transport peut en principe définir plusieurs modalités de transport mais seul le transport web (requêtes HTTP) est défini dans le cœur du protocole.

La conception du protocole est prévue pour être enrichie par de nouvelles modalités de transport permettant de définir l'usage d'autres protocoles (FTP, SSH, etc.).

Cette section présente les développements réalisés autour du protocole CID. Une première section est dédiée au modèle du manifeste, une seconde à la partie client développée dans Scenari et une troisième et dernière dédiée au client et aux serveurs génériques développés comme démonstrateurs.

Manifeste

```
1 datatypes xsd = "http://www.w3.org/2001/XMLSchema-datatypes"
2 namespace cid="http://www.cid-protocol.org/schema/v1/core"
3 namespace local = ""
4
5 grammar{
6
7 #Defininition of a user description
8 user-desc =
9   element cid:label{attribute xml:lang {text}?,text}*,
10  element cid:doc{attribute xml:lang {text},text}*
11
```

```
#Definition of a step
step-def =
  #URL to use in the transport
  attribute url{xsd:anyURI},
  #Mandatory metas
  attribute needMetas{list{xsd:NCName+}}?,
  #Accepted metas
  attribute useMetas{list{xsd:NCName+}}?,
  #Returned metas
  attribute returnMetas{list{xsd:NCName+}}?,
  #Mandatory step
  attribute required{xsd:boolean},
  #URI which define the step
  attribute is{list{xsd:anyURI+}}?

#Definition of a request for web interact
webInteractRequest-def =
  element cid:request{
    #Supported methods
    attribute
method{"GET"|"POST;application/x-www-form-urlencoded"|"POST;multipart/form
    #Supported property storages
    attribute properties{list{("header" | "queryString" |
"post")+}}?,

    attribute excludeHeaderUriEncoding{list{xsd:NCName+}}?
  }

#Definition of a request for web data exchange
webExchangeRequest-def =
  element cid:request{
    #Supported methods
    attribute
method{"GET"|"POST;application/x-www-form-urlencoded"|"POST;multipart/form
    #Supported property storages
    attribute properties{list{("header" | "queryString" |
"post")+}}?,

    attribute excludeHeaderUriEncoding{list{xsd:NCName+}}?
  }

#Definition of a request for web upload
webUploadRequest-def =
  element cid:request{
    #Supported methods
    attribute method{"PUT"|"GET"|"POST"|"POST;multipart/form-data"},
    #Supported property storages
    attribute properties{list{("header" | "queryString" |
"post")+}}?,

    attribute excludeHeaderUriEncoding{list{xsd:NCName+}}?
  }

#Definition of a transport property
prop-def = element property{attribute key{xsd:NCName},attribute
value{xsd:NCName}}

#Definition of chunked file upload
chunk-def =
  element preliminaryChunk{attribute
strategy{"sequential"|"unordered"|"random"},prop-def?}?,
  element finalChunk{prop-def}?
```

```
67
68 #Definition of system wait
69 systWait-def = element systemWait{prop-def?,webExchangeRequest-def+}
70
71 #Definition of user wait
72 userWait-def = element userWait{prop-def?,webInteractRequest-def+}
73
74 #Definition of foreign element (use to valid a CID extension)
75 foreign-elements = element * - (cid:* | local:*) { anything  }*
76 anything = ( element * - (cid:* | local:*) { anything } | attribute
   * { text } | text )*
77
78
79 start =
80 element cid:manifest{
81   #Manifest documentation
82   user-desc,
83
84   #Processes definition
85   element cid:process{
86     #Transports binded to this process
87     attribute transports{xsd:IDREFS}?,
88     #URI which defines this process
89     attribute is{list{xsd:anyURI+}}?,
90     #Process documentation
91     user-desc,
92
93     #Restriction on the meta used by the system
94     element cid:restriction{
95       #Name of the meta
96       attribute name{xsd:NCName},
97       #URI which defines the meta
98       attribute is{list{xsd:anyURI+}}?,
99       #Mandatory value of the meta
100      attribute value{text},
101      #Meta documentation
102      user-desc
103    }*,
104
105    #Definition of a meta
106    element cid:meta{
107      #Name of the meta
108      attribute name{xsd:NCName},
109      #Accepted cardinality of the meta
110      attribute cardinality{"1"|"?"|"+"|"*"}?,
111      #URI which defines the meta
112      attribute is{list{xsd:anyURI+}}?,
113      #Meta documentation
114      user-desc,
115      #Restricted values
116      element cid:value{text}*
117    }*,
118
119    (
120      #Data exchange step
121      element cid:exchange{step-def, attribute
   exchangeTransports{xsd:IDREFS}?} |
122      #User interact step
123      element cid:interact{step-def, attribute
   interactTransports{xsd:IDREFS}?} |
124      #File upload step
```

```
125       element cid:upload  {step-def, attribute
uploadTransports{xsd:IDREFS}?}
126     )+
127   }+,
128
129   #Definition of supported authentications scheme
130   element cid:authentications{
131     #No authentication
132     element cid:noAuthentication{empty}?&
133     #Basic HTTP Authentication
134     element cid:basicHttp{empty}?&
135     #CAS Authentication
136     element cid:cas{attribute url{xsd:anyURI}}?&
137     #Internal Authentication (+cookie or properties)
138     element cid:internal{attribute url{xsd:anyURI},attribute
authProperties{list{xsd:NCName+}}?}?&
139     #CID extension allowed
140     foreign-elements*
141   }?,
142
143   #Supported transport definition
144   element cid:transports{
145     #Web transport definition
146     element cid:webTransport{
147       #Id of the transport
148       attribute id{xsd:ID}?,
149       #Mandatory support of cookies
150       attribute needCookies{xsd:boolean}?,
151       #Mandatory support of session properties. Properties name
definition
152       attribute sessionProperties{list{xsd:NCName+}}?,
153       (
154         #Data exchange web transport definition
155         element cid:webExchange{attribute
id{xsd:ID}?,webExchangeRequest-def+}|
156         #User interact web transport definition
157         element cid:webInteract{attribute
id{xsd:ID}?,webInteractRequest-def+}|
158         #File upload web transport definition
159         element cid:webUpload  {attribute
id{xsd:ID}?,webUploadRequest-def+, chunk-def }|
160         #Asynchronous file upload definition
161         element cid:webAsyncUpload{
162           attribute id{xsd:ID}?,
163           webUploadRequest-def+,
164           ((systWait-def, userWait-def?) | (userWait-def,
systWait-def?)),
165           chunk-def
166         }
167       )+
168     }*,
169     #CID extension allowed
170     foreign-elements*
171   }
172 }
173
174 }
```

Client Scenari

Les sources de Scenari contiennent un client CID générique. Son objectif est d'être le

connecteur privilégié pour toute transaction effectuée par l'intermédiaire du protocole CID. À ce jour, ce client est mobilisé dans trois contextes d'usages différents.

- Le client CID peut être paramétré dans un modèle documentaire afin d'envoyer un document issu de la génération d'un sous-graphe vers une plate-forme tiers. Un bouton *envoyer* est alors proposé à l'issue de la génération et le client CID prend le relais.
- Le client CID peut-être paramétré dans un modèle documentaire pour être appelé lors de l'édition d'une URL. Ainsi, lorsqu'un modèle prévoit le référencement d'une URL depuis un serveur particulier, celui-ci peut proposer un serveur CID comprenant une interface de sélection de l'URL et simplifier la saisie pour le rédacteur.
- Le logiciel de modélisation SCENARIbuilder contient des primitives documentaires de ressources distantes qui s'appuient sur CID. L'enjeu est de proposer des items de ressources (vidéo, audio, image) issues d'un serveur de média (*Digital Asset Management* - DAM) et de permettre leur utilisation au même titre qu'une ressource locale.

Développement

Le client CID intégré à Scenari est développé dans la couche cliente de Scenari. Les sources (aussi bien la partie interface *(http://scenari-platform.org/svn/dev-core/branches/4.2.x/Xul_Wsp/content/Wsp/windows/cid/cid.xul)*, que le contrôleur *(http://scenari-platform.org/svn/dev-core/branches/4.2.x/Xul_Wsp/content/Wsp/utils/cid/cid.jsm)*)

sont disponibles en ligne dans le dépôt de sources de Scenari.

Client et serveur de démonstration

Dans l'objectif de tester, démontrer et diffuser le protocole CID, nous avons développé un client générique et deux serveurs de fichiers de démonstration. Ces trois projets sont accessibles dans un dépôt de sources dédié *(https://github.com/scenari/cid)*.

Client générique

Le client générique est développé en HTML et Javascript. Il est à télécharger et entreposer sur un serveur web classique (un simple serveur web statique suffit).

Aucune installation particulière n'est nécessaire, il suffit de copier le fichier *index.html* et le dossier *js* sur un serveur web.

L'utilisation du client se fait alors dans un navigateur supportant les API File *(http://www.w3.org/TR/FileAPI/)* et Web Messaging *(http://www.w3.org/TR/webmessaging/)* de HTML5. Une fois le fichier *index.html* chargé, il suffit de rentrer l'URL d'un manifeste et de suivre les instructions du client.

Copie d'écran du client HTML générique

SingleCIDServer : un serveur de fichiers simple

SingleCIDServer est un serveur de fichiers écrit en PHP permettant le téléversement de fichiers dans un unique dossier par l'intermédiaire du protocole CID. Il est composé d'un unique

fichier php à téléverser sur un serveur PHP. Une installation en ligne est préalable à toute utilisation.

Processus de déploiement

Le manifeste de SingleCIDServer propose un déploiement en deux étapes :

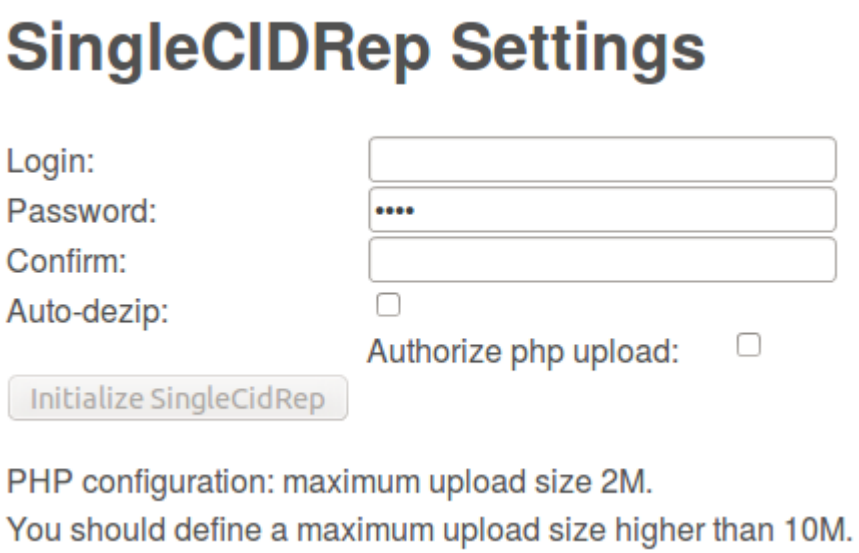
1. une première étape *exchange* facultative dont l'enjeu est de tester l'authentification fournie par le client ;
2. une seconde étape *upload* permettant de réaliser le déploiement.

```
<?xml version='1.0' encoding='UTF-8'?>
<cid:manifest xmlns:cid='http://www.cid-protocol.org/schema/v1/core'
>
  <cid:process>
    <cid:label xml:lang='en'>Remote file repository</cid:label>
    <cid:doc xml:lang='en'>This server accepts any content and
entreposes it on a single repository.</cid:doc>

    <cid:meta name='file-name' cardinality='?' is=
'http://schema.org/name'>
      <cid:label>File name</cid:label>
    </cid:meta>
    <cid:meta name='file-url' cardinality='1' is=
'http://schema.org/url'/>
    <cid:exchange url=
"http://example.com/SingleCIDRep.php?cdaction=testAuth" required=
'false' is='http://schema.org/AuthorizeAction'/>
    <cid:upload url=
"http://example.com/SingleCIDRep.php?cdaction=upload" required=
'true'  useMetas='file-name' returnMetas='file-url'/>
  </cid:process>
  <cid:authentications>
    <cid:basicHttp/>
  </cid:authentications>
  <cid:transports>
    <cid:webTransport needCookies='false'>
      <cid:webExchange>
        <request method='GET' properties='header queryString'/>
        <request method='POST;application/x-www-form-urlencoded'
properties='post header queryString'/>
        <request method='POST;multipart/form-data' properties='post
header queryString'/>
      </cid:webExchange>
      <cid:webUpload>
        <request method='PUT' properties='header queryString'/>
        <request method='POST' properties='header queryString'/>
        <request method='POST;multipart/form-data' properties='post
header queryString'/>
      </cid:webUpload>
    </cid:webTransport>
  </cid:transports>
</cid:manifest>
```

Installation

Une fois le fichier de source téléversé sur un serveur PHP, l'installation se fait en ouvrant le fichier PHP dans un navigateur web (par exemple, à l'URL http://www.example.com/SingleCIDRep.php).



Installation d'un SingleCIDRep

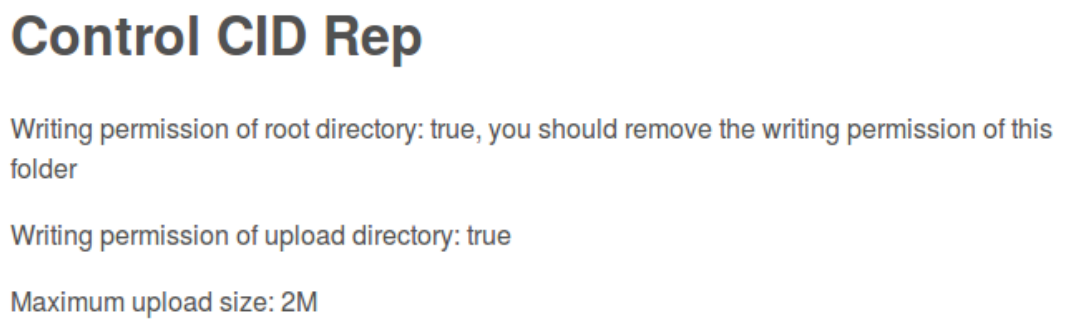
L'installation demande l'initialisation d'un login et d'un mot de passe pour effectuer les dépôts. L'option *auto-dezip* permet de désarchiver automatiquement les fichiers reçus. L'option *php upload* permet d'accepter le téléversement de fichiers php.

Utilisation

Une fois installé, le manifeste est disponible à l'URL du fichier PHP (par exemple http://www.example.com/SingleCIDRep.php). Les fichiers téléversés sont accessibles à l'URL parente (par exemple, URL http://www.example.com).

Administration

Une page de contrôle de l'installation peut être affichée en utilisant le paramètre *cdaction* et la valeur *control* (par exemple http://example.co m/SingleCIDRep.php?cdaction=control).



Copie d'écran de la page de contrôle de SingleCIDRep

Pour réinitialiser le nom d'utilisateur et le mot de passe, il est possible d'éditer le fichier *param.php* créé à l'installation ou simplement de le supprimer et de procéder à nouveau à l'installation.

Avertissement sécurité

Attention, ce serveur est un démonstrateur développé à la fois pour illustrer le fonctionnement de CID et encourager le développement de nouvelles parties serveur.

Il n'est en revanche pas recommandé pour un usage industriel. Le login et le mot de passe définis par l'utilisateur sont stockés en clair dans le fichier PHP.

SimpleCIDServer : un serveur de fichiers

SimpleCIDServer est un serveur de fichiers écrit en PHP permettant le téléversement de fichiers dans un ou plusieurs dossiers par l'intermédiaire du protocole CID. Il est composé d'un unique fichier PHP à téléverser sur un serveur PHP. Une installation en ligne est préalable à toute utilisation.

Processus de déploiement

Le manifeste de SimpleCIDServer propose un déploiement en trois étapes :

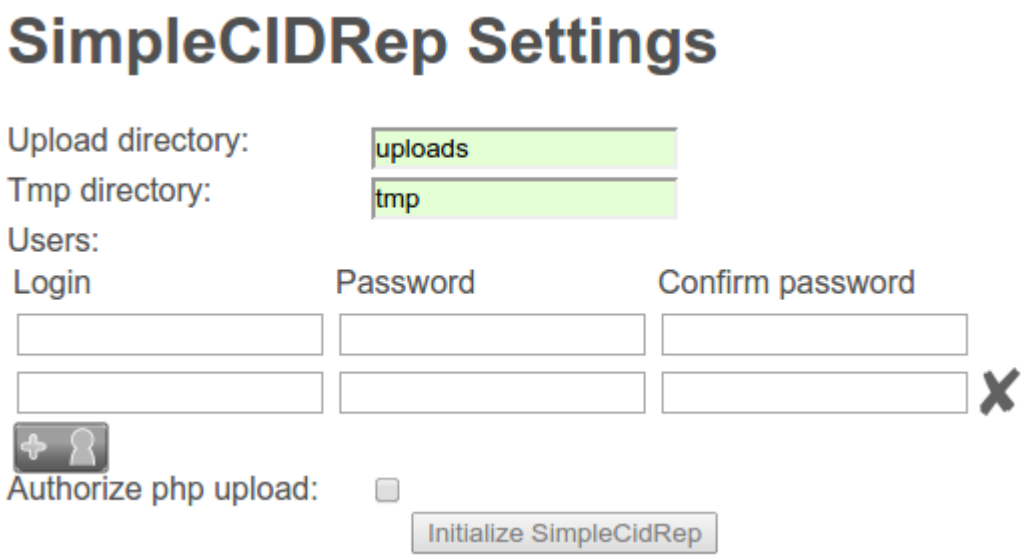
1. une première étape *exchange* facultative dont l'enjeu est de tester l'authentification fournie par le client ;
2. une seconde étape *upload* permettant de réaliser le déploiement ;
3. une étape d'interaction afin de permettre le déploiement du fichier téléversé dans un dossier de l'arborescence.

```
<?xml version='1.0' encoding='UTF-8'?>
<cid:manifest xmlns:cid=
'http://www.cid-protocol.org/schema/v1/core'>
 <cid:process>
   <cid:label xml:lang='en'>Deposit file</cid:label>
   <cid:doc xml:lang='en'>This server accepts any content and
allows the sender to choose the storage directory.</cid:doc>
   <cid:meta name='file-name' cardinality='?' is=
'http://schema.org/name'>
     <cid:label>File name</cid:label>
   </cid:meta>
   <cid:meta name='file-url' cardinality='1' is=
'http://schema.org/url'/>
   <cid:exchange url=
"http://example.com/SimpleCIDRep.php?cdaction=testAuth"
required='false' is='http://schema.org/AuthorizeAction'/>
   <cid:upload url=
"http://example.com/SimpleCIDRep.php?cdaction=upload" required=
'true' useMetas='file-name'/>
   <cid:interact url=
"http://example.com/SimpleCIDRep.php?cdaction=negotiationFrame"
 required='true' returnMetas='file-url'/>
 </cid:process>
 <cid:authentications>
   <cid:basicHttp/>
 </cid:authentications>
 <cid:transports>
   <cid:webTransport needCookies='false' sessionProperties=
'uploadedFile'>
     <cid:webExchange>
       <request method='GET' properties='header queryString'/>
       <request method=
'POST;application/x-www-form-urlencoded' properties='post
header queryString'/>
       <request method='POST;multipart/form-data' properties=
'post header queryString'/>
     </cid:webExchange>
     <cid:webUpload>
       <request method='PUT' properties='header queryString'/>
       <request method='POST' properties='header queryString'
/>
       <request method='POST;multipart/form-data' properties=
'post header queryString'/>
     </cid:webUpload>
     <cid:webInteract>
       <request method='GET' properties='queryString'/>
```

```
31         <request method=
  'POST;application/x-www-form-urlencoded' properties='post
  queryString'/>
32         <request method='POST;multipart/form-data' properties=
  'post queryString'/>
33       </cid:webInteract>
34     </cid:webTransport>
35   </cid:transports>
36 </cid:manifest>
```

Installation

Une fois le fichier de source téléversé sur un serveur PHP, l'installation se fait en ouvrant le fichier PHP dans un navigateur web (par exemple, à l'URL http://www.example.com/SimpleCIDRep.php).



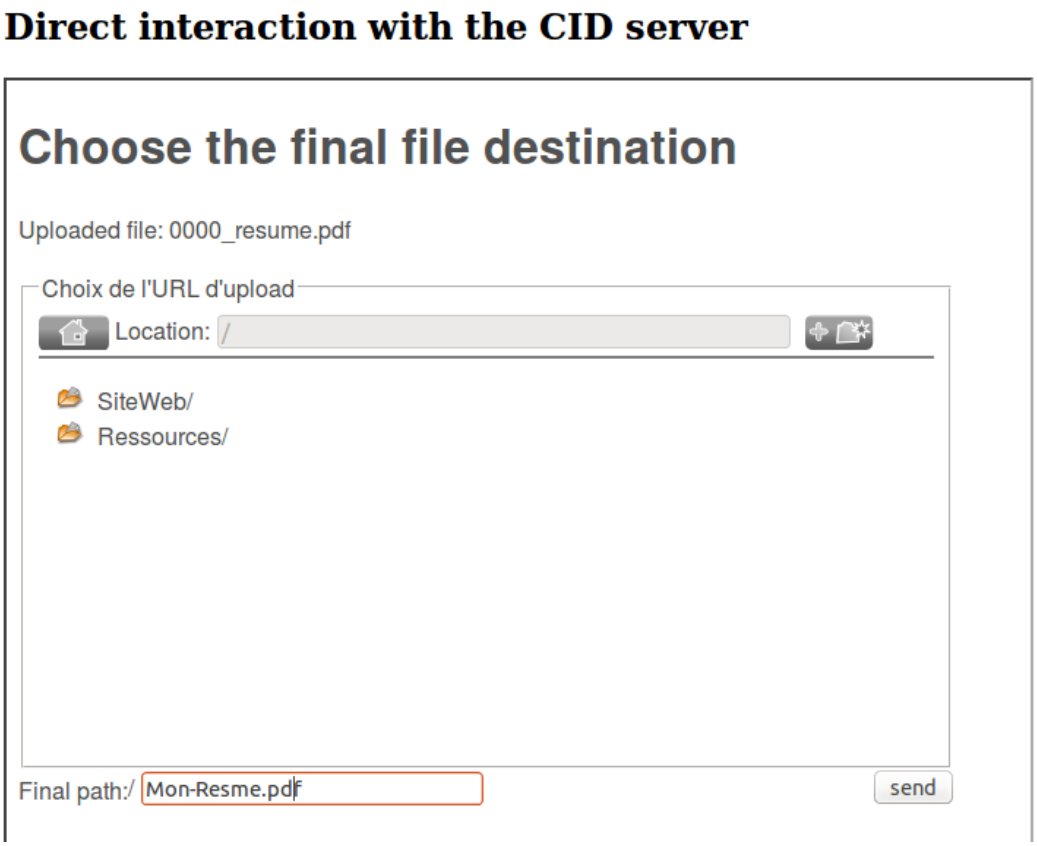
Installation d'un SimpleCIDRep

L'installation demande l'initialisation d'un ou plusieurs jeux de logins et mots de passe pour effectuer les dépôts, le nom du dossier dans lequel téléverser les fichiers et le nom du dossier temporaire (dans lequel un fichier est téléversé avant d'être classé dans l'arborescence du dossier *upload*).

Utilisation

Une fois installé, le manifeste est disponible à l'URL du fichier PHP (par exemple http://www.example.com/SimpleCIDRep.php). Les fichiers téléversés sont accessibles à l'URL parente (par exemple, URL http://www.example.com).

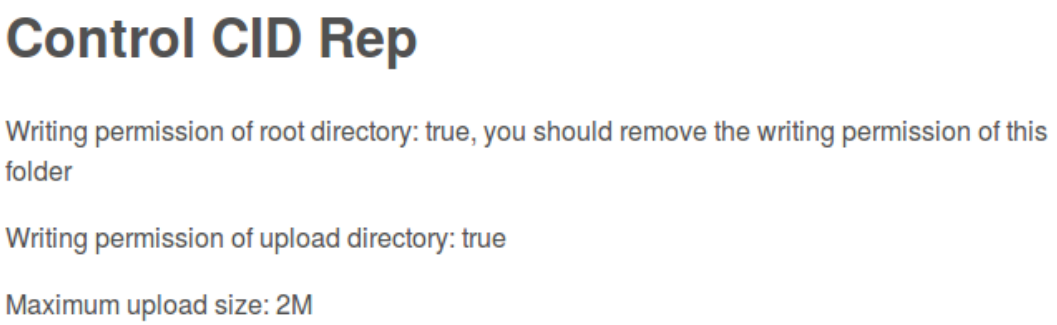
Après un téléversement, le serveur envoie une interaction permettant de sélectionner le chemin (au sein du dossier d'upload spécifié à l'installation) dans lequel déployer le fichier envoyé.



Interaction entre un SimpeCIDRep et un usager à travers le client générique

Administration

Une page de contrôle de l'installation peut être affichée en utilisant le paramètre *cdaction* et la valeur *control* (par exemple http://example.co m/SimpeCIDRep.php?cdaction=control).



Copie d'écran de la page de contrôle de SingleCIDRep

Pour réinitialiser les noms d'utilisateur et les mots de passe, il est possible d'éditer le fichier *param.php* créé à l'installation ou simplement de le supprimer et de procéder à nouveau à l'installation.

Annexe 4

Méthodologie pour la conception de chaînes éditoriales

Principe 253
Structuration des projets de rééditorialisation 253
Structure éditoriale 253
Compléments auctoriaux 254
Documentarisation de l'activité 255

Ce document synthétise la méthodologie proposée pour concevoir des chaînes éditoriales opérant des graphes documentaires complexes. Il se structure en trois parties, une première dédiée au principe général, une seconde à la structuration des projets de rééditorialisation et une troisième et dernière dédiée à la modélisation de l'activité.

Principe

La conception de chaînes éditoriales pour l'exploitation de graphes complexes s'articule autour des ateliers.

Dans un premier temps, le graphe est structuré en différents ateliers. L'objectif est de fournir un atelier par dynamique de rédaction (autant du point du vue des auteurs que des documents finaux). Ainsi chaque projet éditorial se voit attribuer un atelier puis chaque équipe de rédacteurs (voire chaque rédacteur le désirant) se voit attribuer son atelier de travail.

Dans un second temps, chacun des ateliers créés au sein desquels une réelle activité auctoriale est hébergée se voit associer un modèle d'activité dédié. L'enjeu est d'adapter le modèle documentaire existant à l'organisation de l'activité du ou des rédacteurs de chacun des ateliers. Il est donc possible d'aboutir à autant de modèles différents que d'ateliers.

Structuration des projets de rééditorialisation

Structure éditoriale

La conception de la chaîne éditoriale débute par la structuration des projets éditoriaux. L'objectif est de configurer un atelier par projet.

Étape 1, les projets principaux

La première étape consiste à configurer un atelier par **projet éditorial principal**. Un projet principal est un projet au cœur du dispositif éditorial. Il contient les versions de référence d'une documentation

Si les projets principaux mutualisent des fragments, les ateliers peuvent être liés par des fragments proxys.

Étape 2, les projets dérivés

La seconde étape consiste à configurer un atelier calque de dérivation par **projet éditorial dérivé**. Un projet éditorial dérivé est un projet qui mobilise l'ensemble des fragments d'un projet principal pour la publication de sa documentation.

Étape 3, les projets satellites

La troisième étape consiste à configurer un atelier par **projet éditorial satellite**. Un projet éditorial satellite est un projet qui mobilise un sous-ensemble des fragments utilisés dans un projet principal ou un projet dérivé.

Ces nouveaux ateliers sont reliés aux ateliers déjà configurés par des fragments proxys (de partage ou de dérivation).

Étape 4, les projets d'agrégation

La quatrième étape consiste à identifier les projets éditoriaux d'agrégation. Un projet d'agrégation est un projet éditorial dans lequel l'essentiel des fragments publiés est issu d'autres ateliers.

Lorsqu'un projet éditorial est exclusivement constitué de l'agrégation de publications existantes, il convient de se tourner vers l'usage de solutions issues de SCENARIdepot.

Au contraire, lorsque l'édition de fragments originaux ou la surcharge de fragments existants sont nécessaires, l'agrégation peut être faite dans un atelier de la structure. Il convient alors de configurer un nouvel atelier et d'initialiser un fragment proxy pour chacun des sous-graphe à agréger.

Compléments auctoriaux

Une fois la structure éditoriale posée, il faut alors la compléter avec des ateliers dédiés à organiser la rédaction.

Étape 1, protection du graphe

La première étape dans la structuration auctoriale consiste à identifier les contextes à protéger. Il s'agit de protéger les fragments d'un atelier de modifications encore non validées ou à l'inverse de protéger des fragments en cours de rédaction de la visualisation et du référencement d'autres rédacteurs.

Lorsque la portion du graphe à protéger concerne tout un atelier, il convient de configurer un atelier calque de travail.

Lorsque la portion du graphe à protéger concerne une partie des fragments d'un atelier, il convient de configurer un nouvel atelier et un ou plusieurs fragments proxys.

Étape 2, ateliers d'équipes

La seconde étape consiste à identifier les équipes rédigeant dans un même atelier mais ne partageant pas les mêmes modalités d'organisation de l'activité. L'objectif final est de modéliser des fragments pragmatiques au plus proche du contexte métier. Afin de simplifier les modèles, il est possible d'instancier un atelier de travail par équipe.

Lorsqu'une équipe travaille sur l'ensemble des fragments d'un atelier, il convient de configurer un atelier calque de travail.

Lorsqu'une équipe travaille sur une partie des fragments d'un atelier, il convient de configurer un nouvel atelier et un ou plusieurs fragments proxys.

Documentarisation de l'activité

Afin de faire correspondre au mieux la chaîne éditoriale à son contexte d'usage, il est possible de choisir un modèle différent par atelier. Les variations entre modèles peuvent aussi bien porter sur des structures documentaires que sur les modèles d'activité.

Incompatibilités documentaires inter-ateliers

Attention, en introduisant des variations documentaires entre ateliers, il est possible d'introduire également des incompatibilités entre certains fragments et les ateliers dans lesquels ils seront exploités.

Soit un fragment X rédigé dans un atelier A avant d'être envoyé dans un atelier B à l'aide d'un atelier calque ou d'un fragment proxy. Si le modèle documentaire exploité par B ne permet pas l'édition de fragments rigoureusement identiques, le fragment X sera en erreur dans l'atelier B.

Les variations dans le modèle documentaire sont donc à user intelligemment (par exemple, en permettant à un atelier A de verser des fragments dans un atelier B dont le modèle est un sur-ensemble du modèle de A).