Ingénierie hautement productive et collaborative à base de connaissances métier : vers une méthodologie et un méta-modèle de gestion des connaissances en configurations

Julien Badin

N° d'ordre : 167 Année 2011

THESE DE DOCTORAT

Université de Technologie de Belfort-Montbéliard – UTBM

Ecole doctorale Sciences Pour l'Ingénieur et Microtechnique – SPIM

Pour obtenir le grade de

DOCTEUR

SPECIALITE : MECANIQUE

Ingénierie hautement productive et collaborative à base de connaissances métier : vers une méthodologie et un méta-modèle de gestion des connaissances en configurations

Présentée et soutenue publiquement par

Julien BADIN

Le 29 Novembre 2011 devant le jury d'examen :

Président	Prof. Eric BONJOUR	Institut National Polytechnique de Lorraine
Rapporteurs	Prof. Michel TOLLENAERE	Institut National Polytechnique de Grenoble
	Prof. Abdelaziz BOURAS	Université Lumière Lyon 2
Examinateurs	Laurent PETIT	PSA Peugeot Citroën
Directeur et co-encadrant	Prof. Samuel GOMES	Université de Technologie de Belfort-Montbéliard
	MdC. Davy MONTICOLO	Institut National Polytechnique de Lorraine
	MdC. Dominique CHAMORET	Université de Technologie de Belfort-Montbéliard

Préparée au Laboratoire Mécatronique, Modèles, Méthodes et Métiers - M3M-E.A.3318
Rue du Château - 90010 BELFORT Cedex

«Tout ce qui est dans la limite du possible doit être et sera accompli.»

Jules Verne.

Remerciements

Je tenais à remercier tout d'abord les rapporteurs, à savoir Monsieur Abdelaziz BOURAS, Professeur à l'Université Lumière de Lyon 2, et Monsieur Michel TOLLENAERE, Professeur à l'Institut National Polytechnique de Grenoble, pour avoir accepté de rapporter mes travaux de thèse.

Je remercie également le président du jury, Monsieur Eric BONJOUR, Professeur à l'Institut National Polytechnique de Lorraine, ainsi qu'en qualité d'examinateur, Monsieur Laurent PETIT, Ingénieur Calcul de la société PSA Peugeot Citroën, pour m'avoir fait l'honneur de bien vouloir évaluer mes travaux de recherche.

J'adresse toute ma reconnaissance à mon directeur de thèse, Monsieur Samuel GOMES, Professeur des Universités à l'Université de Technologie de Belfort-Montbéliard, pour ses conseils scientifiques, ses encouragements et sa foi indéfectible à mon égard.

Je remercie également mes encadrants de thèse à savoir, Madame Dominique CHAMORET, Maitre de Conférences à l'Université de Technologie de Belfort-Montbéliard, pour son suivi et son intérêt, ainsi que Monsieur Davy MONTICOLO, Maitre de Conférences à l'Institut National Polytechnique de Lorraine pour son regard critique et son encadrement.

Je remercie le laboratoire M3M, et plus particulièrement l'équipe INCIS de l'UTBM, pour m'avoir offert un cadre intellectuel et matériel, adéquat à la réalisation de ces travaux de recherche.

J'exprime toute ma gratitude envers la société DPS, et plus particulièrement, Monsieur Patrick GRIMBERG, Directeur de DPS, ainsi que Monsieur Kevin MAQUIN, Chef d'équipe, pour m'avoir accueilli et m'avoir accordé leur totale confiance tout au long de ces travaux de recherche, notamment dans le cadre du projet ADN.
Une pensée également pour Eddy, Anne, Alister, Sébastien et Abdoulaye, mes collègues de « l'équipe ADN ».

A ce titre, je souhaite remercier tous les acteurs du consortium du projet ADN pour nos riches débats, qui contribuent encore aujourd'hui à faire murir nos idées.

Je souhaite maintenant adresser des remerciements tout particuliers à Madame Emilie BERTOCCHI pour son aide, sa patience et son entrain, Madame Béatrice ROSSEZ pour sa gentillesse, Monsieur Frédéric DEMOLY pour ses conseils avisés et Monsieur Pascal ALDINGER, qui me suit maintenant depuis de nombreuses années.

Une pensée toute particulière à ma femme, Caroline BADIN, pour avoir corrigé mon rapport dans la douleur, pour m'avoir supporté et encouragé durant ces trois années, que ce soit dans les bons moments comme dans les plus difficiles.

Enfin, une dernière pensée, à ma famille et mes amis pour leur soutien, surtout pour ceux, qui aussi, ont la tête dans leurs travaux de recherche et dans leur thèse.

Julien.

Résumé

Ces travaux de recherche concernent le domaine de l'ingénierie des connaissances pour la conception de produits et plus particulièrement les phases amont du couple produit-simulation dans le processus de conception.

Au cours de ces différentes phases amont, les acteurs d'un même projet utilisent simultanément de nombreuses modélisations géométriques et comportementales du produit. Ils peuvent aussi avoir recours à plusieurs outils logiciels hétérogènes, communiquant très difficilement entre eux.
Dans ce contexte, le partage des connaissances entre les différents modèles métiers apparait comme une nécessité. En effet, concevoir un produit implique une gestion des connaissances d'une granularité fine en tenant compte de leur niveau de maturité et de leur cohérence.
Le recours à de nouvelles méthodes et de nouveaux outils est alors nécessaires dans le but de soutenir l'approche globale PLM et continuer à optimiser et rationaliser le processus de conception de produits.

Dans ce cadre, nous proposons une approche qualifiée de KCM – Knowledge Configuration Management, basée sur la gestion des connaissances de granularité fine, en configurations. Cette approche est de nature à favoriser la collaboration entre les acteurs d'un projet, en améliorant la capitalisation, la traçabilité, la réutilisation et la cohérence des connaissances, utilisées simultanément dans plusieurs activités en parallèle du processus de conception.

Les principaux résultats de notre travail de recherche se structurent autour de trois axes :

- Une méthodologie de gestion des connaissances en configurations qualifiée de KCMethod.
- Un méta-modèle, baptisé KCModel, de structuration des concepts manipulés par KCMethod.
- Une maquette de faisabilité sous forme d'outil logiciel ADES, permettant d'expérimenter et valider notre approche.

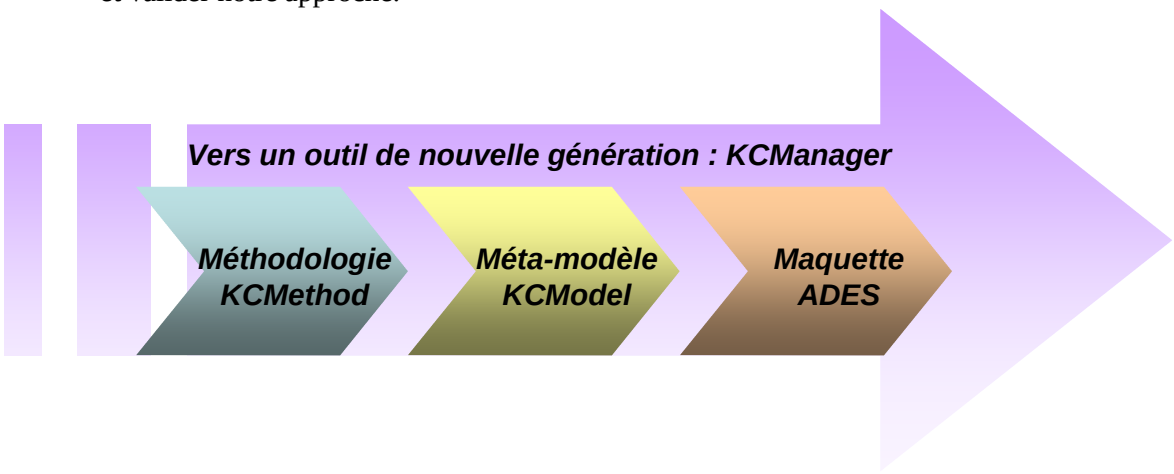


Figure 1: représentation synoptique de la contribution de ces travaux de recherche

L'ensemble des résultats obtenus s'articule autour d'une solution logicielle de nouvelle génération, qualifiée de KCManager, permettant de déployer en entreprise l'ensemble de la démarche proposée Figure 1.

Abstract

This research work deals with the field of knowledge engineering for product design, especially the upstream phases of the design-simulation couple in the product design process.

In these phases, the project participants use many geometric and behavioural models of the product in parallel, while using multiple heterogeneous software tools, with difficulties to communicate between each other. However, especially in the upstream phases of the design process, designing a product requires sharing fine granularity knowledge between the different expert models, taking into account their levels of maturity and consistency. Consequently, new methods and tools are then needed in order to support the overall PLM approach and continue to optimize and streamline the product design process.

Thus, in this research context, we propose an approach referred to as KCM – Knowledge Configuration Management, based on management of fine granularity knowledge in configuration. This approach is likely to improve collaboration between project participants, improving capitalization, traceability, reuse and consistency of the knowledge used simultaneously on several activities in parallel within the design process.

The main results of our research are structured around three axes :

- A methodology for knowledge configuration management qualified as KCMethod.
- A meta-model, called KCModel, structuring concepts manipulated by KCMethod.
- A model of feasibility, namely ADES software tool, allowing testing and validation our approach.

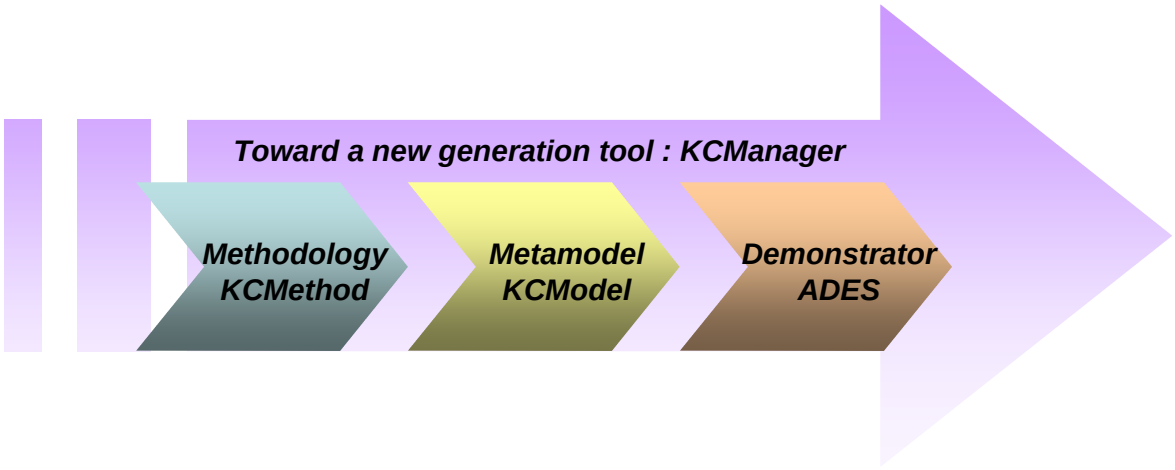


Figure 2: synoptic representation of this research work

The overall results are articulated around a next-generation software solution, described as KCManager, in order to deploy the approach proposed in Figure 2 across the company.

Table des matières

Liste des figures

Liste des tableaux

Liste d'acronymes

ADES : Alliance des Données Elémentaires de Simulation.
ADN : Alliance des Données Numériques.
API : Application Programming Interface.
CAE : Computer Aided Engineering.
CAO/CAD : Conception Assistée par Ordinateur / Computer Aided Design.
CDV : Chargé de projet Fonction Système – Cinématique, Dynamique et Vibratoire.
CC attelage : Chargé des composants de l'attelage mobile.
CC carter : chargé des composants de l'attelage mobile.
CCPC : chargé de projet Fonction Système – Combustion et Conversion Pression Couple.
CFD : Computed Fluid Dynamic, mécanique des fluids.
CoDeKF : Collaborative Design and Knowledge Factory.
CP : Chef ou chargé de projet.
CPM : Core Product Model.
CTRC : Capitalisation, Traçabilité, Réutilisation, Cohérence.
DAIM : Design Analysis integration Model.
DFX : Design For X.
FBS : Function Behavior Structure.
FEM : Finite Element Modelling.
FUI : Fond Unique Interministériel.
GC : Gestion de Configuration.
GMP : Groupe Moto-Propulseur.
HMI : Human Machine Interface
IA : Intelligence Artificielle.
ICE : Information Core Entity.
IDM : Ingénierie Dirigée par les Modèles.
IGES : Initial Graphics Exchange Specification.
KBE : Knowledge Based Engineering, Ingénierie à base de connaissances.
KC : KnowledgeConfiguration.
KCM : Knowledge Configuration Management, Gestion des configurations de connaissances.
KCManager : Knowledge Configuration Manager.
KCMethod : Knowledge Configuration Method.
KCModel : Knowledge Configuration Model.
KD : KnowledgeDomain.
KMS ou KLM : Knowledge Management System ou Knowledge Lifecycle Management.
MED : Mise En Donnée
MDA : Model Driven Architecture.
MDE : Model Driven Engineering.
MD-MV : Multi-Domaines Multi-vues.
MV : Multi-Vues.
NTIC : Nouvelles Technologies de l'Information et de Communication.
OMG : Object Management Group.
PDM : Product Data Management.
PLM : Product Lifecycle Management.
PPO : Produit Process Organisation
SBD : Simulation Based Design.
SDM : Simulation Data Management.
SEED : Simulation Environment for Engineering Design.
SGDT : Système de Gestion des Données Techniques.
STEP : Standard Template for Electronic Publishing.
Théorie C-K : Théorie Concept-Knowledge.
UML : Unified Modeling Language.
VDA-FS : Verban der Automobilindustrie – Flächenschnittstelle.
WIP : Work In Progress.

Introduction générale

A. Origines et motivations des travaux de thèse

Dans l'industrie manufacturière, et plus particulièrement dans l'industrie automobile et aéronautique, les processus de conception des systèmes mécaniques ont évolué d'une ingénierie séquentielle vers une ingénierie concourante afin d'améliorer la qualité des produits, mais aussi de réduire les coûts et les délais de développement. Dans le but de limiter le nombre de maquettes et de prototypes physiques, le recours à des méthodes de modélisation et de simulation numérique complexes (calculs par éléments finis, simulation 0D-1D, intégration CAO/Calcul, etc.) se généralise, intégrant et couplant des domaines d'expertises très diversifiés en relation avec les différents phénomènes, aujourd'hui qualifiés de multi-physiques, auxquels sont soumis les produits (phénomènes mécaniques, thermiques, acoustiques, etc.).
Cependant, il apparaît au sein des bureaux d'études et de calcul, de nombreux problèmes de non-qualité ou de faible productivité en conception-simulation. Ces difficultés peuvent être liées à des manques ou des absences de synchronisation des configurations de paramètres, de règles métier et voire même des jeux de valeurs de ces paramètres (instances de paramètres) ; paramètres en général communs à plusieurs métiers. Malgré l'existence sur le marché et dans les entreprises d'outils et de formats neutres d'échange dédiés à la gestion des données et des modèles du produit (outil Product Data Management - PDM, outil Simulation Data Management - SDM, format d'échange IGES, format d'échange STEP, etc.), ces problèmes de non-qualité et de faible productivité en conception et en simulation subsistent. De plus, ils s'accroissent, dans la mesure où le recours à des modeleurs CAO et des outils de calculs paramétrés se généralisent, sans qu'aucune structure de communication au niveau des paramètres directeurs de conception et de simulation (et des règles métier qui les sous-tendent) n'ait été prévue. Plusieurs facteurs, tels que la durée inégale et le nombre d'itérations entre les différentes étapes de conception et de simulation, aggravent la situation et font que, très souvent, des concepteurs travaillent sur des modèles dont les paramètres fonctionnels (efforts mécaniques, sollicitations thermiques, etc.), géométriques (longueur, largeur, volume, etc.), physiques (module de Young du matériau, coefficient de Poisson, efforts, etc.) ne sont plus à jour, voire ne sont plus synchronisés aux différentes étapes du processus de conception.

Il s'agit, dans le cadre de ce travail de recherche, de développer des méthodes, des modèles et des outils d'ingénierie productive, à base de connaissances métier, autorisant l'interconnexion des différents paramètres de conception-simulation et l'intégration des contraintes multi-métiers nécessaires à la conception d'un système mécatronique complexe. La finalité est d'améliorer la qualité et la productivité en phase amont d'ingénierie du couple Produit-Simulation, non seulement en entrant dans un niveau de granularité plus fin, comparé aux traditionnels travaux autour des échanges de données et de modèles géométriques, mais surtout en faisant intervenir les problématiques de calcul, intégrant les connaissances métier, très en amont dans le processus de conception.

Pour ce faire, nous proposons une approche basée sur la gestion en configurations des connaissances cruciales pour la conception et la simulation numérique de produits, voire de pièces mécaniques. Cette approche permet la capitalisation, la traçabilité, la réutilisation et le maintien en cohérence des données, informations et connaissances utilisées et partagées dans de nombreux modèles métiers, ainsi qu'une meilleure collaboration des acteurs du projet. Cette approche est présentée sous forme d'un modèle de connaissances produit générique, qui est basé sur la notion de configurations de connaissances. Il servira de socle à la définition d'un nouvel outil de gestion de configurations de connaissances pour la conception et la simulation numérique dans le cadre d'un projet de recherche partenariale national.

B. Contexte des travaux de recherche

B.A. Le chercheur en entreprise

Nos travaux de recherche sont réalisés au sein de l'entreprise DPS : Digital Product Simulation, dans le cadre d'une thèse en convention CIFRE, en collaboration avec le laboratoire M3M (Mécatronique, Méthodes, Modèles et Métiers) de l'UTBM (Université de Technologie de Belfort-Montbéliard), qui sera présenté dans un prochain paragraphe.
L'entreprise DPS est une PME d'environ cent personnes, fondée en 1997, avec pour objectif de favoriser l'utilisation de la simulation numérique tout au long du processus de conception, et particulièrement au travers des approches d'intégration CAO/Calcul. L'activité de DPS se focalise autour de l'approche "Digital Product Simulation Driven Early Design Optimization", c'est-à-dire l'utilisation de la simulation numérique au plus tôt pour piloter la conception. Le but recherché par DPS est d'accroitre l'efficacité des activités d'ingénierie en rationalisant et en optimisant le processus de conception.
En effet, coupler finement et très tôt la CAO et les modèles de simulation en les rendant dépendants et associatifs permet d'optimiser les choix de conception, de tester rapidement et efficacement plusieurs configurations du produit, notamment lors de processus créatifs de définition d'architectures. Pour ce faire, l'entreprise DPS propose des méthodologies et des solutions essentiellement basées sur des modélisations paramétriques et associatives du produit intégrant les contraintes de conception, de simulation et d'optimisation. Ces solutions offrent l'avantage de pouvoir lancer rapidement un grand nombre de calculs itératifs.

La Figure 3 ci-dessous illustre différentes étapes de modélisation pratiquées chez DPS en s'appuyant sur des exemples issus du Groupe Motopropulseur automobile. Ces étapes peuvent se résumer de la manière suivante :

Etape 1. Modélisation CAO paramétrée d'assemblages 3D complexes.
Etape 2. Association d'un modèle d'éléments finis à tous les composants de l'assemblage
Etape 3. Etablissement de toutes les interconnexions d'analyses entre les composants.
Etape 4. Génération d'un fichier universel contenant la mise en données calcul complète.
Etape 5. Lancement automatisé de calculs pour tester de nouvelles hypothèses d'architecture

Lors de ces étapes, des modèles d'intégration CAO/Calcul, couplant la modélisation géométrique et la modélisation physique, sont utilisés lors de processus itératifs où chaque modification de la géométrie est rapidement propagée et testée par la simulation afin d'identifier le meilleur concept.

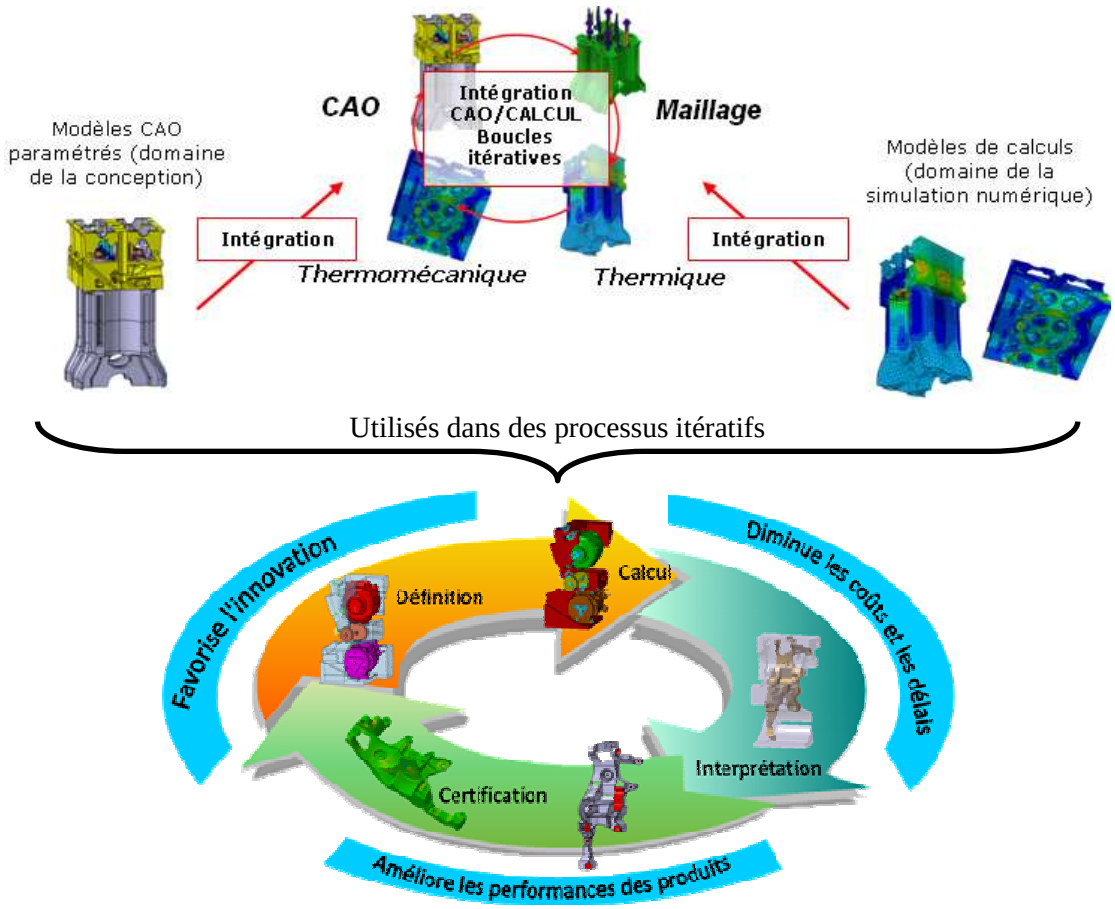


Figure 3 : exemple de modélisation paramétrées CAO/Calcul intégrées [DPS, 2010 b]

Face aux progrès importants des outils matériels et logiciels, dans le domaine de la modélisation et de la simulation numérique, leur utilisation à un niveau pilotant, coïncide avec les besoins et les impératifs industriels actuels et contribue aujourd'hui à accroitre le nombre et la complexité des modélisations multi-physiques, réalisées en entreprise, tout en tenant compte de l'influence de l'environnement des composants étudiés. C'est ainsi, qu'au cours de ces dernières années, l'entreprise DPS a pu constater l'émergence d'une nouvelle problématique chez ses clients industriels qui aujourd'hui utilisent massivement la simulation numérique, particulièrement dans les phases très en amont du processus de conception. En effet, le nombre croissant de modèles et leur complexité engendre des difficultés lors du partage et du maintien en cohérence des données techniques utilisées dans les différents modèles métiers.
C'est ainsi qu'a émergé cette problématique industrielle autour de la gestion conjointe des données de conception et de simulation. Cette problématique concerne des secteurs divers et variés tels que, par exemple : l'industrie manufacturière, l'industrie automobile, ou encore l'industrie aéronautique. Réactifs sur les nouveaux outils et les nouvelles méthodologies de conception et de simulation, ces secteurs d'activité sont touchés de plein fouet par un volume croissant de données techniques à manipuler et à partager. C'est dans ce contexte que DPS a mis en place avec Airbus et PSA Peugeot Citroën, des solutions de formalisation, de capitalisation et de réutilisation des données techniques du produit au travers de plusieurs projets spécifiques, réalisés en interne chez ces grands groupes. Souhaitant aller plus loin dans cette démarche, et grâce à un partenariat fort réalisé entre des grands groupes industriels et des laboratoires universitaires, l'entreprise DPS a coordonné le montage du projet ADN : Alliance des Données Numériques.

B.B. Cadre du projet ADN : Alliance des Données Numériques

Compte-tenu des évolutions des pratiques dans le domaine de la modélisation et de la simulation numérique, le nombre de modèles métiers devant échanger des données de type paramètres, relations mathématiques, règles métiers dans un contexte d'entreprise étendue, devient de plus en plus important. Dans la mesure où ces modèles métiers sont manipulés par des acteurs issus de divers domaines d'expertises, utilisant des outils différents et souvent incapables de communiquer entre eux, le maintien en cohérence des données techniques utilisées dans un contexte d'ingénierie simultanée n'est aujourd'hui pas assuré.
Dans ce contexte, l'entreprise DPS a choisi de monter un consortium regroupant industriels et universitaires pour réaliser le projet ADN (Alliance des Données Numériques) [Systematic, 2010] qui vise à développer des méthodologies et surtout une solution logicielle de nouvelle génération pour la collaboration et le partage des données techniques du couple produit-simulation.
Lancé en septembre 2010, le projet ADN est un projet national financé dans le cadre du neuvième appel à projet du FUI (Fond Unique Inter-ministériel), et a été labellisé par trois pôles de compétitivité :

- Systém@tic en Région Parisienne
- Véhicule du Futur en Région Est
- i-Trans en Picardie

Le projet est prévu pour une période de trente-six mois et implique onze partenaires répartis en trois catégories :

- **Les industriels :** PSA Peugeot Citroën, EADS et FAURECIA
- **Les Universitaires et laboratoires :** Arts & Métiers Paris Tech, UTBM (Université de Technologie de Belfort-Montbéliard) et UTC (Université de Technologie de Compiègne)
- **Les PMI/PME :** DPS (Porteur industriel), Samtech, Soyatec, Eiris Conseil et Deltacad

L'objectif recherché, au terme du projet ADN, est de concevoir et de commercialiser un produit destiné à l'industrie manufacturière répondant à une problématique déterminante : favoriser l'intégration de la simulation numérique dans le processus de conception pour en améliorer la qualité et la productivité. En effet, la gestion de l'ensemble des paramètres devant être échangés entre les différentes applications lors du processus de conception, et la gestion des interfaces entre les applications est un problème récurrent. Il s'agit donc de développer une méthodologie et une solution logicielle générique et intelligente permettant la gestion centralisée de configurations d'informations et de connaissances métier. La solution logicielle du projet ADN permettra la coordination et la traçabilité des échanges de paramètres, de règles expertes et des instances de paramètres entre les différents systèmes inhérents à la chaîne d'ingénierie numérique. L'aboutissement de ce projet est de nature à donner un avantage concurrentiel significatif aux équipes projets impliquées dans de tels processus de conception.

B.C. Le contexte universitaire des travaux de recherche

Nos travaux de recherche se sont aussi déroulés au sein de l'équipe Ingénierie Numérique avancée pour la Conception Intégrée de Systèmes mécaniques (INCIS), du laboratoire Mécatronique, Méthodes, Modèles et Métiers (M3M, E.A. 3318) de l'Université de Technologie de Belfort-Montbéliard (UTBM). Cette équipe travaille essentiellement sur les domaines de la conception hautement productive de produits et de systèmes mécaniques, et de gestion du cycle de vie des produits, intégrant des problématiques d'ingénierie à base de connaissances [Gomes, 2008] [Monticolo, 2008] [Toussaint et al., 2010] [Demoly, 2010]. Les thématiques scientifiques abordées par l'équipe INCIS concernent, en particulier, l'amélioration de la qualité et la réduction des délais d'ingénierie routinière, en s'appuyant sur la gestion collaborative des données, informations et des connaissances en interaction avec des méthodologies de CAO avancées, c'est-à-dire des méthodologies de Conception Assistée par Ordinateur paramétrée et pilotée par des règles métier. Ces règles sont issues de différents domaines métiers d'ingénierie impliqués dans le cycle de vie du produit.

C. Les objectifs du travail de recherche

Dans ce contexte, et face aux problèmes de non productivité due à une mauvaise collaboration et un manque de cohérence des données, informations et connaissances lors du processus de conception, notre objectif de recherche peut être résumé de la manière suivante :

Proposer une méthodologie et un méta-modèle pour la gestion des données, informations et connaissances pour les activités de conception et de simulation numérique, mis en œuvre plus spécifiquement, lors des phases amont du processus d'ingénierie, afin de favoriser la collaboration entre les métiers "Conception" et "Calcul".

Afin d'atteindre cet objectif, nous proposons une démarche de recherche qui comprend quatre étapes et sous-objectifs successifs.
La première étape concerne l'analyse des pratiques industrielles lors des phases de conception de produits, et plus particulièrement l'évolution de l'utilisation des approches de la simulation numérique, afin de mettre en évidence son importance, mais aussi l'impact sur le processus global d'ingénierie.
La deuxième étape consiste à analyser comment les outils actuels de gestion de données sont utilisés dans les processus de conception et de simulation numérique, et mettre en exergue l'intérêt d'une gestion plus fine des données, mais aussi des connaissances issues des processus d'ingénierie afin de favoriser la collaboration entre les différents acteurs d'un projet.
Dans la troisième étape, nous proposons une méthodologie générique baptisée KCMethod-Knowledge Configuration Method permettant la centralisation et la capitalisation des paramètres et règles métiers, afin de créer des configurations de connaissances.

Une configuration de connaissances regroupe un ensemble de connaissances (explicites) structurées et cohérentes pour un objectif donné en relation avec une ou plusieurs activités du processus de conception. Une configuration de connaissances utilise les mécanismes de gestion de configuration, c'est-à-dire, la gestion de la cohérence, la gestion des versions, la traçabilité, la gestion de la maturité, etc.

On s'intéresse ici aux connaissances explicites, c'est-à-dire celles qu'il est possible de formaliser à l'inverse des connaissances tacites. Ces différents types de connaissances sont détaillés dans le chapitre 2. Ainsi, l'approche KCMethod est dédiée à la gestion de ces connaissances issues des activités de conception et de simulation numérique. Elle est ensuite formalisée sous la forme d'un méta-modèle baptisé KCModel - Knowledge Configuration Model.
La dernière étape nous permet de tester la méthodologie et le méta-modèle au travers de plusieurs cas d'études, notamment issus du domaine automobile. Ces expérimentations ont mise en évidence l'intérêt de la gestion des connaissances en configurations pour le suivi, l'analyse et la réutilisation des connaissances, mais aussi pour la collaboration entre les différents acteurs d'un projet.

C.A. Comprendre les approches de conception de produits et de simulation numérique

Le processus de conception de produits, dans sa globalité, et en particulier un de ses domaines clefs que constitue la simulation numérique, sont des domaines absolument prioritaires dans le cadre du développement d'un produit dans l'industrie manufacturière [Devalan, 2009]. L'évolution des pratiques, qu'elles soient organisationnelles ou technologiques, ont autorisé le déploiement de nouvelles approches, de méthodologies et d'outils permettant d'optimiser et de rationaliser le processus de conception, et par conséquent d'améliorer significativement le rendement des activités d'ingénierie en milieu industriel. Afin de comprendre les tenants et aboutissants de ces approches, mais également leurs évolutions et leurs conséquences sur les processus d'ingénierie, nous nous sommes basés, à la fois sur des observations industrielles et sur une approche organisationnelle telle qu'elle a été définie par [Galbraith, 1977], et plus récemment par [Tollenaere et al., 1998]. En effet, une organisation peut être définie comme un ensemble d'entités

réparties et devant travailler en collaboration, afin d'atteindre un but commun tout en se répartissant les tâches, et en mettant en place, en continu, des processus de décision. L'approche organisationnelle met en évidence les interactions entre les acteurs métier ainsi que les connaissances qu'ils vont utiliser pour atteindre leurs objectifs. En effet, chaque acteur possède son autonomie à travers ses propres savoirs et savoir-faire, mais interagit avec les autres acteurs pour contribuer à la réalisation des activités d'ingénierie [Monticolo, 2008]. Le résultat de ce positionnement nous permet de dresser une analyse de l'évolution des activités de conception, et en particulier de la place de la simulation numérique, mais également de dégager une problématique en termes de flux d'informations circulant entre les acteurs métiers et de modèles qu'ils manipulent au travers d'un grand nombre d'outils logiciels.

C.B. Comprendre les mécanismes de gestion des données et des connaissances en conception et simulation numérique

Le développement des NTIC (Nouvelles Technologies de l'Information et de Communication), des méthodes et des outils d'ingénierie collaborative ont favorisé l'accès aux solutions de gestion de données du produit [Large et al., 2009]. Répandues dans le tissu industriel, confrontées de plus en plus à un contexte d'entreprise étendue, elles offrent aujourd'hui un niveau d'interopérabilité et de collaboration désormais en adéquation avec l'augmentation du volume d'information à traiter. Les besoins de structuration, d'organisation et plus récemment de capitalisation sont de plus en plus présents, notamment concernant les informations techniques, mais aussi les connaissances et les savoir-faire métier des entreprises.
Compte-tenu des nouvelles approches émergeantes en conception de produits et en simulation numérique, il nous semble aujourd'hui nécessaire d'étudier comment ces solutions de gestion de données ou de connaissances métier s'adaptent aux problématiques industrielles, c'est-à-dire dans quelle mesure elles répondent ou non au besoin de l'industrie. Cette analyse permet d'identifier des besoins non-remplis en ce qui concerne les plateformes de communication et de gestion des paramètres et des règles de conception encapsulés dans les modèles métiers, besoins déjà constatés par Ducellier [Ducellier, 2008]. L'objectif recherché est donc de proposer une approche répondant à cette problématique, en particulier lors des processus d'ingénierie situés très en amont dans le cycle de développement d'un produit, et plus spécifiquement lorsque l'utilisation de la simulation numérique entraîne l'apparition de nombreux modèles métiers hétérogènes, impliqués dans des boucles itératives, à durée inégale.

C.C. Définir une nouvelle approche et un nouveau modèle de gestion des connaissances en configurations.

Nos travaux de recherche sont découpés en deux sous-objectifs. Le premier consiste en la définition d'une méthodologie, et le second à sa formalisation au travers d'un méta-modèle qui servira de base pour un futur outil développé dans le cadre du projet ADN.
En effet, en se basant sur l'observation des pratiques de conception et de simulation, ainsi que sur le constat de l'utilisation des systèmes de gestion de données et de connaissances, il s'agit de proposer une méthodologie générique répondant aux attentes des utilisateurs en termes de structuration et de gestion des données, informations, mais surtout des connaissances encapsulées dans les modèles métiers et devant être échangées. En effet, le nombre de modèles métiers utilisés, en parallèle, dans de nombreux domaines d'expertise, faisant intervenir plusieurs outils, engendre des incohérences lors du partage et de la réutilisation des règles et des paramètres directeurs du produit développé. Nous proposons donc une méthodologie favorisant la collaboration autour de ces différents modèles métiers, et ceci dans un contexte d'entreprise étendue. La méthodologie proposée se base sur une organisation "en configurations" des connaissances encapsulées dans les modèles métiers et la mise en relation de ces configurations, afin d'identifier les conflits qui peuvent subvenir.
La gestion de configuration (GC) [ISO 10007: 1995] [ISO 10007: 2003] est un domaine de plus en plus utilisé dans les processus d'ingénierie. Initialement dédiée au développement logiciel [Adeli, 2001 a] [Adeli, 2001 b], la gestion de configurations est désormais aujourd'hui utilisée dans le

cadre de problématiques de diversité des produits dans l'industrie [Agard, 2002], en ingénierie système [Aidi, 2007], et aussi dans les Systèmes de Gestion de Données Techniques (SGDT) ou outil de Product Data Management (PDM) [CIMDATA, 2001]. La gestion de configuration permet de connaître à tout moment l'état de l'ensemble des composants du projet en contrôlant leurs évolutions durant la totalité du cycle de vie. Elle permet d'archiver des "états stables" successifs des produits et de vérifier la complétude et la cohérence de ces différents "états". La GC permet également de maîtriser les changements (traçabilité) et de disposer de jalons permettant des restaurations des états antérieurs du système. On peut caractériser l'objectif de la GC comme étant la définition de l'organisation, des méthodes, des moyens instaurés pour assurer et suivre la configuration des produits et être en mesure de livrer un système dont les caractéristiques sont identifiées et les évolutions maîtrisées.
Ainsi, la gestion de configurations fournit un premier cadre pour l'établissement de processus et de procédures permettant de structurer les projets liés au développement de produits complexes. Si, dans un premier temps, cette structuration permet aux acteurs de manipuler les informations relatives au produit de manière cohérente avec les objectifs du projet, elle fournit également un "squelette" du produit sur lequel peuvent être mis en œuvre des mécanismes de gestion des "modifications" [Riviere, 2004].
L'approche que nous proposons, baptisée KCMethod - Knowledge Configuration Method, permet de créer, non pas des configurations de modèles géométriques de produits, mais plutôt des configurations de connaissances centralisées et synchronisées avec les modèles métiers. Notre approche se situe, par conséquent, à un niveau plus fin de granularité, comparé aux approches traditionnelles de gestion de configuration. Elle permet le suivi des modifications et de l'activité dans les modèles, mais offre aussi des possibilités de capitalisation, de gestion des versions, de maintien en cohérence et de réutilisation dans le cadre du développement d'un produit, ou même lors de projets ultérieurs.
Cette méthodologie est formalisée au travers d'un méta-modèle UML[1] générique baptisé KCModel - Knowldge Configuration Model. Ce méta-modèle est complémentaire aux modèles produit traditionnellement utilisés pour soutenir l'approche globale PLM - Product Lifecycle Management, stratégie d'entreprise s'inscrivant dans une évolution des systèmes PDM et de leur utilisation dans l'entreprise [Eynard, 2005], dont l'objectif consiste à prendre en compte l'ensemble des informations du produit tout au long de son cycle de vie. Le méta-modèle KCModel permet de capitaliser les données de type paramètres, règles et contraintes en entités d'informations techniques, puis d'instancier ces entités dans des configurations contextualisées et synchronisées avec un ensemble d'applications logicielles et de modèles métiers.

C.D. Gérer les configurations de connaissances sur un projet

Suite à la description de la méthodologie KCMethod et du méta-modèle KCModel, nous avons pu conduire des expérimentations spécifiques au travers de deux cas d'études dont le premier concerne la conception d'un support de fixation. Il permet d'illustrer les principaux concepts de notre contribution au travers d'un démonstrateur que nous avons spécifié et réalisé, baptisé ADESv1 : Alliance des Données Elémentaires de Simulation.
Le second cas d'étude est appliqué au domaine de l'automobile et a été réalisé avec la participation du constructeur automobile PSA Peugeot Citroën. En effet, depuis le début de ces travaux de recherche, PSA a été un interlocuteur privilégié. La méthodologie et le méta-modèle ont été spécifiés puis réalisés en étroite collaboration avec les ingénieurs du bureau d'études et de calcul de ce grand groupe industriel, en effectuant plusieurs itérations sur les différents besoins identifiés, sur les propositions formulées et sur les résultats obtenus lors de nombreux tests. Ce second cas d'étude, réalisé avec une version améliorée d'ADES, baptisée ADESv2, met l'accent sur la

[1] UML : Unified Modeling Language est un langage de modélisation orienté objet défini par l'OMG : Object Management Group (www.omg.org).

collaboration des acteurs autour des modèles métiers, dans l'objectif de valider la pertinence de notre démarche.

A la suite des deux expérimentations réalisées et de la discussion des résultats obtenus, nous tirons les conclusions afin de déterminer si notre proposition contribue à répondre à la problématique initiale, puis nous proposons des perspectives de recherche visant à enrichir les travaux entamés et nous ouvrir à d'autres domaines d'activité.

D. Plan du manuscrit de thèse

Suite à cette introduction générale, le manuscrit de thèse est organisé en trois parties, elles-mêmes constituées de chapitres (Figure 4). Dans la première partie, intitulée "Problématique et état de l'art", nous introduisons les travaux de recherche ainsi que le contexte dans lequel ils ont été menés. Ensuite, nous identifions la problématique à partir d'un état de la littérature dans le domaine. Cette première partie du manuscrit regroupe le chapitre 1 et le chapitre 2 de la thèse :

- Chapitre 1 Contexte de recherche : Processus collaboratif de conception de produit et simulation numérique

Dans le premier chapitre nous présentons le domaine des projets de conception mécanique et nous abordons notamment la question de la place de la simulation numérique dans le processus global de conception de produits. Ce chapitre nous permet de présenter les pratiques actuelles de conception-simulation et de mettre en évidence les problèmes et les besoins rencontrés par les industriels définissant le cadre de ces travaux de recherche.

- Chapitre 2 La gestion des données et des connaissances pour la conception et la simulation du produit

Le second chapitre aborde les domaines de la gestion des données du produit, de la gestion des connaissances ainsi que les approches de gestion en configuration dans le cadre du processus de conception de produits. Nous proposons une analyse des besoins couverts et des manques des méthodologies et outils traditionnels, afin de dégager notre problématique de recherche.

~~~~~~

La seconde partie du manuscrit, qui correspond à notre "Contribution de thèse", présente nos principaux apports du point de vue des concepts fondamentaux proposés, du méta-modèle de connaissances produit formalisé, ainsi que l'expérimentation de notre approche. Cette partie est constituée des chapitres 3,4 et 5 résumés ci-dessous :

- Chapitre 3 La méthodologie KCMethod

Dans le troisième chapitre, nous présentons la méthodologie KCMethod de gestion des données, informations et des connaissances en configurations pour la conception-simulation de produit. Les principaux concepts sont expliqués et illustrés.

- Chapitre 4 Le méta-modèle KCModel

Le chapitre 4 illustre la formalisation de KCMethod sous forme d'un méta-modèle appelé KCModel. Nous décrivons les différentes classes de ce modèle et nous illustrons son fonctionnement.

- Chapitre 5 Gestion des configurations de connaissances : application au cas de la conception de structures et de l'automobile

Le chapitre cinq présente le démonstrateur ADES (Alliance des Données Elémentaires de Simulation) réalisé à partir du KCModel, et deux expérimentations sur un cas support de fixation et un cas automobile.

~~~~~~

Enfin la dernière partie intitulée "Conclusion et perspectives", conclut sur ces travaux et propose des perspectives de recherche. Elle est composée du chapitre 6.

- Chapitre 6 Conclusion et perspectives

Le sixième et dernier chapitre rappelle la contribution par rapport à la problématique de partage et de gestion des connaissances. Il soulève une discussion sur les perspectives de recherche qu'il conviendrait de réaliser.

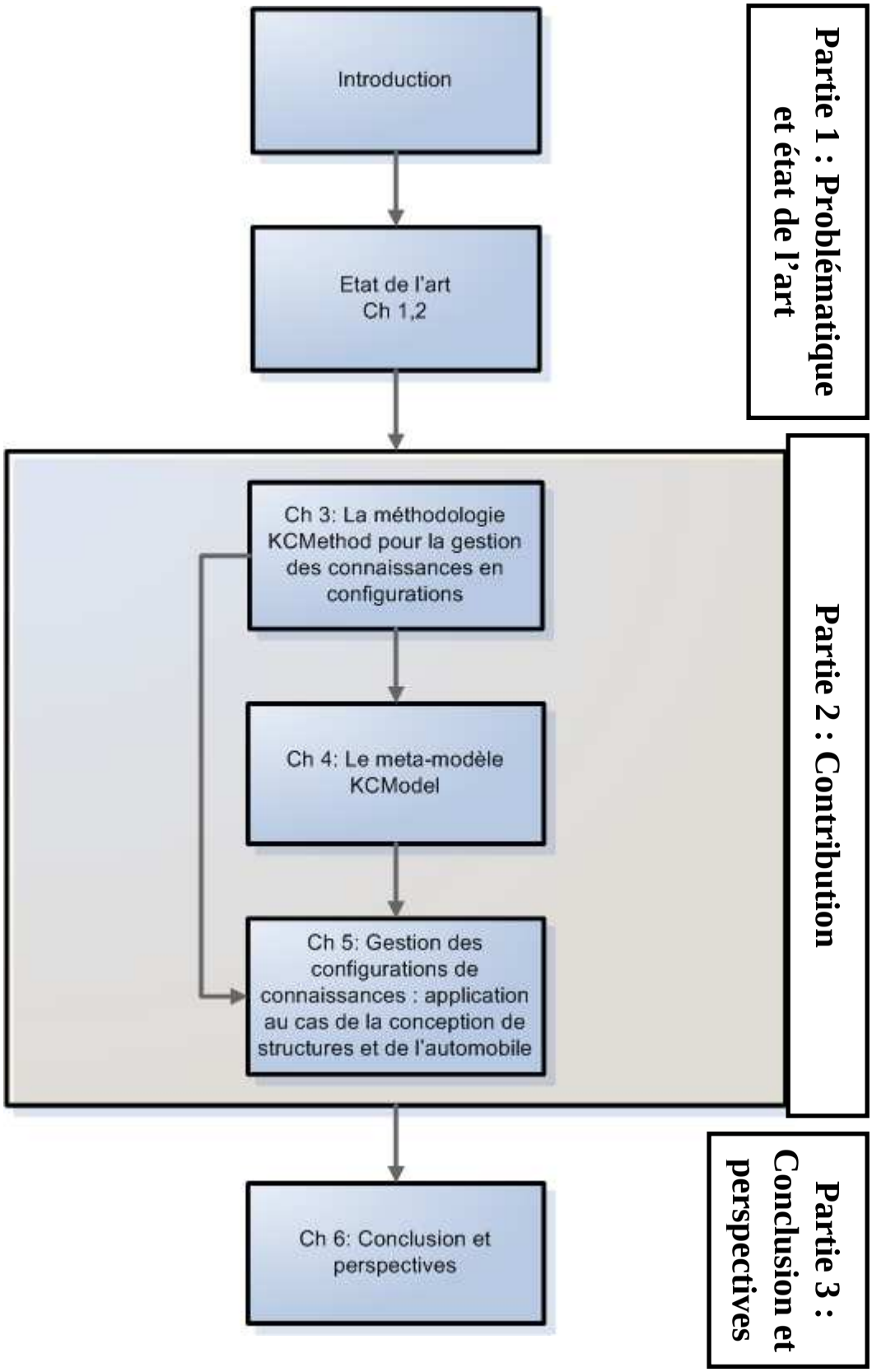


Figure 4 : représentation synoptique du manuscrit de thèse

Chapitre 1. Contexte de recherche : Processus collaboratif de conception de produit et simulation numérique

Dans le cadre de ce premier chapitre, nous étudions les conditions actuelles de réalisation des projets de conception de produits ainsi que les pratiques industrielles dans le domaine de la conception et de la simulation. L'objectif est de mettre en évidence la complexité et l'hétérogénéité des activités, des acteurs, des modèles utilisés, afin de positionner ce travail de recherche sur la gestion des paramètres et des contraintes lors d'un projet de conception.
En effet, l'utilisation simultanée de nombreux modèles métiers de conception et de simulation, et ceci dans plusieurs domaines d'expertises, cause de nombreuses difficultés de partage et de collaboration entre les différents acteurs d'un projet.

1.1 Le processus d'ingénierie en conception de produits

Dans un contexte économique difficile, mondialisé et extrêmement concurrentiel, touchant l'ensemble de l'industrie manufacturière, et plus particulièrement le monde de l'automobile, nous avons assisté, au cours de ces vingt dernières années, à la mise en place d'outils et de méthodologies visant à optimiser le processus de développement des produits en agissant essentiellement sur la réduction du coût et des délais, tout en améliorant la qualité (il est alors souvent question d'utiliser le terme "triptyque coût/qualité/délais") [Miles et al., 1992] [Yeh, 1992] [Burman, 1992]. Cette optimisation passe, en partie, par une rationalisation des processus d'ingénierie et plus particulièrement du processus de conception, en tenant compte des contraintes inhérentes à l'ensemble du cycle de vie du produit [Merlo, 2003] [Merlo et al., 2004]. En effet, le processus d'ingénierie regroupe un ensemble d'activités interconnectées entre elles et impliquant de nombreux acteurs dans des domaines d'expertises différents mais dépendants les uns des autres : la conception, la fabrication, les essais, l'industrialisation, etc.

Tout au long du cycle d'élaboration d'un produit, l'activité de conception, située en amont du cycle de vie, est donc considérée comme un ensemble de tâches interdépendantes (tâches de modélisation fonctionnelle, tâches de recherche de solutions, tâches de modélisation géométrique, tâches de calcul et de dimensionnement, tâches d'optimisation tenant compte des contraintes de fabrication et d'assemblage, etc.) générant des spécifications, des contraintes, et un ensemble de solutions candidates plus ou moins pertinentes [Louhichi et al. 2005]. Le processus de conception de produit est donc une activité spécifique du processus d'ingénierie qui, au même titre que le processus de fabrication par exemple, est absolument essentielle dans le cycle de vie du produit [AFNOR, 1994]. Ainsi, dans la suite de ce manuscrit, nous parlerons indissociablement du processus d'ingénierie et du processus de conception de produit car c'est cette activité qui nous intéresse plus particulièrement dans ces travaux de recherche.

En effet, dans un contexte de réduction des temps de conception, et en particulier par la mise en parallèle de certaines activités du processus d'ingénierie, les pratiques des industriels ont évolué d'une ingénierie dite séquentielle, car découpée, ou en phases qui se déroulent de manière successives dans le temps [Pahl et al., 1996], et correspondant à une vision Taylorienne de l'activité de conception, vers une ingénierie qualifiée de concourante [Sohlenius, 1992], de simultanée ou d'intégrée [Tichkiewitch et al., 1993] [Brissaud et Garro, 1996]. Ces nouvelles visions de la démarche d'ingénierie, enrichies par les méthodes et outils dans le domaine du travail collaboratif, visent à accroitre la réactivité de l'entreprise pour réduire les coûts et les délais, tout en améliorant la qualité des produits conçus. Elles se concrétisent par une mise en parallèle des activités de conception et le développement d'un contexte collaboratif de partage des données entre les ressources et les acteurs de l'entreprise [Jagou, 1993], mettant en œuvre, en particulier, une organisation par projet. En effet, l'approche concourante ou simultanée de la conception consiste à croiser les projets de l'entreprise, et les métiers disposant de compétences et connaissances spécifiques. Les projets portent sur un produit et/ou sur un process spécifique, devant répondre à un besoin client, tandis que les métiers traduisent les connaissances et savoir-faire requis pour le bon déroulement et la réussite du projet.

De ces approches sont nées de nouvelles méthodologies et de nouveaux outils permettant aux concepteurs de réduire les cycles d'ingénierie qualifiés de routiniers, mais aussi d'intervenir de plus en plus tôt dans le processus de conception. Il s'agit par exemple des méthodologies et d'outils tels que :

- le Design For X (DFX : Design For Manufacturing, Design For Assembly, Design For Cost, Design For Environment, etc.),
- le développement massif des modèles paramétriques, grâce aux performances des outils de CAO paramétriques (Conception Assistée par Ordinateur),
- une meilleure gestion des données et informations techniques en ayant recours aux outils de Gestion de Données Techniques,
- ou encore la conception à base de connaissances s'appuyant sur des banques de connaissances partagées ou encore sur des solveurs dédiés aux raisonnements sur des paramètres et des règles d'ingénierie.

En effet, il est admis qu'aujourd'hui les gains les plus importants sont à réaliser sur les phases amont du processus de conception en agissant le plus tôt possible [Clautrier 1991] [Tichkiewitch et al., 1993].
Dans la suite de cette partie, nous détaillons ces différents points, afin d'aborder la place et l'évolution de l'utilisation de la simulation numérique, dans le contexte industriel que nous avons décrit précédemment.

1.1.1 Organisation du projet de conception

L'organisation d'un projet de conception fait appel à la notion de processus et de cycle de vie. Un cycle de vie représente un enchaînement d'étapes qui se succèdent les unes aux autres au travers de phases et de jalons [Ducellier, 2008]. Même si l'on fait généralement référence au cycle de vie du produit, il en existe d'autres faisant référence à l'organisation et aux informations manipulées dans le cadre du développement d'un produit. Ainsi [Boudichon, 1994] formule les différentes propositions suivantes :

- **Le cycle de vie du projet :** il représente les étapes de la conduite de développement d'un projet de conception organisées en phases ponctuées de jalons.
- **Le cycle de vie du produit :** il correspond aux différentes phases de maturité du produit.
- **Le cycle de vie des informations :** il correspond au niveau de maturité des informations utilisées dans les activités du processus de développement d'un produit.

Cette organisation permet de structurer la conduite d'un projet en regroupant les activités du processus de conception dans un objectif commun [Midler, 1993]. Chez les industriels, les jalons sont à la charge d'un "responsable jalon" ou du "responsable projet" qui anime et pilote les tâches assurées par les experts issus de différents domaines métiers. Le jalonnement des activités du processus de conception peut être associé au domaine de l'ingénierie simultanée dans la mesure où il fait intervenir un ensemble d'activités et d'acteurs en parallèle, comme le montrent les travaux de Subrahmanian [Subrahmanian et al., 2005] et de Charles [Charles, 2005] (Figure 5) . Au sein de ce schéma, les flèches symbolisent à la fois les flux d'informations et de données échangées entre les acteurs, mais aussi l'évolution du produit dans son cycle de vie.

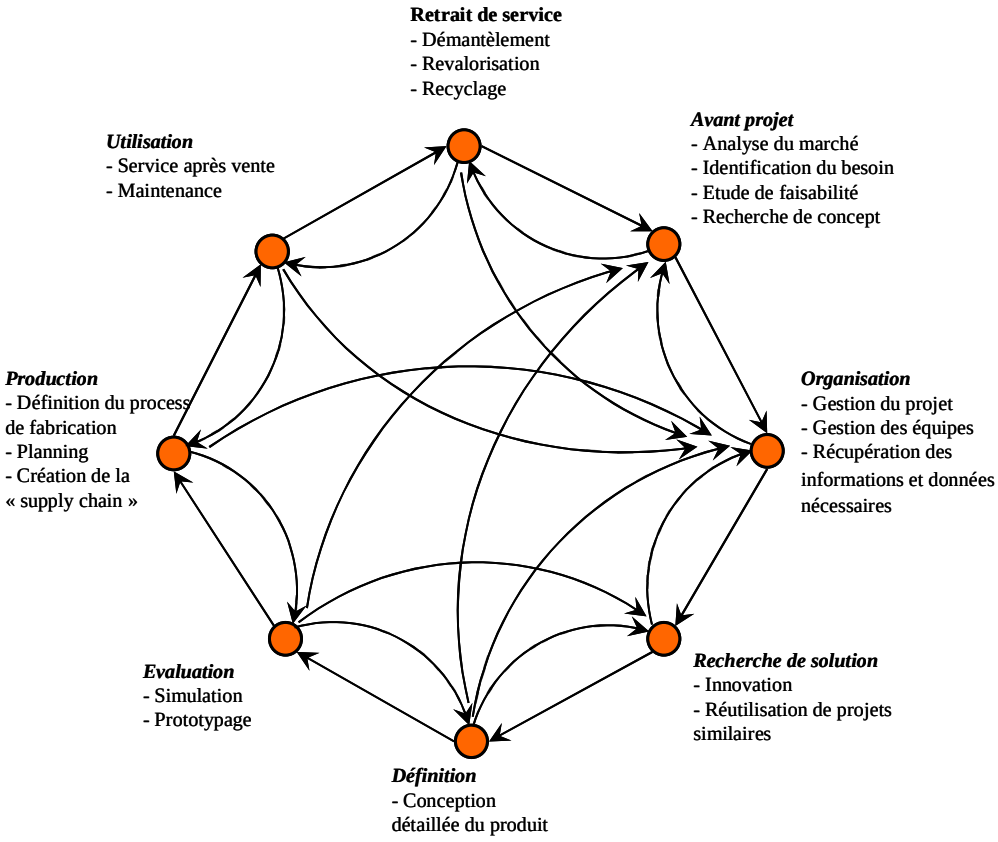


Figure 5 : les activités du cycle de vie du produit [Charles, 2005]

Le cycle de vie du produit caractérise l'état de maturité du produit depuis ses phases de développement jusqu'à ses phases de fin de vie. Il est généralement composé de quatre étapes [Boujut et al., 2003][Grebici, 2007] (Figure 6):

- **Le produit virtuel** (n'existe que sous forme virtuelle : maquette numérique, dessins, etc.)
- **Le produit évoluant** (fabrication et début de commercialisation)
- **Le produit mature** (produit mature et commercialisé)
- **Le produit en déclin** (fin de vie du produit et retrait)

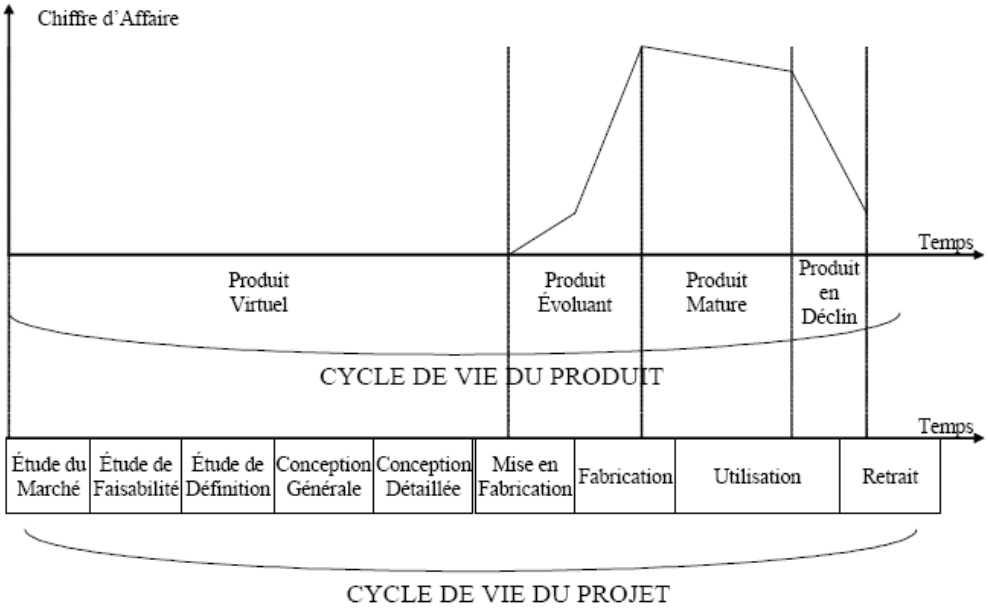


Figure 6 : cycle de vie du produit

C'est au cours de la phase correspondant à l'état de maturité qualifié de "produit virtuel" que les gains à réaliser sont les plus importants, comme l'illustrent les travaux de Clautrier [Clautrier 1991]. Cet auteur met en avant le paradoxe d'un engagement fort des coûts lors des premières phases d'un projet, alors que les dépenses réelles sont encore relativement faibles, contrairement aux dernières phases du projet. Ce constat est par ailleurs repris et vérifié par de nombreux auteurs dans divers domaines, comme celui de la connaissance en conception de produit [Sellini, 1999]. Il constitue un élément principal de la problématique que nous traitons car c'est un facteur déterminant de l'évolution des pratiques de la conception puisqu'il contribue à intervenir le plus tôt possible dans le processus de développement du produit et il est, par conséquent, à l'origine de nombreux nouveaux concepts.
En effet, dans les phases composant la définition du produit virtuel, de nombreuses modélisations du produit sont utilisées au travers d'un grand nombre de domaines d'expertises. Ces modélisations sont de nature très différentes car elles peuvent être géométriques, fonctionnelles, paramétriques, etc., et nécessitent l'utilisation de plusieurs outils hétérogènes.

En conclusion, dans le cadre du développement d'un produit impliquant de nombreuses activités en parallèle, il est admis que les gains les plus importants sont réalisables dans la phase correspondant à l'état de maturité du "produit virtuel", et par conséquent relativement tôt dans le processus de conception. Dans ce contexte, l'échange des données est crucial et nécessite des méthodologies et des outils spécifiques qui interviennent dans le cycle de vie des informations afin d'assurer la cohérence, la traçabilité et le partage des données.

1.1.2 De l'ingénierie séquentielle vers l'ingénierie collaborative

Afin de réduire les temps de développement et de mieux correspondre à la réalité, les pratiques des industriels ont évolué d'une ingénierie séquentielle vers les concepts de simultanéité, d'intégration, voir de collaboration [Lu et al., 2007].

Dans le domaine de la conception, en ingénierie séquentielle, l'un des premiers modèles de référence fut celui de [Pahl et al., 1984] qui considère le processus de conception comme une "succession hiérarchique et séquentielle de phases", dont découle, en particulier, la norme AFNOR [AFROR 2002]. Cette approche qualifiée d'algorithmique, propose une organisation en quatre phases successives [Pahl et al., 1996] [Pahl et al., 2007] : la clarification du besoin, la recherche de concepts, la conception préliminaire, la conception détaillée.
D'autres travaux, tels que ceux de [Aoussat, 1990], [Ullman 2003], font également l'objet d'organisation du processus de conception en phases qui s'enchaînent de manière successive. Néanmoins, ces approches ne semblent pas systématiquement correspondre à la réalité des pratiques industrielles [Charles, 2005], notamment lorsque l'on aborde les notions d'itérations de cycles élémentaires et de convergence des points de vue [Merlo, 2003].

1.1.2.1 La notion d'itération d'un cycle élémentaire :

Quand on analyse finement l'activité d'un concepteur au sein de chaque phase de conception, on s'aperçoit que l'on peut considérer une tâche comme un processus de conception à part entière. En effet, un concepteur dans sa tâche devra rechercher des solutions, les évaluer, en sélectionner une, la formaliser, la diffuser, etc. Le concepteur est donc amené à effectuer plusieurs itérations avant d'arriver à un consensus. C'est ce que proposent [Blessing 1994] puis [Roozenburg et al., 1995] au travers d'une modélisation du processus de conception "itérative basée sur un cycle élémentaire", avec notamment l'utilisation de solutions de prototypage et surtout de simulation permettant au concepteur de tester ses concepts et de valider ses choix. Les itérations sont inévitables dans les processus d'ingénierie et doivent être gérées efficacement afin de maintenir la cohérence, l'intégrité et la validité du modèle informationnel du produit à travers son cycle de vie [Stark et al., 2004] [Liu et al., 2009].

1.1.2.2 La notion de convergence des points de vue :

Si chaque tâche du processus de conception peut être considérée comme un processus à part entière, l'ensemble des tâches et des acteurs doivent néanmoins converger vers une solution unique et ce malgré le fait qu'ils ne partagent pas forcément les mêmes vues métiers. Le processus de conception résulte donc de la construction de compromis entre les différentes fonctions et les différents métiers.
Dans un processus de conception organisé en phases successives, les tâches se succèdent les unes aux autres, les suivantes réutilisant (du moins en partie) les données issues des précédentes. Or, dans la mesure où chaque tâche peut s'inscrire dans un cycle itératif, la "re-définition" d'une tâche amont impose la remise en question des tâches situées en aval et par conséquent des boucles longues et délicates de re-conception à l'origine de nombreuses pertes de temps et d'erreurs. De plus, la notion de convergence implique que les choix des concepteurs convergent vers le même objectif, nécessitant forcément la prise en compte dans une tâche, des contraintes inhérentes aux autres tâches afin de trouver des compromis. Or, en suivant un processus organisé en phases successives, les tâches situées en aval se retrouvent pénalisées car elles sont de plus en plus contraintes par les choix effectués lors des tâches amont favorisant, encore une fois, la re-conception, voire même la prise en compte ou l'apparition tardive de contraintes ou conflits non encore détectés. Il est alors possible que ce type de situation remette en cause tout le développement du produit, car demandant assez tard des modifications sur des décisions importantes prises en amont.

1.1.2.3 La parallélisation des activités du processus de conception

Dans un souci d'amélioration de l'efficacité et de la qualité en ingénierie, et pour faire face aux inconvénients inhérents à l'enchaînement séquentiel des activités liées au cycle de vie des produits, une autre vision du processus global de développement du produit est apparue (Figure 7) [Demoly, 2010]. Proposant de nombreuses déclinaisons, elle peut être qualifiée d'ingénierie intégrée

[Andreasen et al., 1985] [Tichkiewitch et al., 1995], simultanée [Bocquet, 1998], distribuée [Brissaud et Garro, 1996], concourante [Sohlenius, 1992] (Concurrent Engineering – CE), voire collaborative [Li et al., 2005] [Li et al., 2006] [Lu et al., 2007]. Cette vision est apparue dans les années quatre-vingt et est aujourd'hui largement utilisée par les entreprises. Elle propose, en effet, d'intégrer ou de mettre en étroite relation l'ensemble des activités de conception d'un produit dès l'initialisation de sa conception [Prasad, 1997]. Dans le cadre de nos travaux de recherche, nous parlerons plutôt d'ingénierie collaborative et intégrée car caractérisée par deux grands principes : la simultanéité dans les processus d'ingénierie et l'intégration des différents métiers. Le premier principe consiste à réaliser en parallèle différentes activités liées à la conception du produit. L'intégration consiste, quant à elle, à établir des échanges entre tous les acteurs dès les premières phases du projet, par la prise en compte, à chaque phase du développement, des contraintes relatives à l'ensemble des étapes du développement du produit. L'ingénierie collaborative et intégrée est donc une façon différente d'organiser la conception, car elle permet de rompre avec l'organisation séquentielle des activités, et par conséquent, d'améliorer l'efficacité et la qualité des activités d'ingénierie.

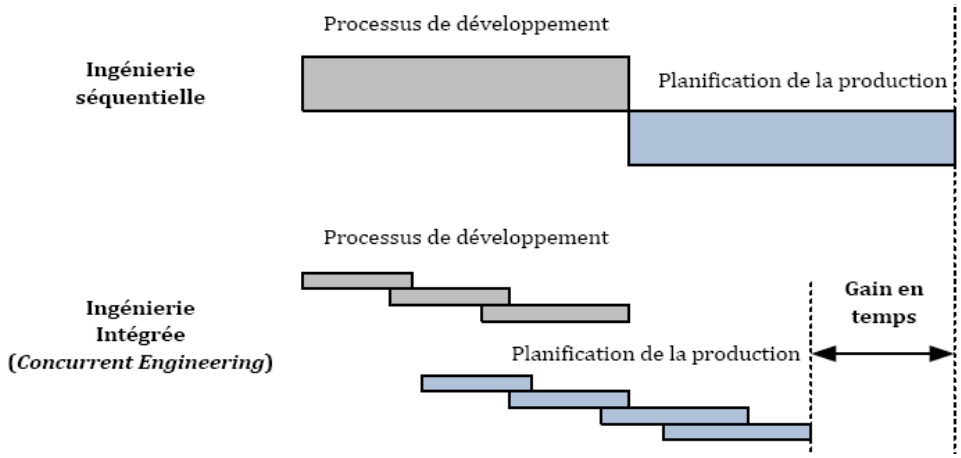


Figure 7 : de l'ingénierie séquentielle vers l'ingénierie intégrée [Demoly, 2010]

Une des premières conséquences de ces organisations de projets de conception est le côtoiement d'une grande diversité d'acteurs qui doivent simultanément concourir au même but. Les situations de partage des données, des informations et des connaissances métier associées au produit, au marché, aux procédés, etc., qui sont par nature, hétérogènes, sont alors multipliées [Zaclad, 2003]. Dans ce contexte, des méthodes et des outils de collaboration se sont développés afin de permettre aux concepteurs de disposer de la bonne information, à la bonne personne, au bon moment, et surtout au bon niveau de maturité.
De plus, un des nouveaux enjeux rendu possible par les processus d'ingénierie intégrés et collaboratifs, est de répondre à un contexte de plus en plus répandu d'entreprise étendue. Cela consiste à substituer une proximité virtuelle "organisationnelle" à la proximité géographique actuelle, et lever les blocages culturels de la collaboration inter-entreprises et inter-disciplinaires [Large et al., 2009].

Nous retenons l'importance des processus collaboratifs et simultanés, notamment dans le cadre des notions d'itérations, inévitables dans le processus de conception. Elles doivent être correctement prises en compte afin de limiter les phases de re-conception et permettre aux ingénieurs de converger plus vite vers une solution optimisée. Dans ce contexte, la disponibilité et l'accès aux informations est un enjeu essentiel des processus intégrés, permettant d'améliorer le développement des produits.

1.1.3 Typologie des activités de conception

La conception de produit peut être abordée selon différents points de vue ou types selon le contexte et l'approche que l'on a choisis d'utiliser dans le processus d'ingénierie. Ainsi, [Aifaoui et al.,

2004] formule, dans sa proposition, cinq grandes catégories qu'il est possible d'aborder sur deux niveaux différents.
Le premier niveau oppose la conception axiomatique [Suh, 1990] à la conception algorithmique [Pahl et al., 1996]. L'une repose sur l'utilisation d'axiomes généraux et l'adoption d'un cheminement récurent entre différents domaines, sans toutefois imposer d'étapes rigoureuses. L'autre se caractérise par un ensemble de tâches séquentielles et distinctes. Ainsi, les tâches sont associées à des étapes permettant de structurer l'enchainement des tâches entre elles.
Le second niveau regroupe trois sous-catégories qui complètent la proposition : la conception routinière qui réutilise des solutions déjà connues et éprouvées issues d'autres conceptions ; la conception à base de cas qui reprend et modifie une conception existante pour qu'elle réponde aux nouveaux besoins fonctionnels ; et la conception innovante qui fait appel à la notion de créativité pour trouver de nouvelles solutions et de nouvelles voies pour la résolution de problèmes de conception.

Dans le même courant, [Brown et al., 1985] proposent une autre organisation des activités de conception en trois classes caractérisées par deux critères : les sources des connaissances et les stratégies de résolution de problèmes (Tableau 1).

Classes	Sources des connaissances	Stratégies de résolution de problèmes
Conception routinière	**Connues**	**Connues**
Conception innovante	**Connues**	**Non connues**
Conception créative	**Non connues**	**Non connues**

Tableau 1 : caractérisation des classes de conception [Brown et al., 1985]

Dans cette proposition, la conception créative se démarque de la conception innovante par le fait que, dans le premier cas, l'activité de conception porte sur un produit inconnu alors que, dans l'autre cas, elle porte sur un produit connu.

Enfin, une autre typologie semble bien caractériser la réalité des processus industriels. Elle est aujourd'hui reconnue et admise par la communauté scientifique et met en avant deux classes de résolution de problèmes d'ingénierie [Durruvu, 1989] [Deneux, 2002]. Cette typologie est représentée Figure 8 (tirée du modèle de gestion des connaissances KnoVA-Méta [Serrafero et al., 2006]).

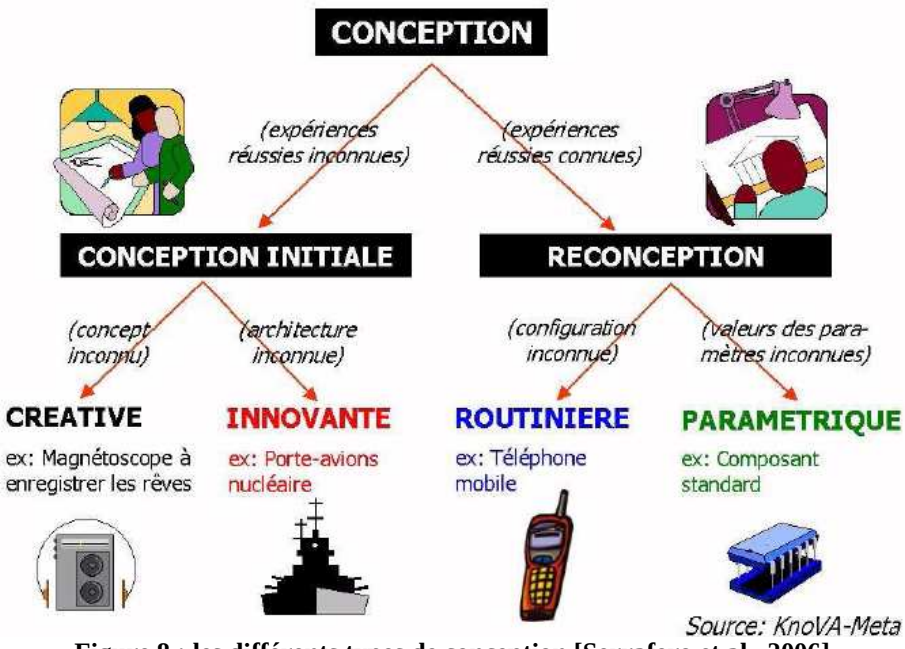


Figure 8 : les différents types de conception [Serrafero et al., 2006]

Ainsi, selon cette typologie, on distingue la "conception initiale" de la "re-conception". La "conception initiale", permet de concevoir et d'inventer un produit pour la première fois et ne nécessite pas de base préexistante. Quant à la "re-conception", elle permet de reconcevoir un nouveau produit à partir d'un objet existant en s'appuyant, par exemple, sur un cahier des charges déjà établi, tout en prenant en compte les questions de coûts, de poids, de volume, de performance, etc.
Dans le cadre de ces travaux, nous nous intéressons plus particulièrement à la "re-conception" et à ses deux sous-catégories :

- **La conception routinière :** à concept et architecture connus à l'avance, l'objet à concevoir est alors à configuration inconnue parmi une grande variété de combinaisons et de topologies possibles [Dorville et al., 1996]. Les grands principes sont alors identifiés et connus mais leur adaptation est nécessaire pour être réutilisés.
- **la conception paramétrique :** à concept, architecture et configuration connus à l'avance, l'objet est conçu à partir d'un ensemble de paramètres et de règles métiers hérités de propriétés connues et identifiés. L'activité de conception revient alors à modifier ces paramètres et ces règles afin de régénérer un nouveau produit par associativité. La conception paramétrique peut s'appuyer sur une modélisation et une configuration numérique permettant de transformer le besoin fonctionnel en solutions géométriques 3D.

L'activité de conception nécessite parfois des allers retours entre les différentes classes de la typologie. Néanmoins, il apparaît qu'aujourd'hui dans l'industrie manufacturière, le temps consacré aux phases de re-conception occupe environ 80% du temps de conception, ne laissant que peu de place à l'innovation et la créativité [Gomes, 2008]. En effet, si l'on considère le cas du développement d'un groupe moto-propulseur automobile, l'activité de re-conception autour des composants élémentaires du moteur (carter, piston, bielle, etc.), qui sont bien connus des concepteurs, représente l'essentiel du temps passé. Ces composants sont repris d'un moteur à l'autre et souvent des solutions déjà existantes et éprouvées sont malheureusement "réinventées". Ce constat bride l'activité d'ingénierie car ce temps passé par les concepteurs sur des activités routinières, fastidieuses et à faible valeur ajoutée, occulte la vision qu'ils pourraient avoir sur de nouveaux concepts, si ces phases de re-conception étaient mieux organisées et plus réduites dans le temps.

Dans le cadre de ces travaux, nous nous focaliserons sur des activités de conception entrant dans le cadre de la conception routinière et paramétrique, dans la mesure où nous ferons appel massivement à des modèles paramétriques du produit.

1.1.4 Les approches pour la rationalisation et l'optimisation des processus routiniers de conception de produits

L'évolution du processus de conception, d'une ingénierie séquentielle vers une ingénierie intégrée et le développement du contexte collaboratif, a encouragé l'émergence de méthodologies et d'outils permettant d'optimiser et de rationaliser le processus de conception.

Ainsi, la conception à base de connaissances ou Knowledge Based Engineering (KBE) [Chapman et al., 2001] [Prasad, 2005] [Milton, 2008] nous semble être une démarche particulièrement bien adaptée aux problématiques d'ingénierie routinière, car elle permet la réutilisation de connaissances capitalisées sur des expériences ou des projets antérieurs [Stokes, 2001]. En effet, l'approche KBE consiste à mettre à disposition des applications logicielles adaptées pour l'acquisition et la réutilisation de connaissances sur un produit de la manière la plus intégrée qu'il soit. Cette approche intéresse plus particulièrement les entreprises dans la mesure où elle permet le management de leur capital intellectuel au travers de la gestion de leurs connaissances et savoir-faire.
Nous pouvons mettre en évidence différents outils ou méthodologies tels que les systèmes entrant dans le cadre de "l'Ingénierie Hautement Productive" et du PLM apprenant [Gomes 2008], les mémoires d'entreprise [Monticolo et al., 2008], [Bekhti et Matta, 2003], [Bidal et al., 2002**],** [Matta et al., 2000], les retours d'expériences [Baizet, 2004] ou encore les méthodologies de conception

telles que la théorie C-K (Concept-Knowledge) [Hatchuel et al., 2004] qui ont pour but de capitaliser les connaissances produites lors des projets de conception et de les exploiter pour générer de nouvelles solutions, qu'elles soient routinières ou innovantes.
En effet, de nombreux projets de recherche tel que MOKA (Mefthodology and tools Oriented to Knowledge Based engineering Applications) [Ammar-Khodja et al., 2005] [Skarka, 2007], DEKLARE (Design Knowledge Acquisition and Redesigning Environnement) [DEKLARE, 1995] [Blessing, 1996] [Arana et al., 2000] [Yvars, 2001], CoDeKF [Gomes 2008] ou ATLAS [Aldanondo et al., 2010] ont contribué à permettre la prise en compte des connaissances tout au long du cycle de vie d'un produit, et de définir un cadre ou de bonnes pratiques pour mieux concevoir.

Une approche complémentaire à la conception à base de connaissances est l'utilisation de modèles métiers paramétrés et associatifs. Le développement de la CAO paramétrique et variationnelle [Lesage, 2003] a permis d'enrichir les modèles conçus sur la base d'informations liées aux connaissances des concepteurs [Charles et al., 2005]. En effet, les concepteurs ne se contentent plus de modéliser géométriquement en 3D un composant, ils introduisent de manière judicieuse des paramètres et des règles métiers qui pilotent cette géométrie. Ainsi, la modification d'un seul paramètre directeur pourra entrainer, via les règles métier sous-jacentes, la modification de plusieurs autres paramètres, conduisant à une modification globale de la solution puis à une mise à jour de la géométrie paramétrique du composant modélisé (ce comportement de propagation et de mise à jour peut-être qualifié de notion d'associativité).
En effet, les modèles métiers paramétrés permettent aujourd'hui d'agir directement sur les modélisations géométriques (2D ou 3D), voire comportementales du produit dans le cas de la simulation numérique. Grâce à de tels modèles, il est possible d'avoir une meilleure prise en compte des contraintes de conception issues de divers domaines métiers impliqués dans la conception et la modélisation d'un composant. L'autre intérêt de cette approche est de permettre la génération rapide d'un grand nombre d'architectures et de tester différents concepts qui n'auraient potentiellement pas été explorés faute de temps. Cela contribue à optimiser le processus de conception, mais aussi à libérer du temps pour l'innovation.
Un exemple intéressant constitue le concept de "CAO 4D" [Serrafero et al,. 2007] [Gomes et al., 2002] [Gomes, 2008] qui présente les mutations dans le domaine des logiciels de CAO qui ont évolué de la 2D vers la 3D, et qui évoluent maintenant vers la 4D. La 4D étant le pilotage et la génération de modèles géométriques 3D à partir de paramètres et de contraintes encapsulés dans les modèles, ou issus de bases de connaissances d'ingénieries routinières.
La Figure 9 met en évidence les avantages du concept de "CAO 4D". A partir de la modélisation d'un produit générique dans lequel sont intégrés des paramètres et des règles métiers, l'utilisateur peut jouer sur la modification des valeurs des paramètres, voire même sur les configurations de paramètres (activation ou désactivation de certains paramètres ou règles) pour générer, à la volée et en un laps de temps très court, différentes architectures d'un produit.

Aujourd'hui, ces approches deviennent courantes dans l'industrie manufacturière. Elles utilisent parfois des approches de modélisation à base de squelettes, c'est-à-dire des modélisations géométriques de composants, voire même d'assemblages complets, qui s'appuient sur des squelettes paramétrés et réglés. Ces squelettes constituent des constructions filaires ou surfaciques sur lesquelles s'appuient les concepteurs pour créer leurs modèles géométriques solides, selon l'approche classique de la modélisation CSG (Constructive Solid Geometry [Laidlaw et al., 1986]).

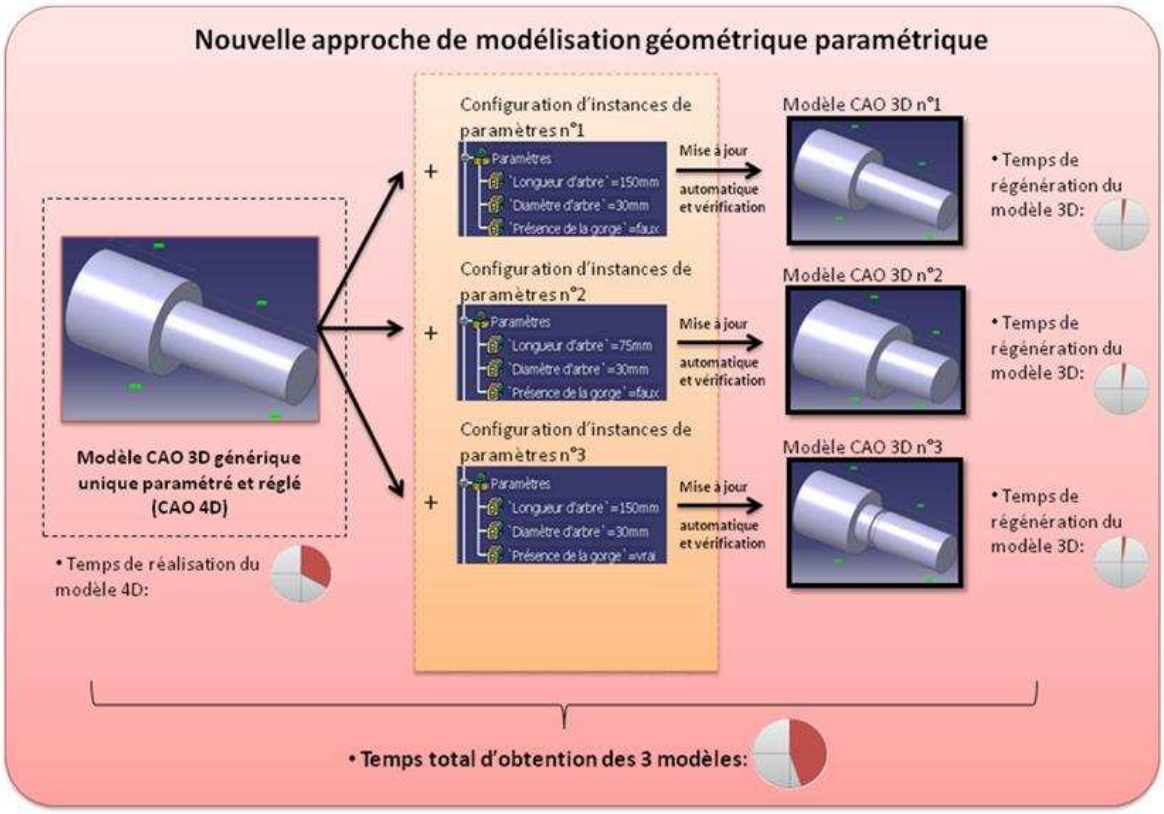


Figure 9 : exemple de conception paramétrée (CAO 4D) à partir de trois configurations de paramètres permettant de générer les trois versions d'un même produit [Gomes, 2008]

Enfin, une autre approche contribuant à la rationalisation et à l'optimisation des processus routiniers d'ingénierie concerne le concept de "Design For X" (DFX) [Holt et al., 2009], qui résulte d'une prise de conscience de l'importance de l'impact des prises de décision en phase amont au sein du processus de conception et de développement, sur les métiers plutôt situés en aval du cycle de vie. Il existe plusieurs approches DFX qui peuvent caractériser, soit un point de vue lié aux métiers intervenants dans le processus de développement (Design for cost, Design for Usability, etc.), soit un point de vue lié aux phases du cycle de vie du produit, tels que la fabrication (Design for Manufacture – DFM), l'assemblage (Design for Assembly – DFA), le désassemblage (Design for Disassembly – DFD), le recyclage (Design for Recycling), etc.
Les méthodes DFX fournissent alors des approches permettant de prendre en compte le maximum d'informations, de connaissances et de procédures (issues des X domaines d'expertises) du cycle de vie d'un produit, et ceci, tout au long du processus de conception. Une solution de conception est le résultat de multiples évaluations d'experts au cours d'un processus collaboratif et multidisciplinaire permettant de converger vers une solution acceptable au regard de l'ensemble des contraintes. Ainsi, le concepteur peut concevoir le produit tout en ayant conscience de l'impact de celui-ci selon la caractéristique ou le métier X [Benhabib, 2003].

Dans ce contexte, chaque modèle métier capitalise et manipule un ensemble de paramètres et de règles métiers qui sont potentiellement partagés avec d'autres modèles métiers. Cela implique une interaction entre les concepteurs afin que ces données, d'une granularité fine (paramètres et règles), soient utilisées de façon collaborative et cohérente. Néanmoins, cette capitalisation pose problème, dans la mesure où les paramètres et les règles ne sont pas accessibles en dehors du modèle qui les utilise. Nous nous intéresserons particulièrement à ce problème de capitalisation et d'accessibilité.

1.1.5 Optimisation et rationalisation du processus de conception dans le cas particulier de la simulation numérique

Tout pousse à penser que l'évolution des différentes approches présentées tout au long de ce chapitre sont complémentaires et tendent à favoriser l'ouverture du processus de conception, afin d'être en mesure d'identifier les conflits au plus tôt, et de dégager des compromis pour fiabiliser le processus de développement d'un produit.
Ainsi, cette démarche d'optimisation et de rationalisation initiée autour des outils de modélisation CAO 3D, tend à s'étendre à l'ensemble des activités du processus d'ingénierie. Un domaine particulièrement impacté concerne le domaine de la simulation numérique, qui à connu cette dernière décennie, des évolutions majeures pour aujourd'hui devenir un moteur de l'innovation dans le développement de produits.

En effet, dans le contexte économique et industriel actuel, l'utilisation de la simulation numérique est devenue un enjeu essentiel, permettant d'ouvrir le champ de l'innovation, d'optimiser et rationaliser le développement d'un produit, tout en permettant des gains de temps et des gains financiers. Néanmoins, de par leurs diversités, tant au niveau des domaines d'expertises, que des outils utilisés, ou encore des représentations des composants, etc., les activités de la simulation numérique sont difficiles à mettre en place et nécessitent le recours à des processus spécifiques de préparation et de traitement des modèles métiers, de gestion et de collaboration des données, ainsi que des informations utilisées par les concepteurs. Cette collaboration devient aujourd'hui cruciale pour les industriels, dans la mesure où le développement d'un produit nécessite l'utilisation simultanée d'un grand nombre de modélisations géométriques et comportementales, hétérogènes et indépendantes. Or, dans chaque modèle métier, les acteurs du bureau d'études et de calculs manipulent un ensemble de paramètres et de contraintes (relation mathématiques, règles métiers, conditions limites, etc.) leur permettant de générer plusieurs configurations du produit et de les tester (numériquement) rapidement, grâce au recours à des approches d'intégration CAO/Calculs. En effet, ces approches autorisent le lancement d'un grand nombre d'itérations, au cours desquelles différentes variantes de produits sont explorées et testées, permettant ainsi de valider des principes d'architectures robustes.

Nous verrons, dans la suite de ce chapitre, l'importance de la place de la simulation numérique. Elle permet des gains significatifs dans le processus de conception, cependant elle contribue à accentuer les problèmes de partage et de collaboration autour des connaissances dans le processus de développement du produit.

1.2 La simulation numérique dans le processus de développement du produit

La simulation numérique est une activité du processus d'ingénierie utilisée lors du développement d'un produit. C'est un domaine à part entière dans le processus de conception qui intervient à de nombreux niveaux et qui assiste les concepteurs dans leur choix et les solutions de conception qu'ils retiennent (Figure 10 et exemple en Annexe 1). Son évolution a été très importante et liée à la fois aux approches d'ingénieries intégrées, et au développement des technologies logicielles et matérielles, autorisant aujourd'hui des puissances de calcul permettant de reproduire finement le comportement physique d'un composant.

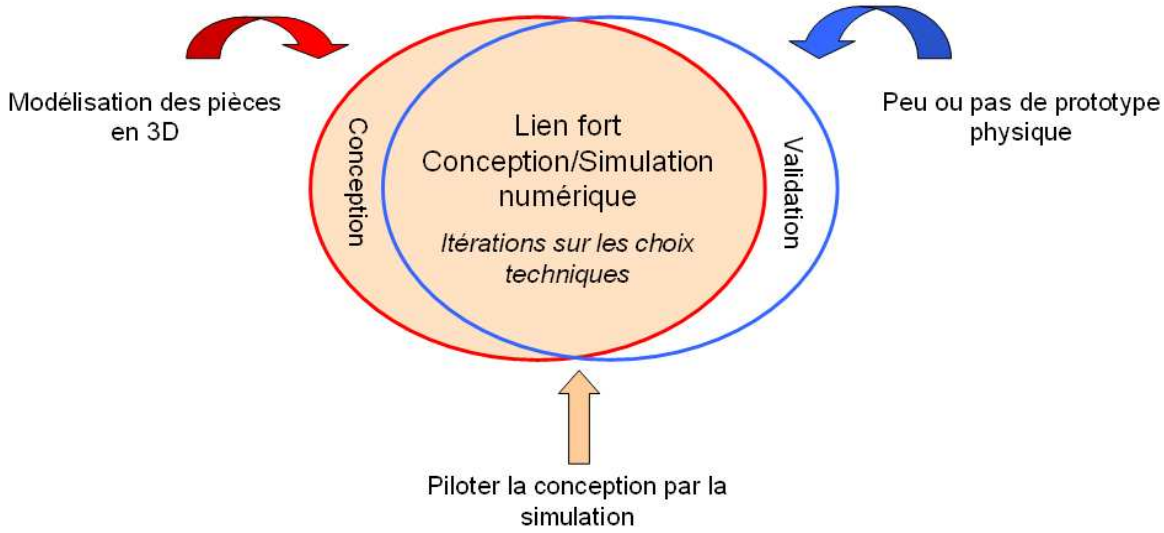


Figure 10 : l'utilisation de la simulation

Néanmoins, dans la mesure où il n'existe aujourd'hui aucune plateforme de partage collaborative des données de granularité fines (paramètres et contraintes) de façon indépendante des domaines d'expertises ou des applications logicielles qui les utilisent, la capitalisation, la traçabilité, le maintien en cohérence et la réutilisation est délicate et entraine du bruit et des erreurs dans le processus de conception. En effet, il arrive trop souvent que des calculs soient lancés sur de mauvaises configurations et/ou versions de paramètres, entre pièces connexes ou non. Les pertes de temps et financières qui en découlent dépendent du moment où l'on s'aperçoit de l'erreur. Dans le cas d'une erreur constatée tardivement, les pertes peuvent être considérables [Badin et al., 2009].
Dans la suite de ce chapitre, consacré au domaine de la simulation numérique, nous décrivons et définissons globalement les activités du calcul. Nous abordons ensuite les pratiques et les enjeux inhérents à l'utilisation de la simulation numérique et son intégration dans le processus de conception. Enfin, nous proposons un bilan de ces pratiques nous permettant de définir le contexte de ces travaux de recherche.

1.2.1 Un ensemble d'activités hétérogènes

Dans le cadre du processus de conception de produit, la simulation numérique a pour objectif de reproduire, numériquement, le comportement physique réel d'un composant soumis à des sollicitations, correspondant à des situations de sa vie future. La réalité du comportement que l'on veut simuler s'obtient à l'aide d'algorithmes ou d'un modèle mathématique qui traduisent ce comportement en un système d'équations aux dérivées partielles à résoudre sous forme analytique ou numérique. Dans ce dernier cas, le système d'équations est résolu de manière approchée par des techniques numériques, notamment des techniques de discrétisation qui permettent d'appréhender le comportement d'un milieu continu par une discrétisation de celui-ci. Les principales méthodes numériques connues sont les méthodes par éléments finis (FEM – Finite Element Modelling, les

plus répandues [Bathe, 1996] [Belytschko, 2000]), par différences finies, par volumes finis, par équations intégrales.

Ainsi, la simulation numérique est un domaine à part entière dans le processus de conception [Devalan, 2009]. Elle est composée d'un grand nombre d'expertises extrêmement variées (aussi appelés domaines métiers) source de sa complexité : par exemple, le domaine du crash, de la thermique, de l'acoustique, de la fatigue, etc. De plus, chaque domaine renferme des sous-domaines en fonction du contexte d'étude ou des phénomènes physiques étudiés: en thermique on peut observer différents phénomènes de types radiatifs, convectifs, conductifs ; en mécanique des fluides on peut observer des phénomènes de laminage, de turbulences, de dilatation, etc. Chaque cas d'analyse, le calcul linéaire, non linéaire, statique, dynamique, nécessite des méthodes de résolution appropriées.

Au travers de ces nombreuses méthodes, la finalité d'utilisation de la simulation numérique est de guider les concepteurs à différents stades du processus de conception, impliquant des échanges de données entre les expertises, et les différents modèles métiers. La diversité de ces domaines traduit l'hétérogénéité des activités de la simulation numérique (Annexe 2). En effet, chaque domaine d'expertise nécessite des connaissances et des compétences pointues sur la physique étudiée, les méthodes de résolution, de représentation des composants, les outils utilisés, etc. Les calculateurs sont alors souvent segmentés dans leurs domaines et utilisent des outils (logiciels notamment) spécifiques d'une activité de simulation à l'autre.
Ainsi, [Mer, 1998] définit cette problématique d'intégration des métiers au sein d'un même projet, à travers la notion de "monde" ; un "monde" étant un ensemble cohérent et structuré composé d'outils, d'objets et d'acteurs qui développent des mêmes logiques d'actions, relèvent des mêmes échelles de grandeur et partagent des connaissances collectives. Cet état de fait rend la coopération entre acteurs de mondes différents complexe et souvent peu efficace pour les projets. L'évolution constante et croissante, de l'activité de la simulation numérique dans les bureaux d'études, et son intégration récente dans les processus de développement de produits a entraîné l'émergence de nouvelles difficultés en termes d'organisation, de suivi et de traçabilité [Aidi, 2007].

Ce constat est d'autant plus révélateur du développement des approches de calcul multi-physique (Annexe 3) visant à coupler plusieurs physiques afin d'étudier plus finement le comportement d'une pièce. En effet, la multiplicité des physiques augmente la complexité du modèle de données [Huynh, 2006]. Un volume important de données doit alors transiter d'un modèle métier à l'autre sans perte d'information, et ce, malgré le caractère hétérogène de ces modèles.

Enfin, couplés aux approches d'ingénierie simultanée (Annexe 4), la mise en place des processus de simulation peut-être délicate. Il est nécessaire de bien l'intégrer dans le processus de conception et de favoriser la collaboration entre les acteurs d'un même projet [Lee, 2005]. En effet, si les activités de simulation sont aujourd'hui bien maitrisées, il n'en n'est pas de même pour les informations et les connaissances partagées [Dreisbach, 2010] [Charles, 2005].

L'hétérogénéité des activités de la simulation numérique, entraine des difficultés de pilotage et des erreurs, comme par exemple des paramètres perdus et non pris en compte, ou des calculs lancés sur de mauvaises valeurs entre deux modèles métiers. Il semble nécessaire de développer des méthodes et des outils permettant de mieux prendre en compte ces informations et ces connaissances, afin de mieux intégrer les métiers de la simulation numérique entre eux et au sein du processus de conception. C'est-à-dire, améliorer le couplage entre processus de conception, de simulation et de gestion des connaissances.

1.2.2 La place de la simulation numérique dans le processus de conception de produit

Suite à ce que nous venons d'exposer, nous pouvons aisément comprendre que les pratiques de la simulation numérique varient beaucoup d'une organisation à l'autre. En effet, certaines organisations privilégient une modélisation spécifique (à l'aide de progiciels dédiés à la simulation),

tandis que d'autres intégreront leurs modèles de simulation directement dans les modèles et les processus de CAO, favorisant ainsi l'intégration conception-simulation ; nous parlerons alors d'intégration CAO/Calcul [Benhafid, 2005] (Figure 11).

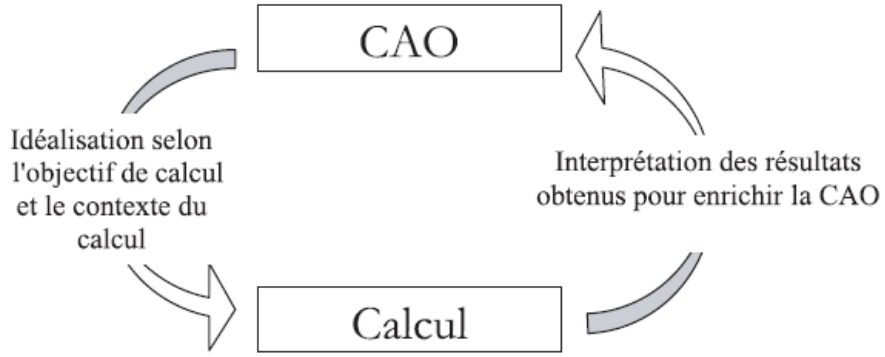


Figure 11 : principe d'utilisation de l'intégration CAO/Calcul [Benhafid, 2005]

Selon ces approches, la simulation numérique est très liée à la modélisation CAO et est utilisée à différents moments dans le processus de conception pour piloter et aider les concepteurs en fonction de leurs besoins et du niveau de maturité du produit.
Ceci nous amène à identifier plusieurs types de simulation numérique et leur positionnement dans le processus global de conception de produit.

1.2.2.1 Typologie des activités de simulation numérique

Etant donné que la simulation numérique s'appuie sur une maquette virtuelle, elle s'affranchit des limitations du prototype réel et autorise l'exploration de multiples solutions qu'il est possible de tester par le prototype virtuel. De nos jours, la simulation numérique est essentiellement utilisée pour s'assurer d'un niveau de performance et optimiser le système mécanique. En ce sens, "le rêve industriel" serait d'utiliser la simulation à tous les niveaux dans le processus de conception, de façon à concevoir juste du premier coup, tout en évitant la réalisation de prototypes physiques extrêmement couteux.

Ainsi, pour répondre à différents besoins tout au long du cycle de vie du produit, [Charles, 2005] et [Amann, 2010] proposent une typologie des activités de la simulation décrite en Annexe 5, positionnant les pratiques de la simulation numérique en amont ou en aval du processus de conception.
En effet, utilisée plutôt en aval et qualifiée de passive, la simulation intervient de façon séquentielle suite à la modélisation géométrique des composants. Elle permet de contrôler, valider, certifier, comprendre sur la base d'une proposition de solution. Historiquement la plus utilisée, elle met en œuvre des modèles métiers très détaillés dont l'objectif est d'être le plus prédictif possible en ce qui concerne le comportement futur de la pièce. Si les modèles de simulation se basent sur la modélisation géométrique de pièces, ils sont totalement décorrélés de ces dernières, de sorte qu'aucune modification tardive de la géométrie ne peut être prise en compte sous peine de reprendre profondément le modèle de calcul. De fait, ces pratiques de la simulation impliquent finalement peu de collaboration.
Utilisée en amont, le rôle de la simulation devient plus proactif et intégré (simulation indicative et optimisation cf. Annexe 5). Généralement couplée à des modèles CAO simplifiés, elle est utilisée pour tester des architectures et aider les concepteurs dans leurs choix [Gardan et al., 2003] [Assouroko et al., 2009]. Elle met en œuvre des modèles métiers utilisés en parallèle suivant de nombreuses itérations permettant d'explorer plusieurs solutions de conception. Ces pratiques ont un impact considérable sur la collaboration des acteurs, dans la mesure où de nombreux paramètres et règles doivent être échangés entre les modèles, de sorte que les concepteurs convergent vers une solution cohérente. Ce type de simulation tend à se généraliser car elle permet d'optimiser le processus de conception.

Compte-tenu des différents types de simulation, plusieurs profils d'acteurs, décrits en Annexe 6, sont nécessaires. Les acteurs, maitrisant un certain nombre de compétences, sont au centre du processus de simulation et interviennent lors de plusieurs phases du processus de conception avec un impact sur la collaboration.

1.2.2.2 L'utilisation de la simulation numérique dans le processus de conception et l'impact sur la collaboration des acteurs

La typologie des activités de simulation numérique met en évidence l'utilisation de la simulation lors de phases amont ou aval dans le processus de conception, avec un impact fort sur la collaboration des acteurs. Ainsi, en amont, on utilise la simulation dans des phases de Trade-off et de pré-dimensionnement, dans les phases aval elle intervient essentiellement en dimensionnement. Ces différentes pratiques sont décrites ci-dessous (pour plus d'information, se référer à l'Annexe 7).

Le Trade-off, phase très amont dans le processus de conception, définit les tendances du produit à réaliser. Elle permet également de définir les spécifications qui devront être prises en compte tout au long du processus de conception. A ce stade, la simulation numérique est utilisée pour simuler de façon globale un système complet, ou de grands ensembles de composants, alors même qu'aucune modélisation géométrique 3D n'existe.

Le pré-dimensionnement, ou conception préliminaire, est une phase de recherche de solution de conception du produit. Lors de cette phase, il est question de tester plusieurs solutions rapidement et de fixer les architectures qui seront développées en détail par la suite.

Le dimensionnement, ou conception détaillée, est une phase où les solutions de conception sont identifiées et il s'agit maintenant de modéliser finement les composants d'un système.

Les phases de Trade-off et de pré-dimensionnement nécessitent le recours à de nombreux modèles métiers simplifiés en parallèle. Chaque modèle métier encapsule un nombre limité de règles et de paramètres directeurs (les paramètres essentiels à la définition des premiers principes d'architectures) nécessaires à la recherche d'architecture de composants. Les acteurs doivent collaborer afin de s'échanger ces données, en assurer la cohérence d'un modèle à l'autre, et ce, malgré les très régulières modifications relatives à l'utilisation de boucles de simulation itérative.
Ainsi, on peut caractériser ces deux phases de la manière suivante :

- De nombreux modèles hétérogènes utilisés en parallèle utilisés pour définir les principes d'architectures
- Un nombre limité de paramètres et de règles devant être partagés (limité aux paramètres et règles directeurs)
- Les paramètres et les règles régulièrement modifiés
- La nécessité d'une forte collaboration

Par opposition, plus les utilisateurs avancent dans le processus de conception, plus le nombre de modèle diminue. La définition des composants évoluant vers un état figé, implique des modifications de paramètres moins fréquentes malgré le fait que leur nombre soit beaucoup plus important en raison du niveau de détail des modélisations.
Ainsi, on peut caractériser la phase de dimensionnement de la manière suivante :

- Moins de modèles métiers car les solutions sont fixées, et il s'agit de définir en détail les composants
- Plus de paramètres et de règles, mais moins de modifications
- Des activités plus segmentées et donc moins de collaborations

Ainsi, notre analyse, schématisée Figure 12, illustre les interactions entre utilisateurs, nécessaires à la conception d'un produit en fonction des phases du processus de conception. Cette figure met en évidence l'importance de la collaboration des acteurs autour des paramètres et des règles encapsulés dans les modèles métiers (particulièrement dans le cadre d'approches paramétriques et

variationnelles) dans les phases amont (Illustration chez EADS en Annexe 8). Nous observons également que l'intersection entre le niveau de collaboration et le nombre de paramètres s'effectue à la fin de la phase de pré-dimensionnement.

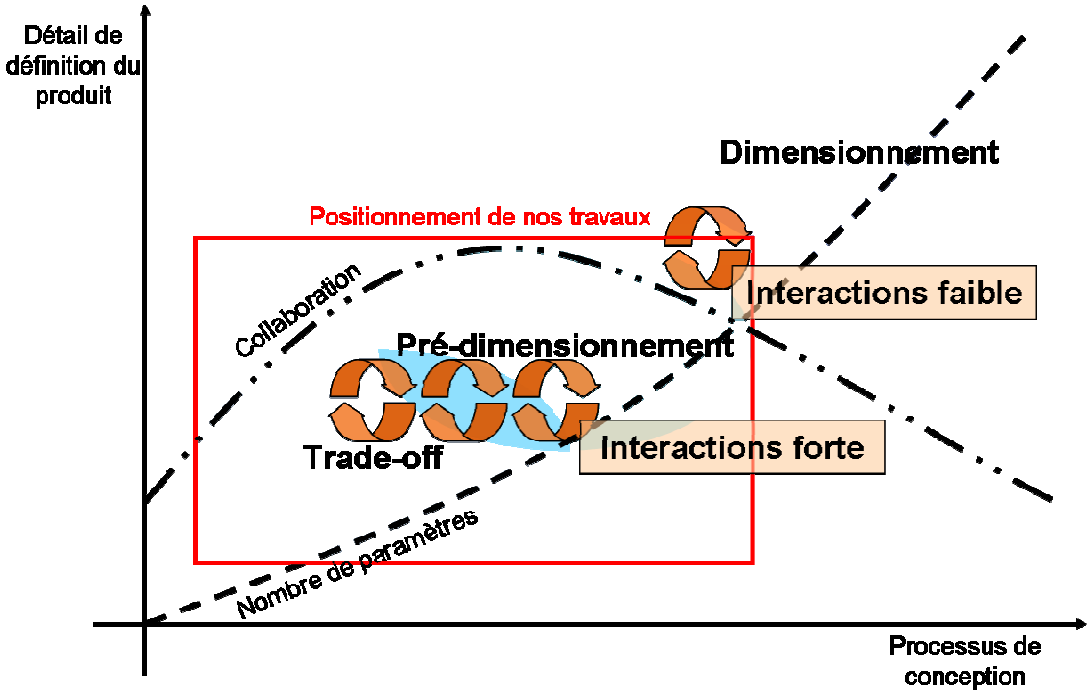


Figure 12 : interactions entre les phases du processus de conception dans lesquels la simulation a un rôle prépondérant

Dans ce contexte, l'utilisation de la simulation, tôt dans les phases de conception, fait émerger des besoins en termes de formalisation et de gestion [Schlenkrich et al., 2004] [Thomas et al., 04]. De plus, dans ces phases, le recours aux approches d'intégration CAO/Calculs (Figure 13) nécessite la mise en étroite relation des modélisations CAO et des modèles de calculs, et donc le partage et la collaboration des acteurs autour des paramètres et des règles à échanger selon plusieurs vues métiers et entre de nombreux outils hétérogènes.

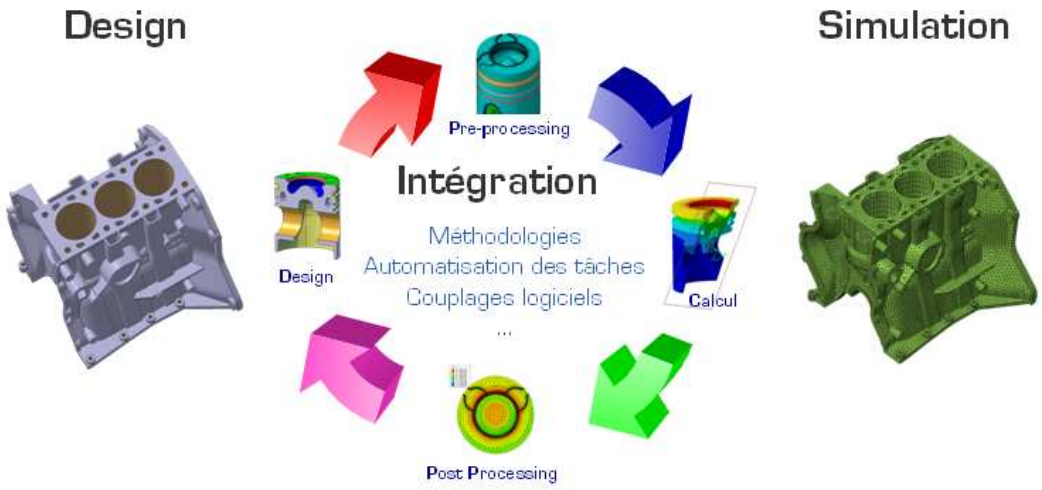


Figure 13: intégration des activités de simulation et de conception

Nous nous intéressons principalement aux phases amont du processus de conception. La simulation numérique y apparaît comme un outil fondamental, accentuant les problématiques de collaboration entre acteurs et d'interopérabilité entre modèles.
Ainsi, la gestion des paramètres et des règles métiers partagées entre un grand nombre de modèles métiers impliqués dans de nombreuses itérations, apparait comme un axe principal d'étude.

1.2.3 Les approches favorisant l'intégration de la simulation numérique

Compte-tenu du périmètre identifié précédemment et de l'importance de l'intégration de la simulation numérique dans le processus de conception, nous pouvons voir émerger trois grandes classes concernant les travaux de recherche autour de ce domaine technique [Baizet, 2004] [Aidi, 2007] [Beylier, 2007]. Ces approches visent globalement à favoriser l'exploitation des modèles numériques et le lien conception-simulation dans un contexte collaboratif [Veron, 2005] [Noël, 2003].Nous allons donc détailler différentes approches, dont la première concerne les outils d'aide à la simulation en conception, la seconde les systèmes de gestion de données, et la troisième les systèmes de gestion de connaissances.

1.2.3.1 Des outils d'aide à la simulation en conception

Ces travaux se concentrent sur le développement de méthodes et d'outils d'aide à la simulation en conception, apportant de nombreuses solutions pour supporter ces activités. Cette approche s'oriente principalement vers une automatisation des tâches d'analyses par les concepteurs, c'est-à-dire portant sur des méthodes d'idéalisation, de maillage, d'optimisation, etc., De nombreuses approches comme celles de [O'Bara et al., 2002], [Benamara, 2006], [Fine et al., 2002], ou encore [Shephard, 2001] [Shephard et al., 2004] (SBD : Simulation Based Design), le projet SALOME (1 et 2) [Salome, 2005], proposent d'automatiser le passage d'un modèle géométrique au modèle de calcul via des procédés de génération automatique (à partir de modèles idéalisés) de maillage ou de discrétisation de modèles d'analyses par éléments finis (parfois en utilisant de l'intelligence artificielle). Néanmoins, certains auteurs [Kurowski, 1995] remarquent que d'importantes erreurs sont générées suite à l'accroissement de l'automatisation du passage à une géométrie idéalisée pour le calcul. De plus, ces recherches se tournent principalement vers l'automatisation des processus d'analyse dans les outils, et réduisent, par là même, l'importance des acteurs dans ces processus. [Eckard, 2000] démontre ainsi que, pour améliorer le temps de développement d'un produit, il devient indispensable d'intégrer la simulation au plus tôt dans le processus de conception, et de faire participer les concepteurs dans les activités de calcul. Il faut fournir des connaissances aux acteurs du calcul pour pouvoir maîtriser les hypothèses utilisées lors de la génération des modèles de simulation à partir de modèles de conception. De plus, les outils actuels de simulation numérique fonctionnent difficilement dans les phases amont du processus de conception (parce qu'ils sont très liés aux éléments finis, et par conséquent nécessitent un modèle CAO déjà assez avancé). Certaines évolutions sont donc nécessaires pour améliorer les réponses de simulation en présence de fortes incertitudes ou variations de paramètres produit [Prudhomme et al., 2001].

1.2.3.2 Les systèmes de gestion de données

D'autres travaux concernent des outils qualifiés de PDM qui réunissent des SGDT et des SIP (Systèmes d'Information Produit). Ces outils permettent le partage des données et le contrôle des flux d'informations, le partage et la documentation des processus de travail, l'assurance de la mise à jour des modifications et de la garantie de la non redondance des données. Ce sont aujourd'hui des systèmes répandus au sein des organisations et des entreprises facilitant la communication dans les cycles de conception. Néanmoins, limités aux seuls processus et informations du métier de la conception, ils ne peuvent prendre en compte les données d'autres métiers complémentaires, dont les données de simulation numérique dans notre cas. Le partage des données, dans le cadre de processus multidisciplinaires, introduit des problèmes de représentation des données et requiert l'utilisation de structures de données adaptées aux besoins [Anderl et al., 2002], d'où l'émergence de systèmes SDM (Simulation Data Management) [Charles, 2005]. Néanmoins, la prise en compte des connaissances de simulation pose des questions. Actuellement, une conclusion s'impose : relativement à la définition proposée par [Prudhomme et al., 2001], ou celle de [Zarifian, 2002], les connaissances ne peuvent être gérées qu'au travers de la gestion d'objets de connaissances, dans lesquels s'inscrivent les données.
Il existe par conséquent divers environnements de travail supportant l'échange des données produit issus de la conception et/ou de la simulation. Cependant, les données conservées sont toujours

relatives au produit, et il est rare qu'une réutilisation soit possible dans d'autres contextes. De plus, ces systèmes n'ont pas pour objectif de gérer des connaissances mais seulement des données produits, *« de ce fait ils ne peuvent pas garantir la livraison d'une connaissance à un acteur pour l'aider dans sa tâche »* [Baizet, 2004]. Nous reviendrons sur la gestion des données et des informations réalisées par les PDM et les SDM dans le chapitre 2.

1.2.3.3 Les systèmes de gestion des connaissances

Finalement, certains travaux sur les systèmes de gestion des connaissances s'intéressent à l'amélioration de l'intégration de la simulation dans la conception en favorisant la coopération entre le concepteur et l'analyste (travaux SG3C) [Troussier, 1999]. Dans ces travaux, le concept de "structuration spécifique" et de "cas d'école" permettent de mettre en évidence les connaissances dites génériques et réutilisables dans d'autres projets. Cette méthode diffère des précédentes puisqu'elle tente d'intégrer la simulation dans les phases de conception en passant par la définition de méthodes de gestion des connaissances pour la simulation.

Nous retiendrons qu'il semble aujourd'hui nécessaire de mieux prendre en compte les connaissances de la simulation et son intégration dans le processus de conception. Dans ce contexte, les acteurs du processus de conception tiennent un rôle prépondérant. Une approche complémentaire et transverse aux trois thèmes évoqués semble intéressante.

1.3 Conclusion

1.3.1 Rappel et analyse

L'analyse, menée tout au long de ce chapitre, a permis de dégager plusieurs points sur lesquels nous positionnons ces travaux de recherche.

- Le flux d'informations partagées entre divers domaines tout au long du processus de conception. Nous nous intéressons plus particulièrement au cycle de vie du produit et des informations concernant la définition du produit virtuel.
- L'ingénierie intégrée et collaborative, en tenant notamment compte des notions d'itération et de convergence. La collaboration des acteurs d'un projet est cruciale, et la cohérence des données, informations et des connaissances utilisées doit être assurée.
- Le contexte d'ingénierie routinière et de modélisation paramétrique.
- La capitalisation et la disponibilité des paramètres et des règles dans les modèles métiers à la source des problèmes de collaborations entre les acteurs du processus de conception.
- L'importance de la simulation dans le processus de conception.
- L'hétérogénéité des activités de la simulation aggravant les problèmes de collaboration entre les acteurs et d'interopérabilité entre les modèles.
- L'importance des phases amont (Trade-off et pré-dimensionnement) couplées à l'intégration de la simulation numérique dans le processus de conception.
- Enfin, un positionnement sur les approches gestion des connaissances pour favoriser l'intégration de la simulation numérique. Les acteurs du processus de conception doivent être au centre de la collaboration.

L'évolution du processus de conception, notamment le développement des processus intégrés et collaboratifs, ainsi que l'évolution des pratiques de la simulation numérique (particulièrement dans les phases amont pour piloter la conception), engendre la manipulation de nombreux modèles métiers hétérogènes. La richesse des applications métiers utilisées au cours du développement d'un produit pose nécessairement des difficultés en termes d'interopérabilité et de collaboration, dans la mesure où chaque application métier encapsule des paramètres et des règles pilotant le produit qui, de part le cloisonnement de ces applications, sont rendus invisibles pour la plupart des acteurs de la conception [Prat et al., 2005]. Ce constat entraîne une multiplicité des paramètres représentant la

même spécification fonctionnelle et pose des problèmes lors de la modification de règles associant ces paramètres.
Cette problématique est de nature à perturber le processus de conception, favorisant les erreurs et la perte de temps. En effet, il n'existe pas aujourd'hui de plateforme de partage et de communication au niveau des paramètres et des règles, obligeant les ingénieurs à échanger par téléphone, au cours de réunion, par mail etc., ne garantissant pas la traçabilité et la cohérence des données techniques. En effet, [Rodiguez et al., 2002] définit que, dans le cadre d'une organisation ne disposant pas de méthodes adaptées à la gestion des données ou des informations techniques, le téléphone est la composante majeure des moyens de communication (46%), suivi par les outils de messagerie électronique (36%), puis enfin par les déplacements et les courriers (18%).

Dans ce contexte, le couplage entre le processus de conception et le domaine de la gestion fait partie des enjeux scientifiques et industriels de cette dernière décennie. Ainsi, le consortium IPPOP (Intégration Produit – Processus – Organisation pour l'amélioration de la Performance en ingénierie) [Roucoules et al.,2006], mettant en œuvre le modèle produit PPO (Produit Process Organisation), avait relevé l'intérêt d'intégrer la vue Process, et plus particulièrement la vue Organisationnelle à la classique vue Produit. Plus récemment, le projet ATLAS propose d'associer un environnement de conception avec un environnement de gestion de projet [Aldanondo et al., 2010]. Les connaissances caractérisant les interdépendances entre les deux environnements permettront alors de propager les décisions ou choix d'un environnement vers l'autre.
Ces travaux opèrent à un niveau macroscopique pour la conception et la gestion de projet. A un niveau plus microscopique, c'est-à-dire entre plusieurs modèles métiers (CAO/Calcul), nous observons des problématiques équivalentes liées au couplage entre les modèles et la gestion des connaissances [Burr et al., 2005]. Les connaissances doivent transiter et se propager d'un modèle métier à l'autre, afin d'assurer la convergence des acteurs du projet vers une solution cohérente.

Ainsi, la Figure 14 illustre, sur un projet de conception de moteur automobile, ces problématiques de partage et de collaboration autour des connaissances métiers. Elle représente quatre modèles métiers utilisés en parallèle, chacun encapsulant un ensemble de paramètres et de règles partagés par d'autres modèles métiers. Malgré les itérations de modifications de paramètres et de règles au sein de chaque modèle, la cohérence doit systématiquement être maintenue et ce, quel que soit le moment où les modèles sont utilisés (exemple appliqué au cas automobile en Annexe 1).

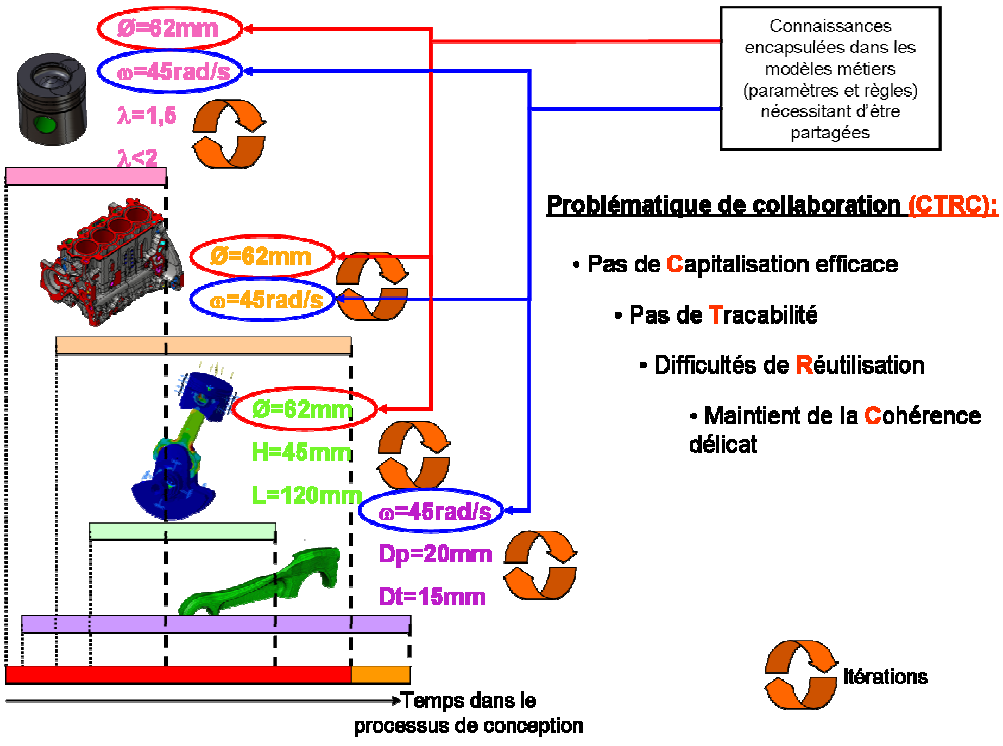


Figure 14 : illustration des problèmes de collaboration autour des connaissances de granularité fine

Par exemple, le paramètre Ø est utilisé en parallèle dans les modèles piston, carter cylindres et attelage mobile. Dans chaque modèle, les valeurs prises par ce paramètre doivent être cohérentes et au bon niveau de maturité, et ce, malgré les nombreuses itérations réalisées par les utilisateurs. Dans le cas contraire, des calculs, parfois très longs, peuvent être lancés sur de mauvaises valeurs de paramètres entre modèles, entrainant des erreurs et des pertes de temps.
Ainsi, au travers de l'illustration proposée par cette figure et des constats réalisés tout au long de ce chapitre, nous proposons une analyse basée sur quatre thèmes, baptisée **CTRC - C**apitalisation, **T**raçabilité, **R**éutilisation, **C**ohérence, regroupant en cascade plusieurs facteurs à l'origine de pertes de temps et d'erreurs dans le processus de conception. Ces thèmes sont détaillés dans le Tableau 2 ci-dessous:

La Capitalisation des paramètres et des règles dans les modèles métiers : Chaque modèle métier fonctionne comme une base de données de ses données techniques, indépendamment des autres modèles métiers. Ce fonctionnement entraine une multiplicité des paramètres et des règles présents dans chaque modèle métier, et par conséquent de potentiels conflits (doublons, paramètres identiques mais portant des noms différent) [Tang et al., 2004]. En plus, du fait que les paramètres et les règles soient inaccessibles en dehors des applications métiers, l'hétérogénéité des outils limite, voire même, rend impossible la communication entre les modèles. De plus, le cycle de vie des données techniques est lié au cycle de vie du modèle métier. Or, pourquoi un paramètre deviendrait obsolète quand un modèle devient obsolète ? Les paramètres ont vocation à posséder un niveau d'abstraction et de généricité plus élevés que les modèles ou les applications dans lesquels ils sont utilisés, ainsi que leur propre cycle de vie. Cette réflexion nous amène à nous poser des questions quant à la nature de ces paramètres et de ces règles : n'est-il pas intéressant de les considérer comme de l'information, voire de la connaissance ?
La Traçabilité des paramètres et des règles : Dans la mesure où la capitalisation des paramètres et des règles n'est pas efficace, il est aujourd'hui impossible de tracer finement l'utilisation des données. En effet, il est impossible de savoir si tel paramètre, ou telle règle, est utilisé par tel concepteur, à tel moment, pour telle activité, avec telle modification. En cas d'incohérence sur l'utilisation des paramètres et des règles entre plusieurs modèles, il est difficile de remonter rapidement à la source de l'erreur.
La Réutilisation des paramètres et des règles entre modèles et d'un projet à l'autre : Compte-tenu des deux points soulevés précédemment, la réutilisation des données lors d'un projet de conception, ou entre différents projets, est complexe. En effet, il est nécessaire de tenir compte du niveau de maturité et de l'évolution des données, afin de limiter les incohérences et les processus routiniers.
La Cohérence des paramètres d'un modèle à l'autre : Enfin, la cohérence des données partagées par plusieurs modèles en parallèle est aujourd'hui cruciale pour continuer à rationaliser et optimiser le processus de conception. Il apparaît, au sein des bureaux d'études et de calculs, de nombreux problèmes de non-qualité ou de faible productivité en conception-simulation. Ces difficultés peuvent être liées à des manques ou des absences de synchronisation des configurations de paramètres, de règles métier et voire même des jeux de valeurs de ces paramètres (instances de paramètres) ; paramètres en général communs à plusieurs métiers. Il nous semble que cette notion de configuration de paramètres et de règles (groupe de paramètres et de règles en tenant compte de leurs versions, maturités, des différentes utilisations) soit un élément essentiel dans la problématique de cohérence.

Tableau 2 : détails des quatre thèmes du CTRC

L'analyse de ces thèmes, ainsi que les problèmes soulevés dans le contexte de la conception de produit, constituent des éléments qui semblent cruciaux pour continuer à optimiser le processus de conception. Nous en tenons particulièrement compte dans notre état de l'art et dans notre proposition.

1.3.2 Synthèse du contexte de recherche

Le positionnement de ces travaux de recherche concerne la collaboration des acteurs autour des paramètres et des règles dans le processus de conception. Nous nous focalisons particulièrement sur le contexte d'ingénierie routinière et paramétrique dans la cadre des processus collaboratifs.
La simulation numérique et son intégration dans le processus de conception illustrent parfaitement les difficultés rencontrées en termes de capitalisation, de traçabilité, de réutilisation et de cohérence des connaissances manipulées. Elles mettent en œuvre un ensemble d'activités hétérogènes permettant aux concepteurs de mieux concevoir, particulièrement dans les phases amont du processus de conception, caractérisant l'ingénierie hautement productive et collaborative.

Le chapitre 2 propose une analyse des outils et des méthodes de gestion de données s'accompagnant d'une réflexion sur la notion de connaissance relativement à notre contexte de recherche. Cette réflexion nous a permis de positionner notre travail par rapport aux besoins non pris en compte par les solutions existantes.

Chapitre 2. La gestion des données et des connaissances pour la conception et la simulation du produit

Dans le cadre de ce second chapitre, nous proposons un état de l'art des outils et des approches relatives à la gestion des données, informations et connaissances pour la conception de produit et l'utilisation de la simulation numérique.
En effet, le contexte actuel et la dynamique du marché font remonter une véritable problématique industrielle autour de la capitalisation et de la réutilisation des connaissances en phase de conception et de développement de produits [El Khalkhali, 2002]. En effet, les projets industriels de grande échelle sont menés en entreprise étendue avec le besoin de coordonner différentes organisations et équipes sur l'intégralité du cycle de vie du produit. Chaque entreprise génère et manipule des données, informations et connaissances sur leurs produits, sur les procédés de fabrication ainsi que sur les modèles de simulation associés. C'est pourquoi, de plus en plus d'outils à "base de données", voire à "base de connaissances", connexes au système d'information émergent, de type PLM (Product Lifecycle Management), PDM (Product Data Management), SDM (Simulation Data Management), afin d'avoir une véritable gestion du patrimoine intellectuel de l'entreprise. En effet, le manque d'interopérabilité entre outils de conception numérique et de simulation constitue un véritable obstacle au déploiement de la stratégie PLM. La dispersion de l'information au travers de systèmes hétérogènes nécessite de nouvelles solutions pour préserver la cohérence du système d'information de l'entreprise [Roucoules et al. 2006].

2.1 La gestion des données et des informations du produit dans le processus de conception

Dans le cadre des processus d'ingénierie intégrés et collaboratifs, la mise en place de systèmes visant à gérer les données, informations et connaissances, est devenue un enjeu crucial pour rationaliser et optimiser le processus de conception.
Ainsi, dans cette section, nous proposons une analyse des solutions méthodologiques et logicielles ayant conduit au développement des SGDT – Systèmes de Gestion de Données Techniques (PDM – Product Data Management et SDM – Simulation data Management), et de leurs évolutions technologiques vers les PLM (Product Lifecycle Management) [Eynard, 2005].

2.1.1 Les SGDT

Le développement des approches d'ingénierie intégrées et collaboratives entraine nécessairement des besoins en termes d'échanges et de communication entre les acteurs des activités du processus de conception. Ces approches s'appuient fortement sur l'essor des technologies de l'information et de la communication (TIC). Elles ont permis le développement des systèmes de gestion documentaires basés sur les informations techniques donnant naissance au SGDT (Systèmes de Gestion de Données Techniques) [Maurino, 1995], [Randoing, 1995]. Utilisés dans un contexte de partage, de gestion et stockage organisé des données techniques liées à la définition du produit et de son processus de production, les SDGT sont communément appelés PDM[2] (Product Data Management) [Liu et al., 2001].
Les PDM, dédiés au domaine de la conception de produits dans l'industrie, sont aujourd'hui les outils de gestion de données les plus utilisés, Ils permettent de gérer les données du produit tout au long du processus de conception et particulièrement, celles relatives à la modélisation géométrique du produit. En conséquence, le PDM propose généralement une décomposition structurelle du produit communément appelée DMU (Digital Mock Up) dans laquelle s'inscrivent les données. L'objectif majeur étant de permettre l'accès à la bonne donnée au bon niveau de maturité, par la bonne personne, au bon moment, favorisant ainsi la collaboration entre les acteurs.
Aujourd'hui, les PDM adressent les vues produit/process/projet et offrent des possibilités étendues comme la planification et la gestion des ressources dans le contexte du processus de conception. Ils répondent aux besoins industriels et aux plateformes d'ingénierie visant à gérer [Peltonen, 2000] [Helms, 2002]:

- le volume de données décrivant le produit et les process associés ;
- les versions des données ;
- les utilisateurs impliqués et leurs accès aux données (gestion des rôles et comptes) ;
- les différents états des données et leurs changements;
- l'archivage des données ;
- l'intégrité des données ;
- la sécurité et la confidentialité des données ;
- la traçabilité des données ;
- l'approbation des données.

Actuellement, avec le développement des technologies WEB, le PDM s'étend à l'ensemble des infrastructures dans un contexte d'entreprise étendue, renforçant les liens de type donneur d'ordre / sous traitant ; on parle alors de "cPDM" (Collaborative Product Definition Management). Ainsi, il peut être présenté comme la première étape vers la gestion de l'intégralité du cycle des produits [Sung et Park, 2007] [Christmas, 2001], donnant naissance à l'acronyme PLM (Product Lifecycle Management) [Rouibah et al., 2007].

[2] PDM : réunion des SGDT et des SIP (Systèmes d'Information Produit)

Néanmoins, limités aux seuls processus et informations du métier de la conception (CAO), les PDM traditionnels ne permettent pas de prendre en compte les données d'autres métiers complémentaires, dont les données issues de la simulation numérique. En effet, le partage des données dans le cadre de processus multidisciplinaires introduit des problèmes de représentations, de processus, de gestion, et requiert l'utilisation de structures de données adaptées.

Ainsi, nous proposons ci-dessous, quelques points pour-lesquels les PDM ne semblent pas être adaptés aux activités de la simulation numérique :

- Les données de simulation ont un cycle de vie spécifique différent du cycle de vie relatif à la conception CAO du produit. Or, un PDM n'est pas capable de gérer simultanément ces deux cycles de façon cohérente.
- La taille des fichiers de simulation est beaucoup plus importante que celle d'un modèle CAO (plusieurs Go chaque fois). De plus, il y a de nombreux types différents de fichiers de calcul.
- La simulation implique l'utilisation d'un workflow spécifique, qui n'existe pas vraiment en CAO. Ce workflow implique une gestion et un partage des données particulier (certaines informations peuvent être considérées comme une entrée ou une sortie dans plusieurs contextes entre différents calculs). Ainsi, il est nécessaire de mettre en place une gestion des échanges de données entre fichiers.
- Il y a beaucoup de diversité dans les types d'objets de calcul à gérer (maillages, données techniques, documents, etc.).
- Les données de simulation ne peuvent pas être facilement structurées dans la DMU. Une organisation spécifique est nécessaire, surtout dans les phases amont du processus de conception où les modèles de simulation peuvent être utilisés, alors qu'aucun des modèles CAO n'est encore connu.
- Les données de simulation doivent être liées à une gamme hétérogène d'outils de simulation.

Tout pousse à penser qu'il est aujourd'hui difficile, avec les outils traditionnels, de gérer l'ensemble des données et des informations du produit/process/projet au sein d'une même structure avec des vues spécifiques. Cela conduit souvent à gérer plusieurs maquettes numériques selon qu'elles se rapportent à la définition, à la collaboration, à la simulation, ou même aux procédés de fabrication ou de maintenance. La collecte des informations contenues dans différentes structures pour l'exécution d'une tâche particulière peut devenir complexe, ainsi que la synchronisation des évolutions.
Ainsi, la prise en compte de différentes vues métiers, comme celles issues des activités de la simulation, nécessite le recours à de nouveaux types de solutions complémentaires aux outils PDM.

2.1.2 L'intégration et la gestion des données de la simulation numérique

Au regard du développement des activités de la simulation numérique dans le processus de conception et du volume de données, continuellement croissant à prendre en compte, il est aujourd'hui nécessaire de se tourner vers de nouvelles solutions.
En effet, la maitrise des données de simulation est un enjeu essentiel visant à favoriser son intégration dans le processus de conception. Plusieurs projets s'intéressent à ces approches et proposent des solutions complémentaires aux traditionnels outils PDM [Krastel et al. 02] [Schlenkrich et al. 2004].
Ainsi, [Shaphard et al., 2004] proposent la démarche SEED - Simulation Environment for Engineering Design (approche de SBD - Simulation Based Design) et rappellent la nécessité de gérer spécifiquement les données de simulation. Ainsi, SEED est une démarche d'intégration de la simulation dans le processus de conception, avec une prise en compte spécifique des activités de simulation et des données manipulées. Elle est plus particulièrement basée sur quatre composantes reliant les PDM d'une part, et l'intégration CAO/Calcul, d'autre part. La Figure 15 illustre les interactions entre les composants:

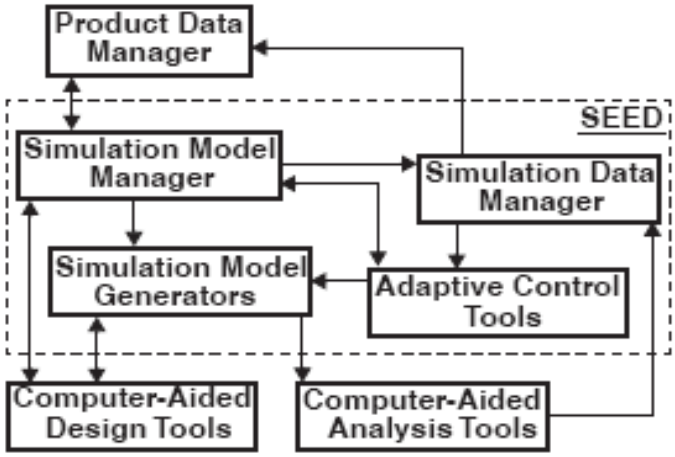


Figure 15 : simulation Environment of Engineering Design [Shephard et al., 2004]

L'approche se focalise principalement sur le contrôle du processus de simulation intégrant divers outils d'analyse, la circulation des flux d'informations, ainsi que sur la définition des modèles de calculs relativement aux informations stockées. Ces approches, qualifiées de SBD – Simulation Based Design, se concrétisent généralement sous forme de boites noires interconnectées avec les outils métiers, les concepteurs n'ayant que peu de contrôle sur le contenu de la boite.
D'autres travaux se focalisent plus particulièrement sur la gestion et surtout sur la collaboration des acteurs d'un projet autour des données de la simulation numérique, en tenant compte des lacunes des systèmes PDM, donnant naissance aux outils qualifiés d'EGDS (Environnement de Gestion de Données de Simulation), aussi appelés SDM (Simulation data Management) [Eynard, et al., 2005]. L'objectif est véritablement de définir un outil se positionnant sur un même plan conceptuel et fonctionnel que les PDM, mais dédié à la simulation numérique, validant la complémentarité des deux outils dans le processus de conception.
Il existe de nombreuses recherches autour de ces approches [Simdat, 2005], dont celle de [Charles, 2005] qui propose une solution particulièrement complète et intéressante. Il définit les concepts nécessaires pour la spécification d'un environnement dédié à la gestion des données de simulation numérique dans une approche PLM [Bouras et al., 2005], en conformité avec les problématiques d'échange de données en formats neutres (format STEP [Arbouy et al., 1994] [Männisto et al., 1998]). Ainsi, il propose un environnement de gestion des données de simulation basé sur un modèle de données générique permettant de combler les lacunes en termes de "fonctionnalités" et "d'interopérabilité". Concernant ses "fonctionnalités", le système apporte des solutions aux problèmes de structuration des données de calculs et la prise en considération des boucles de simulation. En outre, ce système intègre les concepts de gestion alternative, de gestion du cycle vie et des processus. Pour résoudre les problèmes "d'interopérabilité" Charles propose le SDM schéma. Ce modèle est basé sur les caractéristiques de l'AP209 STEP, du PDM schéma [Kim et al., 2005], et des éléments liés à la simulation, tels que les boucles et la gestion alternative (Annexe 9).
Dans un contexte industriel, les systèmes SDM sont utilisés par des personnes ayant des expériences différentes dans le domaine de simulation numérique. Cela implique l'intégration des principes de gestion de la connaissance au sein du concept de SDM.

2.1.3 Bilan sur les outils de gestion de données et d'informations

Basés sur une gestion documentaire, les outils traditionnels de gestion de données possèdent des lacunes, notamment en ce qui concerne la gestion, l'accès et l'utilisation des paramètres et des règles de conception encapsulés dans les modèles métiers.
En effet, en considérant le cas des approches d'intégration CAO/Calcul, on constate que leur mise en place nécessite de tisser des liens étroits et stables entre les modèles métiers. Si les SGDT et les EGDS permettent l'association des fichiers entre eux, cette association est insuffisante pour lier des éléments internes à un fichier à des éléments internes d'un second fichier. Ce n'est pas nécessairement une incapacité technique mais sans doute un problème de niveau de détail [Noël et al., 2005].

Ainsi, ces lacunes sont de nature à engendrer des problèmes de collaboration, les difficultés rencontrées sont alors liées à l'absence de méthode spécifique de partage d'informations, voire de connaissances entre experts. En effet, dans la mesure où les solutions techniques régulièrement adoptées consistent à gérer les documents issus d'activités en faisant abstraction de leur contenu (illustration au travers des notions de check-in, check-out), l'accès à l'information nécessite, non seulement la réservation du document entier, mais également de disposer des logiciels métiers capable de les exploiter.
Il semble alors nécessaire d'enrichir les méthodes d'ingénierie collaborative avec une démarche facilitant la gestion des paramètres et des règles de conception au sein d'une plateforme PLM [Ducellier et al., 2006], mais aussi la prise en compte des connaissances. C'est-à-dire la gestion et l'utilisation des paramètres et des règles métiers dans plusieurs activités simultanées du processus d'ingénierie, de manière cohérente et collaborative. En effet, les systèmes actuels de gestion de données, dans lesquels s'insèrent les environnements intégrés, tendent vers le développement de structures adaptées aux données, sans forcément prendre en compte la notion de connaissances [Aidi, 2007]. Nous revenons sur la notion de connaissance dans la section 2.3 de ce chapitre.
Dans ce contexte, l'intégration des outils de gestion de données, au sein d'une démarche plus globale de plateforme PLM, vise à élargir le spectre des méthodologies à mettre en place afin de favoriser la collaboration et la gestion des informations tout au long du cycle de vie du produit. Ainsi, le PLM peut être vu comme une évolution logique des SGDT [Eynard, 2005].

2.2 L'intégration des systèmes de gestion de données dans l'approche PLM

Le PLM peut être considéré comme une approche stratégique qui utilise un ensemble cohérent de solutions d'affaires intégrées, supportant la création, la gestion, la dissémination collaborative et l'utilisation de l'information de définition du produit. Il a pour principal objectif de fédérer l'ensemble des donnés et des processus tout au long du cycle de vie du produit[3] [Amann, 2002] [Stark et al., 2004] [Säaksvuori et al., 2005].
Cette stratégie tente de répondre aux problématiques de disparité des divers systèmes d'informations parcourant le cycle de vie des produits [Garetti et al., 2007], traduisant les besoins en intégration et en cohérence parmi la quantité considérable de données et de structures de données embarquées dans ces systèmes [Karcher et al., 2003] [Vareilles , 2005] [Bergsjö et al., 2007]. Ces préoccupations font également l'objets de travaux portant sur les questions d'interopérabilité [Merlo, 2009] [Paviot et al., 2009], d'échange de données [Eynard, 2005], de déploiement/implémentation [Eynard et al., 2004] [Le Duigou, 2010], de capitalisation/réutilisation des connaissances [Ducellier, 2008] [Gomes, 2008] (Figure 16) et gestion des processus métiers [Monticolo, 2008] à différents niveaux de l'entreprise [Sudarsan et al., 2005].

[3] Cycle de vie du produit : des phases de définition du besoin jusqu' au retrait et au recyclage du produit

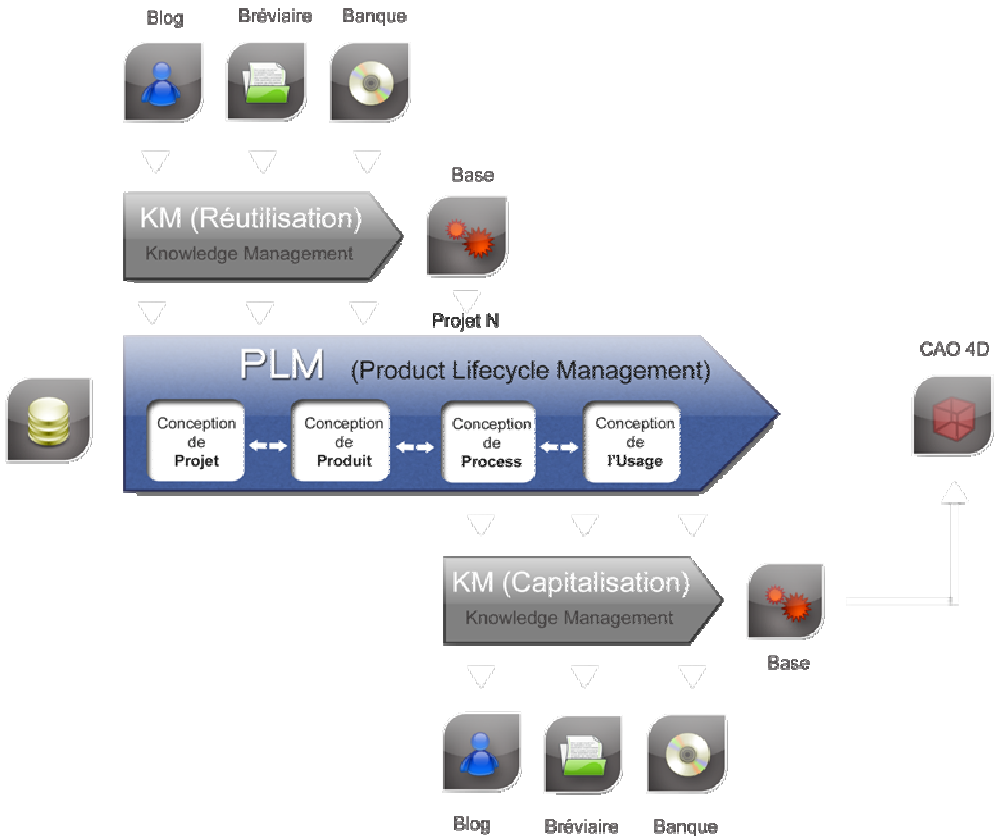


Figure 16 : la réutilisation des connaissances dans le PLM [Gomes, 2008]

Dans ce cadre, le PLM suppose la mise en place d'un ensemble de fonctions [Helms, 2002] (décrites dans [Ducellier, 2008]), accessibles via l'intégration et l'utilisation d'outils tels que les PDM, SDM, mais aussi l'utilisation de format d'échange neutre, de modèles produits, visant à favoriser la collaboration entre les acteurs et l'interopérabilité entre les modèles métiers.

2.2.1 Les modèles à vocation normative

Afin de favoriser la gestion des données et des informations durant l'ensemble du cycle de vie du produit, le recours aux modèles à vocation normative a permis de faciliter les échanges entre des applications métiers hétérogènes.
Ainsi, initialement issus du monde de la CAO, les éditeurs de solutions PLM se sont naturellement basés sur la nomenclature des pièces d'un composant ou du produit pour traiter des problématiques de gestion, de versions et de traçabilité qui surviennent au cours des échanges de données géométriques. Très tôt, la multiplicité des logiciels de modélisation, ainsi que la multiplicité des modèles, a imposé la mise en place de technologies d'échanges des données via l'utilisation de formats neutres comme l'IGES [IGES, 1988], ou encore le VDA [VDA, 1987]. Néanmoins, le périmètre de ces applications ne dépassait que rarement le bureau d'étude et sa gestion des plans, ainsi le PLM était peu outillé pour formaliser des informations techniques qui n'étaient pas directement liées à la géométrie d'une pièce ou à sa spécification.
Dans ce contexte, le développement du format STEP (Standard for the Exchange of Product model data) [ISO 10303-209:2001] a contribué à élargir le champ des types de données qu'il est possible de manipuler et d'échanger entre l'ensemble des applications métiers tout au long du cycle de vie du produit (Annexe 10). Il permet notamment, de définir une représentation non ambiguë des données du produit, interprétable par tout système informatique. Ainsi, un standard pour les PDM sera même développé par la communauté STEP, baptisé PDM schema. Il a pour vocation d'offrir un modèle unique de données supportant celles gérées dans le PDM. Plus récemment, le SDM schema définit un modèle de données commun pour la prise en compte des données de simulation, comme nous l'avons vu dans la section 2.1.2. Le format STEP permet alors d'échanger des

données d'une granularité plus fine que les fichiers eux-mêmes et ainsi, favoriser l'interopérabilité entre application métiers.

Toutefois, il semble que le format STEP n'offre pas totale satisfaction face aux problèmes d'utilisation collaborative des données. En effet, le recours à ce formalisme nécessite la mise en place de protocoles d'échanges complexes et lourds, mais néanmoins indispensables aux transformations entre modèles métiers. Chaque échange est alors lent et des informations sont souvent perdues ou dupliquées [Noël et al., 2007]. De plus, STEP ne tient pas compte du facteur de pluridisciplinarité propre au domaine de la conception intégrée. En effet, dans la mesure où chaque concepteur représente le produit différemment, il est nécessaire de pouvoir décliner selon autant de points de vue les informations autour d'une représentation cohérente du produit [El Khalkhali, 2002]. En outre, STEP ne peut être considéré que comme un formalisme d'échange, dans la mesure où il fournit un modèle statique du produit. Il n'est donc pas en mesure de gérer dynamiquement les informations qu'il manipule afin de prendre en compte, par exemple, la notion de projet, d'organisation, de cycle de vie des informations, etc.
Certains auteurs se sont alors tournés vers la définition de modèles produits, visant à insérer dans un unique modèle tous les types d'informations susceptibles d'émerger au cours de la conception, étendant ainsi les modèles métiers à des informations techniques et technologiques issues des différentes expertises [Noël, 2003]. Ces modèles permettent de centraliser et de gérer finement les informations techniques selon différents points de vue, indépendamment des applications métiers.

2.2.2 Les modèles produits

Un modèle produit décrit les différentes connaissances relatives à un produit [Tollenaere, 1994]. Cette génération de concepts de modélisations a pour but de soutenir l'approche PLM et d'offrir un support à la gestion des données, informations et connaissances du produit (en permettant d'assurer la capitalisation, réutilisation, communication, structuration, etc.) durant l'ensemble des phases de son cycle de vie.
Tout au long du processus de conception, les concepteurs manipulent divers modèles et données associés, mais sont souvent submergés par un flux constant, volumineux et hétérogène d'informations, qu'il faut filtrer, structurer, intégrer, réutiliser et partager. C'est en ce sens que des modèles produits sont définis pour soutenir la collaboration et fournir un environnement d'intégration aux nombreux outils, permettant ainsi de gérer l'hétérogénéité, favoriser la collaboration, maintenir la cohérence, piloter les échanges.
Dans ce cadre, [Demoly, 2010] propose une typologie des modèles produits, parmi eux :

- FBS (Function-Behavior-Structure) [Gero, 1990],
- MV (Multi-Vues) [Tichkiewitch, 1996],
- MD-MV (Multi-Domaines Multi-Vues) [Gomes et al., 2002],
- CPM 1 et 2 (Core Product Model), et ses extensions OAM (Open Assembly Model), DAIM (Design Analysis integration Model), etc., [Fenves, 2001] [Fenves et al., 2004] [Fenves et al., 2008] [Sudarsan et al., 2005],
- PPO (Product Process Organisation) [Noël et al., 2007],
- FBS-PPRE (Process, Product, Resources, External Effects) [Labrousse et al., 2008],
- etc.

Les modèles produits permettent une gestion dynamique et organisée de la structure produit selon différents points de vue. Ils sont généralement formalisés via l'utilisation de langages orientés objets de type UML. Dans la suite de cette section, nous détaillons les grands principes couramment mis en pratique dans les modèles produits.

2.2.2.1 La décomposition et l'approche multi-vues

Il existe plusieurs modèles de structuration des données développés dans le domaine de la conception de produits. L'objectif est de modéliser l'acte de conception et de développer un aspect coopératif entre les acteurs des différents domaines, susceptibles d'intervenir dans la conception du produit et son cycle de vie [Gomes et al., 2004]. Les concepts modernes de modélisations, initiés

par les travaux de [Tollenaere, 1994], puis [Tichkiewitch et al., 1995] [Tichkiewitch, 1996], proposent de répondre à une problématique de mise en commun et de gestion des données définissant le produit. Par la suite, de nombreux auteurs ont repris et enrichi ces modèles pour répondre à des contextes particuliers et permettre de rassembler l'ensemble des données produit, et formaliser les relations qui pouvaient exister entre elles [Yvars, 2001] [Roucoules, 2007]. Ces concepts de modélisations sont définis autour de la structuration des éléments (composants) d'un système et de ses liens entre les composants. Les relations formalisent alors ces liens autour de trois mécanismes (Figure 17) :

- La décomposition (décomposition de composants en autres composants)
- La substitution (remplacer une relation par un ensemble de composants, liens, relations)
- La représentation multi-vues, issue des travaux de [Chapa, 1997], *« cela permet à un composant de pouvoir être multi-décomposé à des niveaux granulaires progressifs, respectivement à la vue que les experts souhaitent avoir sur le produit »* [Roucoules, 2007] : vue ossature, vue géométrique, vue usinage, vue assemblage, etc.

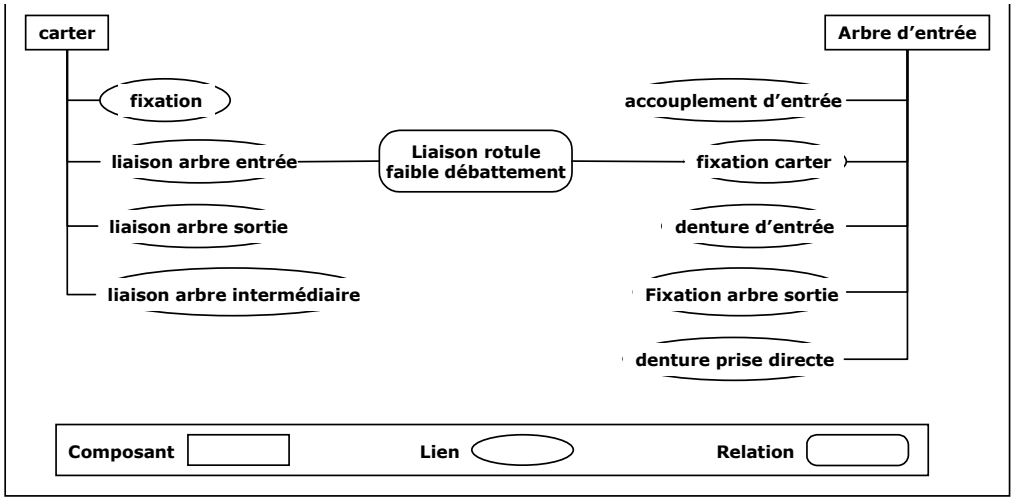


Figure 17 : extrait du modèle produit représentant les liens et les relations entre deux composants d'une boîte de vitesse [Tichkiewitch, 1996]

Il existe aussi de nombreux modèles, comme PPO, basé sur la modélisation du produit et la prise en compte de l'organisation (couplage entre produit, processus et organisation), ou encore le modèle systémique MD-MV basé sur des modèles de conception découpés en mondes et en domaines en interaction, proposé par [Gomes, 2002], s'appuyant sur les travaux de [Le Moigne, 1999] (pour l'approche systémique) et les travaux de [Suh, 1999] et [Sohlenius , 1992] (pour l'approche "Axiomatic Design").
En effet, l'approche "Axiomatic Design" consiste à respecter différentes règles afin d'obtenir une conception optimisée et répondant aux besoins du client. Deux axiomes (axiome d'indépendance et axiome du minimum d'information) et quatre domaines de conception sont proposés :

- Le domaine client (Customer Domain), où les concepteurs définissent les caractéristiques des besoins clients (Customers Attributes = CAs),
- Le domaine fonctionnel (Functional Domain), où les concepteurs déterminent les besoins fonctionnels (Functional Requirement = FRs),
- Le domaine physique (Physical Domain), où les concepteurs établissent les paramètres de conception (Design Parameters = DPs),
- Le domaine du processus (Process Domain), où les concepteurs préparent les processus liés au développement du produit (Process Parameters = PVs).

La conception est alors vue comme un processus itératif, qualifié de "mapping" ou encore de "zig-zagging", entre ces 4 domaines (Figure 18).

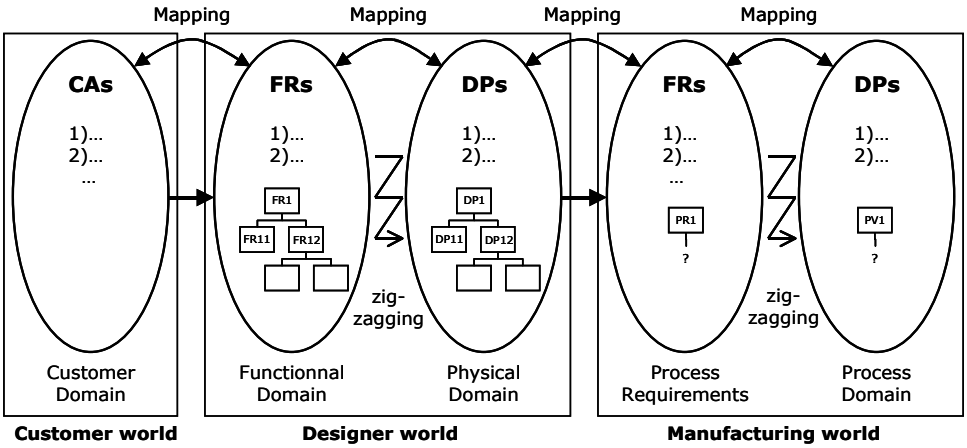


Figure 18 : la conception divisée en mondes et en domaines [Sohlenius, 1992].

Repris par le modèle MD-MV (Figure 19), chaque domaine correspond à une étape du cycle de vie du produit, chaque vue correspond à une représentation du système entier à travers la perspective d'un point de vue. Un point de vue décrit, quand à lui, les conventions et les règles pour construire et utiliser la vue et répondre aux préoccupations des parties prenantes. Le modèle considère le processus de conception comme un processus de création et de transformation de données, et d'informations au sein d'un réseau de domaines de conception en interaction, tel que le domaine projet, le domaine produit, le domaine Process, etc., complété par une vision d'un point de vue systémique. Ainsi chacun de ces domaines de conception, considéré en tant que système à part entière, peut être observé selon cinq types de vues:

- La **vue structurelle,** ou ontologique, considère le système dans sa structure,
- La **vue fonctionnelle,** ou phénoménologique, considère le système dans sa fonction,
- La **vue dynamique,** ou génétique, considère le système dans son comportement temporel,
- La **vue géométrique** considère le système dans ses formes, son encombrement et son positionnement spatial,
- La **vue contextuelle** considère le système dans son contexte lié au cycle de vie.

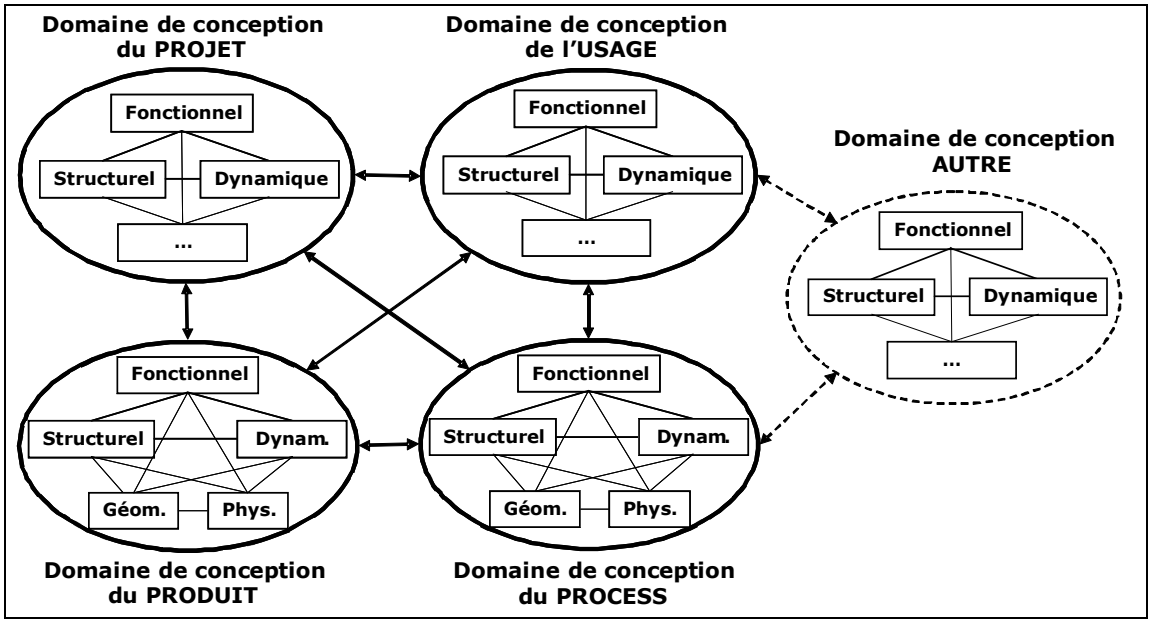


Figure 19 : modèle de données systémique "multi-domaines et multi-vues" comprenant plusieurs aspects appliqués à plusieurs domaines de conception [Gomes, 2002]

Cependant, un des aspects est aujourd'hui très peu développé dans le modèle "multi-domaines et multi-vues" proposé ; l'aspect physique, or il est fortement présent dans les domaines produit et process. Dans ce contexte, il serait intéressant de proposer des améliorations ou un modèle complémentaire visant à combler ces lacunes et permettre au MD-MV, de mieux prendre en compte, par exemple, les données issues de la simulation numérique.

Enfin, le modèle CPM permet d'adresser, en plus des vues "classiques", différentes vues métiers spécifiques via l'utilisation d'extensions. Ces vues permettent une meilleure représentation et une plus grande flexibilité d'utilisation des informations. Néanmoins, la prise en compte d'une nouvelle vue métier impose nécessairement de définir une nouvelle extension. En effet, nativement CPM ne permet pas de générer dynamiquement des vues métiers spécifiques.

Ainsi, suite à cette analyse, il apparaît que ces approches abordent divers points de vue ou domaines et permettent de disposer de **plusieurs niveaux de représentations et de contextualisations** des données et des informations manipulées, mais surtout de favoriser l'interopérabilité lors des processus collaboratifs d'ingénierie. En effet, il semble que toutes les méthodologies, les outils, s'inscrivant dans l'amélioration des processus collaboratifs, convergent vers des modèles ou des approches multi-vues, multi-utilisateurs de la conception du produit.
La décomposition des modèles produits, en vues ou en domaines multi-utilisateurs, coïncide avec le concept général PLM dans lequel un grand nombre d'outils, d'environnements, d'architectures, logicielles ou de méthodologies cohabitent et échangent de volumineuses et hétérogènes données qui doivent être compréhensibles et utilisables par les bons acteurs, au bon moment.

2.2.2.2 Structure conceptuelle des modèles produits

Après avoir mis en évidence, dans la section ci-dessus, l'importance de l'aspect multi-vues/multi-représentations proposé par la majorité des modèles produits, nous nous intéressons à leur structure conceptuelle.
Ainsi, [Harani, 1997] propose un modèle destiné à représenter l'ensemble des informations liées au produit conçu ou à concevoir, construit sur trois niveaux d'abstractions :

- **Conceptuel** (le plus haut niveau d'abstraction caractérisé par un *méta-modèle*)
- **Intermédiaire** (le *modèle,* obtenu par instanciation du méta-modèle, correspondant à un domaine métier)
- **Implémentation** (le *modèle instancié* dans un contexte métier précis)

De nombreux modèles produits, comme le CPM, le modèle de [Sellini 1999], [Yvars, 2001], ou encore le modèle de [Menand, 2002] (projet MULTI), sont structurés selon ce découpage conceptuel. Il permet une définition graduelle, du générique vers le spécifique des données, informations et connaissances utilisées par les concepteurs.
Le niveau conceptuel caractérise le degré d'abstraction le plus élevé. Ainsi, le méta-modèle définit tous les concepts génériques de base.
A partir de ce dernier, un modèle (intermédiaire) est défini pour correspondre à un domaine de conception.
Enfin, ce modèle est instancié pour coïncider avec les activités métiers des utilisateurs dans un projet (implémentation), c'est-à-dire représenter fidèlement tout ou partie d'un système et l'activité de conception dans le cadre du développement d'un produit.

Cette structuration est basée sur des concepts de modélisations et de méta-modélisations génériques, utilisés sur plusieurs niveaux d'abstractions, et sont "classiques" des approches MDE : Model Driven Engineering – MDA : Model Driven Architecture [Combemale, 2008]. Cette structuration est particulièrement bien adaptée à la représentation des connaissances.
En effet, une connaissance peut avant tout être exprimée selon un caractère générique et abstrait, c'est-à-dire en dehors de toutes considérations d'application logicielle, de modèles métiers ou contextes dans lesquels elle est utilisée. A ce titre, [Menand, 2002] a développé un méta-modèle pour formaliser la connaissance relative à un produit (Figure 20). Il est composé de trois niveaux d'abstractions.

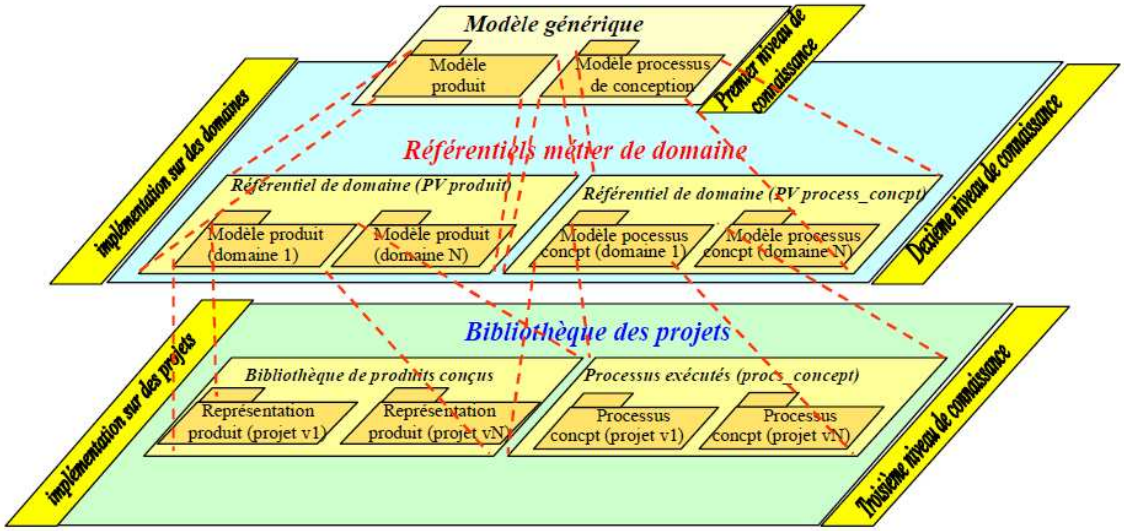


Figure 20 : décomposition conceptuelle des connaissances [Menand, 2002]

- Au plus haut niveau, il permet de décrire la connaissance générique de conception en décrivant les données, informations et connaissances génériques de toutes conceptions de produits. Les connaissances sont décrites à un niveau abstrait, indépendamment des technologies, des outils, des processus.
- L'instanciation sur un domaine donné constitue le modèle décrivant les connaissances propres au domaine. Ainsi, le modèle décrit un référentiel des connaissances métier du domaine. Ce niveau décrit le savoir-faire pour la conception dans un domaine technique spécifique. Il a la capacité d'évoluer quand les nouvelles technologies apparaissent. Cette évolution doit être sous le contrôle de l'acteur responsable [Zouari et al., 2002].
- L'instanciation du modèle sur un projet (bibliothèque des projets) permet de décrire la connaissance du niveau projet. Il offre une description rigoureuse d'un produit (réciproquement son processus de conception) appartenant à un domaine particulier qui est appliqué sur un projet donné selon ses conditions spécifiques.

Cette structuration permet de décomposer la connaissance selon différents niveaux d'abstractions et de représentations successifs, en fonction des besoins des utilisateurs. Elle permet aussi d'avoir une traçabilité complète dans la mesure où une connaissance utilisée dans un modèle métier dérive forcément d'une connaissance définie à un niveau plus générique, ce qui facilite également les propagations en cas d'évolution de cette connaissance générique.

2.2.3 Bilan sur les modèles produits

Les modèles produits apportent une solution complémentaire aux outils de gestion de données, dans la mesure où ils permettent de prendre en compte un spectre plus large de types de données et surtout d'adresser un niveau de granularité plus fin.
Au terme de la description du caractère multi-acteurs, multi-métier de la conception routinière, nous soulignons l'importance de la disponibilité et l'échange d'une grande variété de données, informations et connaissances. Celles-ci concernent à la fois le produit à concevoir et son processus d'obtention. Il est alors fondamental de pouvoir décomposer selon plusieurs vues les informations, afin qu'elles soient plus facilement accessibles et utilisables dans les activités du processus de conception. Ainsi, dans les modèles métiers, les vues les plus couramment utilisées pour structurer les informations sont les vues **structurelle** (décomposition et organisation dynamique des composants du produit), **fonctionnelle** (les fonctions que le produit ou les autres classes du modèles doivent accomplir), **physique** (la façon dont les fonctions sont satisfaites), et **géométrique** (surfaces, formes, features,…). Ces concepts, décrits dans des classes, sont mis en dynamique et en interrelations pour permettre de faire circuler les informations et assister les concepteurs dans leurs différentes tâches tout au long du cycle de conception. Néanmoins, les modèles produits sont souvent incapables de générer à la volée de nouvelles vues métiers correspondant aux différentes

activités du processus de conception à mettre en relation. Une des raisons principales vient de la gestion des informations, obligatoirement organisées selon les vues prédéfinies ci-dessus. Cette approche, que l'on peut qualifier de "top-down" (car elle contraint les informations à une organisation en vue prédéfinie), semble finalement trop statique pour permettre de réorganiser dynamiquement les informations dans de nouvelles vues initialement pas prévues. Ce constat est particulièrement vrai dans le cadre de notre problématique de collaboration au niveau des paramètres et des règles métiers entre modèles CAO et calculs. Dans ce contexte, les utilisateurs ont besoin de structurer en temps réel les paramètres et les règles selon leurs propres vues métier (calcul thermique, calcul acoustique, conception piston, etc.), et de lancer des analyses comparatives afin de vérifier la cohérence des informations partagées. De plus, dans les phases très amont du processus de conception, il n'est pas toujours possible de structurer les informations par rapport à une décomposition structurelle ou fonctionnelle, dans la mesure où elles n'existent potentiellement pas encore. Ainsi, si les approches traditionnellement proposées par les modèles produits sont indispensables, une approche complémentaire, plus focalisée sur les paramètres et les règles, et capable de multi-représentations dynamiques, semble intéressante. Cette approche, plutôt "bottom-up", permet d'organiser les informations et les connaissances génériquement (par exemple constituer un "cloud d'informations" générique) indépendamment d'une quelconque vue. Ces connaissances sont ensuite spécifiées et contextualisées à la volée en fonction des activités du processus de conception, en tenant compte de leurs différents niveaux de représentation et de maturité.
Cette approche complémentaire semble également compatible avec les principes de modélisation sur différents niveaux d'abstractions (conceptuel, intermédiaire, implémentation) et permet une meilleure prise en compte et une meilleure réutilisation des connaissances dans différents projets.

Cette composante réutilisation transverse multi-projets, induit forcément une nécessité de traçabilité et d'évolution des informations et des connaissances au fil du temps, en tenant compte par exemple de leur cycle de vie. Elle est encore plus indispensable si l'on a pour objectif d'assurer leur cohérence. Ce point fera l'objet de la section 2.3.6.

Mais avant, il semble nécessaire de préciser ce qu'il est couramment admis quand on parle de données, informations et surtout de connaissances dans le processus de conception.

2.3 Connaissances et conception de produits

La gestion des connaissances est un domaine qui prend de plus en plus d'importance, particulièrement pour les industriels qui y voient un moyen de sauvegarder et réutiliser leur patrimoine intellectuel et leur savoir-faire, et ainsi être en mesure de concevoir mieux et plus vite. C'est un vaste domaine, qui comporte de nombreuses approches en fonction des objectifs à atteindre et des types de connaissances à manipuler. Ainsi, dans le cadre de la définition de notre problématique, il est nécessaire d'identifier les connaissances qui interviennent lors des processus collaboratifs du couple produit-simulation.

2.3.1 Les origines de la notion de connaissance

La connaissance est une notion aux sens multiples à la fois utilisée dans le langage courant et objet d'études poussées de la part des philosophes contemporains.
Elle fait généralement référence à tout ce qui est tenu pour su par un individu, dans un contexte ou une société donnée. C'est une notion très répandue qui apparaît sous plusieurs formes et dans de nombreux domaines tels que les langues, la sociologie, les traditions ou les légendes.
Très tôt, l'homme a voulu définir et représenter les origines, les moyens, le contenu et les limites de la connaissance, principalement humaine, donnant lieu à l'analyse de la connaissance, c'est-à-dire, la détermination de ses conditions nécessaires et suffisantes (modèle de Platon Figure 21). *« Il s'agit plus précisément d'établir quelles relations entretient la connaissance avec la croyance et la vérité, et quelles procédures de justification permettent de distinguer une simple croyance vraie (qui peut l'être par accident) d'une véritable connaissance »* [Engel, 2002] [Dutant et Engel, 2005].

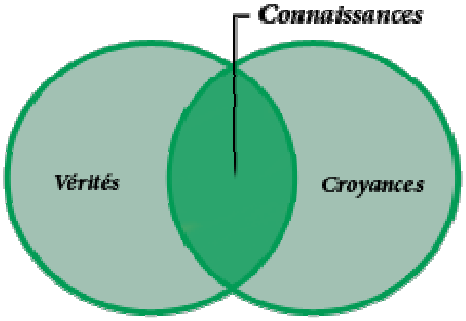


Figure 21 : modèle de Platon

Par la suite, la connaissance s'est rapidement déclinée dans les domaines scientifiques (sciences cognitives), économiques et industriels, pour donner lieu à de nombreuses théories ou caractérisations en fonction de contextes spécifiques. La notion de connaissance est alors souvent associée à la "donnée", "l'information", la "compétence" ou encore la "sagesse", généralement en faisant référence à un capital détenu par une personne, une société ou une organisation. Ainsi, plusieurs natures (tacite, explicite, collective, individuelle), et de nombreux types de connaissances, sont apparus, notamment dans l'environnement de l'entreprise et des différents métiers.
Gérer et capitaliser les connaissances est aujourd'hui devenu nécessaire, dans le but d'améliorer l'efficacité globale des organisations, donnant naissance aux concepts de gestion de la connaissance.

2.3.2 De la donnée à la connaissance

La connaissance est une notion difficile à appréhender, il n'existe pas aujourd'hui de consensus la définissant précisément. [Sanchez et al., 1996] proposent ainsi trois notions à la base d'une première approche du terme connaissance :

- Une connaissance est une idée plus ou moins complète ou précise que l'on a de quelque chose.
- Une connaissance désigne la faculté de connaître, de comprendre.
- Une connaissance est le droit de porter un jugement sur quelque chose qui est lié au niveau de connaissance que l'on a de cette chose.

Toujours selon [Sanchez et al., 1996], la connaissance ou le savoir peuvent alors être caractérisés, comme un ensemble de croyances détenues par un individu à propos d'une relation de causalité entre des phénomènes. Ainsi, certains groupes de personnes au sein d'une organisation partagent le même ensemble de croyances à propos de la même relation causale.
Dans l'industrie, les acteurs d'un projet partagent des connaissances au travers des activités de conception. S'il existe souvent une confusion entre les notions de données, d'informations et de connaissances, il est aujourd'hui admis que la connaissance a pour origine des informations, elles mêmes issues de données crées par les acteurs de l'entreprise. Compte-tenu de cette remarque, et en se basant sur différents travaux, nous proposons la distinction suivante.

Une donnée peut-être caractérisée comme étant une description élémentaire, souvent codée, d'une chose, d'un événement. Par exemple, un fait objectif qui relate un événement, comme une observation simple [Davenport et al., 1998], ou la perception d'un signal ou d'un signe [Weggeman, 1996]. [Tsuchiya, 1995] et [Ermine et al., 1996] caractérisent les données comme des faits de base, apparaissant au cours de la réalisation d'une tâche et pouvant être transcrites sous forme de chiffres, de mots, de symboles ou de figures. Enfin, [Prax, 2000] complète ces définitions avec la notion d'absence d'intention. Ainsi, les données résultent d'une acquisition instrumentale, naturelle ou construite, elles peuvent être qualitatives ou quantitatives. Il n'y a normalement pas d'intention ni de projet dans les données, ce qui leur confèrent leur caractère d'objectivité.

Considérant ces définitions, on peut en déduire qu'une donnée est un fait objectif ou une perception à laquelle aucun traitement cognitif, aucune analyse n'est effectuée. Par conséquent, une donnée ne possède aucun sens particulier, aucun contexte. Par exemple la valeur "500", la chaine de caractère "longueur".
Appliquée au domaine de la conception de produit, nous considérons qu'un paramètre extrait d'un modèle métier, et par conséquent rattaché à aucun contexte, constitue une donnée.

Une information est obtenue à partir d'une donnée ou d'une série de données. Ainsi, on peut considérer une information comme étant une donnée qui à du sens. [Prax, 2000] caractérise l'information comme étant une collection de données organisées pour donner forme à un message pertinent dans un contexte spécifique [Fukuda, 1995]. La façon d'organiser les données résulte alors d'une intention de l'émetteur, ce qui la rend subjective. En conséquence, une information nécessite un émetteur, qui a l'intention d'émettre un message, mais ne nécessite pas obligatoirement de récepteur car l'information est distincte du sujet.

L'information résulte donc de données, organisées, structurées, documentées à partir de conventions, afin de leur attribuer un sens et correspondre à un contexte d'utilisation. A la différence de la donnée, l'information n'est pas objective mais elle a une signification.
On considère que les données extraites des modèles métiers, une fois organisées et structurées pour correspondre à un ou un ensemble de contextes ayant du sens pour les concepteurs, sont de l'information. Par exemple, le paramètre "Longueur=500mm" est un paramètre dimensionnant du piston.

Conformément aux définitions précédentes, la connaissance est donc obtenue à partir d'informations, elles mêmes tirées de données. Elle peut être définie comme étant l'interprétation d'une ou d'un ensemble d'informations par un humain, dans un contexte spécifique [Bender et al., 2000] [Monticolo, 2008].
En effet, dans l'environnement de l'entreprise, la notion de connaissance fait souvent référence à une combinaison d'informations. En structurant de différentes façons restrictives l'information, on obtient alors des connaissances différentes. Ainsi, les connaissances sont définies comme des données et des informations analysées, sélectionnées, partagées et associées à un contexte d'utilisation. Cette connaissance s'intègre dans son système personnel de représentation. L'information reçue subit une série d'interprétations personnelles afin que le sujet puisse construire une représentation qui fasse sens : la connaissance doit faire sens pour le sujet et le récepteur [Prax, 2000] [Zarifian 2002]. En conséquence, la connaissance n'est pas seulement un stock figé, elle est activable selon une finalité. Il y a, dans la connaissance, une notion de processus, de construction d'une situation finale, et donc, elle résulte d'une acquisition d'information et d'une action.
Enfin, les connaissances ne sont pas des vérités éternelles. Elles se périment, se modifient, évoluent, et de ce fait peuvent avoir une importance plus ou moins grande en fonction du temps. La gestion des connaissances ne peut donc se faire que dans le cadre d'un processus extrêmement dynamique basé sur la notion de maturité. De plus, il ne faut pas confondre le degré de maturité des connaissances avec celui des outils et des méthodes les utilisant [Zouari, 2007].

Ainsi, on considère la connaissance comme étant l'utilisation d'une sélection d'informations dans un contexte particulier, nécessitant l'interprétation d'un acteur. La connaissance est une notion dynamique, utilisable dans un but final.
Par rapport à notre contexte de recherche, on considère la connaissance comme étant une sélection et une association de paramètres et de règles métiers nécessitant l'interaction avec un concepteur pour êtres utilisés dans un modèles métiers. Ces paramètres et règles sont régulièrement modifiés afin d'atteindre un objectif dans la conception du produit.
Exemple : Longueur=500mm est un paramètre dimensionnant du piston, il est défini par la relation Longueur= Diamètre/Course afin de respecter le référentiel standard de conception moteur.

Par ces définitions, on comprend que l'information, qui est factuelle, peut être facilement formalisée, capitalisée et transportée dans des documents, des bases, sous forme explicite, alors que

la connaissance, est un item plus humain, subjectif, et souvent tacite [Grundstein, 1995]. Ainsi, nous nous intéressons aux différents caractères de la connaissance pour la conception de produits.

2.3.3 Les différents types de connaissances

2.3.3.1 Connaissances tacites et explicites

Dans le prolongement de la critique Kantienne [Kant, 1781], [Polanyi, 1967] propose une première typologie des connaissances, ensuite reprise par [Nonaka, 1995] et [Grundstein, 1995], dans laquelle il est aujourd'hui admis qu'il existe une distinction fondamentale entre la **connaissance dite tacite** et la **connaissance dite explicite**. En effet, les connaissances verbalisables et explicites ne représentent qu'une partie limitée du savoir humain. Elles reposeraient avant tout sur l'expérience, sur la perception des sens, sur un mode d'apprentissage implicite difficile à exprimer et qui reste le plus souvent tacite, ou "inarticulé".

- **Les connaissances tacites** sont personnelles, car elles "résident dans la tête de l'individu" et ne peuvent pas toujours êtres articulées sous forme codée. En effet, elles sont implicites et font appel à l'expérience et au savoir-faire de la personne qui les possède. Non tangible, elles peuvent être difficiles, voire impossible, à expliciter dans une forme exploitable par d'autres personnes. Les connaissances tacites appartiennent donc au monde des objets mentaux, des représentations mentales. Elles regroupent les compétences innées ou acquises, le savoir-faire et l'expérience de l'individu. Elles sont généralement difficiles à formaliser par opposition aux connaissances explicites.

- **Les connaissances explicites**, par opposition aux connaissances tacites, sont des connaissances codifiées, clairement articulées et transmises dans un langage formel et structuré au niveau d'un document écrit ou d'un système informatique (Wikipedia par exemple). Elles sont transférables physiquement et réutilisables par d'autres personnes car elles apparaissent sous forme tangible. Il est alors généralement plus facile de rendre ces connaissances collectives face aux connaissances tacites qui restent plus individuelles.

De notre point de vue, ces deux types de connaissances ne peuvent pas être traités de la même manière mais sont néanmoins indissociables. Les savoirs implicites dictent une bonne partie de notre comportement, où l'individu lui-même n'est plus conscient de l'étendue de ses savoirs. Nous connaissons plus que ce que nous pouvons exprimer, et la connaissance explicitée en mots et en chiffres ne reflète qu'une infime partie de nos savoirs [Polanyi, 1967]. Ainsi, [Vinck 1997] met en exergue l'inévitable coexistence des connaissances tacites et explicites. Il illustre son concept par la métaphore de l'iceberg de la connaissance (Figure 22) faisant apparaître une partie immergée (implicite) et une partie émergée (explicite,) en rapport avec la perception qu'un individu à de sa propre connaissance.

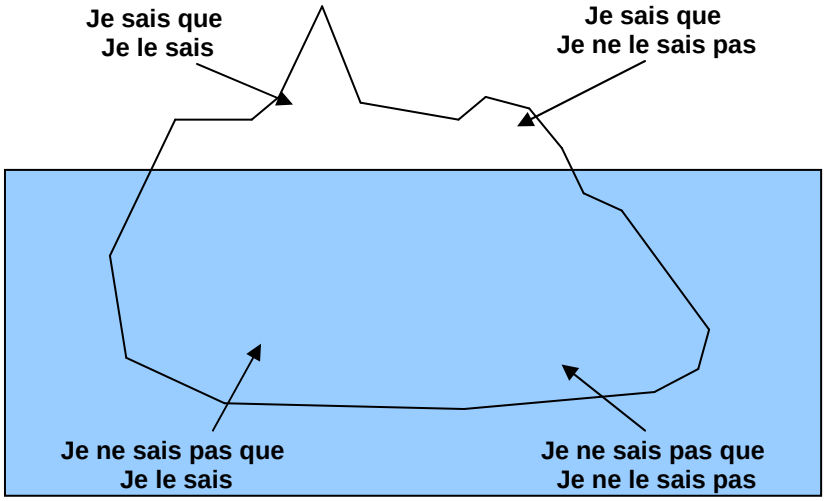


Figure 22 : métaphore de l'iceberg [Vinck, 1997]

Si les connaissances tacites et explicites semblent indissociables, il est néanmoins possible de passer d'un type à l'autre, c'est ce que l'on qualifie de modes de conversion de la connaissance.

2.3.3.2 Les modes de conversion de la connaissance

Dans la réalité quotidienne de l'entreprise, la connaissance est dynamique. Elle passe en permanence d'un état de "connaissances tacites" à celui de "connaissances explicites" et vice-versa. Compte-tenu de la double nature des connaissances, [Nonaka, 1995] propose quatre modes de conversion dans une organisation (Figure 23) :

- **La socialisation :** (du tacite au tacite) cela regroupe tous les modes de transmission d'une personne à une autre, comme du maître à l'apprenti par l'imitation, l'observation, la pratique, l'expérience. Cela permet la création de nouvelles connaissances tacites au travers de schémas ou de modèles mentaux partagés autour d'expertises techniques. Au cours de ce processus, il n'y à pas d'explicitation directe de ce qui est transmis, par conséquent ces connaissances ne pourront être directement exploitées au niveau collectif dans l'entreprise.

- **L'extériorisation :** (du tacite à l'explicite) c'est un processus de conversion où l'individu convertit ses connaissances tacites en connaissances explicites.

- **L'intériorisation :** (de l'explicite au tacite) c'est un processus d'assimilation des connaissances explicites qui, peu à peu, par l'intermédiaire de processus cognitifs, viennent compléter le capital tacite d'un individu ; lors d'un apprentissage par exemple (en anglais "learning by doing").

- **La combinaison :** (de l'explicite à l'explicite) c'est un processus d'émergence de nouvelles connaissances explicites à partir de connaissances explicites déjà existantes et recombinées entre elles.

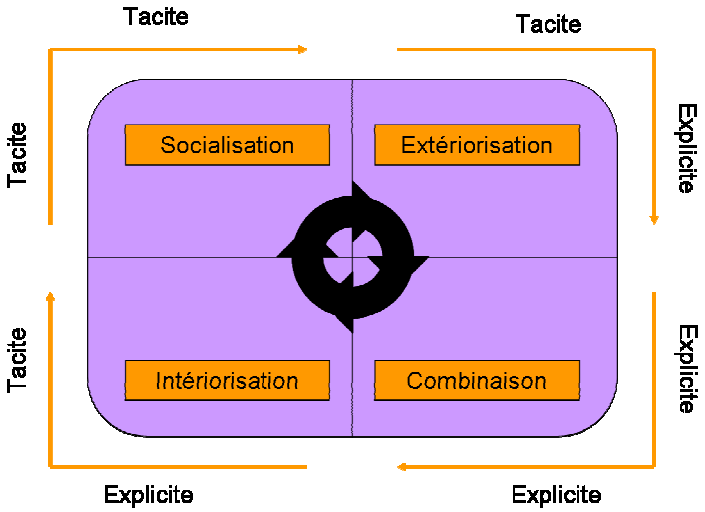


Figure 23 : les quatre modes de conversion de la connaissance [Nonaka 1995]

Cette approche est reprise par de nombreux auteurs comme [Ermine, 2001] qui l'utilise pour fonder son modèle de gestion des connaissances (modèle de la marguerite) dans lequel il décrit un processus de capitalisation et de partage des connaissances tacites nommé "cycle vertueux de la connaissance".
[Grundstein, 1995] réutilise également les travaux de Nonaka pour élaborer son approche basée sur la capitalisation des connaissances cruciales, lui permettant de différencier les savoirs de l'entreprise (les connaissances explicites et formalisées) des savoir-faire individuels et collectifs (les connaissances tacites, stockées dans les têtes des personnes ou utilisant des éléments immatériels).

Etant donné le caractère volatile des connaissances tacites en termes de compréhension, de formalisation, de capitalisation et de dépendance à l'individu, nous nous focaliserons sur une approche de gestion des connaissances explicites.

2.3.3.3 La caractérisation des connaissances

Dans le monde industriel, la construction des connaissances s'effectue au delà du caractère tacite/explicite ou personnel/collectif. Dans ce contexte deux approches s'opposent dans la caractérisation et la construction de la connaissance : l'approche constructiviste et l'approche positiviste.

- Pour **l'approche constructiviste** [Le Moine, 1995], la connaissance est un construit social dont la valeur de vérité dépend du consensus. La démarche consiste à construire la représentation des processus à partir des connaissances partielles qu'en ont les acteurs au travers des activités réelles qu'ils sont amenés à exercer. Tout au long du déroulement de l'étude, les problèmes rencontrés donnent lieu à l'identification de liens informels de communication entre acteurs, non décrits dans les documents, et au repérage des connaissances nécessaires à la résolution de ces problèmes.
- Pour, l'**approche positiviste**, la connaissance est une vérité naturelle dont la valeur de vérité provient d'une démonstration irréfutable contenue dans des axiomes, théorèmes et universaux en tout genre, en général corrélés à l'expérience du réel et indépendante de la volonté des acteurs.

De notre point de vue, la connaissance n'est pas créée par un groupe d'individus, mais par chaque individu. Cette connaissance est ensuite utilisée, et partagée au cours des projets, lors des collaborations, interactions et échanges d'informations entre les individus. Nous nous plaçons donc dans des courants de pensée constructivistes et positivistes où la connaissance est créée par les individus et peut être formalisée et capitalisée au fil de l'eau, à condition qu'elle soit validée et évaluée par les experts du domaine. Elle devient donc, à ce stade, indépendante du sujet humain (connaissance individuelle) ou du collectif (connaissance collective) qui a permis son émergence (Annexe 11).

Il nous semble que la connaissance est donc avant tout un problème de représentation puisque chaque individu possède sa propre connaissance : une même information peut être interprétée de manière totalement différente entre deux individus en fonction de leurs propres schémas mentaux et de leur façon d'interpréter leur environnement. Ainsi, une donnée doit être renseignée afin d'obtenir une information située dans un contexte qui puisse être utilisée par les acteurs métier en vue de créer de nouvelles connaissances [Monticolo, 2008].

2.3.4 La gestion des connaissances pour le développement de produits

La gestion des connaissances apparaît comme indispensable à la résolution productive de problèmes de conception routinière et au développement d'un environnement favorable à l'innovation [Serrafero, 2006]. En effet, l'entreprise regroupe une grande diversité de connaissances (connaissances projets, connaissances de l'organisation, connaissances du produit, etc.) qui, pour être correctement exploitées, nécessite des méthodes permettant de les identifier, les extraire, les formaliser, les classer, les mettre à disposition. En effet, les entreprises ont aujourd'hui conscience de l'importance de la sauvegarde de ce capital intellectuel considéré comme un des derniers leviers face à la concurrence des pays émergents. Compte-tenu de ces objectifs, des propositions ont été faites dans le but de gérer les connaissances ; citons, par exemple, le développement des mémoires d'entreprises et la capitalisation des connaissances.

- **Les mémoires d'entreprises** tendent à définir un système regroupant l'ensemble des données, informations et connaissances (explicites) caractérisant l'entreprise. Si ces

systèmes permettent de créer des archives, elles ne permettent généralement pas de stocker la façon dont elles sont utilisées [Monticolo et al., 2008], [Bekhti et Matta, 2003], [Bidal et al., 2002**],** [Matta et al., 2000].

- **La capitalisation des connaissances** consiste à considérer les connaissances utilisées par l'entreprise dans le but de constituer un capital et le faire croitre. La capitalisation s'appuie sur une succession d'étapes basées sur la notion de connaissances cruciales [Grundstein, 2000]. Elles sont définies comme étant les connaissances suffisantes et nécessaires à l'activité d'une entreprise.

Concernant la capitalisation des connaissances cruciales et le cadre directeur GAMETH, [Grundstein, 2000] propose un cycle du processus de capitalisation organisé en cinq facettes (Figure 24) :

- Repérer (localiser, identifier les connaissances cruciales)
- Préserver (formaliser, modéliser la connaissance)
- Valoriser (diffuser, partager, exploiter la connaissance)
- Actualiser (maintenir la connaissance)
- Manager (concerne les interactions entre les différentes facettes du processus)

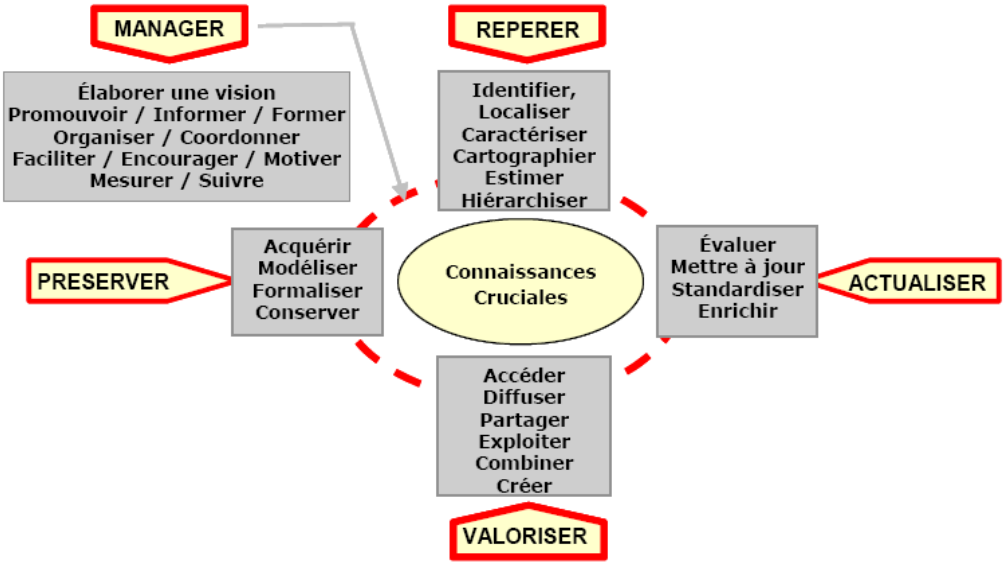


Figure 24 : cycle du processus de capitalisation des connaissances cruciales [Grundstein, 2000]

De nombreuses méthodes existent, visant à cartographier, extraire, classer et capitaliser les connaissances. Elles sont organisées selon des approches qualifiées d'ascendantes (à partir de données on définit des modèles, ex : CommonKADS), descendantes (à partir de modèles, on interprète des données, ex :MKSM [Ermine, 2000]) ou mixtes.

La gestion des connaissances répond donc aux problématiques de capitalisation de connaissances cruciales, permettant de mobiliser les acteurs d'une entreprise autour d'un processus global conduisant à une meilleure prise en compte des contraintes issues des divers domaines métiers, facilitant ainsi la collaboration et optimisant le processus de conception. Dans ce contexte, la notion d'objet de connaissance semble intéressante car elle permet la prise en compte de connaissances très hétérogènes et facilite l'organisation d'un système de capitalisation de la connaissance.

2.3.5 Les objets de connaissances

En conséquence de notre analyse précédente, il apparaît que la gestion des connaissances implique la considération d'un panel extrêmement varié de connaissances réparties dans de nombreux systèmes nécessitant de multiples représentations. En effet, à cause de la complexité des organisations actuelles et de leur aspect dynamique, les contraintes pesant sur un système de capitalisation sont fortes. En conséquence, pour [Barthès, 1997], il est donc peu vraisemblable

qu'un système de capitalisation des connaissances se développe de façon monolithique. Il s'agira plutôt d'une intégration de différents éléments hétéroclites, qui, ensemble, et en relation avec les acteurs humains constitueront le système final. Il semble donc nécessaire de tenir compte du caractère multi-représentable des connaissances et de leurs rapports avec les individus dans l'utilisation et la construction de nouvelles connaissances.
Dans ce contexte, [Prudhomme et al., 2001] proposent la notion **d'objet de connaissances**. Un individu peut alors construire des rapports personnels à ses données, informations et savoirs, vus comme des objets de connaissance, représentables et formalisables, mais de nature différente, en fonction de leurs niveaux de formalisation et de leurs origines. Un objet de connaissance sert donc à expliciter la connaissance par la définition d'artefacts spécifiques constitués à partir de données, et d'informations. L'individu, par une interaction avec des objets de connaissance, génère alors sa propre connaissance individuelle [Baizet, 2004].
En opposition à cette notion, [Grundstein, 2007] décrit la connaissance comme étant non objectivable. Néanmoins, il précise la contradiction existante avec l'idée de connaissances objectivables, portée par l'ingénierie des connaissances, qui conduit à des techniques et des méthodes d'acquisition et de représentation des connaissances. Ces projections, par nature réductrices, ne sont que des informations source de connaissances pour la personne ou l'artefact capable de les interpréter. Par ailleurs, penser que la connaissance n'est pas objectivable tient du paradoxe, dès lors que nous considérons des connaissances techniques (connaissances à caractère descriptif, normatif ou prescriptif portant sur des objets matériels ou immatériels) ou des connaissances scientifiques, ayant valeur de vérité, par nature universelle.

De notre point de vue, nous positionnons ces travaux de recherche dans le domaine de l'ingénierie des connaissances et plus particulièrement des connaissances relatives aux produits. Ces connaissances sont par nature techniques, définies à partir de paramètres et de règles métiers, utilisées et partagées au cours des activités du processus de conception. On peut alors parler de modélisation par entité, les entités étant les objets de connaissances capitalisant les paramètres, et les règles représentant les connaissances utiles à une activité particulière du processus de conception. Dans ce cadre, l'objet de connaissance se positionne alors comme un support à la capitalisation, la représentation et au partage des connaissances entre les acteurs leurs conférant un rôle central. L'objet de connaissance constitue un moyen de formaliser les connaissances explicites. La construction de ces objets se fait par l'interaction d'un individu avec des informations sources de connaissances. Ainsi, dans la mesure où ces objets résultent de cette interaction, on peut considérer qu'une partie des connaissances tacites de l'individu sont capturées, traduisant ses intentions et son expérience.

2.3.6 La gestion de configuration de connaissances

Une des approches qui tend à se développer concerne la gestion de configuration car elle offre de nouvelles possibilités permettant de prendre en compte plus finement les connaissances et leurs évolutions dans le processus de conception. Ainsi, relativement à la description de la gestion de configuration présentée en introduction section C.C, nous soulignons l'intérêt de cette démarche dans le domaine de la gestion de la connaissance.

La gestion de configuration est apparue en même temps que le développement des approches d'ingénierie collaborative (essentiellement en conception routinière). Elle tente de répondre à des problématiques de collaboration, de traçabilité et de cohérence, en proposant des solutions basées sur la gestion de la diversité de produits [Agard, 2002], la gestion de modifications [Rivière, 2004], ou encore la gestion des versions [Zouari, 2007].
Ainsi, [Agard, 2002] propose une méthode basée sur trois niveaux de diversité (fonctionnelle, technique et process) et le passage d'un niveau à l'autre. L'objectif est d'assurer une diversité fonctionnelle totale du produit fini en s'appuyant sur la description du produit et du process. Cette méthodologie est complétée par la définition d'un outil d'aide à la décision pour la conception de faisceaux électriques automobiles.
Plus proche de notre problématique, les travaux de [Rivière, 2004] sur la gestion de modifications appliquée au cas de l'aéronautique permettent de mettre en évidence les difficultés rencontrées lors

de l'évolution et du changement au niveau des paramètres au cours des activités du processus de conception. Ces modifications inévitables sont de nature à désynchroniser la définition du produit de ses spécifications fonctionnelles et techniques, et générer des incohérences entre les acteurs des activités d'un projet. Basée sur les mécanismes de propagation, une proposition est formulée, permettant lors de l'analyse des impacts, de tenir compte des liens de dépendance organisationnels et fonctionnels et la définition d'une stratégie de traitement des processus de modification et d'évaluation. Une application à été développée, baptisée CM-EC : "Collaborative Management of Engineering Changes" prenant en compte :

- L'analyse des impacts
- La supervision des processus de traitement
- L'évaluation des solutions
- Un support à la collaboration

Enfin, à partir d'une étude des modèles produits et de la littérature portant sur l'ingénierie des connaissances, [Zouari, 2007] propose des mécanismes génériques de gestion des versions des connaissances. En effet, la pratique actuelle des systèmes de gestion documentaire fournit le contrôle des versions, mais seulement au niveau du document. Pour la conception collaborative, le contrôle des versions est exigé à un niveau de détail plus fin [Van Leeuwen et al., 2003]. Ainsi, dans le cadre de la parallélisation des tâches, [Zouari, 2007] au travers de sa proposition, favorise le partage des connaissances et des ressources de conception et la formalisation des échanges et des décisions.
Relativement à ces différentes approches, tout pousse alors à penser que la notion de cohérence est une composante forte de la gestion de configuration. Appliquée aux paramètres et aux règles métiers, elle permet de favoriser la collaboration entre les acteurs en permettant de gérer le niveau de maturité, notamment lors d'activités simultanées. Des approches proposées par [Yang et al., 2008] ou encore [Vareilles, 2005] utilisent des moteurs de résolutions de contraintes afin d'assurer le respect des règles manipulées dans les configurations. Ces approches contribuent non seulement au respect des référentiels de conception mais aussi, guident les concepteurs dans les décisions à prendre au cours d'un projet.

Nous considérons la gestion de configuration de connaissances comme une composante essentielle de la gestion de la connaissance. En rapport avec notre contexte de recherche, elle constitue un axe principal répondant aux problématiques de capitalisation, de traçabilité, de réutilisation, mais aussi de cohérence des connaissances lors des processus collaboratifs. De plus, compte-tenu du caractère très dynamique des connaissances, que ce soit au niveau de leur cycle de vie ou au niveau de leurs modifications, il est nécessaire que des mécanismes permettent de les prendre en compte afin d'assurer leur pérennité.

2.4 Différents outils pour la gestion des connaissances au niveau des paramètres et des règles

Pour terminer ce chapitre, et faire suite à l'analyse des outils de gestion de données des modèles produits et du domaine de la gestion de la connaissance, nous présentons quelques outils et travaux que nous avons identifiés comme les plus représentatifs par rapport à notre contexte de recherche.

2.4.1 GREC²

Les travaux de [Ducellier, 2008] portent sur la définition d'une plateforme de gestion des paramètres et des règles associés aux documents gérés dans un outil de gestion de données en contexte PLM. En effet, le constat de l'incapacité des outils de gestion de données à prendre en compte les données et les informations contenues dans les fichiers a été identifié par Ducellier. Pour étayer sa thèse une application appelée GREC² (**G**estion de **R**ègles **E**xpertes pour la **C**onception **C**ollaborative) a été développée. Elle permet de mettre en relation les paramètres et les règles (stockés dans une base indépendante) avec les documents du PLM dans lesquels ils sont utilisés. Ainsi, les différentes fonctionnalités de GREC² s'articulent autour de quatre blocks : Administrer, Interagir, Gérer et Capitaliser (Figure 25).

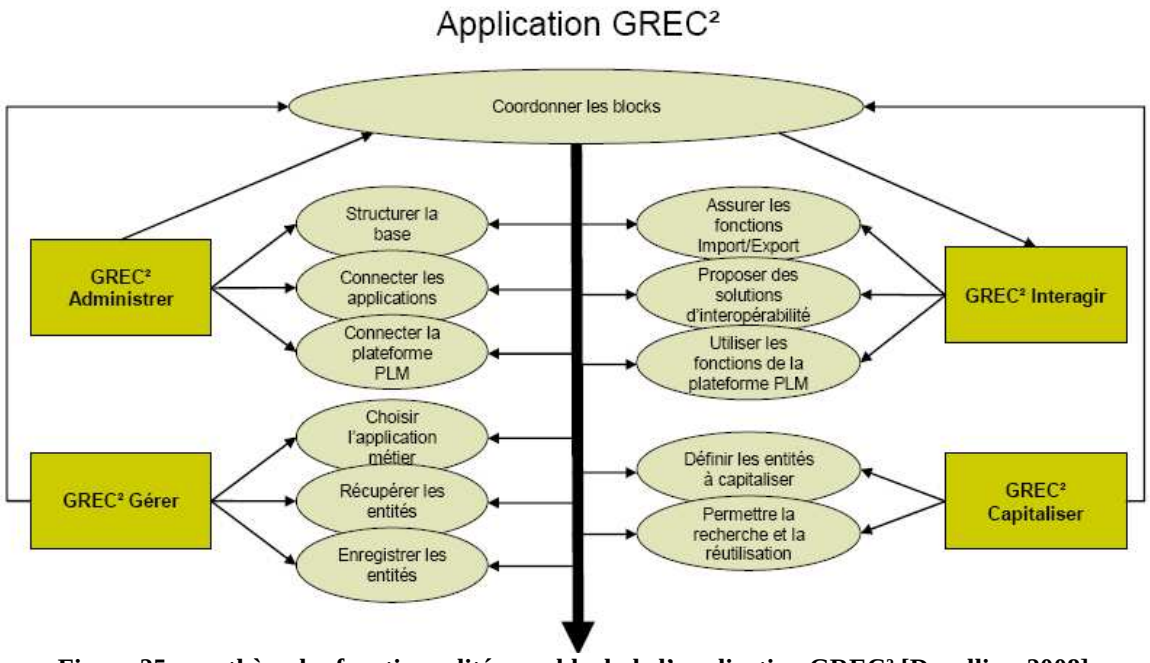


Figure 25 : synthèse des fonctionnalités par block de l'application GREC² [Ducellier, 2008]

De part ses objectifs, GREC² est très liée au fonctionnement de l'outil de gestion de données. Cela permet de constituer un référentiel de paramètres et de règles, afin d'améliorer l'interopérabilité entre les applications métiers. Néanmoins, il n'est pas possible d'effectuer des traitements sur ces connaissances, comme des comparaisons ou des raisonnements sur les règles, qui permettraient aux concepteurs de maintenir une cohérence globale. Ce point fait d'ailleurs l'objet des perspectives de recherches dans lesquelles l'auteur fait référence, à la nécessite d'une orientation sur les notions de contraintes permettant de mettre en places des mécanismes de propagation ou d'optimisation.

2.4.2 Colibri

[Kleiner, 2003] propose un modèle produit paramétrique basé sur la prise en compte des contraintes de conceptions. Ce modèle diffère des approches classiques car il est complètement dédié aux contraintes liant des paramètres entre les applications métiers hétérogènes (Figure 26), offrant une alternative aux processus unidirectionnels ou aux échanges documentaires utilisant des formats neutres.

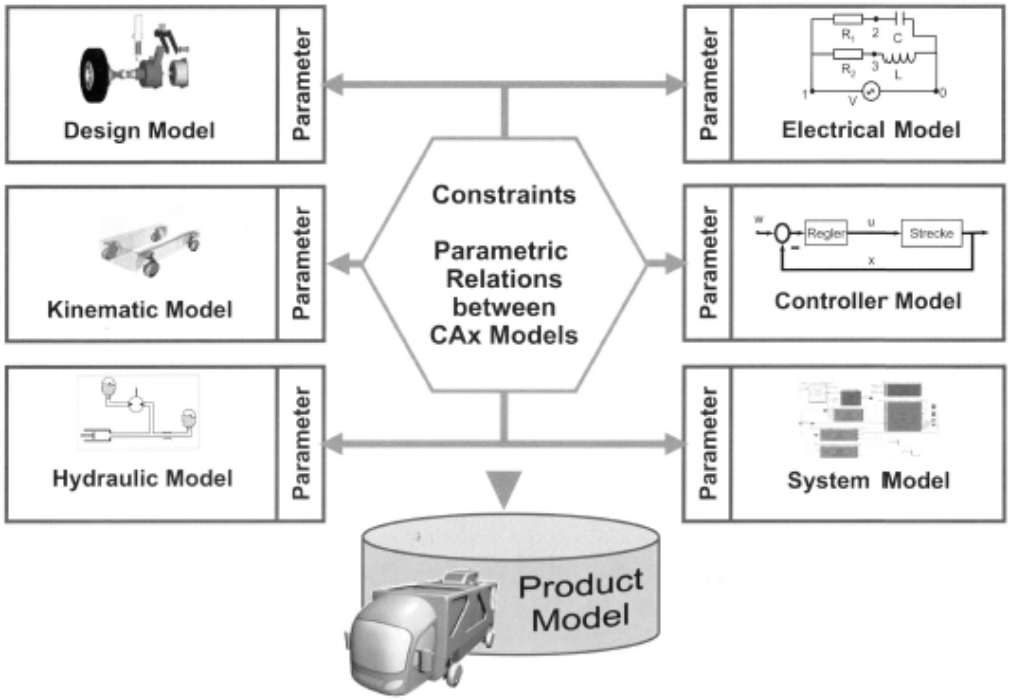


Figure 26 : modèle produit paramétrique [Kleiner, 2003]

Le modèle produit est expérimenté au travers d'une application baptisée COLIBRI (Constraint Linking Bridge). Elle permet concrètement de mettre en relation les paramètres encapsulés dans des modèles hétérogènes via l'intégration de contraintes, en tenant compte de la structure du produit et des documents du projet. L'utilisation de mécanismes de vérification et de résolution permet de mieux tenir compte des contraintes selon les différents points de vues liées aux activités du processus de conception. L'évolution des objets, réels ou virtuels (pièce assemblage ou modèles métiers), utilisant les paramètres, est assurée par des mécanismes de gestion de versions.

La mise en œuvre d'un modèle produit étendu aux données paramétriques, et l'application COLIBRI, constituent une première démarche validant le concept d'intégration des données du produit basé sur les contraintes. Initiée dans les années 2000 (explosion des processus collaboratifs et du nombre de modèles métiers), l'approche se tourne nécessairement vers l'amélioration de l'interopérabilité entre des modèles hétérogènes. Il est alors possible d'intégrer des modèles CAx différents, simplement en les reliant et en utilisant les contraintes précisant leurs relations pour le développement du produit.
Néanmoins, la proposition semble assez limitée quant à la capitalisation, la réutilisation, l'organisation et la gestion des paramètres et des contraintes et leurs évolutions (cycle de vie), issues de préoccupation plus récentes. Dans la mesure où les paramètres restent encapsulés dans les modèles, ils sont par définition inaccessibles en dehors des applications métiers capables de les utiliser. Une des perspectives de ces travaux de recherche serait de s'abstraire des modèles, de manière à fournir une méthodologie de gestion, non seulement des contraintes mais aussi des paramètres, en intégrant la prise en compte de leur cycle de vie, le suivi des modifications et la comparaison de groupes de paramètres et de contraintes partagées par plusieurs acteurs.

2.4.3 Krossroads

Dans le cadre du projet IngéProd actuellement en cours (mené en collaboration par l'UTBM et FAURECIA), un outil prototype a été développé, baptisé KrossRoads [Vernier et al., 2011]. Les problématiques d'IngéProd sont assez proches des problématiques du projet ADN, à la différence près qu'il s'intéresse au couple produit-process, alors qu'ADN s'intéresse avant tout au couple produit-simulation en phases amont du processus de conception.
Dans sa version actuelle, l'outil KrossRoads se limite à représenter des informations porteuses de connaissances métier issues de plateformes PLM et d'outils de CAO. L'outil est connecté à des outils d'exploitation des connaissances (moteur d'inférence, KBE) et des systèmes de gestion des connaissances (KMS) assurant la gestion du cycle de vie des connaissances (Figure 27).

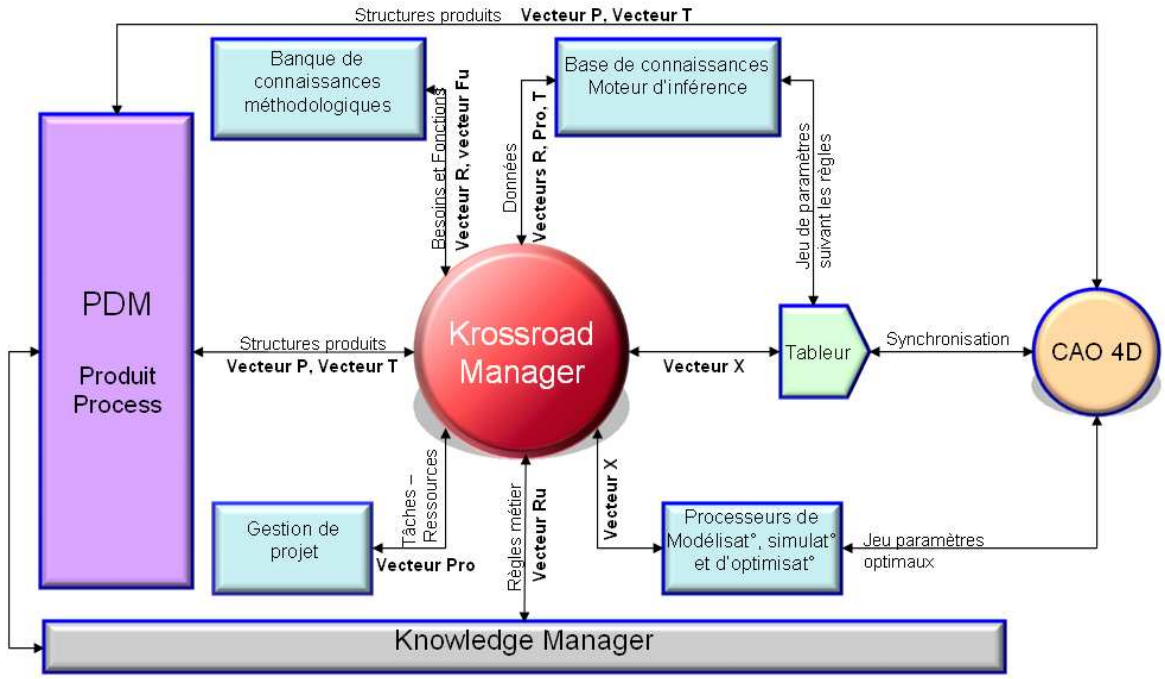


Figure 27 : l'outil KrossRoads et ses connections avec d'autres types d'applications

L'objectif de KrossRoads est d'améliorer l'intégration du savoir-faire d'une entreprise afin de garantir sa faisabilité au plus tôt dans son cycle de développement. Pour cela, dans sa première version, KrossRoads est constitué de trois fonctionnalités principales.
Un premier atelier spécifique est dédié aux experts de l'entreprise permettant de formaliser leurs connaissances sous forme de check-lists de règles métier incluant paramètres (non capitalisés dans l'outil) et formules. L'accent est porté sur les règles de fabrication ou d'assemblage permettant de garantir la faisabilité d'un produit, mais également sur les règles permettant de réduire la diversité d'un composant dans le but de définir des standards.
Le second atelier s'adresse aux acteurs projet de l'entreprise dans lequel ils créent des regroupements de règles spécifiques à leur projet. L'objectif, dans cette étape, est de déterminer les différents points bloquants dans le projet, afin d'établir le plus rapidement des plans d'action.
Quant au dernier atelier, il permet de générer des applications métier et modèles 3D paramétrés à partir des regroupements définis par les acteurs métier, permettant d'aider les concepteurs dans la réalisation du modèle 3D.

KrossRoads fonctionne donc comme un référentiel dynamique de règles spécifiquement organisées et gérées en version, permettant d'aboutir à la définition de produits standards. Au travers de sélection par filtres en relation avec un produit à réaliser, les concepteurs ont accès à une sélection de règles qu'ils doivent impérativement satisfaire dans leur modèles métiers. En revanche, il n'existe pas, dans KrossRoads, de gestion de la cohérence automatique, laissée à la discrétion des utilisateurs. En effet, l'application est d'avantage tournée vers un déroulement séquentiel des activités nécessitant moins de collaborations entre les acteurs.

2.4.4 E2KS

E2KS (Engineering Enterprise Knowledge System) est un outil américain développé par la société Emergent Systems. E2KS répertorie et organise le savoir-faire technique grâce à un réseau très structuré et évolutif. Il associe les utilisateurs et les applications logicielles à ce savoir-faire permettant ainsi :

- La représentation unique des connaissances dans une finalité de réutilisation à objectifs multiples,
- Le suivi de la conformité avec les règles de conception,
- La connexion directe aux logiciels CAO de type Unigraphics ou CATIA

E2ks est basé sur le concept de "Knowledge Packet" ou KPac. C'est une approche spécifique architecturée comme un atome (modélisation par entité) de gestion du savoir-faire technique et du déploiement rigoureux des standards de l'ingénierie. Les KPac permettent alors de capitaliser le savoir-faire des concepteurs sous forme de paramètres, de règles, de graphiques, de tableaux, etc., et de les réutiliser dans des check-lists correspondantes à des vues métiers particulières. Ainsi, chaque concepteur dispose de sa check-list contenant des instances des Kpac sélectionnées sur lesquelles il se base pour son activité de conception. Les check-lists peuvent être synchronisées avec les applications métiers de manière à faire évoluer dynamiquement les connaissances.

De notre point de vue, E2KS semble être la solution la plus aboutie en rapport avec notre problématique, néanmoins l'outil est limité en termes de collaboration, dans la mesure où il est incapable de multi-représenter les connaissances qu'il capitalise, mais aussi incapable de comparer plusieurs check-lists entre elles afin d'identifier les conflits entre utilisateurs. Dans le cadre de notre proposition, nous tiendrons particulièrement compte des concepts et des avantages mis en avant par cet outil.

2.5 Conclusion

Dans ce second chapitre, nous proposons un état de l'art basé sur des recherches et solutions existantes portant sur différents aspects : la gestion des données, les modèles produits, la gestion des connaissances, ainsi que la gestion de configuration, conformément au contexte des travaux de recherche évoqués au chapitre un.

Ainsi, il existe bien sur le marché des outils de gestion de données de simulation (SDM) ou de données produits (PDM), mais ces outils ne sont pas faits pour gérer les paramètres et les règles encapsulés dans les différents modèles métiers. Il est certes possible d'y stocker de façon structurée des résultats de calculs et des ensembles de paramètres, mais toutes les fonctions de regroupement dynamique de paramètres, de propagation de contraintes entre paramètres via des règles métier, de recherche avancée dans des domaines de valeur bornés mais non fixés, etc., sont absentes de ces outils.
Les modèles produits constituent alors une approche complémentaire permettant de soutenir les outils de gestion de données en prenant en compte un plus large spectre de types de données, mais aussi d'informations, selon différents points de vue. En effet, dans la mesure où ils sont capables de structurer les informations selon des points de vue structurel, fonctionnel, processus, etc., indépendamment des outils métiers, les modèles produits permettent une meilleure gestion du projet et favorisent la collaboration des acteurs du processus de conception. Néanmoins, la prise en compte des paramètres et des règles semble encore insuffisante pour leur permettre d'avoir un cycle de vie en totale indépendance avec les outils métiers ou un projet de conception en particulier, limitant ainsi leur réutilisation.
Dans ce contexte, la prise en compte des connaissances permet d'ouvrir le champ à d'autres principes de gestion et de capitalisation qui, couplés à des mécanismes de structuration de modèles, de gestion en configuration, ou encore des concepts d'objets de connaissances, semblent permettre d'atteindre un niveau d'abstraction plus élevé et donc une meilleure prise en compte des paramètres et des règles métiers.

Enfin, l'analyse de différents outils ou approches, identifiés comme adjacents à notre contexte de recherche, met en évidence des manques dans les méthodologies ou les plateformes de gestion et de communication au niveau des connaissances d'une granularité fine. Néanmoins, une volonté commune de capitaliser et partager autour de ces dernières subsiste, afin d'offrir une solution, non pas alternative, mais plutôt complémentaire aux outils de gestion de données ou aux modèles produits.

Ainsi, face aux différents problèmes et constats, évoqués aux chapitres précédents, plusieurs interrogations émergent :

- Comment pallier les problèmes d'incohérence lors du partage de paramètres et de règles métiers entre modèles métiers hétérogènes ?
- Comment fiabiliser et rationaliser le processus de conception, en phase amont, particulièrement lors de la recherche et de la définition de principes d'architectures du produit et de son comportement, face à des sollicitations multi-physiques ?
- Comment améliorer la prise en compte et la gestion des informations et des connaissances afin de favoriser leur réutilisation dans les projets utilisant les approches d'intégration CAO/Calculs ?
- Comment mettre en place une gestion du niveau de maturité du cycle de vie des paramètres et des règles partagés entre plusieurs modèles métiers en parallèle, en tenant notamment compte des notions de versions ?

Notre problématique de recherche consiste donc à développer une nouvelle approche regroupant des méthodes et des outils permettant une meilleure collaboration entre les acteurs d'un projet et une meilleure interopérabilité entre les modèles métiers. Cette approche a pour finalité l'optimisation de la qualité et de la productivité des processus d'ingénierie liés au couple produit-simulation, afin d'améliorer la traçabilité des choix de conception au niveau des informations et des connaissances manipulées. Pour ce faire, nous orientons notre travail de recherche vers une gestion à un niveau de granularité plus fin des informations et des connaissances en configurations, et ceci très en amont du processus de conception.

Notre problématique est en adéquation avec les besoins actuels des industriels, tels qu'exprimés par PSA – Peugeot Citroën et EADS en Annexe 12.

Ainsi, au cours des deux prochains chapitres (chapitre trois et quatre), nous présentons notre contribution portant sur la gestion des connaissances en configurations pour le domaine de la conception de produits. Ensuite, dans la cadre du cinquième chapitre, nous expérimentons notre approche sur deux cas d'études à partir desquels nous tirons les conclusions quant à la pertinence de notre proposition au regard de la problématique formulée. Le sixième et dernier chapitre synthétise notre contribution et soulève la discussion sur les perspectives de recherche qu'il conviendrait de réaliser.

Chapitre 3. La méthodologie KCMethod pour la gestion des connaissances en configurations

Dans le cadre de ce troisième chapitre, nous décrivons notre approche de gestion des connaissances en configurations centrée sur les modèles métiers. Dans un premier temps, nous présentons notre démarche en nous appuyant sur des observations du milieu industriel. Ces observations nous ont permis d'identifier les données, les informations et les connaissances utilisées, et de définir comment elles sont manipulées. Ensuite, nous décrivons, de manière globale, la méthodologie KCMethod que nous proposons pour gérer les connaissances en configurations. Nous expliquons en détails les différents concepts et fonctionnalités.

3.1 Contexte et démarche générale

La présentation de notre contribution se fait en deux temps (Figure 28). La première étape, décrite dans le cadre de ce chapitre trois, présente la méthodologie supportant la gestion des connaissances en configurations. La seconde étape, quant à elle, présentée dans le chapitre quatre, met en évidence la formalisation de cette méthodologie sous forme d'un méta-modèle UML appelé KCModel. Nos travaux ont été menés de manière "incrémentale". En effet, il a d'abord été nécessaire de définir les premiers éléments de la méthodologie. Ensuite, il est apparu fondamental de chercher un formalisme, et c'est ici que la modélisation en langage objet s'est imposée. Au fur et à mesure de leur apparition, tous les concepts ont donc été modélisés et intégrés dans le méta-modèle. Ainsi, nous avons pu "stresser" le méta-modèle afin de s'assurer que, premièrement les concepts élaborés participaient à la résolution de la problématique, mais aussi que le méta-modèle était correctement construit et représentatif de la méthode.

La méthodologie KCModel, aussi appelée KCMethod proposée dans ce chapitre, se focalise sur la gestion des paramètres et des règles encapsulés dans les modèles de conception et de simulation utilisés dans un contexte d'ingénierie collaborative. Ces connaissances nécessitent d'être manipulées dynamiquement selon de nombreux points de vue relatifs aux différents métiers, acteurs et outils utilisés lors du processus de conception. En effet, elles sont à l'origine de très nombreuses décisions de conception, cependant à l'heure actuelle, très peu d'outils semblent capables de les prendre correctement en compte et ainsi d'avoir une gestion des conflits lors du partage de ces connaissances.

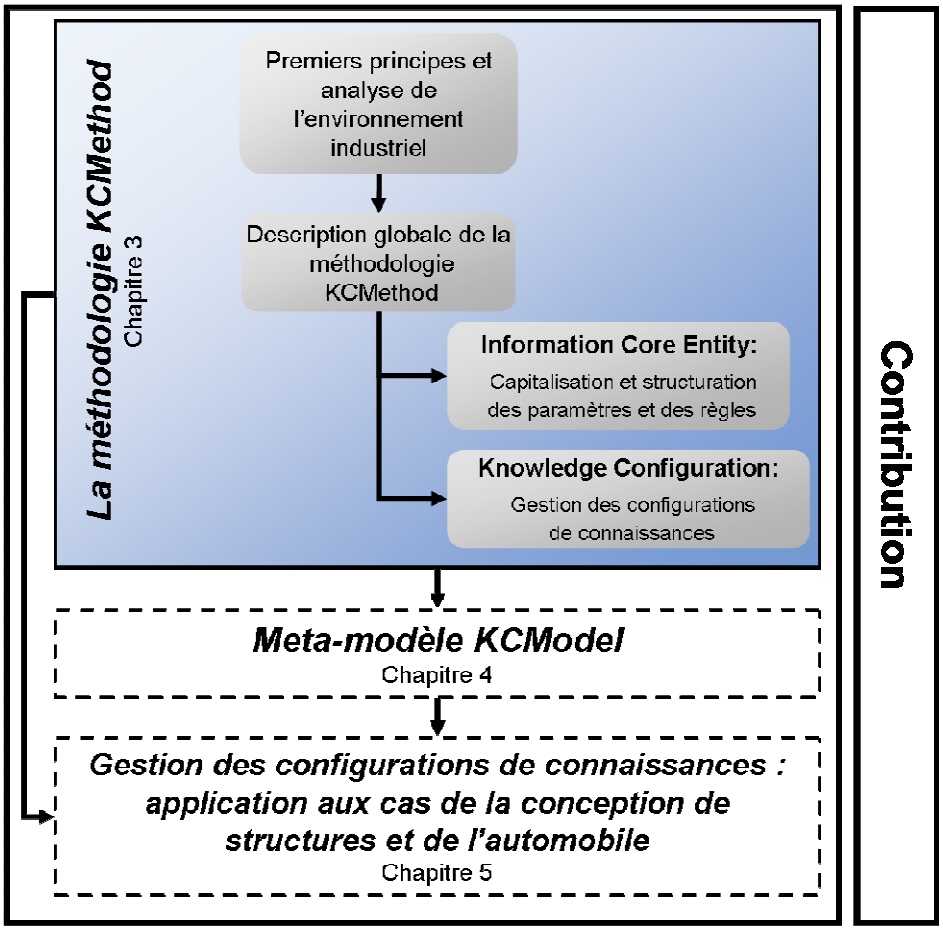


Figure 28 : organisation de la contribution pour la définition de la méthodologie KCMethod

Ainsi, dans une première partie de ce chapitre, nous définissons nos premiers principes méthodologiques, issus d'observations de l'environnement industriel. Cette première réflexion nous permet de tracer le contour de notre approche, décrite de manière globale dans la seconde partie de ce chapitre. La démarche de capitalisation des connaissances, de construction et

d’utilisation des configurations est étudiée afin d’introduire et de mettre en place toutes les notions et concepts fondamentaux de l’approche : le concept d’ICE (Information Core Entity) et de KC (KnowledgeConfiguration).
Dans une troisième partie, ces concepts fondamentaux sont détaillés, et un intérêt particulier est porté à leurs apports dans le cadre de notre problématique. Cette dernière partie a vocation à entrer dans le détail de la méthodologie et peut s’apparenter à une forme de spécification, dans la mesure où ces travaux de recherche sont de nature à aboutir au développement d’une application logiciel qualifiée de KCManager pour la gestion des connaissances en configurations du couple produit-simulation.

3.2 Analyse de l’environnement industriel et premiers principes

Cette partie est dédiée à l’identification des premiers principes de solutions pour la définition de notre méthodologie. Ces premiers principes sont issus, à la fois des constats effectués aux chapitres précédents (chapitre un et deux), mais aussi issus d’analyses de l’environnement industriel, notamment au sein du groupe PSA Peugeot Citroën. Ces principes sont repris dans le développement de la méthodologie KCMethod.

3.2.1 Cartographie des connaissances

Dans le cadre de notre proposition, l’objectif que nous poursuivons est d’aller plus loin que l’échange des connaissances directement entre modèles métiers (via des formats neutres d’échanges par exemple). Nous voulons disposer d’une plateforme centralisant ces connaissances assurant leur gestion en configuration, en tenant compte de leur cycle de vie, leur maturité, etc. Notre objectif de centralisation et de gestion n’est atteignable qu’avec une bonne compréhension des processus d’utilisation et de partage des connaissances.
Ainsi, pour aller au delà du simple interfaçage de modèles métiers utilisant différents outils, il nous semble nécessaire de cartographier les connaissances et les outils utilisés dans le processus de conception, plus particulièrement en phase amont du couple produit-simulation. Plus précisément, la cartographie nous permet d’identifier les connaissances candidates au partage et à la collaboration, et de définir des "principes" pour une meilleure prise en compte de ces connaissances. Pour établir cette cartographie, nous avons réalisé des réseaux sémantiques (Annexe 13) permettant d’identifier les connaissances en relation avec notre contexte de recherche, défini dans le chapitre 1. La Figure 29 illustre un exemple de réseau sémantique, permettant d’identifier certains objets manipulés par les concepteurs et les analystes, ainsi que leurs relations mutuelles dans un processus classique de conception de produits. Par exemple, un "paramètre géométrique" se voit systématiquement associé à une unité, des bornes inférieures, supérieures et une valeur. Autre exemple, "un paramètre calcul" est lié à une "filière de simulation" généralement découpé en paramètre pré et post simulation, eux-mêmes découpés en divers types de paramètres.

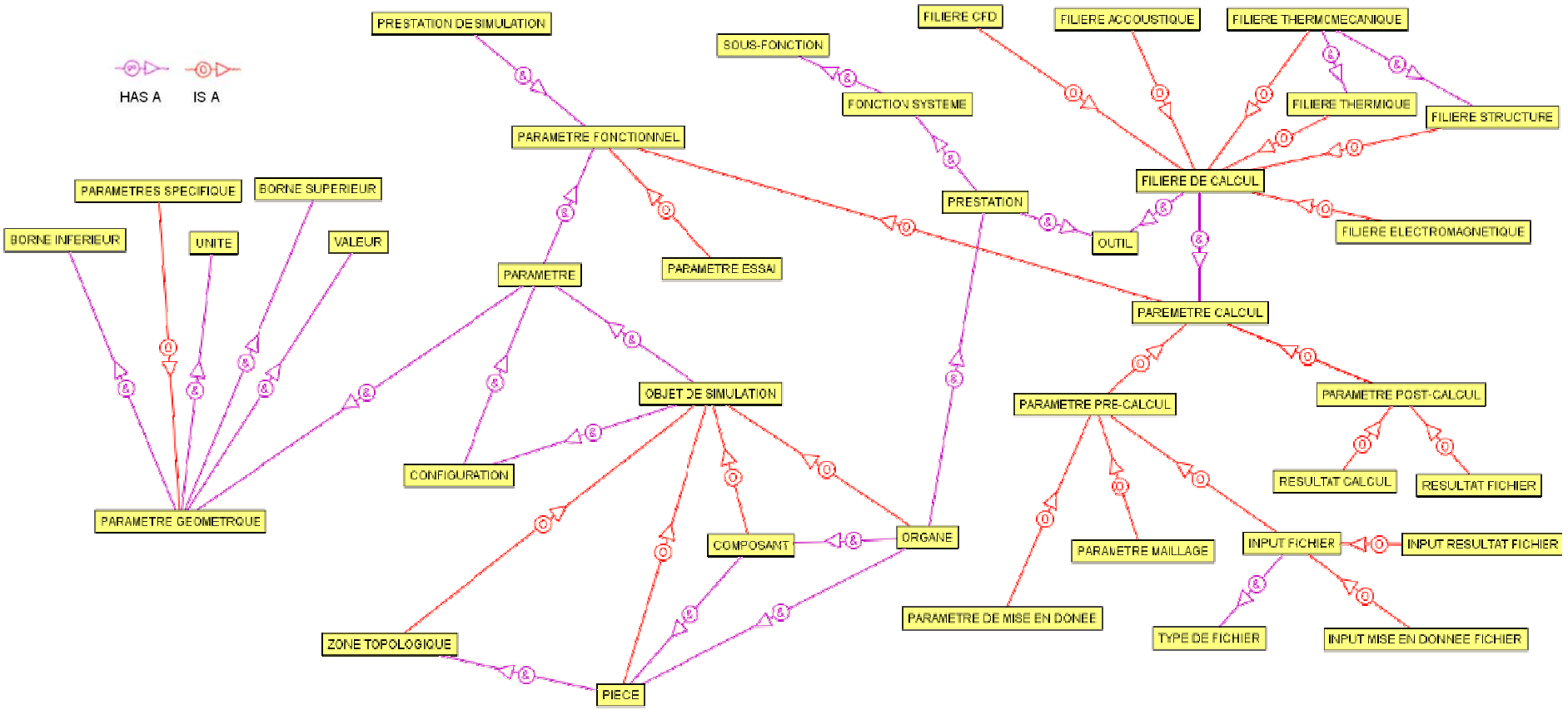


Figure 29 : exemple de réseau sémantique réalisé lors de notre analyse

Notre analyse, menée par des réseaux sémantiques, nous permet de tirer quelques grandes conclusions, résumées sous forme de cinq constats (Tableau 3), qui servent de support à l'élaboration de notre méthodologie.

3.2.1.1 Premier constat

Le premier constat réalisé, suite à l'analyse des réseaux sémantiques, concerne le nombre et les types de paramètres et de règles à prendre en compte dans le cadre de notre problématique :

Il n'est pas nécessaire de gérer la totalité des paramètres et des règles de conception pour assurer l'interopérabilité et la cohérence entre les modèles métiers de conception et de simulation.

En effet, le nombre de paramètres et de règles pour un produit peut être considérable. Par exemple, on estime à environ 10000 ce nombre pour la conception d'un Groupe Moto-Propulseur automobile. Or, parmi ces paramètres et règles, il apparaît que pour une grande partie, ils ne **participent à aucun processus de partage ou de collaboration**. C'est-à-dire qu'ils sont utilisés localement dans un modèle métier, par un acteur donné, et nulle part ailleurs.
Second point, lors des phases amont du processus de conception, les modèles de pré-dimensionnement, par exemple, **encapsulent un nombre limité de paramètres**, souvent jugés directeurs, car représentatifs de spécifications et des principes d'architectures majeurs du produit (cf. 1.2.2.2). Ces paramètres "directeurs" sont partagés par de nombreux acteurs dans plusieurs filières métiers, et sont fréquemment modifiés afin d'identifier les meilleurs concepts.

Ces différentes observations nous permettent d'estimer, pour la conception d'un GMP, qu'il est possible de diviser, au minimum de moitié, le nombre de paramètres à prendre en compte.
Nos observations en entreprise, ainsi que notre modélisation à l'aide des réseaux sémantiques, nous ont permis d'identifier cinq classes de paramètres à prendre en compte, qui sont jugés comme les paramètres cruciaux [Grundstein, 2000] partagés dans les modèles métiers :

- ***les paramètres partagés par plusieurs modèles métiers*** (paramètres liés à des interfaces entre pièces, paramètres d'entrée/sorties pour différentes études, etc.)
- ***les paramètres directeurs*** qui sont réellement les paramètres clefs du produit ou des données à absolument prendre en compte, car issues d'exigences ou de normes à respecter (norme Euro 6 par exemple)
- ***Les paramètres matériaux*** (les données matériaux sont souvent des données clefs et très protégées lors de la conception de produits innovants)
- ***Les paramètres géométriques*** (paramètres principaux liés à la géométrie d'un ou plusieurs composants)
- ***Les paramètres physiques*** (paramètres principaux liés à la physique appliquée lors d'études sur différents composants comme des forces, températures etc.)

Un paramètre peut appartenir à plusieurs classes, et c'est d'ailleurs souvent le cas pour les deux dernières qui regroupent des paramètres qui rentrent également dans au moins une des trois premières classes.
Les paramètres ne sont pas les seuls éléments dont il faut tenir compte lors de la collaboration des acteurs ou de leurs choix d'architectures. En effet, les règles métiers utilisées dans les modèles des concepteurs impactent un ou plusieurs paramètres contraignant les valeurs ou les domaines de valeurs qu'ils peuvent prendre. Nous avons identifié quatre classes sur lesquelles nous reviendrons lors de la définition de notre proposition :

- ***Les relations mathématiques***
- ***Les règles logiques***
- ***Les tableaux de valeurs discrètes***
- ***Les conditions limite***

3.2.1.2 Second constat

Le second constat réalisé met en lumière la nécessité d'associer aux paramètres et aux règles, certaines informations :

Un paramètre, ou une règle, doit toujours être indissociable de certaines informations qui sont absolument nécessaires à son utilisation, et qui, si elles sont absentes, peuvent être source d'incohérences.

Ainsi, lors de la collaboration des acteurs dans le cadre de leurs activités métiers, un paramètre doit être caractérisé par les informations suivantes :

- un nom intelligible pour chaque acteur en fonction de son domaine d'expertise,
- une unité (liée à la grandeur du paramètre),
- des bornes,
- une personne responsable ou experte sur ce paramètre,
- un contexte d'utilisation, par exemple dans quelle phase du processus de conception il est utilisé, par quelle personne, dans quel objectif, etc.

En contexte industriel, lorsque des paramètres sont partagés par différents acteurs, il est rare que l'ensemble de ces informations leur soit systématiquement attaché, et ce, quel que soit le vecteur de communication utilisé (téléphone, tableur, fichier partagé sur un réseau, etc.). Même les outils de type SGDT, plus focalisés sur les contenants, c'est-à-dire les modèles qui encapsulent les paramètres, ne gèrent pas ce niveau de granularité d'information pour chaque paramètre (ou règle).

3.2.1.3 Troisième constat

Le troisième constat, réalisé suite à l'analyse des réseaux sémantiques, porte sur la multi-représentation d'un paramètre :

Les paramètres et règles ont plusieurs niveaux de représentations dont il faut tenir compte dans les processus de collaboration.

Il apparait souvent que certains paramètres, utilisés dans plusieurs modèles métiers, soient en réalité le même paramètre, même s'ils portent des noms différents. Par exemple, un même paramètre sera appelé "diamètre piston" dans un modèle piston, et sera appelé "diamètre alésage" dans un modèle de carter cylindres. Cette multi-représentation du nom du paramètre est due aux vues métiers dans lesquelles il est utilisé, mais provoque des incohérences quand il s'agit de mettre en relation deux modèles.
De même que pour le nom, un même paramètre peut être utilisé avec différents "types" ; sous forme scalaire dans un modèle métier, sous forme de courbe ou de surface de réponse dans un autre. C'est d'ailleurs un cas d'école chez PSA Peugeot Citroën avec le paramètre "pression cylindre". Dans ce cas, il est difficile de comparer facilement ces paramètres afin d'assurer la cohérence des modèles, surtout quand le nombre de paramètres est important. Pour conclure sur ce troisième constat, nous pouvons affirmer qu'un même paramètre doit pouvoir posséder plusieurs représentations au niveau de son nom et de son type, de manière à faciliter la traçabilité et le maintien de la cohérence.

3.2.1.4 Quatrième constat

Le quatrième constat observé concerne la notion de version :

Un utilisateur doit pouvoir créer plusieurs versions en fonction de la modification de la valeur des paramètres ou de leur utilisation.

Il est essentiel de prendre en compte le niveau de maturité, le cycle de vie et l'évolution des paramètres et des règles. C'est une notion qui revient souvent quand on interroge les concepteurs. Ils souhaitent pouvoir tester plusieurs jeux de valeurs de paramètres dans un modèle, et en garder une trace sous forme de version. Aujourd'hui, il est possible de sauvegarder le fichier complet du modèle métier et de le gérer en versions dans un SGDT, pour garder un historique de ces modifications. Néanmoins, ce mode de sauvegarde est plutôt opaque quand on veut observer l'impact de la modification de certains paramètres, en particulier sur une architecture. En effet, les paramètres et les règles étant stockés dans le modèle métier, il est indispensable, pour les visualiser et les modifier, d'emprunter le fichier, et nécessairement de disposer des outils permettant de l'ouvrir. La gestion des versions doit pourvoir être réalisée à un niveau plus fin, sur chaque paramètre (ou plutôt groupes de paramètres), afin de garder une trace des différentes valeurs testées, ou bien même une trace des associations de paramètres dans un modèle. Par exemple, dans le modèle "Piston", un concepteur utilise les paramètres "A", "B" et "C". Suite à des modifications, il n'utilise plus le paramètre "B" qu'il remplace par une relation "A=C+10" et un nouveau paramètre "D". Le système de version doit permettre à l'utilisateur d'avoir une trace de ces modifications, voire même l'autoriser à revenir sur des versions antérieures, ou maintenir plusieurs versions en parallèle.

3.2.1.5 Cinquième constat

Enfin, le cinquième et dernier constat, observé suite à l'analyse de l'ensemble des réseaux sémantiques, concerne la possibilité, voire la nécessité de grouper des paramètres ou des règles, afin de faciliter leur utilisation :

Les paramètres et les règles métiers sont très souvent associés les uns aux autres et ce, quel que soit leur utilisation. Il faut alors pouvoir grouper les paramètres et/ou règles quand c'est possible, sous forme d'entité indissociable

C'est-à-dire que certains paramètres et règles peuvent être groupés et toujours utilisés ensemble. Cette organisation facilite la manipulation de ces paramètres et règles. A titre d'exemple, deux paramètres "pas de vis" (p) et "hauteur de filetage" (H) sont utilisés ensembles dans les modèles de conception. De plus, une règle NF ISO 724 les lient telle que H= 0,866*p. Cette règle doit être systématiquement vérifiée lors de l'utilisation de ces paramètres. On peut donc créer un groupe, de sorte à ce que ces deux paramètres et cette règle soient considérés comme étant **une seule entité indissociable et utilisable dans plusieurs modèles.** De cette façon, on fiabilise le processus de conception en obligeant les utilisateurs à prendre en compte et respecter certaines règles dans leurs modèles métiers. Ce groupement peut-être très utile pour définir des standards ou des normes à impérativement satisfaire lors du processus de conception.

3.2.1.6 Synthèse

Ces constats sont synthétisés sous forme de tableau (Tableau 3) résumant notre analyse de l'environnement industriel.

Premier constat	*Il ne faut tenir compte que des paramètres et des règles jugés cruciaux*
Deuxième constat	*Un paramètre (et une règle) doit toujours porter certaines informations pour éviter les incohérences*
Troisième constat	*Les paramètres et règles ont plusieurs niveaux de représentations*
Quatrième constat	*Il doit être possible de créer plusieurs versions en fonction de la modification de la valeur des paramètres ou de leur utilisation*
Cinquième constat	*Il doit être possible de grouper les paramètres et/ou règles quand c'est possible, sous forme d'entité indissociable*

Tableau 3 : conclusions à l'issue de l'analyse des réseaux sémantiques

En plus de ces cinq constats, les réseaux sémantiques, et surtout les difficultés rencontrées pour les réaliser, nous ont permis de nous rendre compte que la structuration des connaissances et leur utilisation, particulièrement dans les phases amont, est plutôt désorganisée. En effet, ces phases impliquent généralement des activités de recherches de solutions par les concepteurs et donc, les outils et les modèles métiers utilisés ne sont pas systématiquement supportés par un processus ou même stockés dans un SGDT. Cette non-organisation est à l'origine d'importantes pertes de temps car il est très difficile de définir quel utilisateur manipule telle connaissance, à tel moment et donc de s'assurer que les connaissances partagées sont cohérentes. A ce titre, l'analyse des réseaux sémantiques nous permet de faire des recoupements avec les chapitres un et deux dans lesquels nous avions déjà identifiés que les problématiques de collaborations étaient plus importantes dans les phases amont à cause de l'absence de support aux premières activités de conception.

Ainsi, pour assurer une meilleure prise en compte et une meilleure gestion des connaissances en phase amont du processus de conception, nous nous appuyons sur la cartographie réalisée et les cinq constats pour bâtir les fondations de notre méthodologie.

Si l'analyse menée par les réseaux sémantiques permet d'identifier des objets et leurs relations mutuelles, elle ne permet pas de mettre en évidence la façon dont les utilisateurs les partagent au sein des activités du processus de conception. Il est alors nécessaire d'étudier le modèle d'activité d'une organisation sur l'aspect particulier des échanges de paramètres et de règles, afin de définir les premiers principes de solution d'une base commune des informations techniques du produit.

3.2.2 Vers une base commune des informations techniques du produit

Ainsi, voici donc la première approche de mise en commun des informations au travers de la notion de base commune des informations techniques du produit.

La première étape de définition d'une base commune commence par l'observation du modèle d'activité des concepteurs dans un projet de développement de produit. Dans ce contexte, on peut considérer, de manière globale, qu'il existe des interactions entre des acteurs issus de différents "domaines métiers", impliqués dans plusieurs "activités de conception et de simulation" (Figure 30). Dans ce contexte, un acteur d'un domaine métier peut être amené à travailler sur plusieurs activités

simultanées, et une activité peut être impactée par plusieurs domaines métiers à la fois. Ainsi, les échanges de données entre domaines et activités doivent se faire dynamiquement et impliquent donc de nombreux acteurs dans le processus de conception.

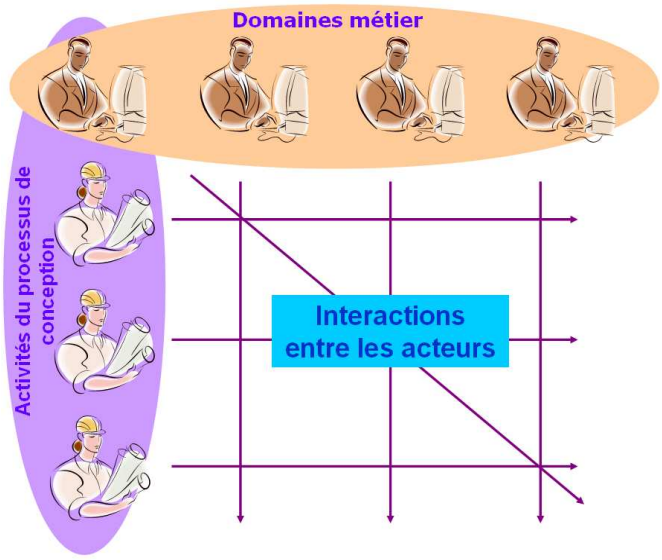


Figure 30 : interactions des domaines et des acteurs d'une organisation

Le modèle d'activité souvent observé dans les organisations est représenté (Figure 31). Il illustre la problématique de partage et de synchronisation d'un paramètre appartenant à des activités et des métiers différents. En effet, les informations nécessaires à une activité en particulier, qu'elle soit mono-métier ou transverse à plusieurs domaines d'expertises, sont souvent cloisonnées dans cette activité. Ainsi, si une autre activité dans un autre domaine métier a besoin de cette information, elle se retrouve dupliquée, et la plupart du temps sans lien avec la première. Cela signifie également que, si dans une activité une information vient à évoluer, propager cette évolution à une autre activité est loin d'être aisé.
Dans l'exemple Figure 31, le paramètre 2 est dupliqué afin de pouvoir être utilisé par deux concepteurs, favorisant l'apparition de disfonctionnement comme des incohérences ; le paramètre mène, en parallèle, deux "vies" différentes (différentes valeurs, différents évolutions, etc.). De plus, ce fonctionnement rend difficile la création de relations (mathématiques ou règles logique) entre des paramètres appartenant à deux activités distinctes, ce qui limite la prise en compte de contraintes inter-métier pourtant inévitables.

Dans ce contexte, le premier principe de solution que nous envisageons serait de faire en sorte que les informations nécessaires à plusieurs utilisateurs soient centralisées et gérées de façon globale quelle que soit l'activité (à droite de la Figure 31).

De cette manière, on évite les phénomènes de duplication et la modification d'une information peut directement être prise en compte dans l'ensemble des activités du processus de conception. La mise en commun des informations permet également de créer des relations entre paramètres d'activités et de métiers différents (relations 8 et 10), difficiles à prendre en compte auparavant, et garantissant une meilleure cohérence et une meilleure prise en compte des contraintes issues de divers domaines d'expertises.

En conséquence, les paramètres et les relations ainsi partagés permettent de satisfaire aux différentes interactions au sein de l'organisation. On considère que cette base de paramètres et de règles communes à toute activité de conception ou de simulation, quel que soit le domaine métier, constitue une "base centralisée d'informations techniques du produit".
On passe alors d'un modèle d'activité où les informations sont totalement cloisonnées à une mise en commun des informations utilisables par les acteurs d'un projet.

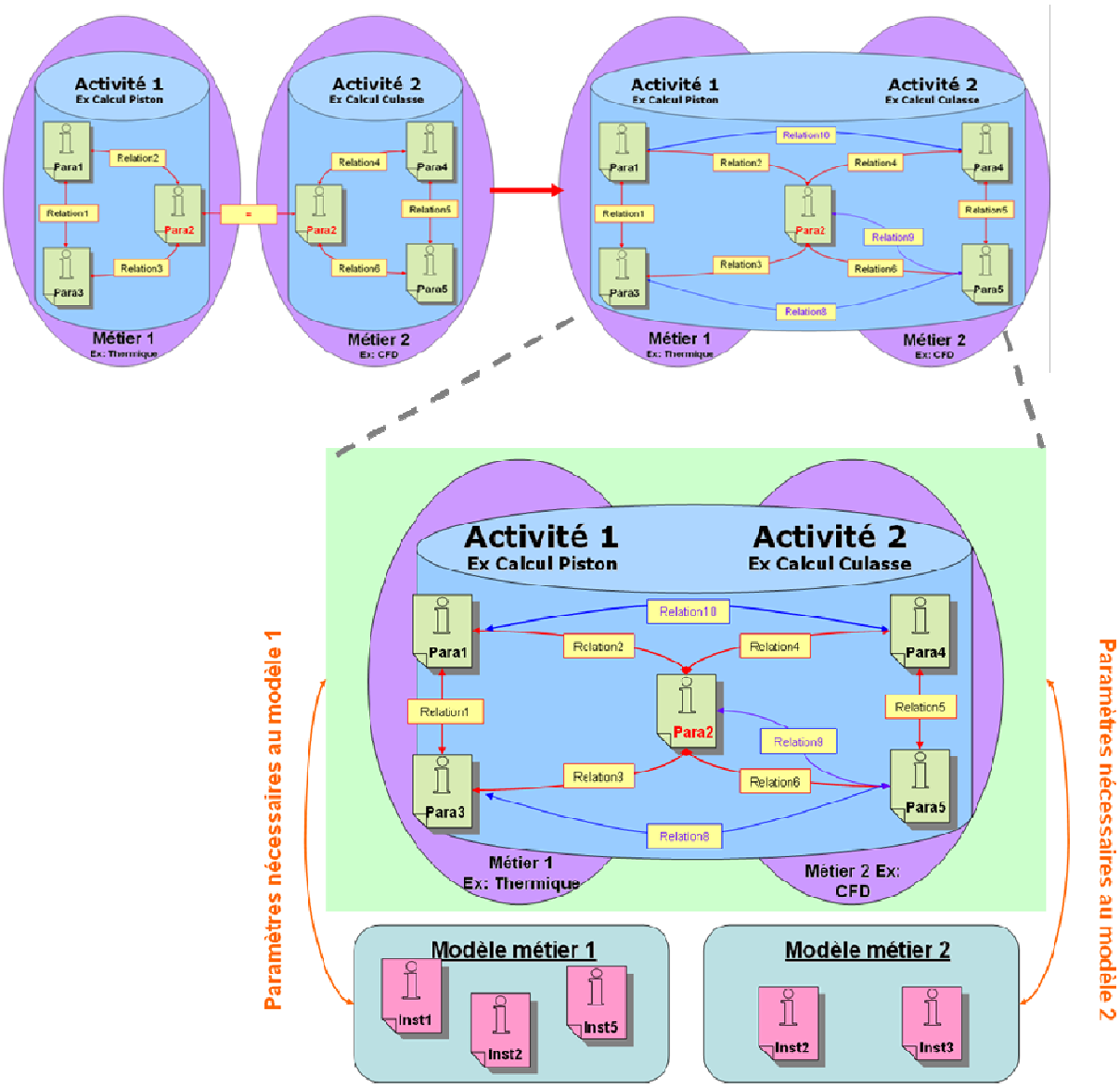


Figure 31 : vers un modèle d'activité favorisant la partage des objets. Création de configuration d'un produit par l'intermédiaire d'instances héritées des objets du socle de données

Ensuite, pour réutiliser les informations dans un modèle métier particulier en rapport avec chaque activité, il est nécessaire de sélectionner depuis la base, les informations utiles et de créer *des instances* vers les modèles métiers (Figure 31).
L'instance possède une valeur en lien avec son contexte d'utilisation bien particulier, ce qui signifie que l'information stockée dans la base doit posséder un caractère suffisamment générique pour être utilisée selon différents points de vues dans plusieurs activités de conception et de simulation. La synchronisation et la cohérence des instances sont assurées par "l'objet parent", de ce fait nous utilisons la notion d'héritage pour mettre en relation différentes instances d'une information partagée par différents acteurs, dans différents modèles métier. Si un objet est modifié, l'ensemble des instances pourront être impactées par ce changement, inscrivant de cette façon le modèle d'activité dans un processus dynamique.
Ainsi, une équipe de concepteurs travaille de manière collaborative à l'instanciation d'un même artefact depuis une base générique et transverse. Le système doit être capable de capter en temps réel les décisions des concepteurs concernant les variables du produit, de les propager et de gérer les incompatibilités entre décisions, de manière à converger vers un artefact totalement instancié en minimisant le nombre de décisions remises en cause. Dans ce contexte, on considère qu'une sélection d'instances réalisée depuis la base d'information, pour être utilisée dans un modèle métier particulier,

préfigure un autre principe de la méthodologie. En effet, ce paquet d'instances a pour vocation d'avoir un cycle de vie propre et est de nature à subir des modifications de manière synchrone avec l'utilisation du modèle métier. De plus, il est nécessaire d'assurer la cohérence globale de toutes les instances lors de ces processus dynamiques, et de prendre en compte les modifications subies par les instances. Cette notion rappelle les principes de gestion de configuration décrits dans les normes ISO10007 :1995 et 100007 :2003.

Ainsi, en plus de permettre la capitalisation et la centralisation les paramètres et les règles, de sorte à créer une base technique générique d'informations, nous définissions un objet "configuration" correspondant à une sélection de paramètres et de règles instanciées et synchrone avec un modèle métier.

3.2.3 Une première solution

Compte-tenu des constats réalisés, des orientations prises quant à la définition d'une base centralisée des informations techniques du produit, ainsi que de l'utilisation des mécanismes de gestion de configuration, nous proposons un premier positionnement d'un outil de gestion des paramètres et des règles en configurations qualifié de KCManager (Figure 32).

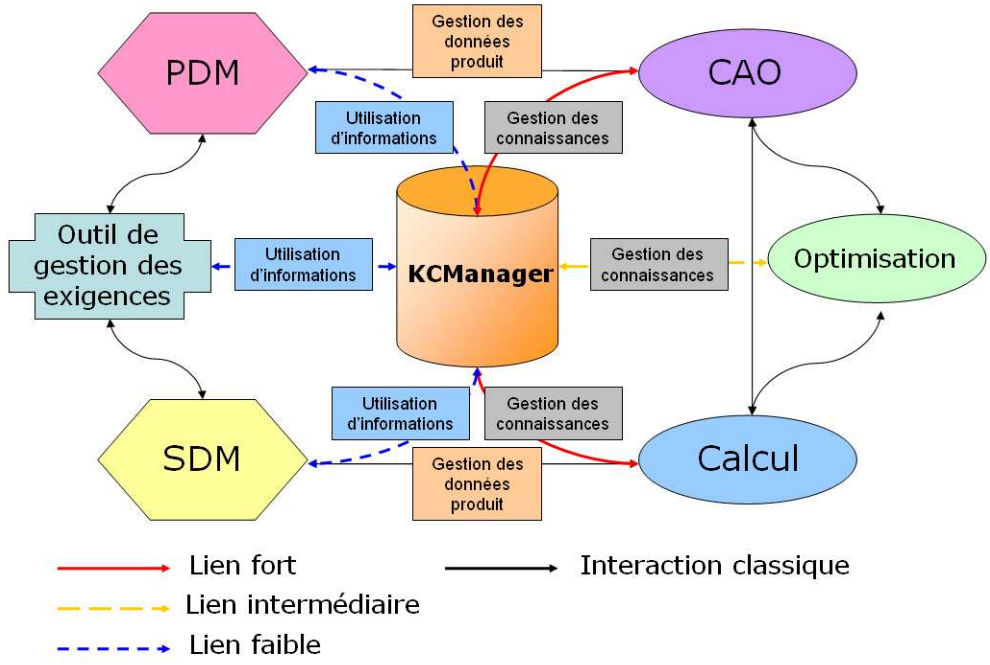


Figure 32 : premier positionnement d'un outil de KCManager

Cet outil KCManager (Knowledge Configuration Manager) sert de base pour la gestion centralisée des connaissances de type paramètres et règles métiers. Il est en mesure de les capitaliser, d'assurer une traçabilité et de permettre leur réutilisation dans différents modèles métiers, selon différentes vues, tout en analysant les connaissances utilisées afin d'assurer la cohérence entre les modèles. Cet outil est à l'interface, à la fois des applications de gestion de données de type PDM et SDM contenant les informations du produit (gestion documentaire des modèles etc.), des outils métiers pour la modélisation ou la simulation (CAD, CAE, Outils d'optimisation, etc.), mais aussi d'outils dédiés à des processus d'ingénierie spécifique comme la gestion des exigences.

3.3 Méthodologie de gestion des données, des informations et des connaissances en configurations : KCMethod

La partie 3.2 permet d'illustrer le cheminement intellectuel et scientifique réalisé à la suite de l'analyse du contexte et de la problématique de ces travaux de recherche. Ces premières déductions et orientations nous permettent de mettre en avant les origines de notre proposition de méthodologie de gestion des connaissances en configurations.
Cette méthodologie, baptisée KCMethod - Knowledge Configuration Method, est focalisée sur la gestion des paramètres et des règles encapsulés dans les modèles métiers que nous considérons comme de la connaissance (cf. section 2.3.2). KCMethod est basé sur une gestion en configurations de ces connaissances encapsulées afin d'assurer une bonne capitalisation, traçabilité, réutilisation et cohérence dans les activités du processus de conception.

3.3.1 La gestion des connaissances dans KCMethod

L'objectif de ces recherches est de nature à définir une méthodologie pour aider les concepteurs d'un produit à assurer la cohérence des données, informations et des connaissances qui sont partagées dans de nombreux modèles de conception et de simulation hétérogènes.

Dans ce contexte, nous considérons comme étant (cf. section 2.3.2) :

- De la **donnée**, des paramètres et des règles extraits des modèles métiers sans aucune analyse ou contexte associé.
- De l'**information**, des données qui prennent un sens et qui sont organisées en vue d'être exploitées. Par exemple, quand les données sont capitalisées, structurées et contextualisées. (cf. constat 2 de la partie 3.2.1).
- De la **connaissance**, la réutilisation des informations, dans un contexte précis correspondant à une activité du processus de conception et nécessitant l'interprétation d'un concepteur.

L'approche globale peut être présentée en quatre grands axes (Figure 33), relatifs à la fois aux modèles de gestion des connaissances comme ceux proposé par [Monticolo et al., 2008] ou [Grundstein, 2002], mais aussi relatifs à la problématique **CTRC** (**C**apitalisation, **T**raçabilité, **R**éutilisation, **C**ohérence) expliquée dans le chapitre 1 [Badin et al., 2011].

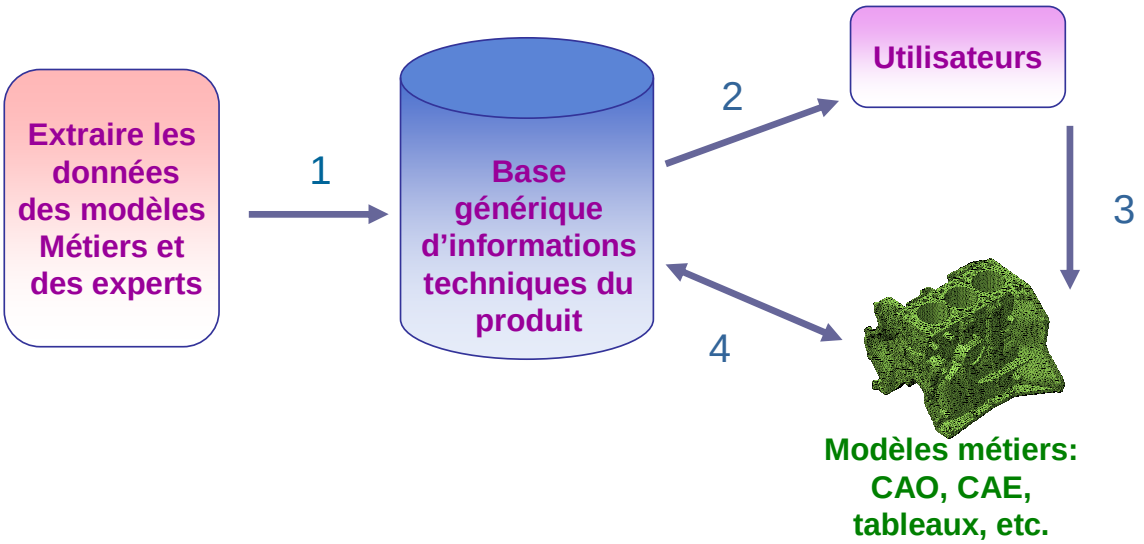


Figure 33 : processus global de la méthode

Première étape, on extrait des modèles métiers et des experts les données de type paramètres et règles jugés cruciaux. C'est-à-dire, uniquement les données qui ont réellement un rôle dans les problématiques de collaboration entre les activités du processus de conception. Comme nous l'avons vu dans la première partie de ce chapitre, ces données concernent cinq classes de paramètres :

- *les paramètres partagés,*
- *les paramètres directeurs,*
- *les paramètres matériaux,*
- *les paramètres géométriques,*
- *les paramètres physiques,*

et quatre classes de règles essentielles à la définition d'un produit :

- *les relations mathématiques,*
- *les règles logiques,*
- *les tableaux de valeurs discrètes,*
- *les conditions limite.*

Par opposition, les paramètres et les règles appartenant à un modèle métier unique, qui ne sont pas partagés ou qui ne relèvent pas de spécifications fonctionnelles ou stratégiques, ne sont pas pris en compte dans la méthodologie.
Une fois ces données extraites, on les capitalise de façon générique (par exemple : nom "diamètre piston", unité en "mm", famille de paramètre "géométrique d'interfaces", etc.), et on les organise de manière à créer une base centralisée et générique d'informations techniques d'un produit. Ainsi, cette base contient l'ensemble des paramètres et des règles cruciaux, nécessaires à la conception de n'importe quel type de produit en particulier. Par exemple, une base d'informations techniques regroupant les paramètres et les règles pour concevoir n'importe quel type de Groupe Moto-Propulseur thermique.

Deuxième étape, permettre à des utilisateurs, concepteurs, experts, chefs de projet ou autre, de parcourir la base d'informations techniques afin de sélectionner les paramètres et les règles qui les intéressent pour leurs activités dans le processus de conception. Cette sélection donne lieu à la création d'une configuration contenant des instances de paramètres et de règles (c'est-à-dire des paramètres avec des valeurs) créées depuis la base générique. Nous considérons, à ce moment, que les utilisateurs manipulent des connaissances car la création des configurations relève de la sélection et de la manipulation des paramètres et des règles instanciés dans un contexte précis du processus de conception. Nous reviendrons en détail sur la notion de configuration.

Troisième étape, permettre aux utilisateurs de réutiliser les connaissances dans leurs modèles métiers quels qu'ils soient. A cette étape, il est aisé d'avoir une traçabilité complète des paramètres et des règles utilisés, c'est-à-dire de savoir quel paramètre est utilisé, par qui, dans quel modèle et dans quel but.

Enfin, dernière étape, analyser la cohérence des connaissances partagées entre plusieurs modèles métiers par différents utilisateurs et capitaliser de nouvelles connaissances et de nouvelles informations qui pourront être réutilisées.

La gestion centralisée des données, informations et connaissances, favorise la collaboration entre les activités métiers gravitant autour de la conception du produit. Cette gestion facilite la réutilisation des connaissances, permet une meilleure compréhension du produit de la part des différentes équipes métiers, et garantit la cohérence au fur et à mesure des modifications du produit.

L'objectif final de cette méthodologie consiste à transiter, d'une situation anarchique d'échange de paramètres et de règles entre les modèles métiers, à une situation où les connaissances sont centralisées, redistribuées avec traçabilité et analysées, afin d'assurer la cohérence des activités du processus de conception.

3.4 Principes et concepts fondamentaux de KCMethod

Dans cette section, nous présentons les deux concepts principaux de KCMethod nous permettant, par la suite, de décrire globalement le processus d'utilisation de la méthodologie. Puis, nous revenons plus en détail sur les différents concepts et leurs fonctionnements.
KCMethod est une méthodologie de gestion en configurations des connaissances encapsulées dans les modèles métiers. Cette méthodologie permet d'avoir un suivi dynamique des évolutions de paramètres et de règles dans le processus de conception d'un produit. Elle est basée sur deux concepts majeurs : le concept d'Information Core Entity (ICE) et le concept de KnowledgeConfiguration (KC).

Une ICE est une entité générique indécomposable qui permet de capitaliser des données cruciales extraites de modèles métiers et d'experts afin de passer de l'état de donnée à celui d'information. Elle est composée de paramètres et de contraintes génériques, de manière à ce que l'ensemble des ICEs constitue une base générique d'informations techniques d'un produit (un référentiel technique). Par exemple, une ICE contenant un rapport de réduction "r", une vitesse d'entrée "Ve", une vitesse de sortie "Vs" et une relation "r=Vs/Ve". La base d'informations est dynamique, évolue et s'enrichie au fil du temps. L'ICE est l'entité de plus bas niveau manipulée dans la méthodologie KCMethod. Pour utiliser les ICEs dans un contexte précis, comme une activité du processus de conception, il faut créer une KnowledgeConfiguration et instancier les ICEs dans cette dernière.

Le concept de KnowledgeConfiguration (KC) est le deuxième élément fondamental de la méthodologie, il caractérise l'originalité de cette approche. En effet, pour réutiliser les ICEs dans un contexte de conception précis (par exemple, l'étude thermique d'un carter cylindre dans un projet moteur), un utilisateur fait une sélection dans la base d'ICEs, de manière à ne retenir que celles dont il à besoin pour son activité. Ensuite, il instancie cette sélection dans une KnowledgeConfiguration synchronisée avec un modèle métier. Ainsi, une KnowledgeConfiguration est donc constituée d'instances d'ICEs, elle est représentative des connaissances encapsulées du modèle avec lequel elle est synchronisée. Cette notion de configuration de connaissances fait référence aux mécanismes de gestion en configuration, c'est-à-dire qu'elle offre tous les services de gestion des versions, de gestion des modifications et de gestion de la cohérence. Ainsi, dans une configuration, un utilisateur pourra sauvegarder plusieurs versions en fonction de la modification des valeurs des paramètres ou de l'ajout ou de la suppression d'instances d'ICEs. L'intérêt des configurations, est d'avoir un mode de représentation homogène des connaissances qui peuvent être utilisées dans différentes activités du processus de conception. C'est par elles que la cohérence des connaissances partagées est assurée. En effet, s'il est très difficile de comparer directement plusieurs modèles métiers entre eux, sachant qu'ils utilisent différents outils parfois incapables de communiquer ensemble, nous comparons les configurations pour donner aux utilisateurs l'information sur l'existence d'éventuels conflits. Par exemple, un paramètre partagé dans deux modèles avec deux valeurs différentes, ou alors une règle métier qui n'est pas respectée.

3.4.1 Processus global KCMethod

L'articulation globale du processus KCMethod est représentée Figure 34 et résumé Tableau 4.

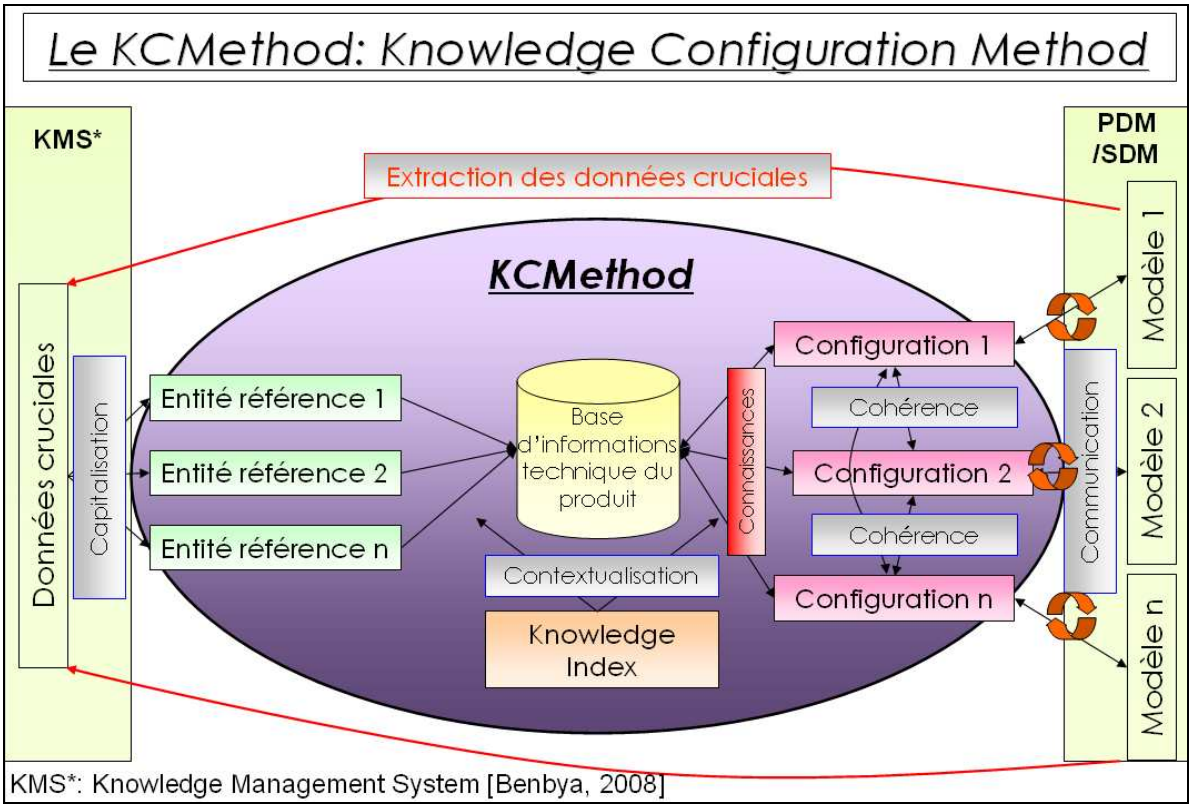


Figure 34 : processus global KCMethod

Une fois les données cruciales extraites des modèles métiers, il est nécessaire de les structurer et de les documenter. Nous utilisons alors les ICEs pour capitaliser les paramètres et les règles de façon générique. Dans la méthodologie KCMethod, l'ICE est l'entité de plus bas niveau manipulée, elle porte un nom, une description, elle a un responsable, un cycle de vie qui lui est propre et peut évoluer dynamiquement. Dans une ICE, on peut capitaliser un ou plusieurs paramètres ou règles, mais aussi créer des relations entre plusieurs ICEs. L'ensemble des ICEs génériques constitue une base d'informations techniques d'un produit (au centre sur la Figure 34).

Si les ICEs sont génériques, c'est-à-dire qu'on ne sait pas à priori dans quel cas précis elles vont être utilisées, elles sont néanmoins contextualisées en fonction de l'ensemble de leurs potentielles utilisations grâce aux Knowledge Index. Il permet d'attribuer des informations (sémantiques) à l'ICE en fonction de ses contextes d'utilisations, c'est-à-dire les composants ou les types d'études de simulations dans lesquels elle peut être utilisée, les équipes métiers la manipulant souvent, ou tout autre critère significatif pour une entreprise. Ainsi, à partir des Knowledge Index, les utilisateurs pourront effectuer des recherches par filtres ou mot clefs pour ne retrouver que les ICEs qui concernent un contexte précis d'utilisation. Le Knowledge Index peut aussi bien contextualiser des ICEs que des configurations.

Pour réutiliser les paramètres et les règles, donc les ICEs, dans un modèle métier, les utilisateurs créent des configurations. En effet, l'utilisateur effectue une recherche dans la base d'ICEs et sélectionne celles qui l'intéressent et qui sont porteuses des connaissances encapsulées dans son modèle métier, pour enfin les instancier dans une configuration. Une configuration est donc composée d'instances d'ICEs (paramètres plus valeurs) créées depuis la base générique d'informations techniques, et cette configuration est synchronisée (en bidirectionnel) avec le modèle, c'est à dire que la valeur des paramètres dans le modèle est synchronisée avec la valeur des paramètres dans la

configuration. L'utilisateur peut donc travailler sur son modèle métier et, quand il le souhaite, le synchroniser avec la configuration qui devient alors représentative de l'architecture (par exemple, mais pas seulement) sur laquelle il travaille. De cette façon, un utilisateur peut créer plusieurs versions de configuration pour sauvegarder plusieurs architectures intéressantes.

Chaque utilisateur du processus de conception peut créer sa propre configuration indépendamment des autres utilisateurs. Ainsi, une ICE pourra être instanciée dans plusieurs configurations en même temps, et chaque configuration est synchronisée avec un modèle métier. Pour assurer la cohérence des connaissances, on analyse les configurations entre elles et on remonte, aux utilisateurs, l'information sur d'éventuels conflits.
A titre d'exemple, une ICE contenant le paramètre "diamètre piston" est instanciée dans une configuration associée à un modèle de calcul thermique du piston, et est aussi instanciée dans une configuration associée à un modèle de calcul sur un assemblage d'attelage mobile (piston + vilebrequin + bielle) pour une analyse en fatigue polycyclique. L'utilisateur travaillant sur le piston entre un diamètre piston de 50,5mm, et l'utilisateur travaillant sur l'attelage mobile entre un diamètre piston de 50mm. Nécessairement, l'analyse des configurations fera remonter l'information de cette inégalité ente ces deux instances d'ICEs utilisées en parallèle dans deux activités de conception. Les utilisateurs disposent alors de toutes les informations pour se contacter et itérer pour converger vers une solution.
Cet exemple est volontairement trivial, les conflits peuvent apparaitre sur des valeurs de paramètres utilisés dans différents modèles en parallèles ou alors sur des règles métiers non respectées. La gestion des conflits s'apparente donc à un processus itératif où l'information d'incohérence est donnée aux utilisateurs. Enfin, une fois les activités de conceptions terminées, les configurations peuvent être utilisées au travers d'autres activités, voire d'autres projets, et ainsi limiter les processus routiniers.
Le Tableau 4 résume globalement le processus KCMethod en reprenant les grandes étapes de la méthodologie telle qu'illustrée sur la Figure 34.

La capitalisation en entités références	Capitaliser génériquement les données cruciales extraites des experts et des modèles métiers pour construire une base d'informations techniques.
Contextualiser	Le concept de Knowledge Index permet d'attribuer des contextes génériques aux ICEs et des contextes spécifiques aux configurations.
Création des configurations de connaissances	Réutiliser certaines ICEs pour construire une configuration de connaissances synchronisée avec un modèle métier
Gérer les conflits et validation	Gérer les conflits sur les paramètres et les règles utilisées dans chaque configuration et entre les configurations. Valider les connaissances d'une phase pour les réutiliser.

Tableau 4 : processus global KCMethod

Concernant l'utilisation des configurations, relativement aux activités du processus de conception, elles sont naturellement organisées en jalons ou en phases, conformément à l'organisation d'un projet de développement de produit tel que nous l'avons vu au chapitre 1.

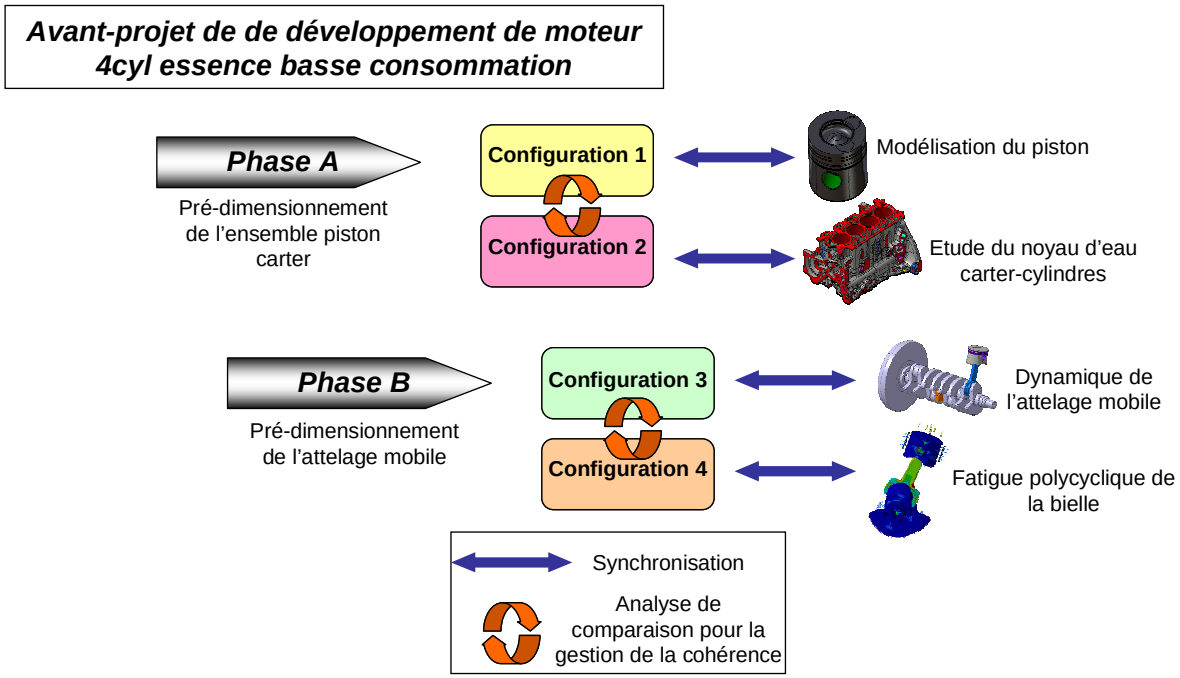


Figure 35 : organisation des configurations dans le processus de développement de produit

La Figure 35 illustre un exemple d'organisation des configurations dans le cadre d'un avant-projet de développement moteur. Le projet est découpé en phases, et chaque phase contient plusieurs activités de conception et de simulation (deux activités par phase dans l'exemple). Pour chaque activité, une configuration est associée et les configurations d'une même phase sont analysées en cohérence.

KCMethod permet de créer un lien entre les systèmes de gestion de données de type PDM et SDM, qui stocke et gère les modèles métiers et les systèmes de gestion de la connaissance (KMS – Knowledge Management System). Ces deux types d'outils sont théoriquement dépendants l'un de l'autre, mais sont rarement couplés dans la réalité des processus industriels, faute de plateforme d'échange et de gestion assurant le suivi et la traçabilité entre les deux types d'outils. KCMethod est de nature à favoriser ce lien, de sorte à mieux prendre en compte et réutiliser les connaissances dans les modèles, mais aussi entre n'importe quel outil d'ingénierie.

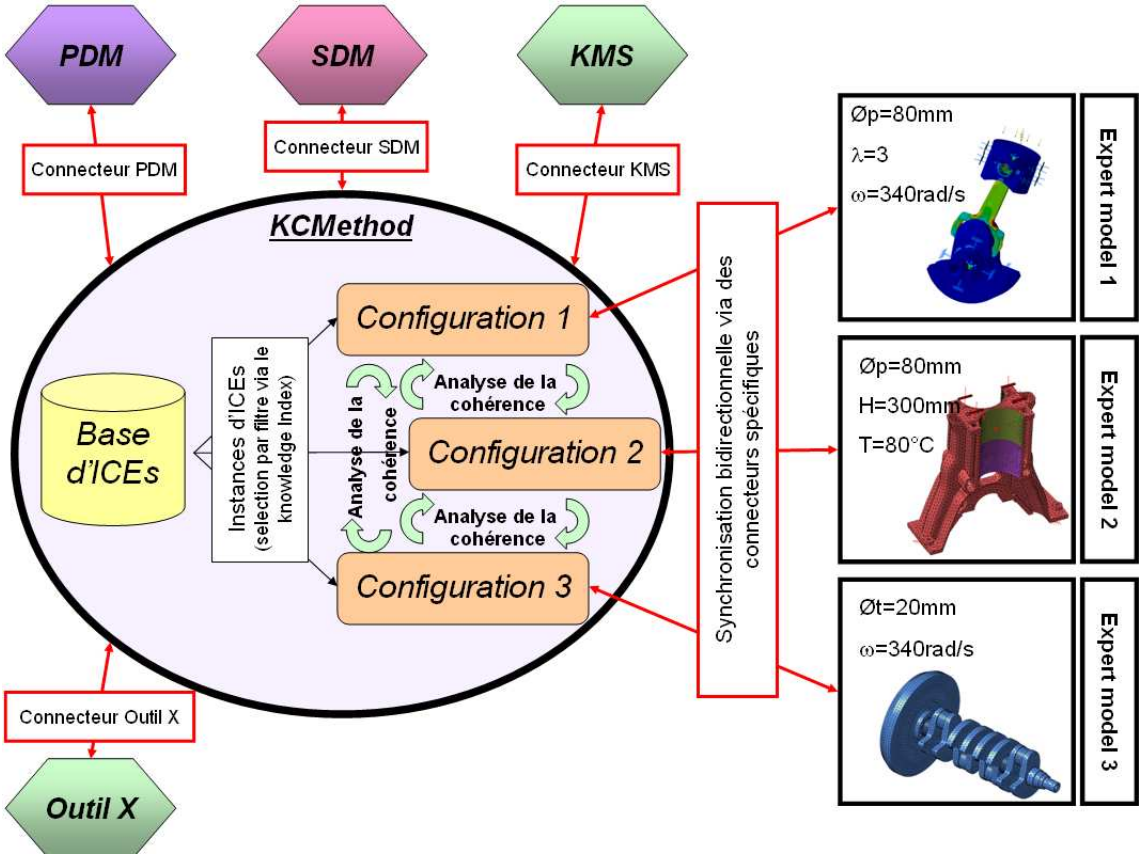


Figure 36 : illustration du périmètre KCMethod et ses interactions avec les autres outils

Ainsi, KCMethod vient en complément des outils de gestion d'informations et de données de type SGDT, car elle se focalise sur les connaissances encapsulées dans les modèles métiers gérés dans les PDM et les SDM.

D'un point de vue opérationnel, le périmètre de la méthodologie, par rapport aux outils utilisés dans une entreprise, comprend la centralisation et la capitalisation des ICEs ainsi que la création et l'utilisation de configurations. La synchronisation des configurations et des modèles métiers utilisant divers outils nécessite le développement de connecteurs spécifiques avec chaque outil (Figure 36).

Dans la suite de ce chapitre, nous détaillons les différents concepts définis dans le cadre de la méthodologie KCMethod. Un intérêt particulier est porté à leurs fonctionnements et leurs apports dans le cadre de notre problématique de recherche.

3.4.2 Détail du concept d'ICE : Information Core Entity

Le concept d'ICE fait référence à la modélisation par entités qui est considérée, depuis les années 90, comme un facteur clef de la représentation des informations relatives au produit et au processus d'élaboration des produits manufacturiers [Taylor et al., 1994] [Krause et al., 1995]. Une définition générale proposée par [Tollenaere, 1998] de la notion d'entité, l'identifie comme étant : *«un groupement sémantique (atome de modélisation) caractérisé par un ensemble de paramètres, utilisé pour décrire un objet indécomposable utilisé dans le raisonnement relatif à une ou plusieurs activités liées à la conception des produits et des systèmes de production ».*
En conséquence, le concepteur travaille à partir d'une bibliothèque d'entités à instancier en précisant les valeurs des paramètres des entités choisies pour son modèle. L'approche semble naturelle et la manipulation aisée. De plus, la sémantique métier peut être encapsulée au sein d'une entité, ce qui à pour avantage de pouvoir la représenter selon différents point de vues [Yvars, 2001].

3.4.2.1 Structure des ICEs

La création des ICEs résulte d'une analyse métier afin d'identifier les données et les informations cruciales à capitaliser et à partager dans le processus de conception, et plus particulièrement entre les modèles métiers. L'ensemble des ICEs forme une base dynamique qui est enrichie au fur et à mesure du temps et évolue au fil de l'eau. L'ICE possède son propre cycle de vie, sa maturité, son niveau de criticité, ses versions (avec un historique), etc.
Chaque ICE possède par défaut un IDentifiant unique et un ensemble d'attributs répartis dans 4 classes principales : *Les informations générales, les Informations spécifiques, les paramètres capitalisés et les Contraintes capitalisées* (Figure 37). Des attributs intrinsèques à l'ICE sont renseignés automatiquement à la création ainsi qu'à chaque modification :

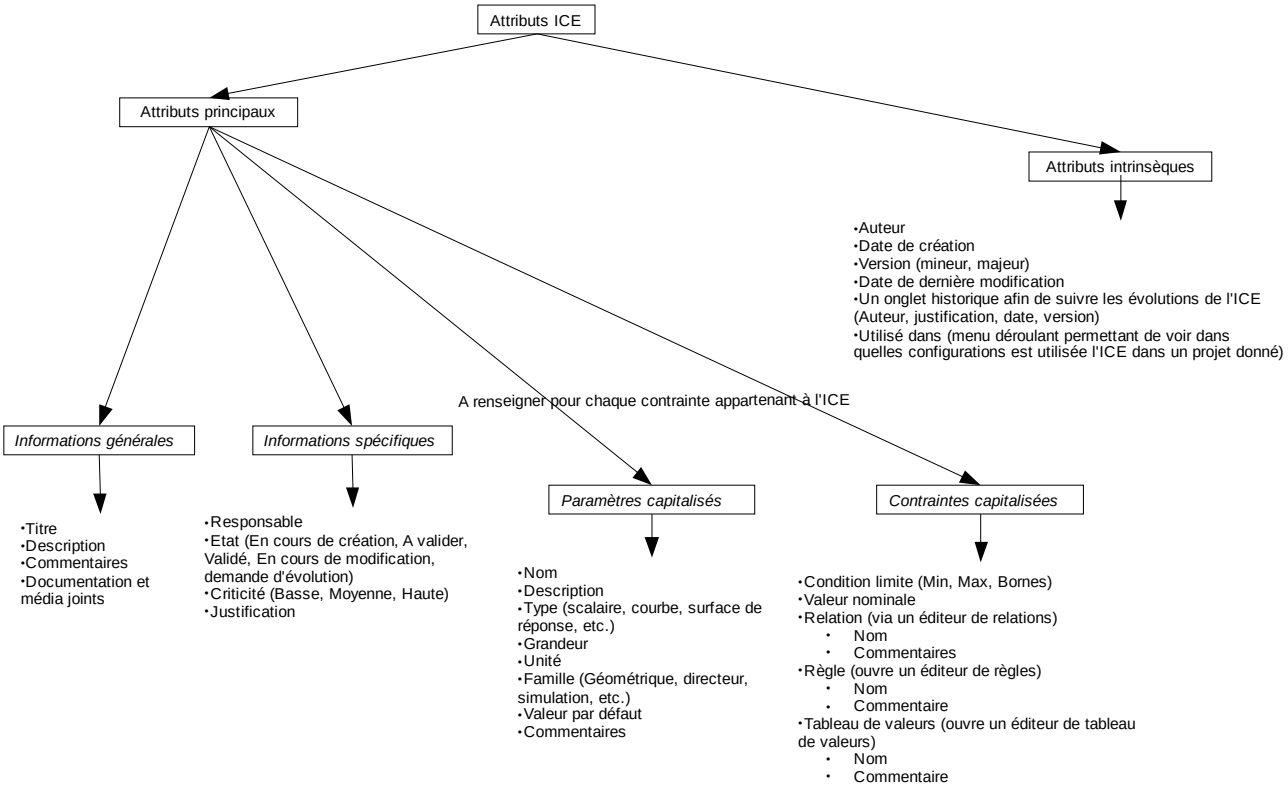


Figure 37 : attributs des ICEs

Une ICE est "paramétrable", ainsi, il est possible de créer de nouveaux attributs afin d'adapter la capitalisation au contexte de l'entreprise.
La création des ICEs s'effectue en différentes étapes qui permettent, successivement, de renseigner les attributs, de créer les paramètres, de créer les contraintes et de les lier aux paramètres concernés, pour finalement visualiser l'ICE avant enregistrement. La Figure 38 illustre un exemple de visualisation d'une ICE. On retrouve les informations générales et spécifiques en bas de la figure, et différents onglets représentants des paramètres capitalisés. Ces paramètres, capitalisés de façon générique (sans valeur), sont renseignés via l'utilisation de différentes informations (grandeur physique, unité, famille, etc.), et possèdent éventuellement des contraintes spécifiques (bornes, une valeur nominale, etc.).

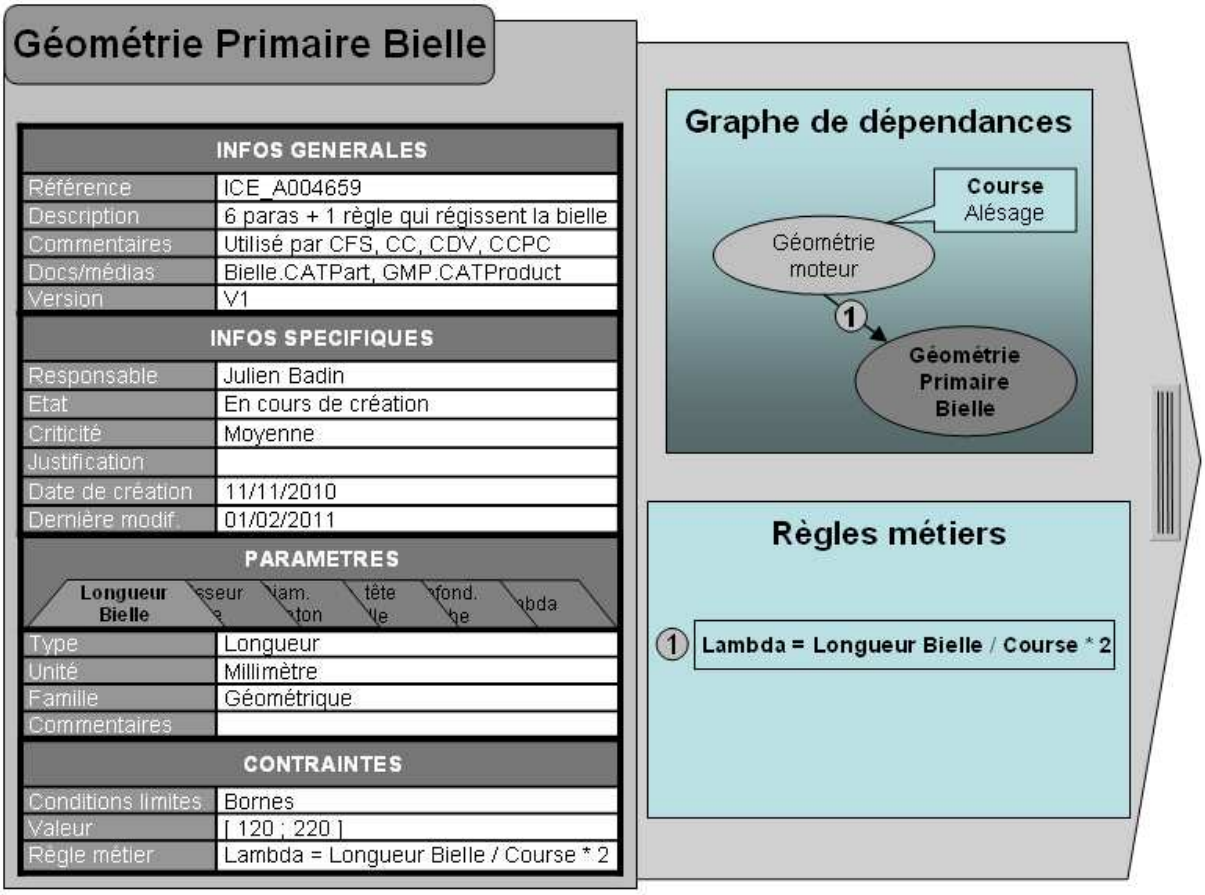


Figure 38 : exemple de visualisation d'une ICE contenant plusieurs paramètres et contraintes

En plus de la sauvegarde des paramètres, la Figure 38 illustre une règle métier capitalisée dans l'ICE qui lie trois paramètres :

- Le paramètre "Lambda" (capitalisé dans l'ICE : "Géométrie Primaire Bielle")
- Le paramètre "Longueur" Bielle (capitalisé dans l'ICE : "Géométrie Primaire Bielle")
- Le paramètre "Course" (capitalisé dans l'ICE : "Géométrie moteur")

Les paramètres impliqués dans la règle peuvent tous appartenir à l'ICE qui capitalise la règle, ou bien appartenir à différentes ICEs, ce qui signifie que des liens entre plusieurs ICEs génériques sont possibles (représenté par le graphe de dépendance). En effet, le paramètre "course" est capitalisé dans une autre ICE nommée "Géométrie moteur" ce qui implique un lien de dépendance entre cette ICE et l'ICE "Géométrie Primaire Bielle". A la création d'une règle, l'utilisateur pointe les paramètres utilisés. Ainsi, une règle connait toujours l'ICE qui contient les paramètres qu'elle utilise, que ce soit dans une même ICE ou entre plusieurs ICEs.
Enfin, à la fin de la création d'une ICE, celle-ci est contextualisée en fonction de l'ensemble de ses utilisations grâce à la notion de Knowledge Index sur laquelle nous reviendrons par la suite.
La création des ICEs génériques, est exclusivement réservée à un public d'utilisateurs experts ou de chefs de projets. Ce sont eux qui sont garants des connaissances et du savoir-faire de l'entreprise.

3.4.2.2 La capitalisation des données dans les ICEs

On capitalise essentiellement deux types de données cruciales dans les ICEs (Figure 39) :

- Les paramètres
- Les contraintes

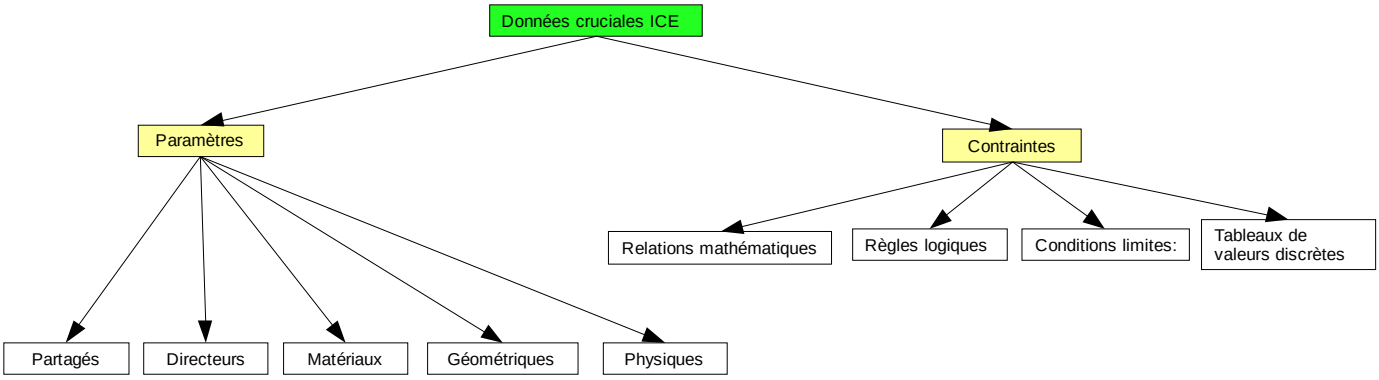


Figure 39 : données cruciales ICEs

La typologie de contrainte au même titre que celle des paramètres présentée, est suffisamment générique pour être utilisée dans différents domaines industriels sur plusieurs produits. Néanmoins, des adaptations peuvent être nécessaires pour mieux coïncider avec des besoins spécifiques à certaines organisations.

3.4.2.2.1 Les paramètres :

Nous considérons un paramètre comme étant une donnée exprimée selon un point de vue d'utilisation d'une caractéristique mesurable ou quantifiable. Les types de paramètres capitalisés font référence aux classes identifiées dans la partie 3.2.1. Cela concerne les paramètres jugés cruciaux et nécessaires dans le cadre de la problématique de partage entre les modèles métiers (Figure 39).
Une ICE peut contenir un ou plusieurs paramètres en fonction de la façon dont les utilisateurs veulent les capitaliser (grouper des paramètres et des règles dont leur utilisation les rend indissociables, cf. section 3.2.1. constat 5). Si l'ICE est renseignée par différents attributs, chaque paramètre reçoit également des informations nécessaires à sa définition et son partage entre plusieurs activités du processus de conception (cf. section 3.2.1. constat 2).
Enfin, un paramètre est capitalisé de façon générique dans une ICE, néanmoins il peut se voir attribuer une valeur par défaut, ce qui signifie qu'à chaque instanciation de l'ICE, le paramètre instancié s'initialisera avec la valeur définie par défaut. Cette valeur pourra évidemment être modifiée.

3.4.2.2.2 Les contraintes :

Si dans les pages précédentes de ce document nous avons souvent fait référence à la capitalisation des paramètres et des "règles métiers", il est plus correct et plus général de préférer la notion de contrainte englobant les règles métiers, mais pas seulement.
Nous considérons une contrainte comme étant un concept d'obligation ou de restriction créé par les règles en usage dans un milieu, ou par les lois propres à un domaine. Sous cette même notion, nous regroupons quatre classes identifiées partie 3.2.1.

- **Les conditions limite** :

Les conditions limite sont des contraintes affectées à un paramètre en particulier, qui restreignent son utilisation à un domaine. Elles sont de différentes natures :

  - *Valeur nominale :* signifie que le paramètre capitalisé dans l'ICE ne peut porter que cette valeur (un module d'Young par exemple). Si le paramètre instancié n'est pas égal à la valeur nominale définie dans l'ICE générique, cela génère un conflit.
  - *Bornes :* signifie que le paramètre n'est valide que pour un ensemble de valeurs appartenant à un domaine. Les bornes (fermée ou ouvertes) peuvent encadrer complètement le paramètre ou seulement définir un max ou un min, ou même exprimer une tolérance. Concrètement, dans la méthodologie, cette notion d'encadrement est exprimée sous forme de fonction d'appartenance (Figure 40)

autorisant une plus grande flexibilité lors de la résolution des contraintes et favorisant l'utilisation de méthodologies dédiées à l'aide à la décision comme la logique floue [Massa et al., 2005].

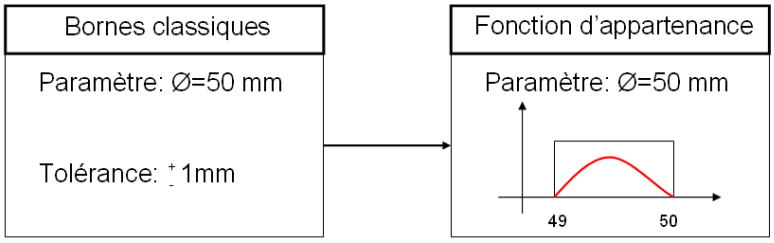


Figure 40 : différence entre des bornes classiques et une fonction d'appartenance sur la valeur prise par un paramètre

La contrainte de type condition limite possède une particularité. Si elle est générique au moment de la capitalisation dans une ICE, elle peut être affinée lors de son instanciation dans une configuration, de manière à créer une contrainte locale en rapport avec une activité du processus de conception.
A titre d'exemple, une ICE de la base d'information contient un paramètre "ω" et une contrainte générique de type condition limite [200;300].
Un utilisateur "A" instancie cette ICE dans sa configuration de "cinématique vilebrequin" et donne une valeur au paramètre ω=250rad. Or, dans son modèle il à besoin d'affiner cette contrainte (tout en respectant son expression générique), il peut alors définir une contrainte locale : [245;270].
Ensuite, un utilisateur "B" instancie également cette ICE dans sa configuration "effort max bielle" et définit une condition limite locale : [260;290].
A partir des conditions limites locales définies par les deux utilisateurs, il est possible de croiser les domaines d'utilisation des paramètres et d'identifier les espaces de solutions, ou les ensembles vides quand il n'y a pas d'intersection entre les domaines (Figure 41).

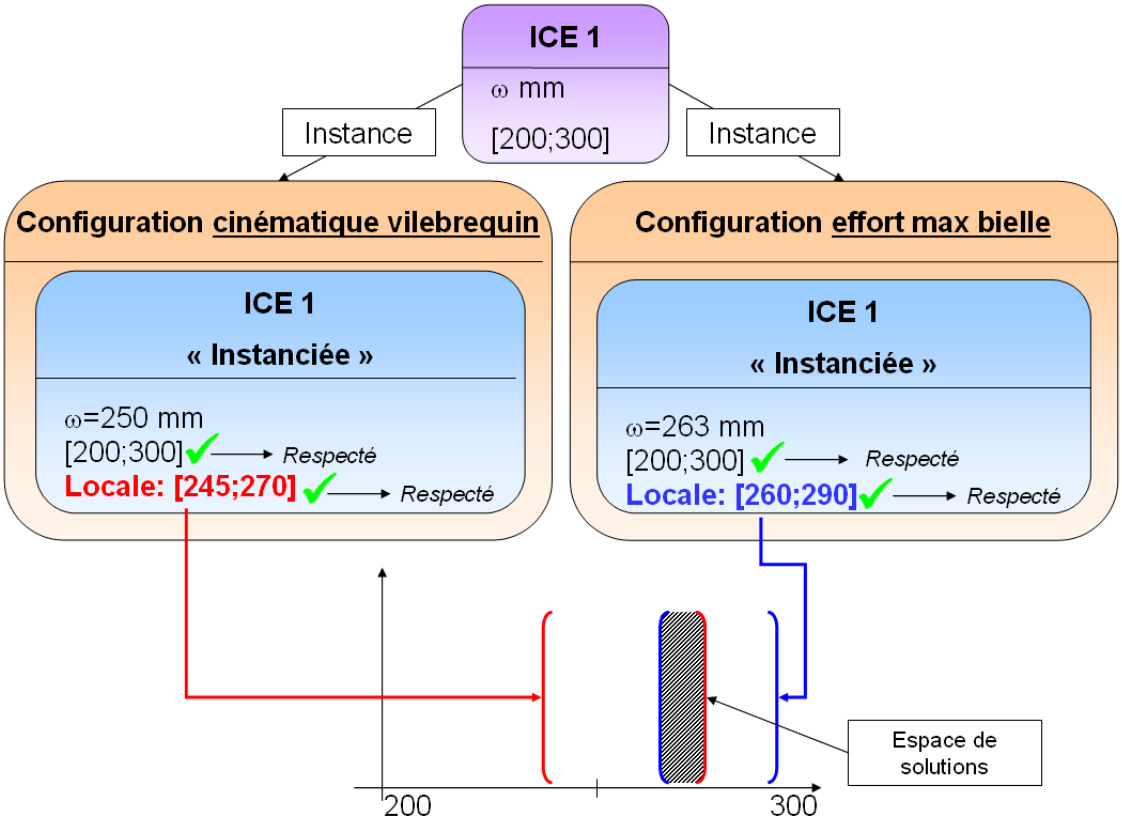


Figure 41 : exemple de condition limite locale dans deux configurations

- **Les relations mathématiques** :

Elles expriment une relation entre paramètres en utilisant des opérateurs mathématiques. Par exemple, Z=X+Y.

- **Les règles logiques :**

Elles expriment une relation entre paramètres ou groupes de paramètres de type logique ou condition. Par exemple, SI Z=(X+Y)/2, ALORS T=Z/2, SINON T=Z.

- **Les tableaux de valeurs discrètes** :

Ce sont des tableaux de correspondances de valeurs. Un tableau de valeurs discrètes comporte plusieurs paramètres organisés en colonne regroupant un ensemble de valeurs. Par exemple, pour une valeur en particulier (en entrée) du paramètre de la première colonne, le paramètre de la seconde colonne prendra une valeur propre (en sortie).

Si cette typologie de contrainte n'est pas exhaustive, elle est suffisante pour traiter un nombre considérable de conflits dans les modèles de conception, soit en vérifiant leur respect (vérification du respect des contraintes et retour de type ok/ko), soit en utilisant des mécanismes de propagation de contraintes afin de fixer ou limiter les domaines de valeurs de certains paramètres, en définissant la valeur d'autres et ce, de façon acausale.

3.4.2.2.3 Les différentes façons de capitaliser les paramètres et les contraintes :

La façon dont les paramètres et les contraintes sont capitalisés dans les ICEs est très importante et représentative de leur utilisation. En effet, considérant que l'ICE est l'entité de plus bas niveau manipulée, si plusieurs paramètres et règles sont capitalisés dans une même ICE, ou s'ils sont capitalisés dans des ICEs différentes, le comportement à l'instanciation dans les configurations va varier de façon importante. Nous illustrons, ci-dessous, ces deux comportements via l'utilisation de plusieurs exemples simples.

- *Capitalisation dans une même entité :*

Plusieurs paramètres capitalisés dans une ICE les rendent indissociables lors de l'instanciation dans une configuration. Concernant les contraintes, cela veut dire qu'elles devront impérativement être respectées.

- *Capitalisation dans plusieurs entités :*

Capitaliser des paramètres et des contraintes dans des ICEs différentes permet d'avoir plus de souplesse lors de leur utilisation dans les configurations. Même s'il existe des liens entre plusieurs ICEs génériques, celles-ci peuvent être utilisées séparément en fonction du contexte d'utilisation dans lequel la configuration est crée.

Pour illustrer ces propos, nous proposons un exemple[4] de capitalisation de deux paramètres "A" et "B" et de deux contraintes "A<20mm" et "A=2*B".

[4] Dans les exemples, la représentation des ICEs est volontairement très simplifiée afin de faciliter la compréhension des concepts présentés.

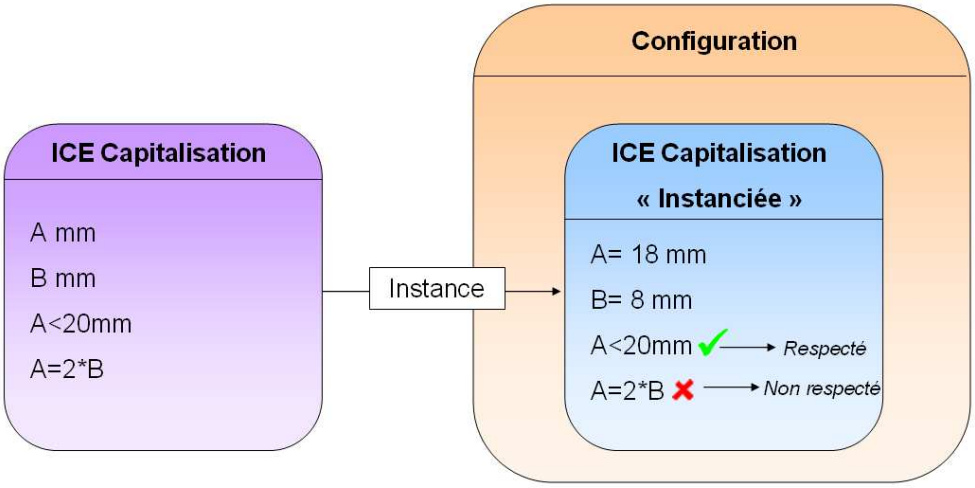


Figure 42 : exemple 1 de capitalisation dans une ICE

Sur le premier exemple, Figure 42, une seule et même ICE est utilisée pour capitaliser les deux paramètres et les deux contraintes. Pour réutiliser cette ICE, l'utilisateur va l'instancier dans une configuration et donc obligatoirement utiliser l'ensemble de son contenu. Ainsi, il devra absolument respecter les deux contraintes, sous peine de se retrouver avec un conflit dans sa configuration. Dans ce cas, les différents composants de l'ICE sont indissociables, quel que soit le contexte d'utilisation (quelle que soit l'activité de conception ou de simulation).
On peut dire que quand une contrainte est capitalisée dans une ICE et porte sur des paramètres capitalisés dans la même ICE, elle est considérée comme un référentiel métier à respecter impérativement.

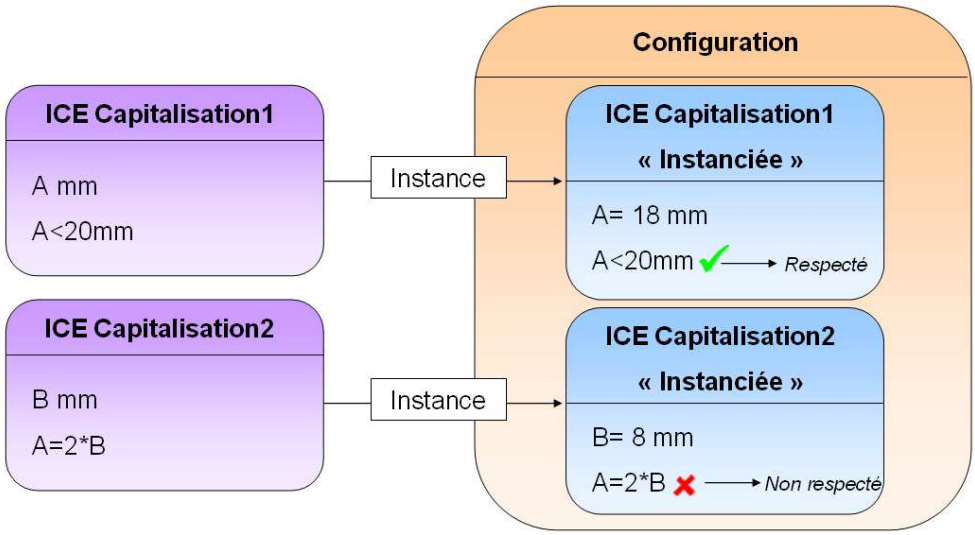


Figure 43 : exemple 2 de capitalisation dans des ICE différentes.

Dans ce second exemple Figure 43, on obtient le même résultat que dans l'exemple 1 mais avec deux ICEs utilisées pour capitaliser les paramètres et les contraintes. Le paramètre "A" est toujours lié à la contrainte "A<20mm" car ils sont toujours capitalisés dans la même ICE. En revanche, "B" est capitalisé à part avec la contrainte "A=2*B".
Dans ce contexte, l'ICE 1 peut être utilisée indépendamment de l'ICE 2, néanmoins l'inverse n'est pas possible. En effet, dès que "B" est utilisé dans la configuration, la relation doit être respectée et fait appel à l'ICE 1. La souplesse de cette organisation vient du fait que l'utilisateur peut choisir de ne pas utiliser l'ICE 2 dans sa configuration. S'il travaille sur une activité du processus de conception qui ne nécessite que l'utilisation du paramètre "A", il peut instancier l'ICE 1 sans tenir compte de l'ICE 2.

Un troisième exemple est illustré Figure 44. Le résultat est toujours identique aux deux exemples précédents. Néanmoins, la contrainte "A=2*B" est capitalisée seule dans une ICE. L'utilisateur peut donc ne pas prendre en compte cette relation mathématique s'il le souhaite dans la configuration, alors qu'il peut utiliser les paramètres "A" et "B". En revanche, à partir du moment où l'ICE 3 est instanciée dans la configuration, elle fera nécessairement appel à l'ICE 1 et 2 qui devront aussi être instanciées.

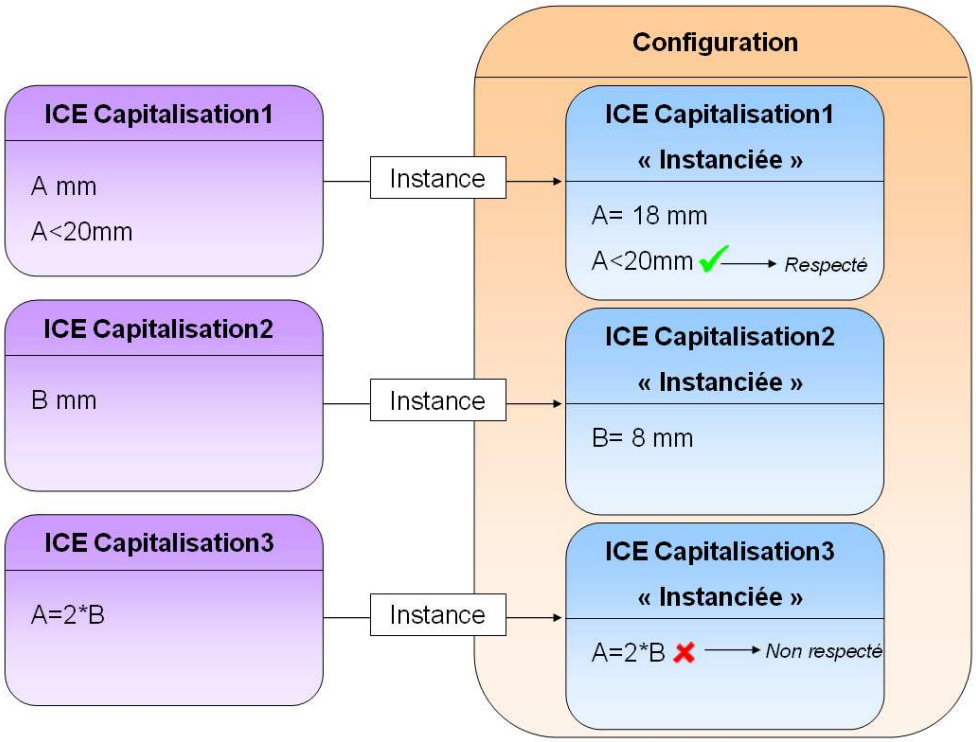


Figure 44 : troisième exemple de capitalisation dans plusieurs ICEs

Séparer les paramètres et les contraintes dans plusieurs ICEs implique qu'elles peuvent être utilisées indépendamment les unes des autres en fonction de l'activité de conception ou de simulation en lien avec une configuration. Par conséquent, il existe bien des liens entre ICEs génériques dans la base d'informations techniques du produit mais cela ne veut pas dire que toutes les ICEs seront instanciées dans une configuration.

3.4.2.3 Cycle de vie et impacts des modifications des ICEs

Précédemment, dans ce chapitre, nous avons insisté sur le fait qu'une ICE est capable d'évoluer avec le temps. En effet, une ICE n'est pas une entité statique, c'est à contrario une entité dynamique qui va pouvoir changer en fonction de l'évolution des paramètres et des contraintes qui la composent. L'ICE possède son propre statut, son historique, et elle est gérée en version, permettant à son contenu d'avoir son cycle de vie indépendamment d'une activité de conception ou d'un modèle métier. Dans ce contexte, étant donné qu'il existe des liens entre plusieurs ICEs, il est nécessaire de prendre en compte les impacts que peuvent générer la modification d'une ICE sur une autre.

3.4.2.3.1 Le cycle de vie des ICEs :

Traditionnellement uniquement encapsulés dans les modèles métiers, les paramètres et les contraintes sont dépendants du cycle de vie du modèle en question (cf. CTRC chapitre 1).
Pour répondre à cette problématique, les ICEs permettent aux paramètres, ainsi qu'aux contraintes, de posséder leur propre cycle de vie, indépendamment des modèles ou des activités du processus de conception. La Figure 45 illustre un processus classique de cycle de vie d'une ICE.

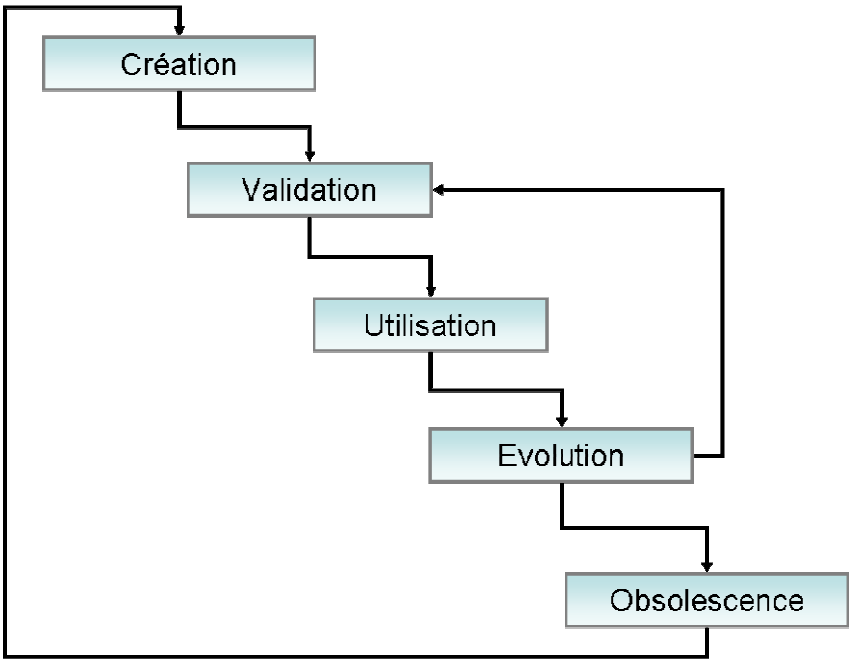


Figure 45 : cycle de vie d'une ICE

Les ICEs sont crées et modifiées au fil des projets de conception et entérinées suivant un "état" caractérisé par un statut de validation. De plus, une ICE possède une version et un historique, ainsi un utilisateur peut la modifier, c'est-à-dire, changer ses attributs, mais surtout modifier des paramètres ou des contraintes, en ajouter, ou encore en supprimer.
A chaque modification, l'utilisateur peut créer une nouvelle version (mineure ou majeure) de l'ICE qui deviendra alors la nouvelle référence. Les anciennes versions sont conservées dans un historique qui répertorie les informations sur les modifications de l'ICE, l'auteur de ces modifications, la raison et la date.
Enfin, l'ICE porte également une autre information utilisée dans la gestion de son cycle de vie : "la période d'inactivité". En effet, quand le nombre d'ICEs est important, il est nécessaire de fournir une aide aux experts pour continuer à évaluer les ICEs, et particulièrement celles qui sont peu utilisées. Les ICEs possèdent toutes un indicateur portant sur leur utilisation, de cette façon si une ICE n'est plus utilisée pendant un laps de temps trop important, le responsable de cette ICE reçoit une demande de réévaluation. L'objectif est de déterminer si cette ICE doit continuer à exister dans la base d'informations techniques, auquel cas l'utilisateur peut réinitialiser son compteur, ou alors la modifier et la faire évoluer. Il peut aussi la supprimer, ou plutôt la déclarer obsolète si c'est le cas.
Ce processus garantit une gestion dynamique des ICEs et une bonne corrélation entre les connaissances réellement utilisées dans les modèles métiers et les informations de la base d'informations techniques. Néanmoins, le cycle de vie d'une ICE ne dépend pas que de son propre chef, en effet les ICEs peuvent être liées les une aux autres, ce qui implique une gestion des impacts lors de la modification d'une ICE liée.

3.4.2.3.2 Gestion des impacts entre ICEs génériques :

Plusieurs ICEs génériques peuvent être liées par des contraintes, ce qui implique que la modification d'une ICE peut avoir des répercussions sur d'autres ICEs. Ainsi, quand un utilisateur décide de modifier une ICE, un graphique de dépendance lui présente les liens vers les autres ICEs génériques potentiellement impactées par cette modification. Tant que tous les liens de dépendance ne sont pas vérifiés l'utilisateur ne peut pas sortir du processus de modification.

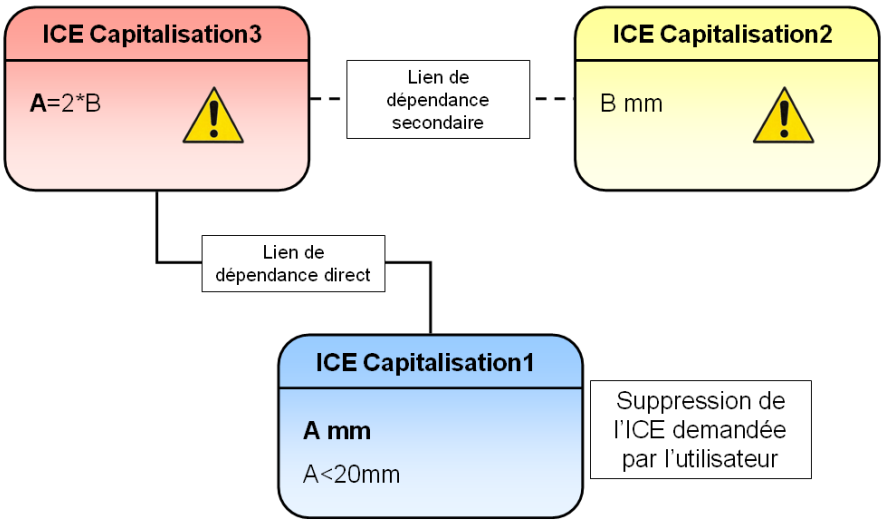


Figure 46 : affichage des dépendances pour la suppression d'une ICE liée

Comme illustré par l'exemple Figure 46, un utilisateur veut supprimer l'ICE capitalisation 1 contenant le paramètre "A". Le graphe lui montre les liens de dépendance vers d'autres ICEs et les éléments concernés. Les dépendances directes et secondaires indiquent les liens à prendre en compte et à satisfaire si l'utilisateur supprime "A". S'il veut aller au bout de sa démarche, il devra soit supprimer l'ICE Capitalisation 3, soit la modifier pour la rendre indépendante de l'ICE Capitalisation 1.

3.4.2.4 La multi-représentation des ICEs

La multi-représentation des connaissances est un enjeu crucial des problématiques de collaboration. En effet, il faut être en mesure de disposer des connaissances selon différentes vues en relation avec chaque domaine métier. Coïncidant avec ce contexte, les paramètres et les contraintes, capitalisés de façon générique dans les ICEs, ont vocation à être utilisés dans de nombreux contextes de conception nécessitant l'interaction et l'interprétation des concepteurs. Ces contextes dépendent des domaines métiers, des outils, de l'expérience et du ressenti des acteurs, donc autant dire qu'ils sont très hétérogènes. Cela se traduit par le fait qu'une même connaissance doit pouvoir être compréhensible selon différents points de vue caractéristiques de ses nombreux contextes d'utilisations. Ainsi, selon le métier de l'observateur, une pièce sera sémantiquement différente et pourra être considérée selon plusieurs points de vue. Chacun de ces points de vue pourra correspondre à une représentation des entités capitalisant les connaissances, et nous obtiendrons ainsi un modèle dynamique multi-vue de notre objet [Yvars, 2001].
En conséquence, une ICE peut avoir différents niveaux de représentations pour correspondre à des normes, des habitudes ou des contextes particuliers. Par exemple, en fonction des domaines métiers, un paramètre peut avoir différents noms. Il peut en être de même pour le type d'un paramètre qui sera utilisé tantôt sous forme d'une valeur numérique, tantôt sous forme de liste de valeurs ou encore d'une courbe.

Dans ce contexte, nous définissons les différents niveaux de représentations des ICEs portant sur :

- Le nom des ICEs
- Le nom des paramètres
- Le type des paramètres
- Le nom des contraintes

L'objectif est de faire en sorte qu'une ICE apparaisse à un utilisateur avec des noms et une sémantique adaptée à son contexte d'utilisation en particulier et avec lesquels il à l'habitude de travailler. Ainsi, une même ICE ou un même paramètre pourront avoir plusieurs noms en fonction d'un contexte ou d'un autre, comme dans l'exemple de diamètre piston et de diamètre alésage Figure 47. Les deux

noms sont affectés au même et unique paramètre mais sont visibles différemment selon que les utilisateurs travaillent sur le composant piston ou carter cylindres (Figure 47).

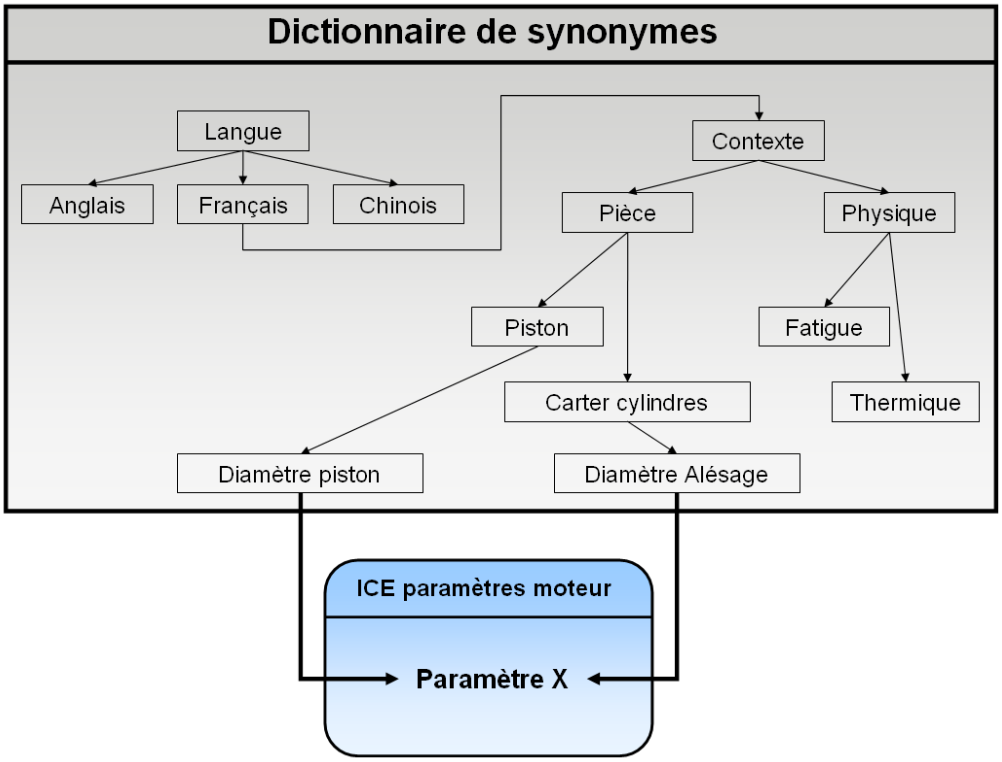


Figure 47 : multi-représentation du nom d'un paramètre

Dans le cas particulier de la multi-représentation de noms sur un paramètre, nous définissons un dictionnaire de synonymes contenant les noms des paramètres organisés en fonction de leurs contextes d'utilisation (en réalité c'est plutôt un graph) et des langues utilisées. Les contextes d'utilisation peuvent dépendre d'un domaine métier, d'une pièce, etc. Ainsi, lors de la création ou de la modification d'une ICE, l'utilisateur créé un paramètre puis lui affecte un ou plusieurs noms issus du dictionnaire.

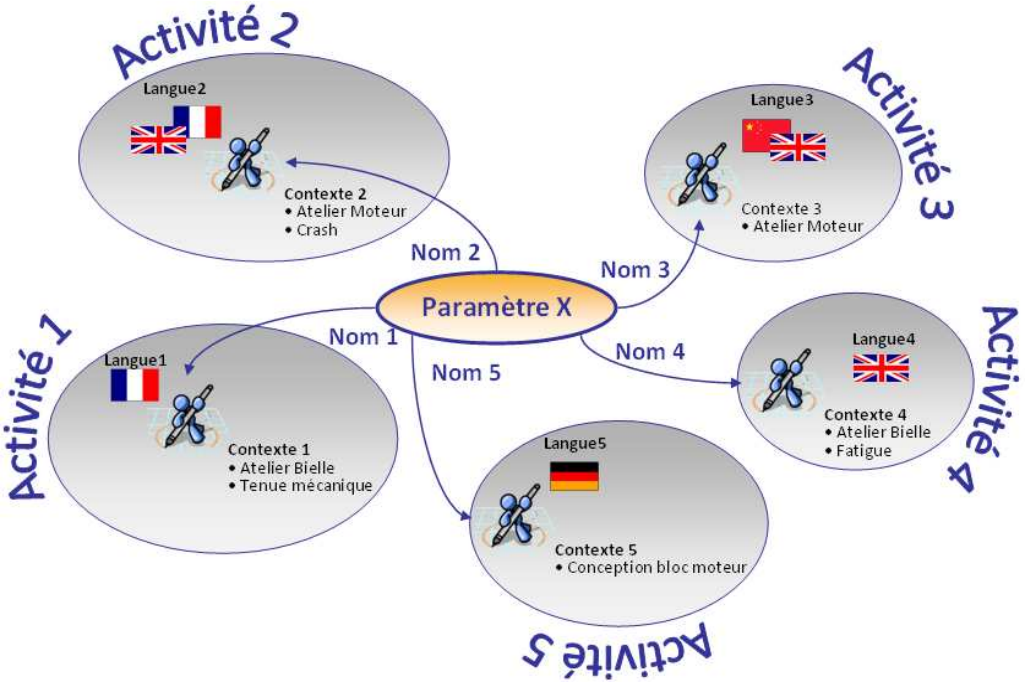


Figure 48 : différentes représentations du nom d'un paramètre en fonction des contextes et des langues

Quand un utilisateur veut utiliser une ICE, il fait une recherche par contexte d'utilisation afin de filtrer celles qui l'intéressent. En fonction des contextes définis dans ce filtre, les ICEs apparaîtront avec une représentation adaptée. Le paramètre pourra donc avoir différents noms dans plusieurs activités du processus de conception, qui dépend à la fois du contexte de conception et des langues utilisées (Figure 48).

3.4.2.5 Bilan sur l'intérêt des ICEs

Les ICEs sont des entités qui permettent de formaliser, capitaliser et structurer les données pour les transformer en informations. Les paramètres et les contraintes sont centralisés et possèdent leur propre organisation (groupes de paramètres et de contraintes), et leur propre cycle de vie, indépendamment des activités ou des modèles utilisés dans le processus de conception. Si la capitalisation et la gestion des informations est plus efficace, l'utilisation des ICEs permet d'avoir une grande flexibilité lors de la réutilisation des paramètres et des contraintes, notamment quand il s'agit de tracer leurs utilisations. Ainsi, on peut suivre l'activité d'une ICE, savoir dans quelle configuration elle est instanciée, par qui, dans quel but. Cette traçabilité est fondamentale et indispensable si l'on veut ensuite assister les utilisateurs dans leurs tâches en leur donnant l'information sur la présence de conflits entre plusieurs modèles métiers.

Si l'ICE permet de structurer les informations, l'ensemble des ICEs forme un nuage utilisable dans de nombreux contextes. L'utilisation de ce nuage, via la création de configurations, permet de générer à la volée et très dynamiquement de nombreuses vues métiers différentes (Annexe 14).

Ainsi, on peut caractériser cette approche de "bottom-up" dans la mesure où ce ne sont pas les vues métiers qui servent de structures aux connaissances mais les connaissances qui se structurent pour définir une vue métier particulière. Ce point traduit de l'originalité de notre approche par rapport aux modèles produits traditionnels.

En effet, comme nous l'avions vu dans la section 2.2.3 du chapitre 2, les modèles produits (multi-vues [Tichkiewitch et al., 1995]) proposent généralement de structurer les données et les informations selon différentes vues préétablies (vue produit, process, organisation, etc.), traduisant d'une approche de type plutôt "top-down". Ainsi, il est plus difficile de générer dynamiquement, à la volée, d'autres vues dans lesquelles les informations seraient utilisées.

3.4.3 Détail du concept de Knowledge Index : principe et fonctionnalités

Le nombre d'ICEs stockées dans la KCMethod peut devenir important, ainsi, il doit être possible de retrouver rapidement, parmi l'ensemble des informations capitalisées, uniquement celles dont un utilisateur a besoin pour son activité. Dans ce contexte, nous avons défini le concept de Knowledge Index.
Le Knowledge Index est un concept qui permet de donner un contexte et de la sémantique à un objet, dans notre cas une ICE ou une configuration. La création de Knowledge Index est similaire à la définition de métadonnées[5] et de données associées, que l'on attribue comme des "tags", de sorte à faciliter la recherche et la réutilisation d'un objet. De cette façon, il est possible de donner à une ICE l'ensemble de ses contextes d'utilisations potentiels pour qu'ensuite, via des recherches par filtres, l'utilisateur ne réutilise que les ICEs correspondantes à son activité. La Figure 49 illustre un exemple de Knowledge Index. Les métadonnées correspondent à un contexte global et les données à des éléments particuliers de ce contexte. Les Knowledge Index sont définies en fonction du domaine d'activité de l'organisation, et leur rôle est bien d'indexer les informations et les connaissances.

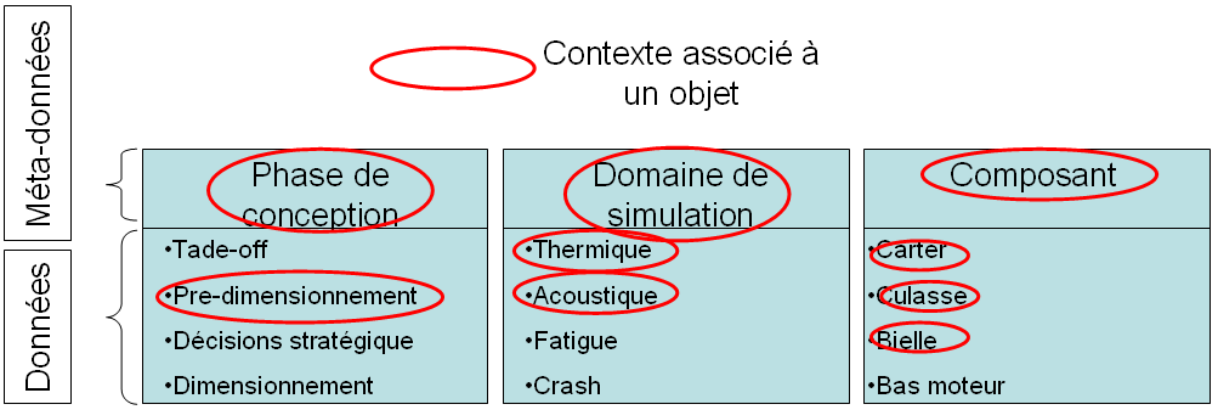


Figure 49 : exemple de représentation de Knowledge Index

Par exemple, une ICE est utilisée dans des modèles portant sur des culasses, et/ou carters, et/ou bielle, mais aussi dans des modèles de calculs thermique et acoustique, pour du pré-dimensionnement. A sa création elle sera taguée avec la métadonnée "composant" et l'ensemble de ses données correspondant à ses utilisations (cercle sur la Figure 49). Il en est de même pour les autres métadonnées. Pour réutiliser les ICEs, un utilisateur effectue plusieurs recherches par filtres dans les Knowledge Index et constitue un "panier" dans lequel il sélectionne, au fur et à mesure, les ICEs qu'il estime intéressantes à instancier dans sa configuration. Ce système de recherche est un mécanisme classique que l'on retrouve sur les sites marchands par exemple.
Les Knowledge Index peuvent être utilisées pour définir n'importe quel type de contexte, que ce soit en rapport avec une pièce, un type d'étude, des noms d'utilisateur ou des équipes projets, tout ce qui est utile à une organisation. Chaque métadonnée et donnée associée porte sa propre sémantique essentiellement basée sur une description. Il est possible de construire plusieurs types de Knowledge Index selon que l'on veut laisser de la flexibilité aux utilisateurs ou non. Par exemple, créer des listes fixes de Knowledge Index (uniquement modifiables par les experts) ou laisser les utilisateurs les compléter au fil de l'eau.

[5] Métadonnées : [ISO/CEI 11179 : 2007] registre sur les métadonnées

3.4.4 Détail du concept de KnowledgeConfiguration

Après avoir abordé en détail le concept d'ICE, nous détaillons l'autre concept principal de la méthodologie KCMethod ; les KnowledgeConfigurations. Le concept de configuration de connaissance est l'élément fondamental de la méthodologie KCMethod, et c'est essentiellement lui qui constitue l'originalité de l'approche. L'objectif consiste à réutiliser les ICEs depuis la base d'informations techniques, de manière à construire des configurations qualifiées de connaissances. Chaque configuration est fortement contextualisée car elle correspond à une activité du processus de conception et est synchronisée avec un modèle métier (Figure 50). Par conséquent, on considère une configuration de connaissance comme étant une représentation des connaissances encapsulées dans un modèle métier.

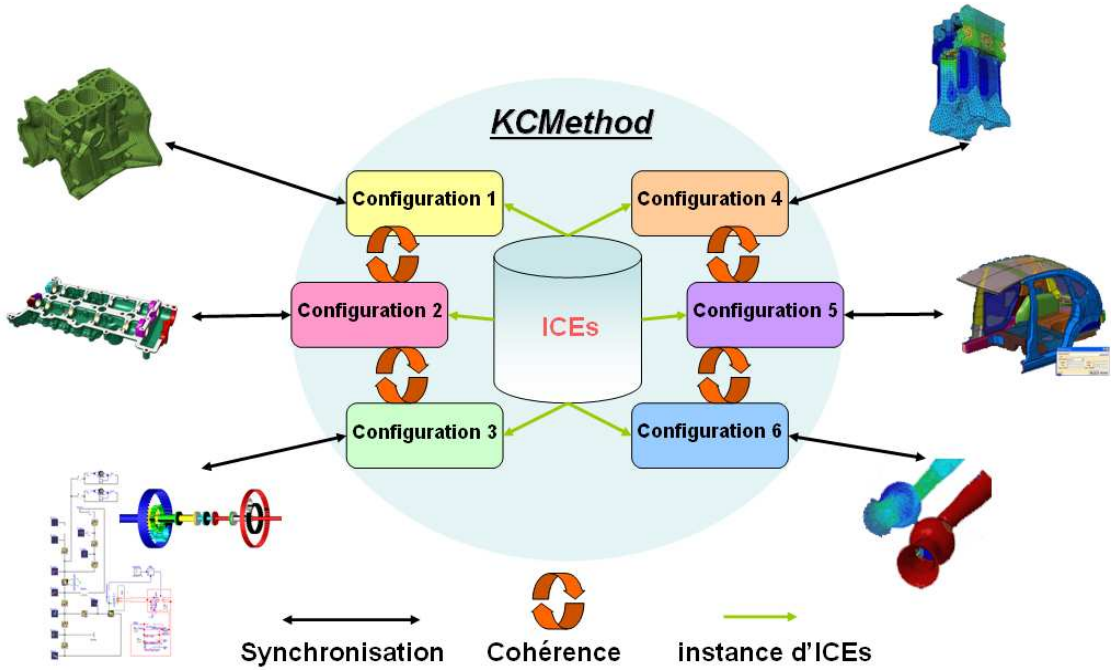


Figure 50 : principe d'utilisation des KnowledgeConfigurations synchronisées avec des modèles métiers

Dans ce contexte, nous justifions le concept de configurations de connaissances de la façon suivante :

En rapport avec la notion de connaissance, Prudhomme et al., expliquent qu'elle nécessite la définition et l'utilisation d'objets de connaissance pour pouvoir être explicitée, artefacts spécifiques constitués à partir de données, informations et savoirs. Ainsi, l'explicitation des connaissances ne peut s'effectuer qu'à travers leurs représentations dans des structures particulières, composées d'objets de connaissances dans lesquels s'inscrivent, entre autres, les données [Prudhomme et al., 2001] [Prudhomme et al. 2007].

KCMethod est en adéquation avec cette définition de la connaissance, les configurations étant des objets adoptant une structure particulière dans lesquelles s'inscrivent les données, les informations, donc les ICEs. Les configurations sont des objets fortement contextualisées et construites via l'interaction et l'interprétation d'un utilisateur pour une activité du processus de conception. En effet, construire une configuration revient à effectuer "un câblage" dynamique des informations présentes dans la base d'ICEs afin d'obtenir un ensemble cohérent de connaissances pour un objectif donné. Cette vision est également en accord avec la définition de Grundstein dans le cadre directeur GAMETH qui considère que les connaissances n'existent que dans la rencontre d'un sujet avec une donnée ou une information [Grundstein, 2007] [Grundstein, 2009].

Au delà de la réutilisation des ICEs dans les modèles métiers, les configurations permettent surtout de suivre l'activité de conception au niveau des paramètres et des contraintes et de détecter, dynamiquement, les conflits qui peuvent subvenir entre plusieurs modèles métiers. Cette information

sur la cohérence ou l'existence des conflits, au niveau des connaissances partagées, est primordiale car elle est aujourd'hui source d'erreurs et de perte de temps considérable dans le processus de conception.

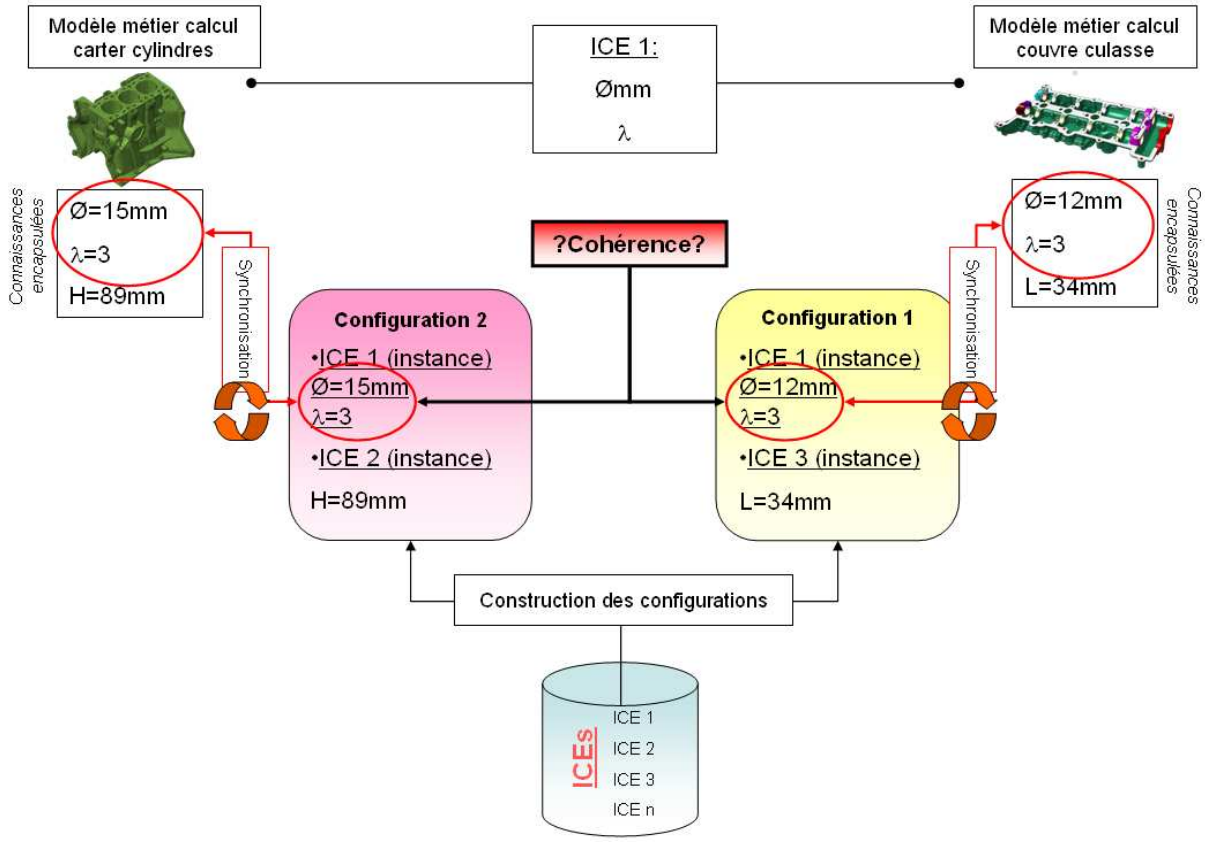


Figure 51 : exemple d'utilisation des User Configurations pour détecter les conflits

La Figure 51 illustre un exemple d'utilisation des configurations pour la gestion de la cohérence des connaissances partagées par deux modèles métiers. Dans cet exemple, un modèle "couvre culasse" et un modèle "carter cylindres" doivent partager des paramètres capitalisés dans l'ICE générique 1 stockée dans la base d'ICE. Ainsi, chaque utilisateur pour chaque modèle créé sa configuration, chacune composée d'une instance de l'ICE 1, et la synchronise. Les deux configurations sont alors représentatives des connaissances encapsulées et les conflits sont alors mis en évidence, sur la valeur du paramètre "Ø" en l'occurrence.

En plus de permettre la gestion des conflits, les configurations permettent l'accès aux mécanismes de gestions de configuration adaptés aux connaissances. Il est par exemple possible de "versionner" une configuration, c'est-à-dire créer différentes versions majeures ou mineures, en fonction de la modification des instances d'ICEs (modification des valeurs des paramètres par exemple) ou alors en fonction de la modification de sa structure (ajout ou retrait d'instances d'ICEs).
Dans ce contexte, nous considérons que les configurations de connaissances correspondent aux principes de gestion de configuration tels qu'ils sont énoncés dans les normes ISO : 10007 de 1995 [ISO 10007: 1995] et 2003 [ISO 10007: 2003]. Dans cette norme, une configuration est décrite comme étant un ensemble des caractéristiques fonctionnelles et techniques d'un produit, définies par l'information de configuration du produit. L'ambition de la gestion de configuration réside dans la limitation des phases de re-conception ainsi que la mise au point de produits non-conformes. Elle participe à la maitrise des risques et à la réduction des délais en permettant aux équipes de développement de travailler, à un instant donné, sur une définition unique et cohérente du produit (cf. chapitre 2, section.2.3.6)

Pour assurer la gestion de configurations de connaissance dans la méthodologie KCMethod, nous avons défini deux types de configurations :

- *La configuration utilisateur qui est la configuration directement synchronisée avec un modèle métier.*
- *La configuration squelette qui sert de support à la création, la gestion et la collaboration des configurations utilisateurs.*

Ces deux types de configurations possèdent leur propre cycle de vie et sont utilisés conjointement dans le processus de conception.

3.4.4.1 Les configurations utilisateurs (User Configuration)

Les configurations utilisateurs sont les plus classiques, elles proposent différentes vues métiers en fonction des modèles avec lesquels elles sont synchronisées

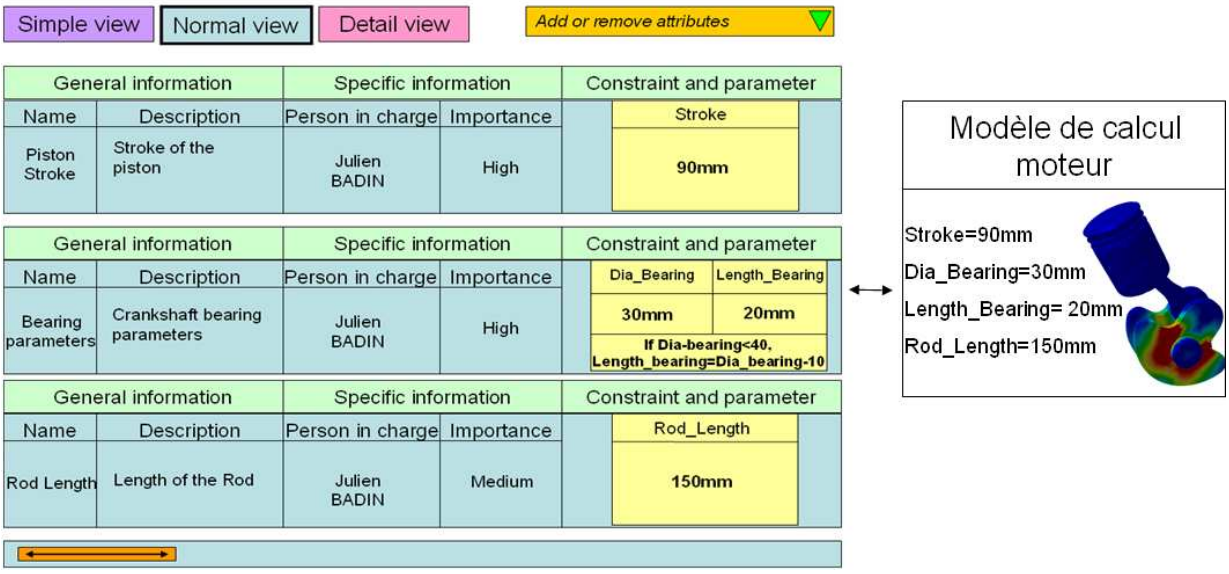


Figure 52 : exemple de User Configuration synchronisée avec un modèle métier

La Figure 52 illustre un exemple de configuration utilisateur en relation avec un modèle métier créé par un concepteur. Chaque bloc représente une instance d'ICE avec ses attributs et surtout les paramètres et les contraintes instanciés. Les valeurs des paramètres sont synchronisées avec les valeurs des paramètres dans le modèle métier.

3.4.4.2 Les configurations squelettes (Skeleton Configuration)

La configuration squelette assure la gestion et la collaboration de l'ensemble des configurations utilisateurs pour un jalon dans un projet (Figure 53). Elle est représentative de l'ensemble des activités du jalon, et par conséquent elle contient toutes les instances d'ICEs utilisées dans chaque configuration utilisateur du jalon.

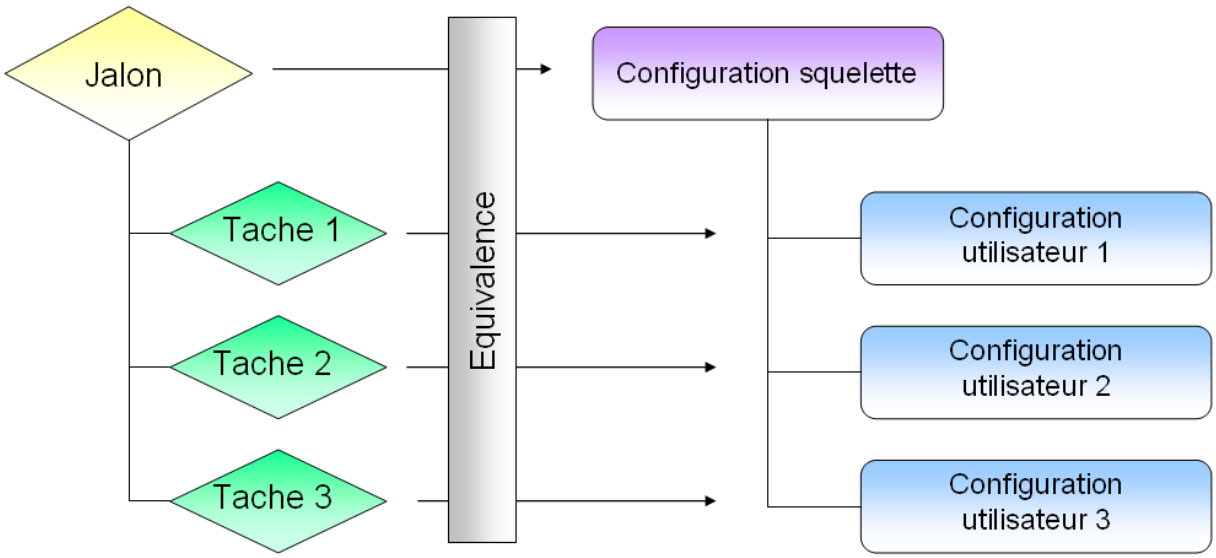


Figure 53 : organisation des configurations de connaissances

Le responsable du jalon est chargé de la création, du suivi et de la validation de la configuration squelette. **Elle est constituée de trois parties** (Figure 54) :

- **Une partie référence** contient les instances d'ICEs préparées par le responsable du jalon. Elles sont, soit directement instanciées depuis la base générique et valuées (ou pas), soit récupérées dans d'autres configurations (de jalons précédent par exemple). Les instances de la partie référence servent de support de référence pour la création des configurations utilisateurs.
- **Une partie conflits** propose de visualiser les conflits existants entre les différentes instances d'ICEs (les paramètres et les règles) utilisées dans les configurations utilisateurs, plus précisément dans les versions publiées des configurations utilisateurs vers la configuration squelette. Le responsable de jalon peut ainsi redistribuer les tâches afin que les utilisateurs parviennent à un compromis (lors d'un processus itératif entre la configuration squelette et les configurations utilisateurs).
- **Une partie conformité** affiche l'ensemble des instances d'ICEs utilisées dans les configurations utilisateurs sur lesquelles il n'existe pas de conflits.

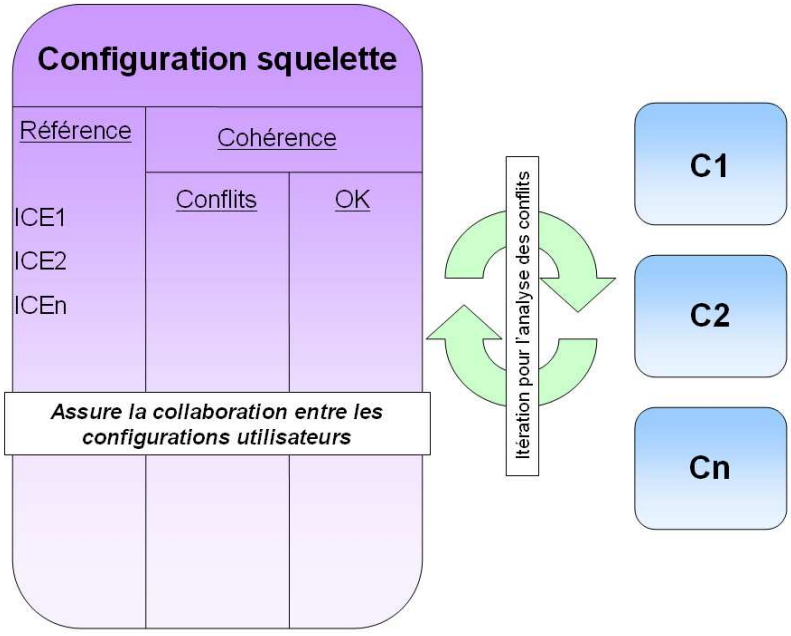


Figure 54 : représentation des différentes parties d'une configuration squelette

La configuration squelette offre une vision tableau de bord de l'ensemble des activités du jalon, elle est visible par tous les utilisateurs, mais seul son responsable peut directement la modifier ou modifier les configurations utilisateurs via la configuration squelette. Par exemple, s'il existe un paramètre "Longueur bielle" instancié, dont la valeur est égale à 42mm dans la configuration utilisateur 1, et une autre instance de ce même paramètre dans la configuration utilisateur 2, dont la valeur est de 45 mm, un conflit apparaît dans la configuration squelette et le responsable jalon pourra, à partir de la vue conflit, modifier les instances du paramètre de la configuration 1 ou 2

C'est donc de l'interaction entre ces deux types de configurations que la gestion des conflits est réalisée (Figure 55).

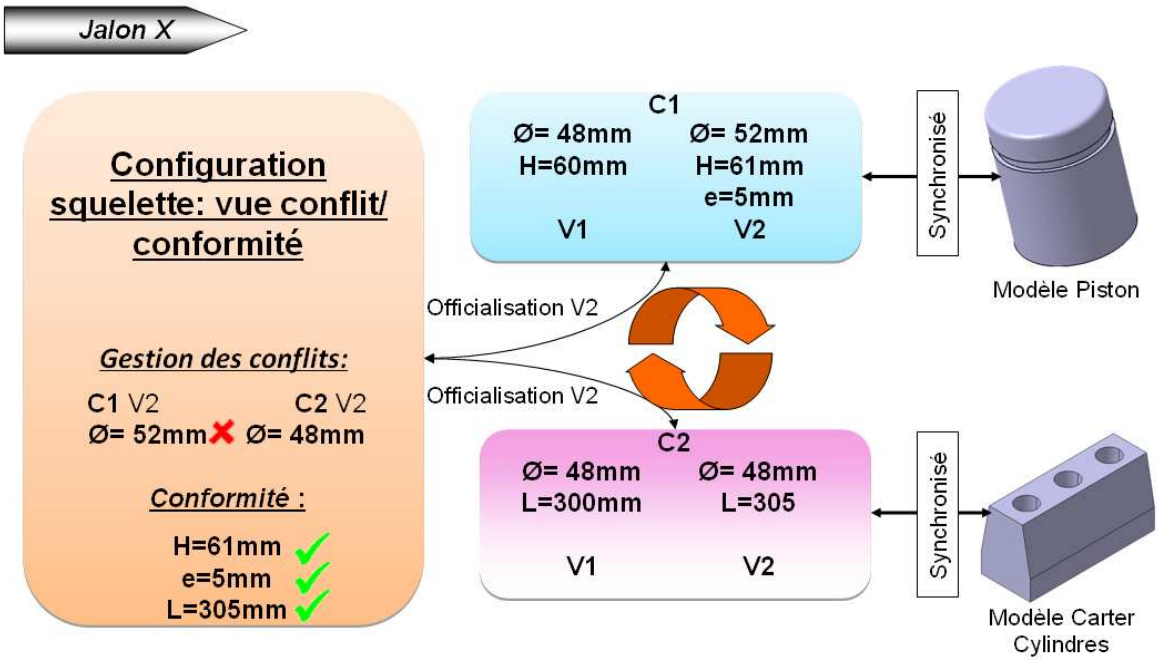


Figure 55 : gestion des conflits par la configuration squelette

Sur cette figure[6], deux modèles différents sont synchronisés avec deux configurations utilisateurs contenant plusieurs versions. Les versions 2 de chaque configuration utilisateurs sont alors publiées vers la configuration squelette pour être analysées en conflits. Ensuite, la configuration squelette présente sur la vue conflits les incohérences qui existent, ici entre deux valeurs du paramètre "Ø". Elle affiche aussi la vue conformité qui présente l'ensemble des paramètres sur lesquels il n'existe pas de conflits. La finalité de l'utilisation des configurations squelettes est de fournir, à la fin du travail dans un jalon, une vue de l'ensemble des connaissances utilisées, validées et cohérentes dans des modèles métiers hétérogènes.

3.4.4.3 Illustration d'utilisation de la méthodologie

Relativement aux concepts d'ICE et de configurations précédemment détaillés, nous proposons cette section dans laquelle KCMethod est illustrée sur la base du scénario proposé Figure 55, c'est-à-dire, utilisant une configuration squelette pour un jalon X, et deux configurations utilisateurs synchronisées avec deux modèles métiers.

[6] Pour faciliter la lecture sur la Figure 55, nous n'avons pas représenté les instances d'ICEs mais uniquement les paramètres contenus dans ces instances.

Il existe plusieurs possibilités pour créer les configurations, notamment de manière semi-automatique à partir des modèles métiers, en réutilisant des configurations types (Template de configuration) ou bien même, en effectuant des copies complètes ou partielles de configurations déjà utilisées lors de jalons précédents (Annexe 15). Le scénario que nous proposons de décrire met l'accent sur la création manuelle des configurations par les utilisateurs, sans récupérer d'informations issues d'autres jalons ou tâches.

Première étape, le responsable du jalon ou assimilé, crée la configuration squelette et l'affecte au jalon X. Une fois créée, le responsable du jalon fait plusieurs recherches par filtres dans la base d'ICEs afin de sélectionner celles qu'il souhaite instancier dans la partie "Référence" de la configuration squelette et value les paramètres s'il le souhaite.

Seconde étape, les utilisateurs créent chacun leur configuration utilisateur indépendamment les uns des autres et l'affecte, dans le jalon, à leur tâche en lien avec leur modèle métier. Ensuite, ils sélectionnent dans la partie référence de la configuration squelette les instances d'ICEs qu'ils souhaitent utiliser et les copies, dans leurs configurations utilisateur, en initialisant la valeur des instances par rapport à la valeur proposée par le responsable du jalon. Les utilisateurs ont aussi la possibilité d'aller directement piocher dans la base d'ICEs. Dans ce cas, les nouvelles ICEs instanciées sont automatiquement ajoutées dans la partie référence de la configuration squelette avec la première valeur donnée par le premier utilisateur (Figure 56). Si ces instances d'ICEs peuvent être directement utilisées dans les configurations utilisateurs, le responsable du jalon devra quand même les valider sous peine de bloquer le processus final de validation des configurations. Une même ICE peut être instanciée plusieurs fois dans une configuration, et/ou dans plusieurs configurations simultanément avec différentes valeurs sur les paramètres.

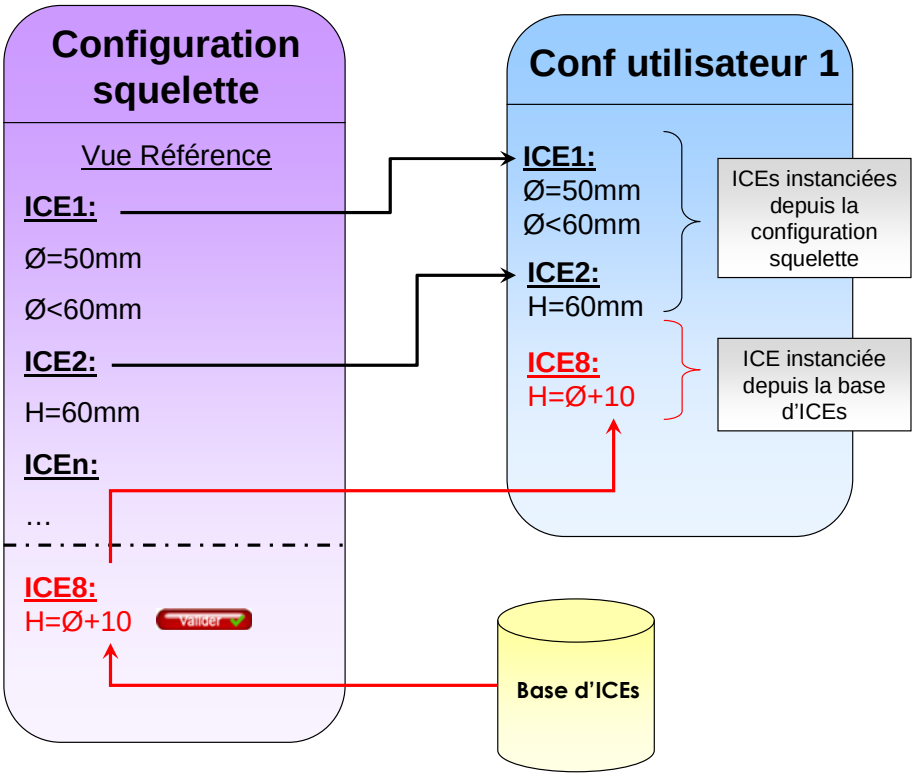


Figure 56 : création d'une configuration utilisateur à partir de la configuration squelette et de la base d'ICEs

Troisième étape, les utilisateurs synchronisent leurs configurations avec leurs modèles métiers et travaillent sur ces modèles. Au fur et à mesure de leur avancement, ils resynchronisent la configuration et créent plusieurs versions afin de sauvegarder différents états ou concepts explorés dans les modèles. A ce stade, au sein de sa propre configuration, un utilisateur visualise si des conflits existent sur des contraintes non respectées, comme des conditions limites ou des règles.

Quatrième étape, quand un utilisateur est satisfait d'une version de sa configuration, il la publie vers la configuration squelette afin quelle soit analysée en conflits avec les autres versions de configurations utilisateurs déjà publiées. Une seule version à la fois par configuration peut être publiée dans la configuration squelette. Cette publication fige la version de configuration en question, n'empêchant tout de même pas l'utilisateur de travailler sur d'autres versions en parallèle.

Cinquième étape, la configuration squelette analyse les instances, entre elles, de l'ensemble des configurations utilisateurs publiées et affiche les conflits. C'est-à-dire, les contraintes non respectées mais aussi les instances d'ICEs utilisées dans plusieurs configurations utilisateur à la fois, avec différentes valeurs sur les paramètres partagés. La gestion des conflits s'effectue de manière itérative, bouclant entre les configurations utilisateurs et les modèles, ainsi qu'entre les configurations utilisateurs et la configuration squelette, jusqu'a éliminer l'ensemble des conflits.

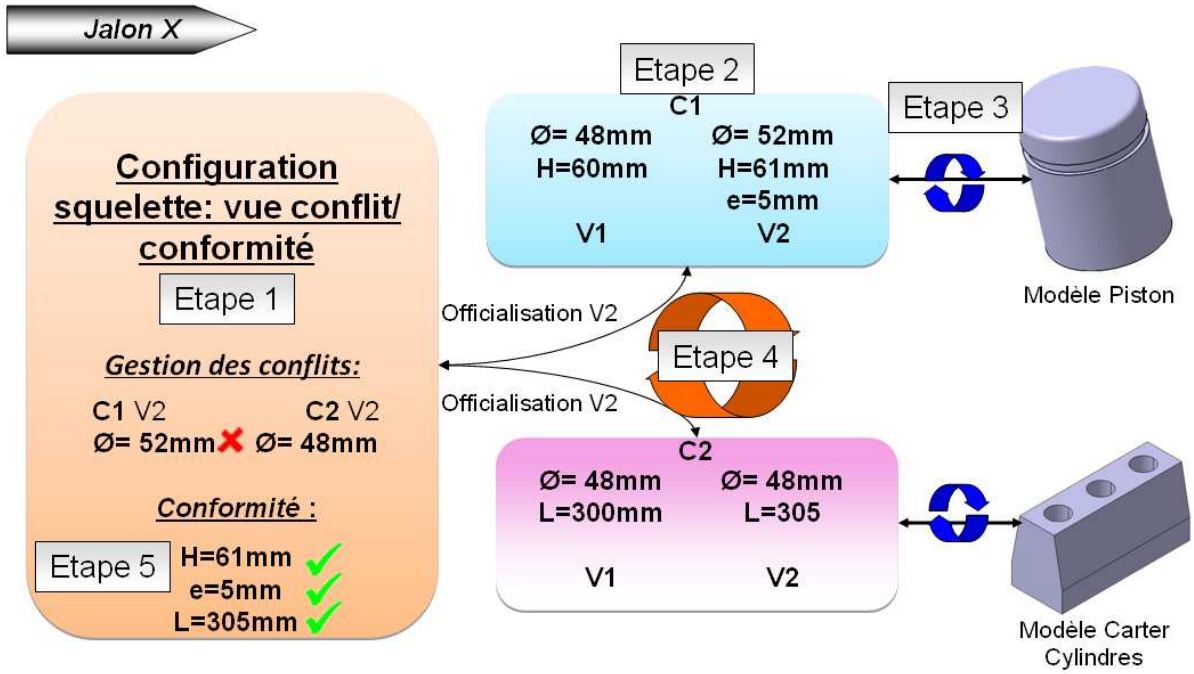


Figure 57 : processus KCMethod

A l'issue de ce processus illustré Figure 57, les configurations suivent un cycle de validation qui permet de valider avec confiance l'ensemble des connaissances utilisées au travers des activités du jalon. Ces connaissances peuvent alors être réutilisées pour une deuxième étape du jalon (création d'une version 2 de la configuration squelette), dans un jalon différent ou même sauvegardées pour être réutilisées dans un autre projet.

Durant tout le processus de gestion des configurations de connaissances les utilisateurs manipulent les différents objets de KCMethod pour converger vers uns solution cohérente. Dans ce contexte, les concepteurs ainsi que les responsables de jalons ont accès à certaines fonctionnalités leur permettant une gestion plus fine des connaissances et une plus grande souplesse d'utilisation. Nous proposons de décrire, dans la section suivante, certaines de ces fonctionnalités :

- Liées à la création des configurations
- Liées à la manipulation des instances dans les configurations
- Liées à la gestion des conflits entre deux configurations utilisateurs

3.4.4.4 Fonctionnalités liées à l'utilisation des ICEs et des configurations des connaissances

La première fonctionnalité décrite, concerne l'utilisation des ICEs. Les ICEs permettent de capitaliser plusieurs paramètres et règles à la fois, les rendant indissociables. Néanmoins, dans certains cas particuliers, il arrive que les utilisateurs aient besoin de ne pas utiliser, ou du moins de différer l'utilisation de certains paramètres de l'ICE dans une configuration. Ils ont alors la possibilité de désactiver ces paramètres dans les instances d'ICE présentes dans les configurations (exemple au chapitre 5 sur le cas automobile). Si des contraintes portent sur ces paramètres, elles seront aussi désactivées, qu'elles soient dans la même ICE ou dans une autre. L'utilisateur est alors averti de cette propagation de désactivation à laquelle il doit prêter attention.

Autre fonctionnalité concernant l'utilisation des instances d'ICEs dans les configurations, les utilisateurs disposent de deux possibilités leurs permettant de lier des paramètres d'une configuration utilisateur à une autre :

- L'abonnement : c'est une fonctionnalité qui permet à un utilisateur de s'abonner à une instance d'ICE utilisée dans une autre configuration utilisateur. Ainsi, il sera averti des modifications réalisées qu'il pourra ensuite prendre en compte.
- Le lien maitre/esclave : c'est une fonctionnalité qui permet à un utilisateur de lier son instance d'ICE en esclave à une instance d'ICE utilisée dans une autre configuration utilisateur qui sera considérée en maitre. Ainsi, l'instance esclave sera directement pilotée par la maitre sans que l'utilisateur de cette instance soit perturbé.

Le responsable de jalon dispose également de fonctionnalités lui permettant de mieux traiter les conflits. Ainsi, il peut, à partir de la configuration squelette, geler des instances d'ICEs utilisées par les concepteurs. De la même manière, il à la possibilité de geler une configuration entière. Ces fonctionnalités participent à l'amélioration de la convergence des utilisateurs vers une solution de conception commune et cohérente.
Au niveau de la gestion de la cohérence, le responsable de jalon a la possibilité de choisir de négliger un conflit (exemple au chapitre 5 sur le cas automobile). En effet, certains conflits ne peuvent être résolus à l'instant "t" ou ils apparaissent. La résolution de ces conflits nécessitent, par exemple, de récupérer des résultats issus d'autres jalons. Ainsi, ces conflits pourront être identifiés comme "en attente" et traités ultérieurement sans qu'ils ne bloquent le processus de validation des configurations de connaissances.

Enfin, il est possible d'effectuer une analyse de gestion de conflits sans passer par la configuration squelette. En effet, afin de décharger le responsable de jalon de la gestion d'un trop grand nombre de conflits, les utilisateurs ont la possibilité de directement comparer leurs configurations entre elles. En effet, quand un utilisateur manipule une instance d'ICE, il a la possibilité de visualiser si une autre instance de la même ICE est partagée par une autre configuration, et avec quelle valeur (Figure 58).

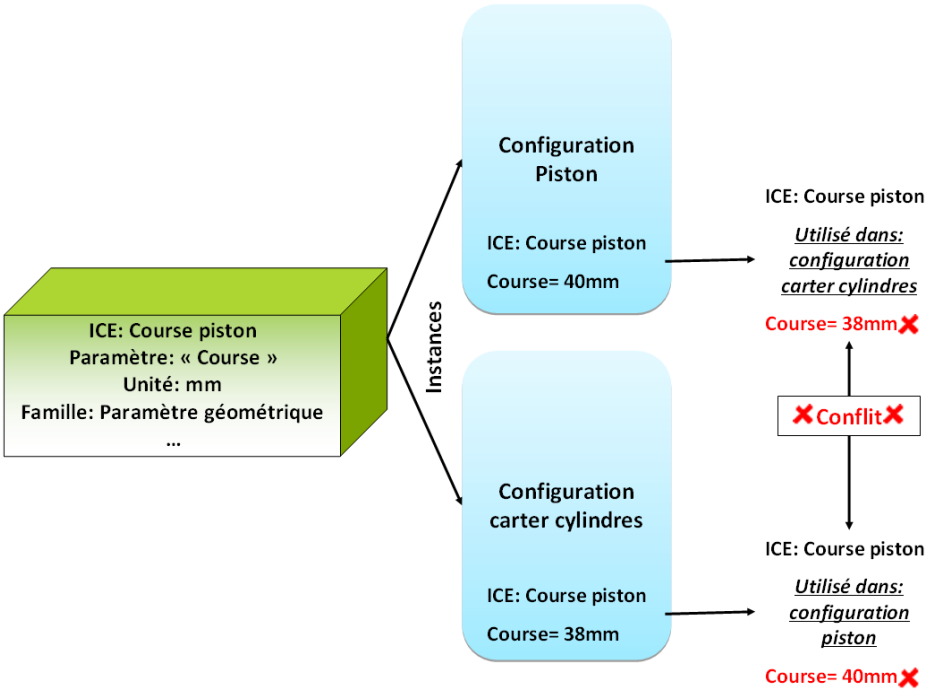


Figure 58 : gestion des conflits entre configurations utilisateurs

Sur cette figure, l'utilisateur de la configuration piston est informé que l'utilisateur de la configuration carter cylindres utilise également l'ICE "course piston" dans sa configuration avec une valeur du paramètre course égale à 38mm, alors que sa valeur du même paramètre est de 40mm. Sur la base de ce constat, les deux utilisateurs peuvent se contacter afin de trouver une issue à cette incohérence.
Ainsi, les conflits qui surviendront dans la configuration squelette sont ceux qui nécessiteront plus d'approfondissement ou la prise d'une décision par le responsable du jalon. La gestion des conflits est un élément essentiel de notre méthodologie que nous proposons de décrire plus en détail.

3.4.5 La gestion des conflits

Nous avons largement abordé la notion de cohérence et de gestion des conflits lors de la définition du processus KCMethod et de la description des différents concepts. Ainsi, nous allons préciser, avec plus de détail, comment les conflits sont identifiés et quels moyens les utilisateurs ont d'agir dessus.

La gestion des conflits porte sur l'analyse des instances d'ICEs et plus particulièrement sur les paramètres et les contraintes instanciées dans les configurations. Elle s'effectue à deux niveaux dans KCMethod :

- En interne dans une configuration utilisateur : dans ce cas, l'analyse des conflits porte sur le respect des contraintes en fonction de la valeur des paramètres
- Entre plusieurs configurations : dans ce cas, l'analyse des conflits porte sur le respect des contraintes mais aussi sur la comparaison des instances de paramètres utilisées dans plusieurs configurations. Dans ce contexte, c'est la configuration squelette qui est chargée de cette analyse.

L'objectif de la gestion de conflits est de détecter les incohérences et remonter cette information aux utilisateurs pour leurs permettre d'itérer, prendre des décisions et finalement converger vers une solution complètement instanciée cohérente (cf. notion d'itération et de convergence, chapitre 1, section 1.1.2.2).

3.4.5.1 Cohérence sur les instances de paramètres

Concernant les paramètres, la gestion de la cohérence porte essentiellement sur la vérification et la comparaison des instances utilisées dans différentes configurations. Illustré Figure 57, "Ø" a une valeur de 52mm dans la Configuration 1 (C1) et 48mm dans la Configuration 2 (C2). Cette différence stigmatise l'existence d'un conflit entre les instances utilisées dans deux configurations synchronisées avec des modèles métiers. La comparaison des instances d'ICEs, dans le cadre de la vérification des paramètres, s'effectue dans la configuration squelette. Elle présente les instances d'ICEs issues des versions publiées des configurations utilisateurs vers la configuration squelette, sur lesquels un conflit existe (Figure 61).

3.4.5.2 Cohérence sur les contraintes

La gestion de la cohérence sur les contraintes s'effectue à la fois dans les configurations utilisateurs et dans la configuration squelette. Elle permet aux concepteurs de s'assurer de bien prendre en compte et de respecter les conditions limites et les règles dans leurs modèles, en fonction de la valeur des paramètres entrés. On détecte également les incohérences issues de l'utilisation de règles incompatibles.
Pour travailler avec les contraintes, les utilisateurs disposent de deux fonctionnalités leur permettant de les assister dans leur choix et d'assurer la cohérence. En effet, les contraintes de chaque ICE peuvent prendre deux états : "vérification" ou "propagation" avec des comportements bien différents. En fonction de leurs besoins, les concepteurs peuvent faire passer les contraintes d'un état à l'autre, sachant que "vérification" est l'état par défaut.

Le premier état "vérification" permet de littéralement vérifier le respect des contraintes en fonction de la valeur des paramètres. La contrainte retourne une information de ok/ko à l'utilisateur qui devra alors manuellement modifier les valeurs des paramètres pour satisfaire la contrainte (Tableau 5).

A=20mm **B=10mm** **A=B+10 OK**	**A=25mm** **B=10mm** **A=B+10 NO**

Tableau 5 : vérification des contraintes

Le second état "propagation" permet de propager les contraintes au fur et à mesure que l'utilisateur définit les valeurs des paramètres. Par exemple, si le concepteur fixe la valeur du paramètre "A", la valeur du paramètre "B" sera automatiquement fixée ou bornée dans un intervalle en accord avec la contrainte. Cet état nécessite l'utilisation d'outils de type moteur de CSP - Constraint Satisfaction Problems (Annexe 16) permettant d'utiliser des mécanismes résolution et de propagation sur les contraintes (sur des domaines non bornée et continus) [Vareilles, 2005]. Ces moteurs ont la capacité de résoudre une équation de manière acausale, c'est à dire sans limiter la "directionnalité". De manière générale, cela permet de mieux prendre en compte les contraintes et d'assister les utilisateurs dans le choix des valeurs (Tableau 6).

A [5mm] **B [5mm;+∞]** **C [0mm; +∞]** **A<=B** **B<=A+C**	**A [-1mm; 4mm]** **B [4mm]** **C [5mm]** **A<=B** **B<=A+C**	**A [-∞; +∞]** **B [-∞;+∞]** **A=B+10** **B=A+2** **Incompatibilité des règles**

Tableau 6 : propagation des contraintes

- Dans la première colonne, l'utilisateur commence par fixer la valeur de A (5mm), ensuite les valeurs de B et C sont proposées dans des intervalles possibles.
- Dans la deuxième colonne, l'ingénieur commence par fixer la valeur de C (5mm), puis la valeur de B (4 mm), enfin la valeur de A est proposée dans l'intervalle [-1mm; 4mm].
- Dans la dernière colonne, l'ingénieur a choisi deux règles qui sont incompatibles.

L'analyse des contraintes permet aussi d'identifier redondance cyclique et les incompatibilités de règles. Par exemple, la Figure 59 illustre un modèle de conception de bielle simplifié. Elle est définie à partir de six paramètres directeurs liés par cinq relations (règles dans le cadre).

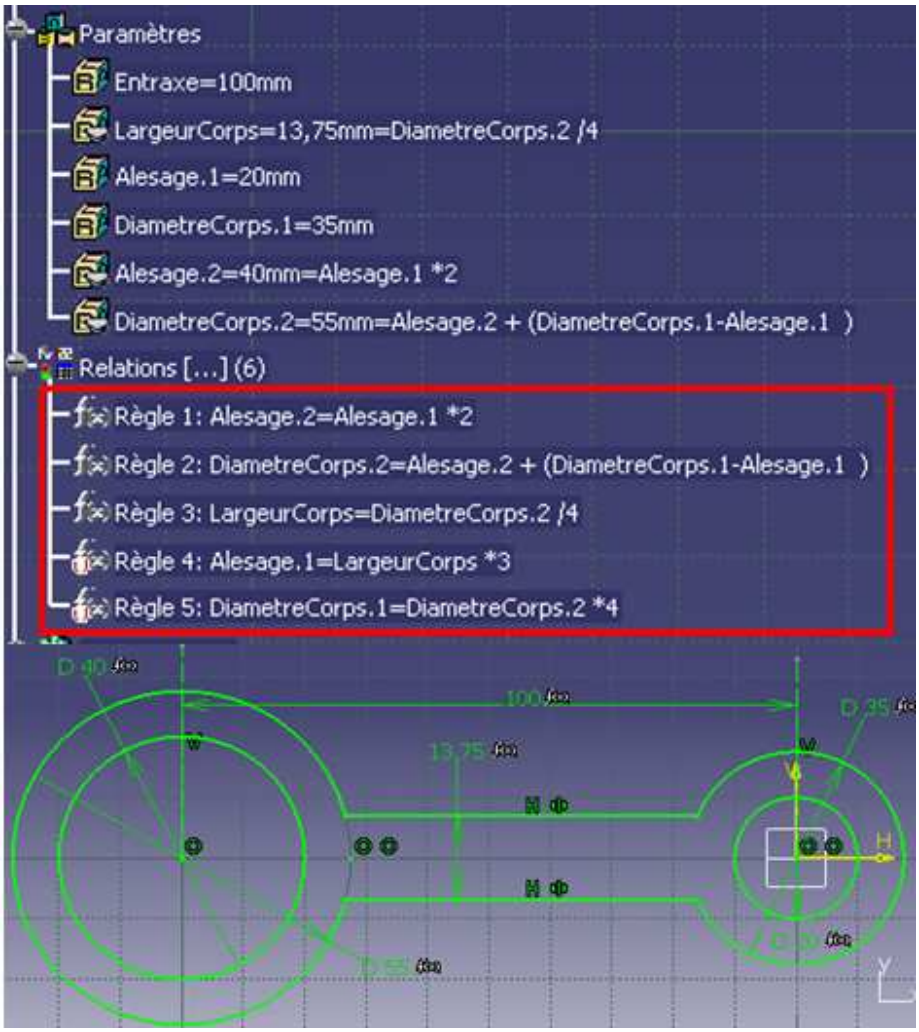


Figure 59 : exemple de bielle paramétrée et contrainte

En analysant les relations, on remarque que des incohérences existent et donc que des paramètres sont sur-contraints (Figure 60). Par exemple, le paramètre 2 est ici clairement sur-contraint, à la fois parent et enfant et impliqué dans de nombreuses relations.

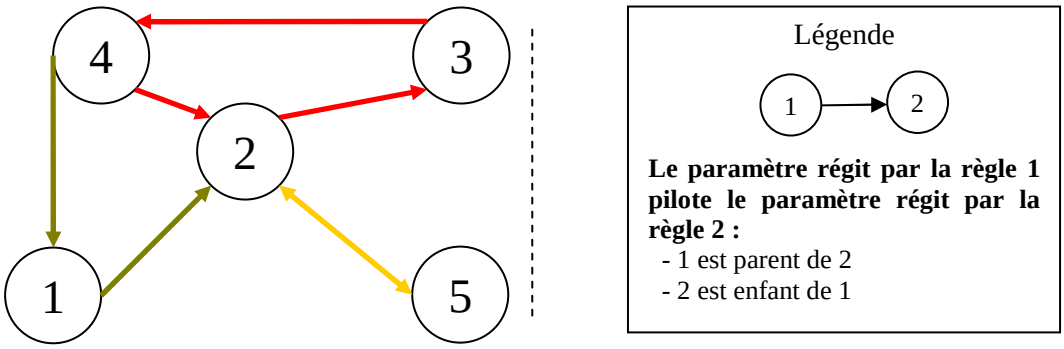


Figure 60 : relations entre les paramètres

- Boucle 1 : 2 ⇔ 5
- Boucle 2 : 2 → 3 → 4 (→ 2)
- Boucle 3 : 2 → 3 → 4 → 1 (→ 2)

Donner l'information sur les règles impliquées dans ces boucles, ainsi qu'identifier les paramètres sur-contraints, permet aux utilisateurs de prendre des décisions afin de maintenir la cohérence des connaissances partagées par plusieurs modèles métiers.
Ces mécanismes préfigurent une base nécessaire pour l'utilisation de systèmes d'aide à la décision plus poussés, explorés dans plusieurs travaux au sein de l'équipe INCIS ainsi que dans le consortium ADN.

3.4.5.3 Visualisation des conflits dans la configuration squelette

Une fois les conflits détectés, il s'agit de remonter l'information aux différents utilisateurs. Dans le cas de l'analyse des conflits entre configurations utilisateurs, c'est la configuration squelette qui est chargée d'afficher ces informations.
La Figure 61 illustre un exemple de la vue conflit de la configuration squelette, organisée par ICEs. La première ICE "course piston" contient un paramètre unique "course" utilisé dans deux versions de configurations utilisateurs officialisées : Calcul bielle polaire V2 et Calcul bielle tenue V1.1. Ainsi, il existe un conflit entre ces deux instances d'ICEs car la valeur des instances n'est pas la même.

		Config Calcul bielle polaire V2	Config Calcul bielle tenue V1.1	
Course piston	Course	**79,5 mm**	**80 mm**	Conflit
Responsable: Julien BADIN				

		Config Calcul CC pression V1.1	Config Calcul Vilo vibratoire V1	
Entre axe futs	`Entr'Axe`	**89 mm**	**88 mm**	Conflit
Responsable:				

		Config Calcul bielle polaire V2	Config Calcul tenue bielle V1.1	
Rapport λ bielle/course	Lambda	**1,7484**	**1,75**	Conflit
Responsable:	>3	**Non respecté**	**Non respecté**	Conflit
	λ='Long_Bielle'/'Course'	Respecté	Respecté	
	Course	79,5 mm	80 mm	
	Long_Bielle	139 mm	140 mm	

Figure 61 : visualisation des conflits dans la configuration squelette

Autre exemple de visualisation de conflits avec l'ICE "Rapport λ bielle/course". Dans cet exemple, un conflit existe sur le paramètre "λ", il est utilisé dans deux versions de configurations avec des valeurs différentes. De plus, les contraintes de cette ICE sont en état "vérification", donc la configuration squelette retourne la valeur "Non respecté" pour la contrainte λ>3. En revanche, "λ" respecte bien la relation qui lie ce paramètre à "course" (issu de l'ICE précédente) et "long_Bielle" qui apparaissent en gris car ils appartiennent à d'autres ICEs.
La configuration squelette évolue dynamiquement au fur et à mesure des itérations avec les configurations utilisateurs. Ils peuvent visualiser toutes les informations de la configuration squelette, mais pas directement intervenir dessus. C'est le rôle du responsable de jalon qui peut directement effectuer des modifications depuis la configuration squelette ou geler des instances d'ICEs, ou même des configurations entières.

La résolution des conflits est de nature à valider l'ensemble des connaissances utilisées dans un jalon. Un processus de validation est défini pour gérer les interactions entre les deux types de configurations.

3.4.6 Le processus de validation des configurations

L'objectif de ce processus est de valider une configuration squelette et donc figer l'ensemble des configurations du jalon. Cela signifie, pour ce jalon projet, que les différentes activités de conception-calcul sont terminées, validées, et surtout que les connaissances utilisées sont cohérentes (avec une traçabilité des modifications et une capitalisation des configurations). Pour illustrer ce processus de validation, nous définissons deux points de vue, avec pour chacun d'eux, des droits particuliers (détail du processus en Annexe 17) :

- Le point de vue utilisateur
- Le point de vue responsable jalon

Pour qu'une configuration squelette soit validée, il faut respecter deux conditions majeures :

- Il ne doit plus y avoir de conflits (ou s'il en reste, qu'ils soient déclarés comme ignorés)
- Le statut des configurations utilisateurs doit être en état "à valider", ce qui signifie qu'un concepteur a terminé le travail sur son modèle et que sa configuration ne subira plus de modification.

Si une de ces conditions n'est pas respectée, la configuration squelette ne pourra pas être validée (soit il faut résoudre les conflits, soit les configurations peuvent changer et de nouveaux conflits peuvent apparaitre).

Chaque type de configuration possède des statuts différents :

- Trois pour la configuration squelette :
    - Travail en cours - Work in Progess (WIP)
    - A valider - Ready
    - Validé - Over (ou Terminated)

- Quatre pour les configurations utilisateur :
    - A justifier - New
    - Travail en cours - Work in Progress
    - A valider - Ready
    - Validé - Over (ou Terminated)

Ces différents attributs interviennent dans un processus global de validation liant l'utilisation des configurations utilisateurs à la configuration squelette.

3.4.6.1 Processus global de validation

De manière globale, le processus de validation suit le raisonnement présenté en Figure 62 qui met en parallèle les deux cycles de validation des deux types de configurations et leurs liens.

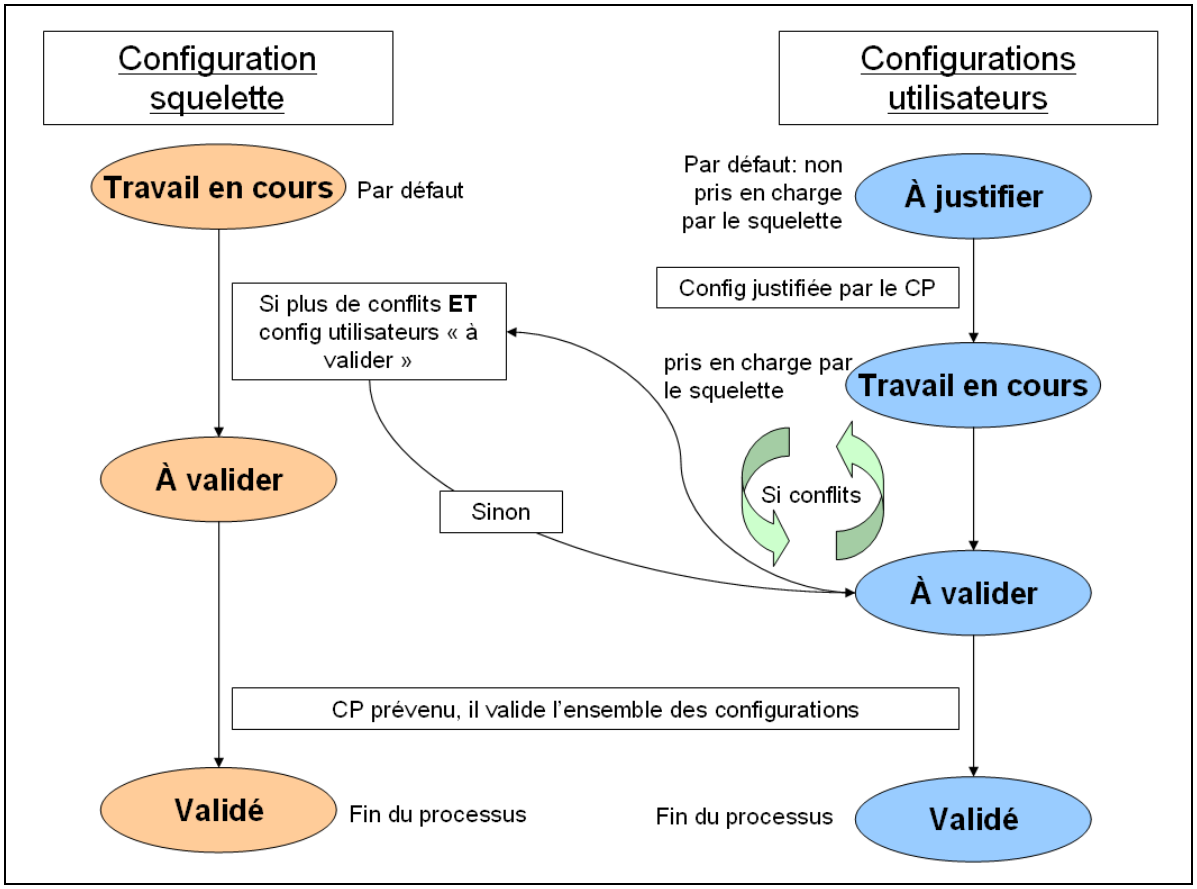


Figure 62 : cycle global de validation des configurations

Suite à sa création, la configuration squelette possède le statut "WIP" (Work In Progress). Le passage en statut "Ready" se fait automatiquement quand les deux conditions préalablement citées (plus de conflits et configurations utilisateurs en statut "Ready") sont respectées. Le responsable jalon est alors prévenu et peut faire un dernier point sur l'ensemble des configurations, et valider le tout en donnant à la configuration squelette le statut "Over". A ce moment, toutes les configurations utilisateurs passent également en statut "Over" et l'ensemble des configurations du jalon est officiellement validé, garantissant la cohérence de toutes les connaissances utilisées dans toutes les activités du jalon.

Concernant les configurations utilisateurs, elles apparaissent par défaut en "New", ce qui signifie qu'elles peuvent être utilisées mais qu'elles ne sont pas prises en charge par le squelette (pas de publications possible). Une fois que le responsable de jalon valide leur création, leur statut passe en "WIP" et l'utilisateur peut publier des versions vers la configuration squelette. Quand l'utilisateur est satisfait et qu'il estime en avoir terminé avec son modèle métier, il fait passer le statut en "Ready" et la configuration est figée. Ainsi, l'utilisateur ne peut plus publier d'autres configurations ou faire de modification (mais il peut continuer à travailler sur d'autres versions de configurations utilisateur). Finalement, les configurations utilisateur passent en statut "Over" automatiquement quand le responsable de jalon valide la configuration squelette. Il est alors possible de créer une nouvelle version de la configuration squelette, de sorte à la réutiliser (tout ou partie) dans le projet.

3.4.6.2 Versionnement de la configuration squelette

Le versionnement d'une configuration squelette à pour effet de créer une nouvelle version dans laquelle la partie référence sera initialisée par la partie "Conformité" de la version précédente. Elle peut être placée dans un nouveau jalon ou alors dans le même jalon. Dans ce dernier cas, cette nouvelle version se comporte comme LA configuration squelette de référence du jalon (l'ancienne version étant figée mais toujours consultable). Bien entendu, cette nouvelle version peut être modifiée à souhait, c'est-à-dire, ajouter ou enlever des instances d'ICEs ou simplement modifier les valeurs des instances des paramètres.

3.4.6.3 Bilan sur le processus de validation

Ce processus global de validation et de droits en fonction des utilisateurs garantit une bonne utilisation et la cohésion des configurations entre elles. Par ce biais, la traçabilité et la cohérence des connaissances manipulées est garantie.
Le processus décrit ici est un processus global qui doit subir des aménagements en fonction de l'organisation qui utilise cette méthodologie. Des rôles peuvent être ainsi définis plus précisément avec des droits d'accès spécifiques, notamment si une entreprise utilise un outil KCManager avec des sous-traitants qui ne doivent pas avoir accès à toutes les informations capitalisées.

3.4.7 La réutilisation des connaissances

KCMethod facilite la réutilisation des connaissances au sein d'un projet ou au travers de différents projets. Au sein d'un projet la réutilisation des connaissances peut être réalisée par la création de liens entre les configurations via des mécanismes d'abonnements ou de maitres/esclaves tels que décrits précédemment. Elle peut aussi être envisagée par une réutilisation de connaissances issues de configurations en cours d'utilisation, ou validées pour créer de nouvelles configurations.
On peut ainsi, à partir d'une configuration existante, créé de nouvelles configurations en copiant tout ou partie de ses instances d'ICEs. Ce processus peut s'effectuer entre configurations utilisateurs dans un jalon ou bien être utilisé pour créer une nouvelle configuration squelette. Les configurations (squelette ou utilisateur) sont aussi capitalisables sous forme de Templates. Dans ce cas, elles sont stockées dans une libraire accessible par plusieurs projets de conception de produits. En effet, bien souvent deux projets de conception d'un même type produit sont très similaires et par conséquent des configurations issues d'un projet son réutilisables dans l'autre permettant de réduire les processus routiniers et de gagner du temps. Ainsi, on peut donc considérer qu'en plus de la base d'ICEs qui s'enrichit au fil du temps, on crée une base de Templates de configurations de connaissances réutilisables qui peut être vue comme la définition de "standards" de connaissances ré-applicables.
Afin de pouvoir maitriser la taille de ces bases et d'optimiser la réutilisation des informations et des connaissances d'un projet à l'autre, nous définissons le concept de Knowledge Domain.

3.4.8 Détail du concept de Knowledge Domain

Le concept de Knowledge Domain permet de définir un contexte macroscopique pour la définition de la base d'ICEs et l'utilisation des configurations. En effet, on peut qualifier un Knowledge Domain d'environnement global d'utilisation des connaissances. Par exemple, dans le cas de l'industrie automobile, un constructeur produit un système complet qui est une voiture. Ce système est décomposé en sous-systèmes majeurs, tels que le groupe-moto-propulseur (GMP), le train avant, le train arrière, l'habitacle, etc. Dans cet exemple, ce constructeur veut recréer ce découpage afin de capitaliser les connaissances respectives à chaque système. Ainsi, le concept de Knowledge Domain permet de recréer ce découpage au niveau de l'utilisation des informations et des connaissances, de sorte à limiter la taille des bases d'ICEs et de configurations. Ainsi, quand un nouveau projet est créé, il est positionné dans un Knowledge Domain en particulier. Chaque Knowledge Domain possède sa propre base d'ICEs et de configurations, et tout ce qui est créé dans un Knowledge Domain reste dans

le Knowledge Domain en question. Des cas particuliers sont considérés pour des ICEs ou des configurations qui appartiendront aux frontières des Knowledge Domains. Concrètement, une ICE interface de deux Knowledge Domains existera dans les deux bases d'ICEs avec un lien pour la synchronisation.
L'autre intérêt du concept de Knowledge Domain réside dans le fait qu'il pourra être customisé pour correspondre aux activités réalisées dans les projets. C'est-à-dire qu'il est possible de créer des canevas de projets avec des configurations déjà prêtes, des équipes projets déjà définies et des fonctionnalités particulières pour chaque Knowledge Domain (Figure 63). Par exemple, un Knowledge Domain GMP thermique contiendra l'ensemble des ICEs potentiellement utiles à la conception de n'importe quel GMP thermique, ainsi que des Configurations Templates réutilisables dans les différents projets. Il contiendra également les profils des personnes travaillant dans les activités de conception des GMP, des Knowledge Index préétablis, etc.
L'utilisation du Knowledge Domain est de nature à favoriser la réutilisation des connaissances d'un projet à l'autre, de sorte à limiter les processus routiniers.

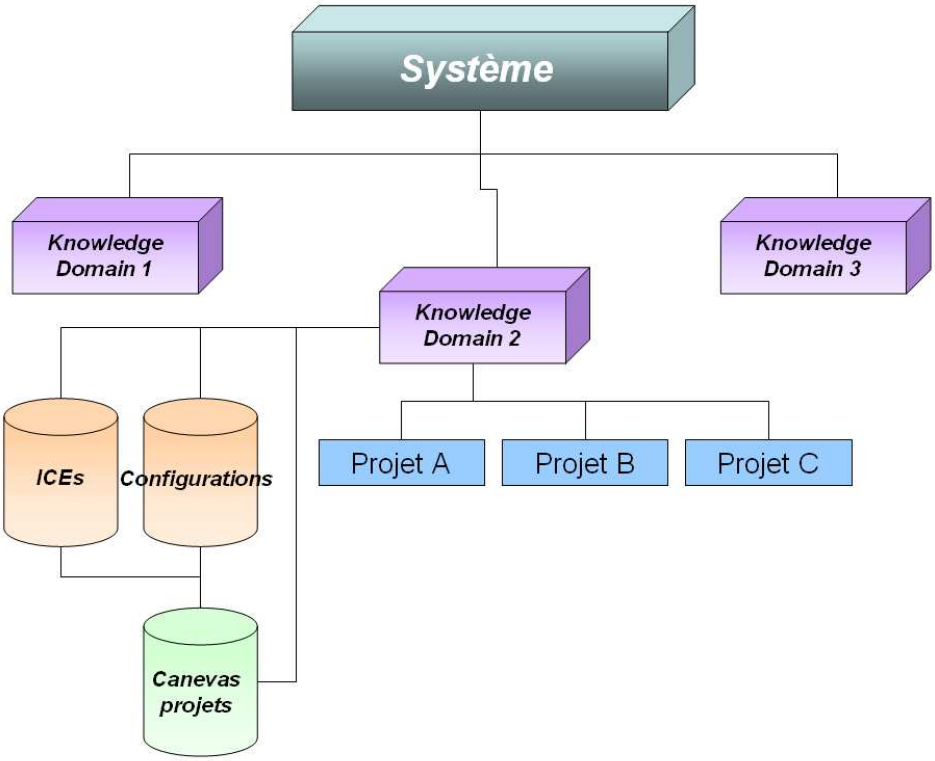


Figure 63 : exemple de décomposition en Knowledge Domain

Si dans l'exemple ci-dessus, le concept de Knowledge Domain est utilisé pour recréer un découpage structurel, c'est bien sur un cas particulier. En effet, le Knowledge Domain représente un domaine d'utilisation d'informations et de connaissances et ne correspond pas nécessairement à une décomposition structurelle d'un système. Par exemple, on peut très bien créer un Knowledge Domain pour la mécanique quantique regroupant l'ensemble des connaissances relatives à cette science. Autre exemple, créer un Knowledge Domain "équipe de recherche X" regroupant l'ensemble des connaissances manipulées par cette équipe, etc.

3.4.9 L'aspect collaboratif

Pour terminer sur la méthodologie KCMethod, nous proposons de brièvement décrire les principes de collaborations qui permettent aux différents utilisateurs de communiquer. Ce sont des mécanismes classiques déjà employés dans les solutions PLM. Deux types de collaborations sont possibles : la collaboration statique et la collaboration dynamique.

3.4.9.1 Collaboration statique

La collaboration statique consiste à définir des actions à entreprendre appelées Review. Une Review est caractérisée par différents principaux attributs :

- Un émetteur
- Un ou plusieurs destinataires
- Une date de début
- Une date de fin
- Un objectif
- Un statut
- Une version
- Un historique

Ainsi, un utilisateur crée une Review pour remplir un objectif donné et l'affecte à un autre utilisateur. Ce dernier reçoit un mail et peut visualiser la Review qui lui a été attribuée. Quand il à terminé son travail, il retourne la Review à son émetteur pour validation avec explication de ses actions. Si l'émetteur valide, la Review est fermée.
Une Review peut être crée par le responsable de jalon afin que deux utilisateurs résolvent un conflit. Des utilisateurs peuvent aussi créer des Review pour demander la création ou la modification d'ICEs génériques. L'intérêt des Review c'est de disposer d'un suivi des actions à entreprendre avec une notion de traçabilité car elles disposent d'un cycle de vie et d'une version. De manière globale, des Review sont créées pour différents contextes :

- Déclarer un problème
- Faire une demande de création ou de modification
- Créer une tâche à réaliser

3.4.9.2 Collaboration Dynamique

La collaboration dynamique consiste à réaliser des revues de projet en direct à distance. Par exemple, le responsable de jalon souhaite résoudre un conflit entre deux configurations, si les deux utilisateurs responsables de ces configurations son connectés il peut les inviter à une revue de projet en temps réel avec partage d'écran. Ils peuvent ainsi résoudre en direct les conflits, ou travailler ensemble sur les configurations.

3.5 Conclusion

Au cours de ce chapitre, nous avons présenté la première partie de notre contribution sous forme de description de la méthodologie KCMethod et de ses concepts fondamentaux. Cette méthode s'adresse à la conception de produits et concerne essentiellement les phases amont, avec pour objectif de favoriser le partage et la cohérence des connaissances échangées dans les modèles métiers.
Nous pouvons conclure ce chapitre sur la nécessité de mieux prendre en compte les connaissances encapsulées dans les modèles métiers via l'utilisation des ICEs et des KnowledgeConfigurations afin d'apporter une réponse au polyptique CTRC identifiée au chapitre 1. L'organisation des connaissances en configurations confère à la méthode son originalité et permet une gestion très flexible et dynamique, nécessaires dans les phases amont du processus de conception. Cette définition de gestion des connaissances en configurations s'intègre dans le contexte PLM et nécessitera une évolution des approches actuelles par l'introduction d'un nouvel outil KCManager complémentaire aux applications existantes, pour une meilleure prise en compte des données, informations et des connaissances dans la conception de produits.
Dès lors, nous avons formalisé la méthodologie sous forme d'un méta-modèle générique, de sorte à proposer un outil informatique nous permettant de tester l'approche sur différents cas issus de l'industrie manufacturière.

Chapitre 4. Le méta-modèle KCModel

Dans le cadre de ce quatrième chapitre nous proposons une formalisation de la méthodologie KCMethod expliquée au chapitre trois, sous forme d'un méta-modèle ; KCModel (Knowledge Configuration Model).
Dans un premier temps, nous expliquons quelle approche de modélisation nous avons suivie pour formaliser le KCModel. Ensuite, nous décrivons l'architecture du méta-modèle, c'est-à-dire sa décomposition en différents packages, puis nous expliquons les différents concepts en les illustrant via des diagrammes de structures et des diagrammes d'activités. Enfin, nous terminons par les objectifs d'implémentation et d'utilisation de ce méta-modèle pour définir un outil basé sur la gestion des connaissances en configurations qualifié de KCManager.

4.1 Introduction

La contribution présentée dans ce chapitre concerne la définition d'un méta-modèle pour la gestion des connaissances en configurations. Baptisé KCModel pour Knowledge Configuration Model, il consiste en une formalisation des concepts de la méthodologie KCMethod.

La Figure 64 illustre l'articulation du chapitre quatre organisée en trois parties.

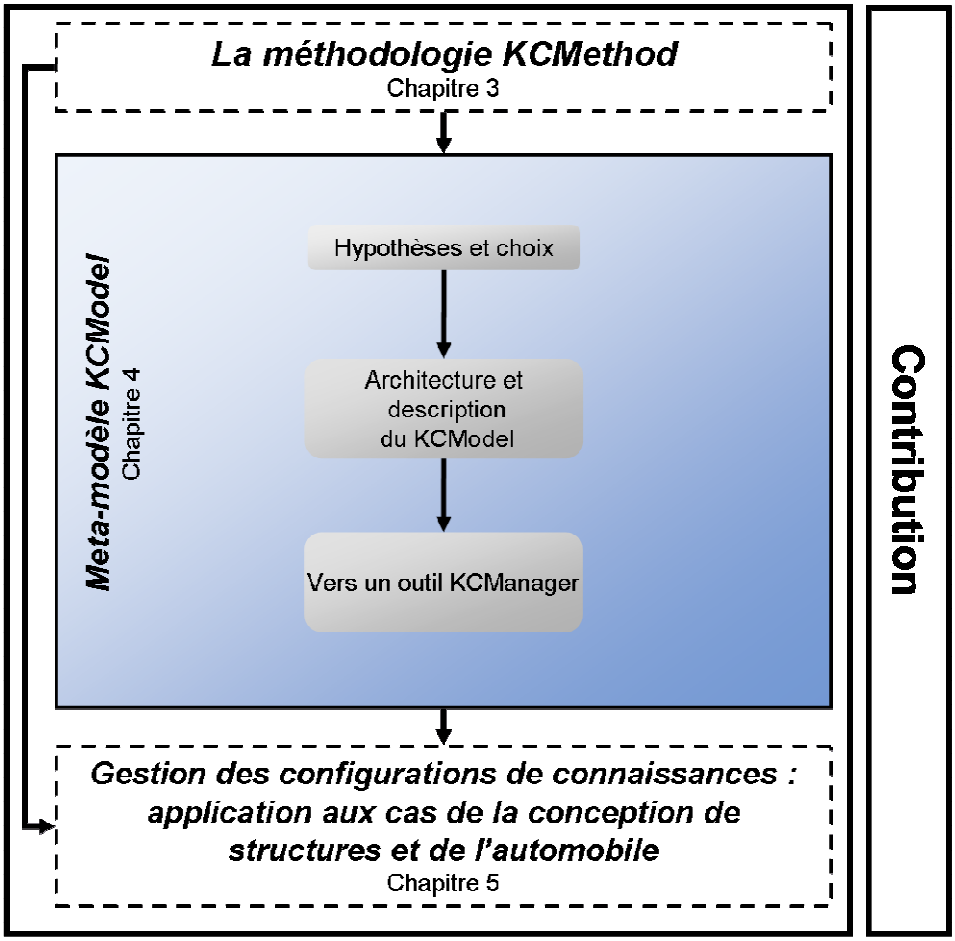


Figure 64 : organisation de la contribution pour la définition du méta-modèle KCModel

Dans la première partie, nous expliquons quel langage est utilisé et quelle philosophie de modélisation a été adoptée. En effet, le KCModel est basé sur les principes de méta-modélisation utilisant les langages orientés objets. Nous avons utilisé le standard UML (Unified Modelling Language) proposé par l'OMG (Object Management Group) [OMG, 2007a], [OMG, 2007b] pour formaliser les concepts et illustrer leur fonctionnement. Le méta-modèle est également basé sur les approches d'Ingénierie Dirigée par les Modèles (IDM[7]) visant à atteindre un haut niveau d'abstraction et donc de définir des modèles, indépendamment des considérations techniques telles que des plateformes ou des langages de programmation.

[7] MDA/MDE en anglais: Model Driven Architecture/Model Driven Engineering sont des approches basées sur des principes d'ingénierie générative visant à générer du code directement à partir de modèles tissés et transformés, mais représentatifs des concepts que l'on souhaite exprimer.

Dans la seconde partie, nous présentons l'architecture du KCModel puis, nous décrivons les différents diagrammes qui définissent nos concepts fondamentaux tels que les ICEs ou les KnowledgeConfigurations. Afin de mettre en évidence le fonctionnement du méta-modèle et d'illustrer les fonctionnalités majeures, nous avons réalisé des diagrammes de structures et d'activités mettant en évidence l'application du KCModel relativement à KCMethod.

Enfin, dans la troisième partie, nous abordons nos objectifs d'implémentation du méta-modèle pour développer un outil KCManager. Nous présentons la structure envisagée pour cet outil de gestion des connaissances en configurations. Puis, nous revenons sur le positionnement d'un tel outil dans l'environnement de l'entreprise.

4.2 Hypothèses et choix

4.2.1 La structure du KCModel

Pour définir le KCModel, nous nous sommes basés sur la définition d'un "modèle" proposée par [Combemale, 2008] : « *Un modèle est une abstraction, une simplification d'un système qui est suffisante pour comprendre le système modélisé et répondre aux questions que l'on se pose sur lui. Un système peut être décrit par différents modèles liés les uns aux autres* ».
En ce sens, nous considérons "un modèle d'application du KCModel", comme étant un modèle de connaissances, dans la mesure où il permet de représenter les connaissances utilisées dans les modèles métiers. C'est-à-dire, répondre aux questions en lieu et place des modèles métiers eux-mêmes en ce qui concerne la manipulation des paramètres et des contraintes encapsulés.
Par exemple, dans le cadre d'un projet de conception, un modèle d'application KCModel définit un ensemble de configurations de connaissances, elles mêmes composées d'instances d'ICEs contenant les valeurs des paramètres pour ce projet. Ces configurations sont synchronisées avec les modèles CAO et CAE et évoluent au fur et à mesure du travail des concepteurs.

La définition de modèles requiert la mise en œuvre de méta-modèles : un ensemble de concepts et leurs relations à utiliser pour exprimer des modèles. Ceci définit l'activité de méta-modélisation, c'est-à-dire suggérer un formalisme de modélisation, chaque modèle se conformant à un méta-modèle prédéfini. **Ainsi, pour définir un modèle d'application KCModel, nous avons réalisé un méta-modèle KCModel qui exprime les concepts et les relations, c'est-à-dire le langage qui permet de construire le modèle pour un contexte donné (cf. 2.2.2.2).**

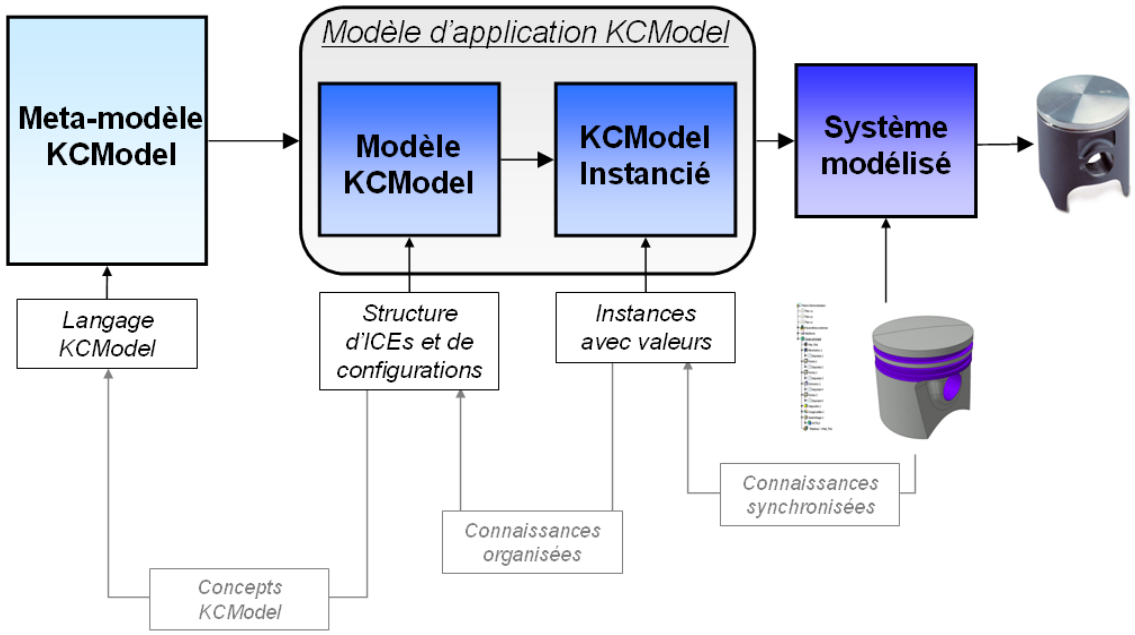


Figure 65 : principe de méta-modélisation du KCModel

Ainsi, la Figure 65 illustre notre contribution. Au premier niveau, on trouve le méta-modèle KCModel qui porte la structure et la définition sémantique des concepts KCModel (décrits dans KCMethod), c'est à dire le langage KCModel. La méta-modélisation est une problématique clé des approches d'IDM [Combemale, 2008].
Au second niveau, on trouve le modèle KCModel obtenu par instanciation du méta-modèle pour un domaine donné.
Enfin, au dernier niveau, on trouve le modèle instancié (ou l'implémentation du modèle) obtenu par instanciation du modèle KCModel. Ce modèle comporte les valeurs des paramètres et les instances de contraintes, et est représentatif du système modélisé, c'est-à-dire, des modèles métiers représentant les composants réels (avec les valeurs des paramètres).
Cette décomposition d'un modèle de connaissance n'est pas nouvelle mais classique des approches de méta-modélisation telles que décrites par [Harani, 1997], [Sellini, 1999] [Menand, 2002]. En effet, cette architecture à trois niveaux est basée sur les préconisations de l'OMG [OMG, 2001] en termes d'IDM (MDA), comme nous l'avons vu, elle est composée :

- du premier niveau appelé méta-modèle,
- du second appelé modèle
- du troisième niveau, l'implémentation du modèle (modèle instancié)

Dans ces travaux, afin de faciliter la compréhension, notamment quand des exemples seront présentés sous forme de diagrammes, et afin de ne pas manipuler trop de niveaux de modélisations, nous proposons de grouper le modèle KCModel et le KCModel instancié sous la notion de modèle d'application KCModel (Figure 65). Il constitue un modèle instancié définissant les structures d'ICEs et de configurations contenant les valeurs de paramètres pour un projet. Le modèle d'application KCModel représente donc le modèle opérationnel utilisable dans un projet par des concepteurs.
En conséquence, à partir d'un même méta-modèle, on peut générer plusieurs modèles d'applications correspondants à différents contextes (Figure 66). Sur cette figure, le méta-modèle permet d'instancier des modèles d'application KCModel correspondants à différents projets de conceptions. Chaque modèle d'application contient un ensemble d'ICEs et de configurations de connaissances (avec valeur des paramètres) correspondant aux activités du processus de conception.

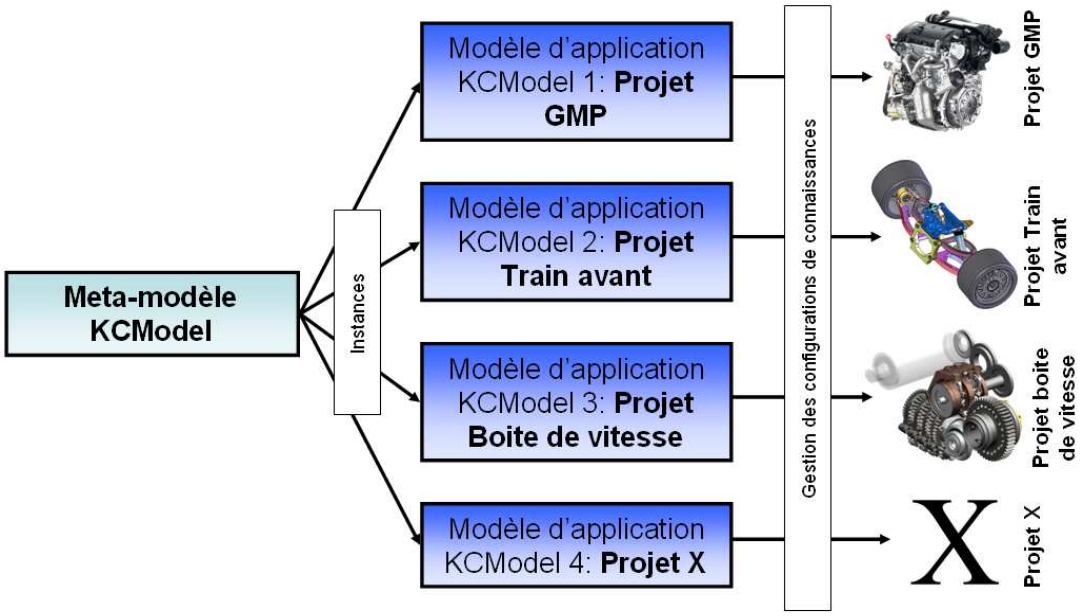


Figure 66 : génération de plusieurs modèles d'application KCModel à partir du méta-modèle

Dans ce contexte, le modèle d'application KCModel doit, pour être utilisable, être conforme au méta-modèle KCModel. Le méta-modèle est alors considéré comme représentant cet ensemble de modèles d'applications auquel chacun d'entre eux doit se conformer. C'est sur ces principes de base que s'appuie l'OMG pour définir l'ensemble de ses standards.

Nous avons utilisé le langage UML (Unified Modeling Language) [OMG, 2007a, OMG, 2007b] pour définir notre méta-modèle et illustrer son fonctionnement. UML permet d'avoir une grande souplesse de modélisation et d'explication, dans la mesure où il est possible de créer de nombreux diagrammes (classe, activité, structure, etc.).

La Figure 67 illustre schématiquement l'articulation des modèles entre eux sur fond d'exemple. Dans le cadre du méta-modèle KCModel, on retrouve la définition du concept d'ICE et de KnowledgeConfiguration au niveau de la méta-modélisation.
Le cadre suivant illustre une instanciation du méta-modèle pour une application sur un projet GMP. Ainsi, le modèle d'application KCModel – GMP propose plusieurs ICEs créées et instanciées (avec valeurs) dans une configuration squelette et plusieurs configurations utilisateurs (pour un piston et un bloc moteur).
Le modèle d'application est synchronisé avec les modèles CAO du piston et du bloc moteur, il est donc représentatif des connaissances encapsulées dans ces modèles.

4.2.2 L'implémentation pour la définition d'une application logicielle

Cette approche de méta-modélisation traduit des préconisations issues de l'IDM visant à fournir un grand nombre de modèles pour exprimer séparément chacune des préoccupations des utilisateurs, des concepteurs, des architectes, etc. L'IDM inclut pour cela la définition de plusieurs standards, notamment UML. Le principe clé de l'IDM consiste à s'appuyer sur ces standards pour décrire séparément des modèles pour les différentes phases du cycle de développement d'une application. Plus précisément, l'objectif majeur de l'IDM consiste en l'élaboration de modèles pérennes (PIM - Platform-Independent Model), indépendants des détails techniques des plateformes d'exécution (J2EE, .Net, PHP, etc.), afin de permettre la génération automatique de la totalité des modèles de code (PSM - Platform Specific Model) et d'obtenir un gain significatif de productivité.

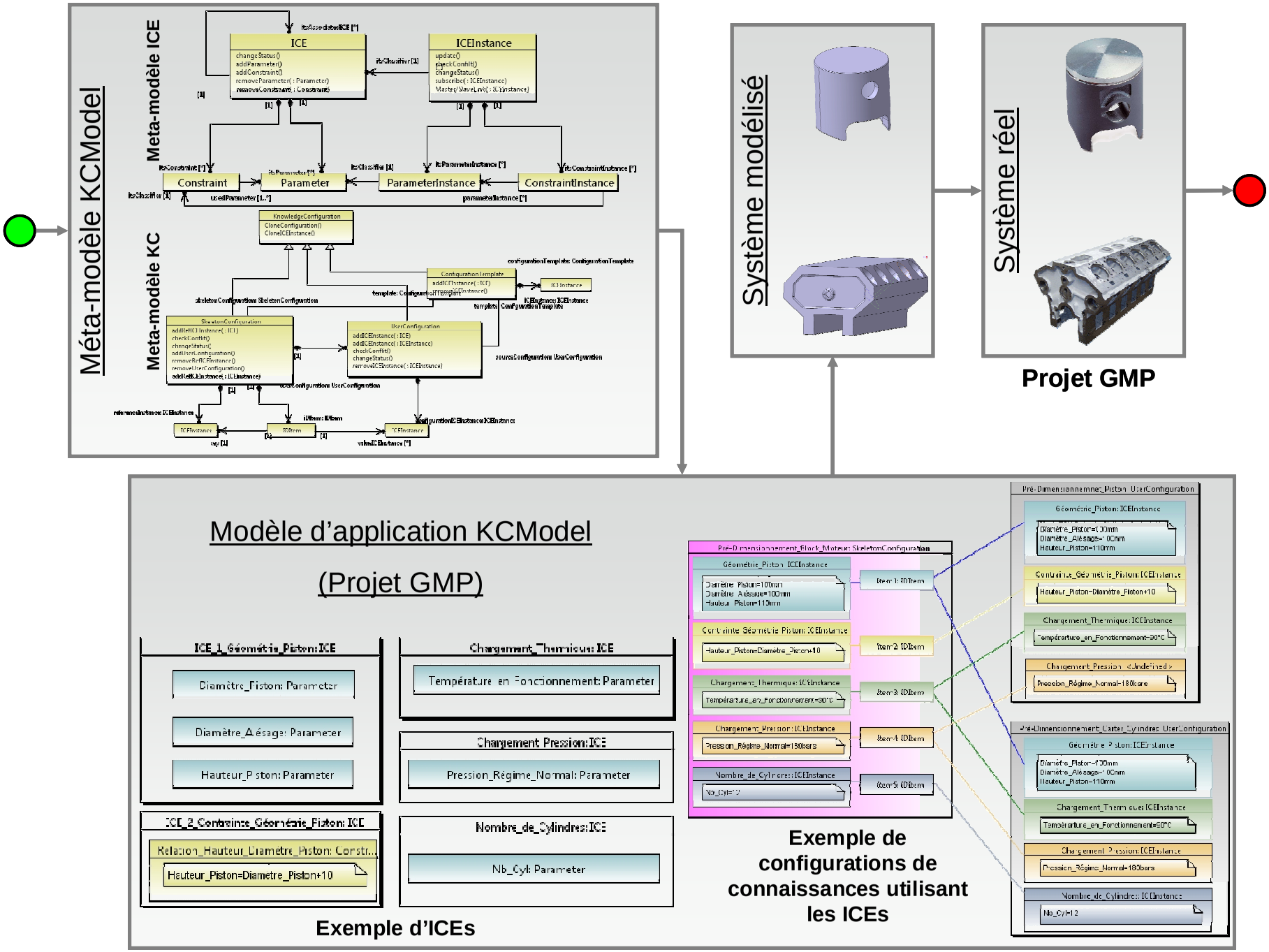


Figure 67 : exemple des principes de méta-modélisation du KCModel

4.3 Architecture conceptuelle du KCModel

4.3.1 Vue d'ensemble

KCModel est un méta-modèle générique décomposé en trois parties (Figure 68), chacune groupant plusieurs packages correspondant à des préoccupations différentes. Ces packages sont génériques, on les lie les uns aux autres (par des relations de binding[8]) pour former un tout cohérent, c'est-à-dire dans l'objectif d'avoir un comportement représentatif de la méthodologie KCMethod.

La première partie est appelée **KCMCore**, il s'agit du cœur du méta-modèle KCModel. Il donne les principales définitions sémantiques des concepts et décrit la structure principale du KCModel et ses relations. C'est ce package qui décrit les concepts essentiels de la méthodologie KCMethod.

La seconde partie, **KCMTool**, fournit une description d'une plateforme informatique permettant d'implémenter le KCMCore afin de concrètement réaliser les fonctionnalités. Ce modèle définit la structure de la plateforme et les fonctionnalités transversales d'outillage[9].

La troisième partie, nommée **KCMImplementation,** permet d'expliquer l'implémentation du KCMCore dans la plateforme KCMTool. Contrairement aux deux parties précédentes qui ne contiennent que des diagrammes de classes, cette troisième partie contient des diagrammes de structures et d'activités permettant d'illustrer le fonctionnement du KCModel.

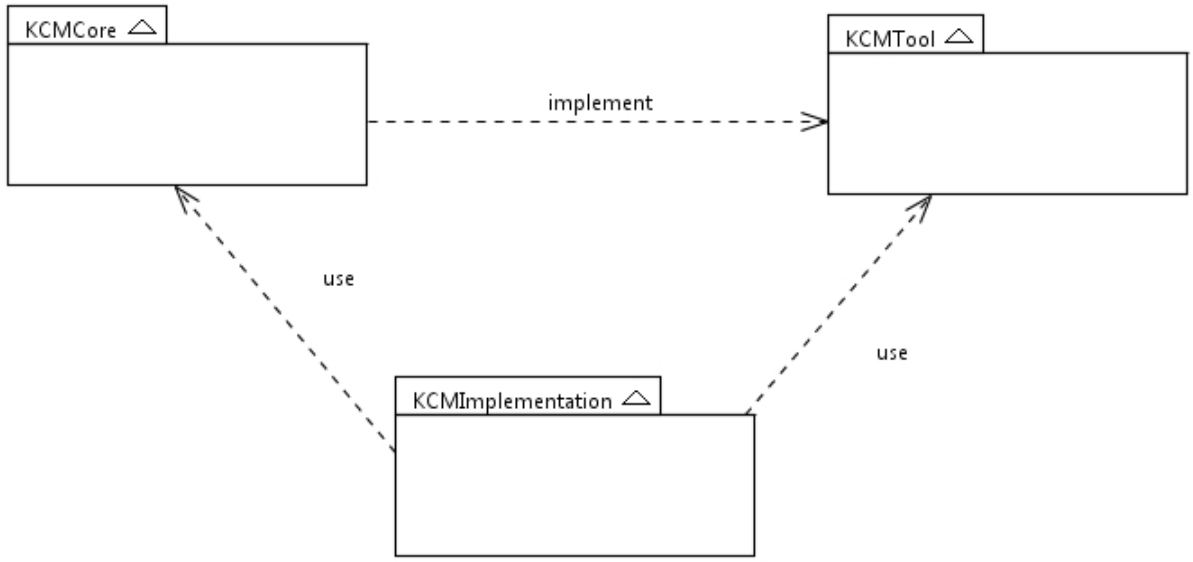


Figure 68 : les trois parties du méta-modèle KCModel

[8] Binding : permet de lier des modèles sur des plans conceptuels identiques ou différents. Concrètement, cela permet d'associer les propriétés d'une classe aux propriétés d'une autre classe.

[9] Ex : l'IHM (Interface Homme Machine), gestion d'accès, gestion de configuration, gestion de représentation, etc.

4.3.2 KCMCore

KCMCore est composé de cinq packages définissant les principaux concepts KCModel :

- Le package **InformationCoreEntity** contient la structure des ICEs ainsi que la définition d'un paramètre et d'une contrainte.
- Le package **KnowledgeConfiguration** définit la structure des configurations de connaissances.
- Le package **Knowledge Domain** définit la structure des Knowledge Domain.
- Le package **KCMProject** définit l'organisation des configurations dans une structure projet.
- Le package **KCMType** contient, à son tour, plusieurs packages définissant des types primitifs par défaut dans le KCModel, tels que les opérateurs nécessaires pour les contraintes ou les différents types de contraintes prises en compte

4.3.2.1 Package Information Core Entity

Ce package (Figure 69) définit la structure permettant la centralisation et la capitalisation générique de paramètres et de contraintes dans des entités Information Core Entity. Les ICEs permettent de donner aux paramètres et aux contraintes leur propre structuration (leur propre organisation) et cycle de vie. Afin de ne pas surcharger le diagramme, les attributs des ICEs énoncés au chapitre 3 section 3.4.2.1, ne sont pas représentés.

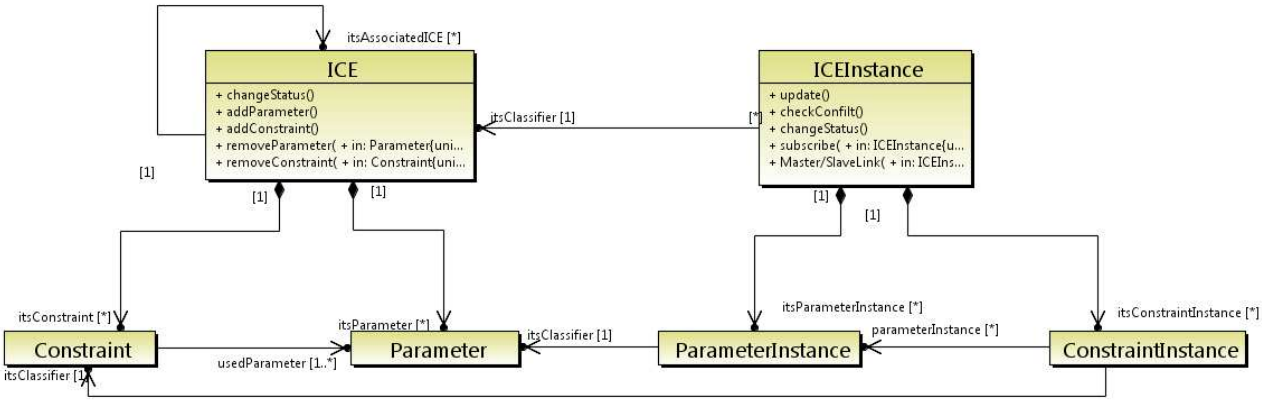


Figure 69 : package ICE

La classe ICE

Une ICE est une entité générique indécomposable qui permet de capitaliser et de structurer les données cruciales extraites des modèles métiers et des experts, afin de passer de l'état de donnée à celui d'information (cf. chapitre 3 section 3.4.2)

Elle est composée de paramètres et de contraintes génériques, de manière à ce que l'ensemble des **ICEs constitue une base générique d'informations techniques d'un produit**. L'ICE est l'entité de plus bas niveau manipulée dans la méthodologie KCMethod. Des liens peuvent exister entre plusieurs ICEs (liens entre paramètres ou entre contraintes et paramètres appartenant à des ICEs différentes). L'ICE est une entité dynamique, elle possède un cycle de vie et par conséquent elle peut prendre différents statuts. Elle évolue au fil des modifications à prendre en compte sur les paramètres et les contraintes.

La classe Parameter (Figure 70)

Un paramètre est une donnée exprimée selon un point de vue d'utilisation d'une caractéristique mesurable ou quantifiable.

Un paramètre ne peut être défini que dans une ICE. Il est typé par une grandeur physique en relation avec des unités standards.

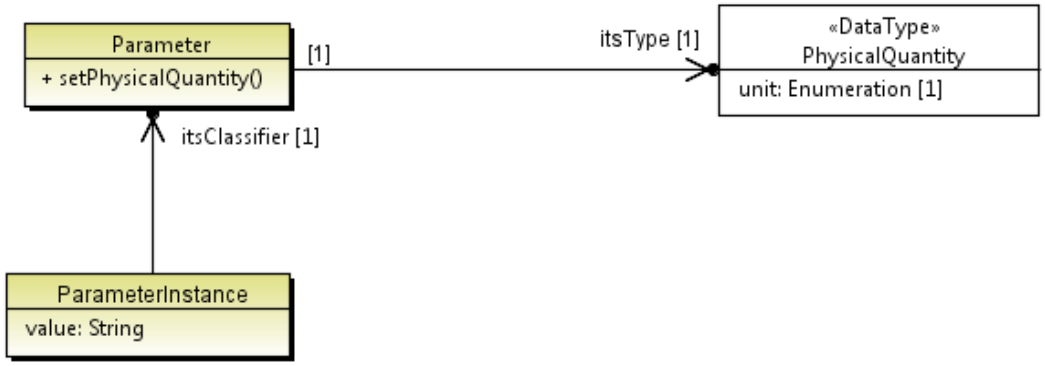


Figure 70 : définition d'un paramètre

Par exemple, un paramètre nombre de cylindres (Nb_cyl) est associé à une quantité physique de type "Entier", et n'a pas d'unité.
Autre exemple, un paramètre Diamètre alésage (Ø_Alesage) est associé à une quantité physique de type "Longueur" et une énumération d'unité correspondante (mm, cm, etc.)

La classe PhysicalQuantity

Toute propriété de la nature qui peut être quantifiée par la mesure ou le calcul, et dont les différentes valeurs possibles s'expriment à l'aide d'un nombre généralement accompagné d'une unité de mesure[10].

Une grandeur peut être exprimée à partir de plusieurs unités respectives. Ainsi, la masse et la longueur sont des grandeurs qui s'expriment respectivement en kilogramme et en mètre (ou en multiple de ces unités de base), alors que l'indice de réfraction d'un milieu s'exprime à l'aide d'un nombre sans unité et constitue une grandeur sans dimension.

La classe Constraint (Figure 71)

Contraint est un concept d'obligation créé par les règles en usage dans un milieu ou par les lois propres à un domaine.

Une contrainte peut seulement être créée dans une ICE. Chaque contrainte contient sa propre expression sous format textuel et connait les paramètres utilisés.

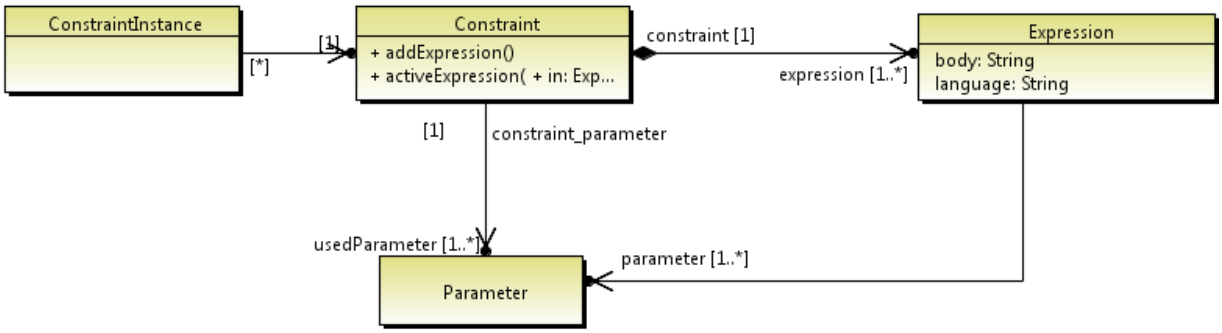


Figure 71 : définition d'une contrainte

[10] Référence Wikipedia, définition de quantité physique

La classe Expression

Texte structuré d'après un langage qui permet d'exprimer une spécification. Une contrainte possède un corps textuel qui peut être exprimé en plusieurs langages.

Une contrainte peut être exprimée en langage littéraire ou en langage de programmation.
Par exemple, une contrainte possède une expression mathématique peut être exprimée en langage java, C++ ou autre.

La classe ICEInstance

ICEInstance est une application d'une ICE qui contient les valeurs des paramètres (ParameterInstance) et les instances de contraintes (ConstraintInstance).

Chaque ICEInstance d'une ICE connait son propre classificateur ICE, c'est-à-dire que pour chaque ICEInstance il est possible de retrouver l'ICE source.
Attention, l'ICEInstance n'est pas une instance d'ICE. En effet, l'ICE peut être considérée comme un **référentiel qualité ou métier** (c'est-à-dire transverse) alors que l'ICEInstance correspond d'avantage à **un référentiel projet.**
L'ICEInstance vérifie la cohérence de ses contraintes en fonctions des valeurs des paramètres.
Ce concept existe pour différentes raisons. Premièrement plusieurs ICEInstances (dérivées d'une même ICE) peuvent simultanément exister dans plusieurs configurations mais avec des valeurs différentes de paramètres. Une ICEInstance peut d'ailleurs être instanciée plusieurs fois dans une même configuration. Il est alors nécessaire de savoir quelle ICEInstance comparer à quelle autre entre plusieurs configurations. Ensuite, si une ICE évolue, l'ICEInstance facilite la propagation dans les configurations permettant aux utilisateurs d'utiliser la nouvelle version ou de rester sur l'ancienne.

La classe ParameterInstance

ParameterInstance est une instance de la classe Parameter

ParameterInstance contient des valeurs spécifiques des paramètres pour un contexte d'utilisation donné.

La classe ConstraintInstance

ConstraintInstance est une instance de la classe Constraint.

ConstraintInstance contient des contraintes exprimées pour un contexte donné.

Exemple d'ICE :
Scénario 1 : création d'une ICE contenant des paramètres et des contraintes
Conformément à KCMethod, un utilisateur souhaite créer une ICE appelée "My ICE", contenant trois paramètres "a", "b" et "c" et deux contraintes : "form" et "condition" (Figure 72).
Les trois paramètres possèdent une quantité physique de type longueur à laquelle est rattachée une énumération d'unités correspondantes (m, cm, mm, etc.). L'unité choisie pour ces trois paramètres est le "mm".
La première contrainte utilise les paramètres "a", "b" et "c" et possède l'expression "(a+b)/c=350", exprimée en langage Java.
La seconde contrainte n'utilise que le paramètre "c" et possède l'expression "0<c<10" exprimée également en langage Java.

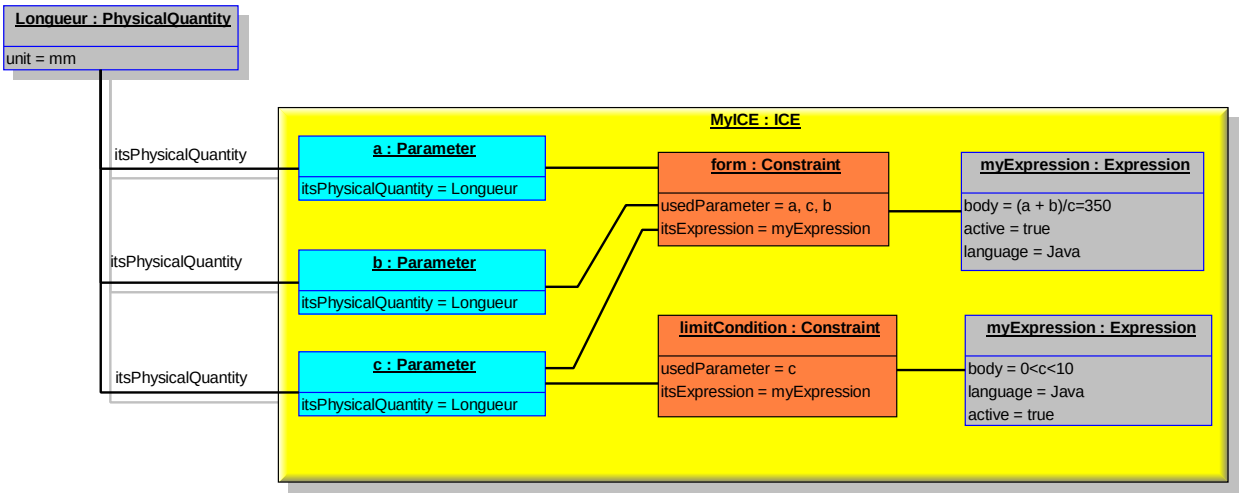


Figure 72 : exemple d'ICE

Scénario 2 : création de plusieurs ICEs liées par des contraintes

Une ICE peut être liée à d'autres ICEs (cf. chapitre 3 section 3.4.2), c'est-à-dire que les paramètres ou les contraintes capitalisées peuvent dépendre d'autres paramètres ou contraintes contenus dans plusieurs ICEs. Par exemple, sur la Figure 73, l'ICE_1 dépend de l'ICE_2 et de l'ICE_3 dans la mesure où la contrainte "c1" est définie à partir de "p1" et "p2" (capitalisés dans l'ICE_1), mais aussi de p3 et p4 capitalisés dans l'ICE_2, ainsi que "p7" appartenant à l'ICE_3.

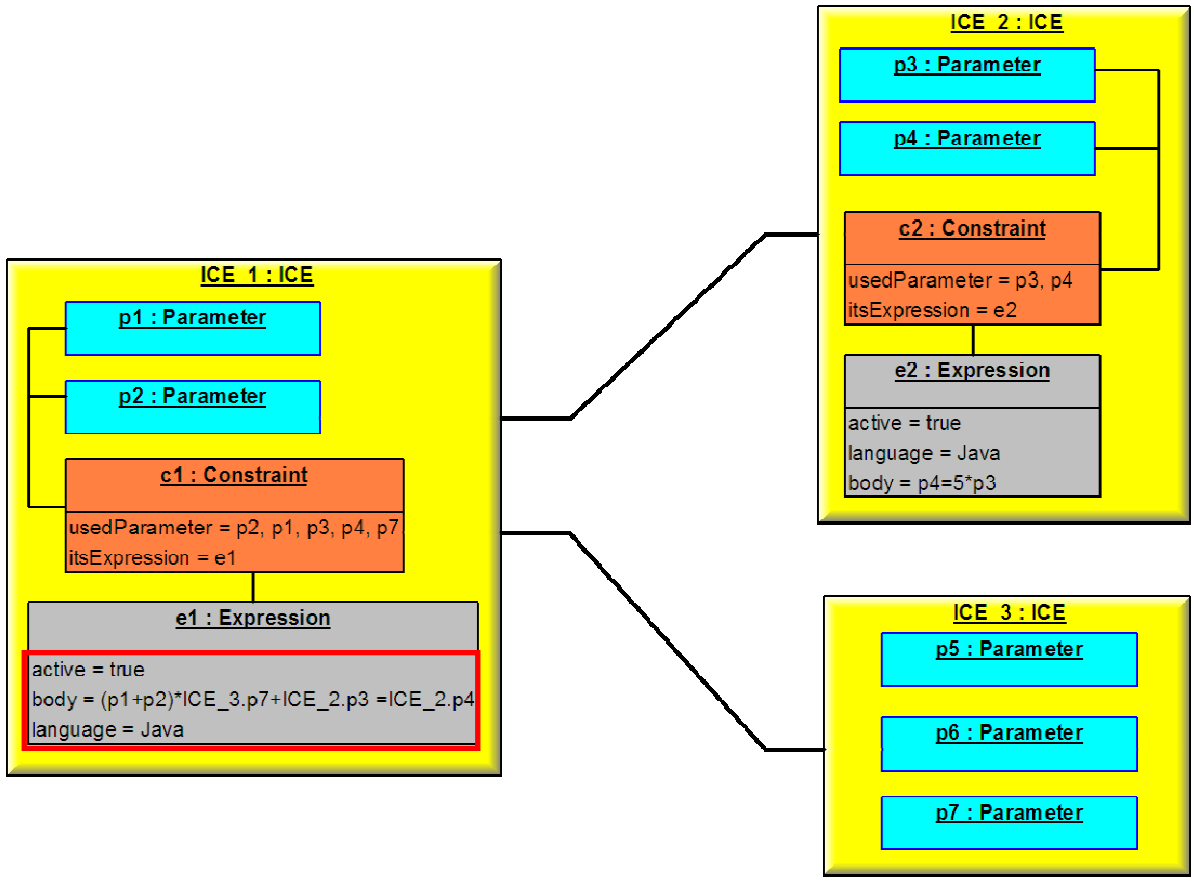


Figure 73 : diagramme de structure illustrant les dépendances entre ICEs

Cette dépendance entre ICEs a un fort impact dans le cadre de la modification de l'une d'elles. L'utilisateur doit pouvoir visualiser les dépendances et les prendre en compte afin de ne pas créer

d'incohérences. Par exemple, si "p4" est supprimé de l'ICE_2, l'utilisateur doit impérativement modifier la contrainte "c1" de l'ICE_1.
Ces dépendances sont également importantes lors de l'instanciation des ICEs dans les configurations. En effet, si l'ICE_2 et l'ICE_3 peuvent être utilisées en totale indépendance, l'ICE_1 pour être utilisée, requiert impérativement l'instanciation des deux autres.

4.3.2.2 Package KnowledgeConfiguration

Ce package (Figure 74) définit la structure des concepts de configurations de connaissances. Les différents types de configurations (squelette, utilisateur, Template) contiennent des ICEInstances. Les configurations permettent de définir une représentation homogène des connaissances, quel que soit l'outil ou le domaine métier. Elles permettent le suivi des modifications des paramètres et des contraintes, ainsi que leur maintien en cohérence.

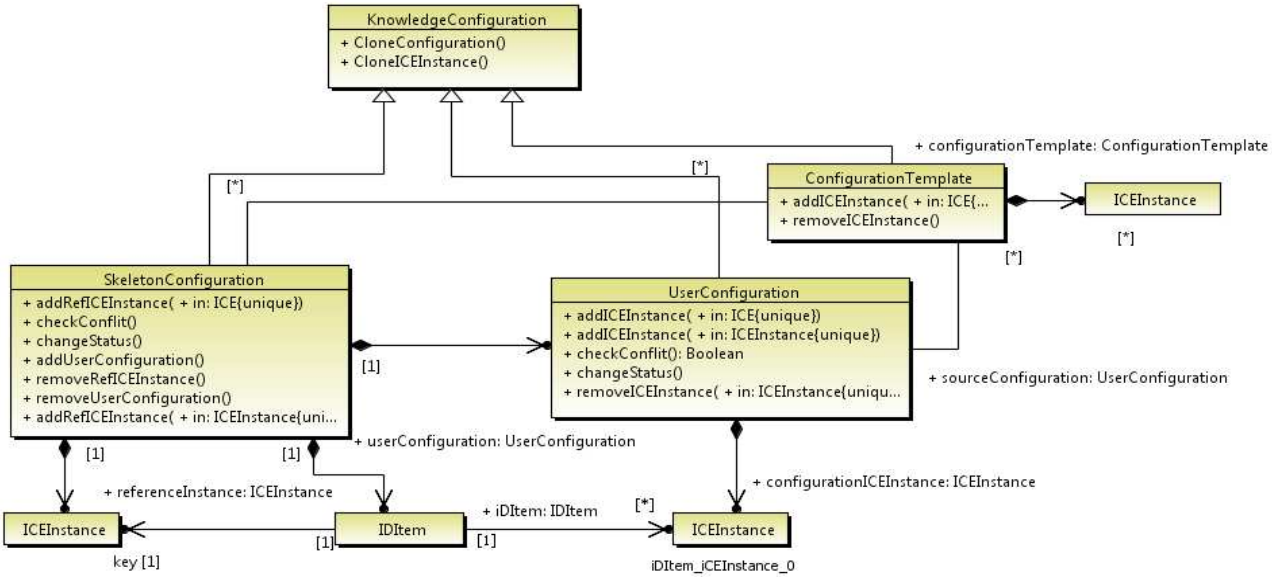


Figure 74 : package KnowledgeConfiguration

La classe KnowledgeConfiguration

KnowledgeConfiguration regroupe un ensemble de connaissances (explicites) structurées pour un objectif donné en relation avec une ou plusieurs activités du processus de conception (cf. section C).

A partir de cette classe abstraite, trois types de configurations sont décrites dans le méta-modèle :

- La classe Skeleton Configuration
- La classe User Configuration
- La classe Configuration Template

La classe SkeletonConfiguration

SkeletonConfiguration assure la gestion et la collaboration de l'ensemble des connaissances utilisées dans plusieurs activités de conception dans un objectif commun. Elle est composée de configurations utilisateur et est représentative de l'ensemble des activités d'un jalon.

Par conséquent, elle contient des ICEInstances de référence en lien avec l'ensemble des ICEInstances contenues dans toutes les configurations utilisateur d'un jalon. Ainsi, elle permet d'assurer la cohérence des connaissances en comparant les ICEInstances **entre** plusieurs UserConfiguration.
Elle est également composée d'ICEInstances de références utiles à la création d'UserConfiguration.
Une SkeletonConfiguration possède un cycle de vie, elle est versionnée au fur à et mesure de l'évolution des UserConfigurations.

La classe UserConfiguration

UserConfiguration est une configuration de connaissances contenant les connaissances utiles pour un contexte de conception donné. Par conséquent, elle peut être considérée comme une représentation des connaissances du contexte en question.

Une UserConfiguration est composée d'ICEInstances, copiées, soit depuis la configuration squelette, soit directement depuis la base d'ICE générique.
Elle permet d'assurer la cohérence des connaissances en comparant les ICEInstances la composant.
Une UserConfiguration possède un cycle de vie, elle est versionnée au fur à et mesure de l'évolution des ICEInstances la composant (avec un suivi des modifications).

La classe ConfigurationTemplate

ConfigurationTemplate est un modèle (au sens canevas) de configuration de connaissances.

La classe ConfigurationTemplate est composée d'ICEInstances. Il est possible d'instancier un Template de configuration pour créer une configuration squelette ou utilisateur. Inversement, il est possible de créer une ConfigurationTemplate à partir d'une configuration squelette, utilisateur, ou à partir de la feuille blanche.
L'ensemble des ConfigurationTemplate est stocké dans une base de configurations de connaissances réutilisables et transverses à plusieurs projets (c.a.d. stockées dans un Knowledge Domain).
Il est possible de créer des Templates de configurations génériques, c'est-à-dire avec des ICEInstances sans valeurs, ou spécifiques, en conservant les valeurs des paramètres.

La classe ICEInstance

La classe ICEInstance correspond à la classe issue du package InformationCoreEntity.
Attention, les ICEInstances contenues dans les trois types de configurations sont différentes. En effet, les ICEInstances composant la configuration squelette sont contenues dans sa partie référence. Ainsi, une ICEInstance de référence peut pointer vers plusieurs ICEInstances dans plusieurs configurations utilisateurs. En revanche, chaque ICEInstance des UserConfiguration pointe forcément vers une unique ICEInstance de référence. Par ce biais, on garantit que la configuration squelette possède bien l'ensemble de connaissances utilisées dans les configurations utilisateurs.

La classe IDITEM

Item d'identifiant de relation d'équivalence entre différentes ICEInstances.

IDItem permet de faire le lien entre les ICEInstances de la configuration squelette et des différentes configurations utilisateurs.
C'est-à-dire que l'item d'identifiant sert d'index de vérification entre ces ICEInstances. Ainsi, les ICEInstances des configurations utilisateur connaissent les ICEInstances de références de la configuration squelette, garantissant une identification et un suivi permanent de l'ensemble des connaissances utilisées.

Exemple de KnowledgeConfiguration :

KCMethod permet à des utilisateurs de créer des configurations utilisateurs synchronisées avec leurs modèles métiers. Ces configurations sont supervisées par des configurations squelettes qui assurent la collaboration et avertissent les concepteurs en cas de conflits sur les paramètres ou les contraintes (cf. chapitre 3 section 3.4.4.2).
Pour illustrer le fonctionnement des configurations de connaissances, prenons cinq ICEs illustrées Figure 75 :

- Géométrie Piston
- Contrainte Géométrie Piston
- Chargement Thermique
- Chargement Pression
- Nombre de Cylindres

Les concepteurs souhaitent les utiliser dans le cadre de deux activités de conception "Pré-Dimensionnement_Piston" et "Pré-Dimensionnement_Carter_Cylindres". Ces activités appartiennent à une même phase du processus de conception appelée "Pré-dimensionnement_Bloc_Moteur".

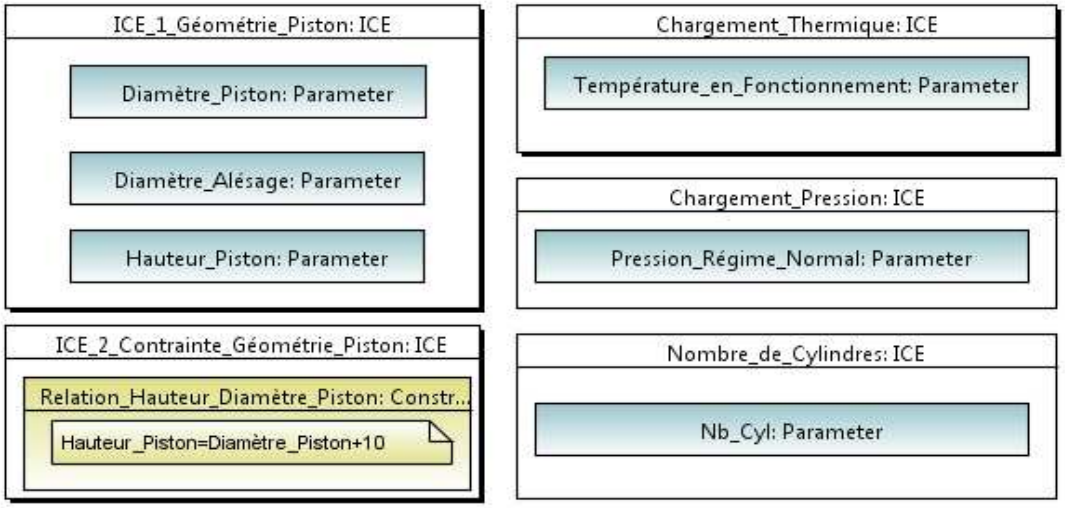


Figure 75 : diagramme de structure représentant plusieurs ICEs

Les ICEInstances sont d'abord instanciées dans une configuration squelette nommée "Pré-Dimensionnement Bloc Moteur", et des valeurs sont attribuées aux paramètres (Figure 76 : cadre correspondant à la SkeletonConfiguration).

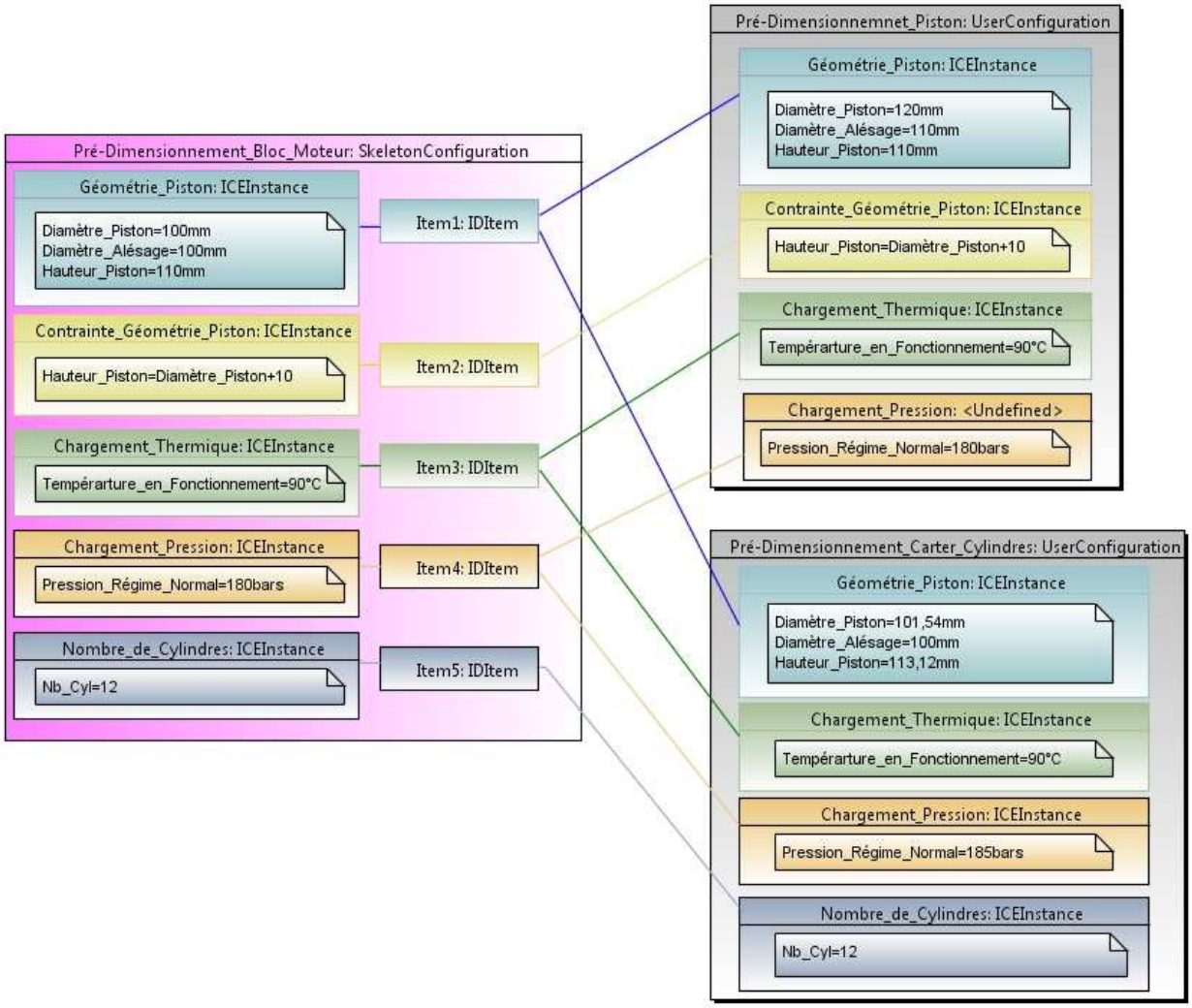


Figure 76 : diagramme de structure représentant l'utilisation d'ICEInstances dans les KnowledgeConfiguration

Les concepteurs peuvent alors créer leurs configurations utilisateur "Pré-Dimensionnement Piston" et "Pré-Dimensionnement Carter Cylindres" en ajoutant des ICEInstances depuis la configuration squelette. Par exemple, l'ICEInstance "Géométrie Piston" est utilisée dans les deux configurations utilisateur avec des valeurs différentes entre les paramètres. Chaque ICEInstance utilisée dans une configuration utilisateur est référencée à l'ICEInstance de la configuration squelette via l'IDItem. L'IDItem garantit la traçabilité de l'ensemble des ICEInstances mais il permet également de comparer toutes les instances lors du processus d'analyse des conflits.

4.3.2.3 Package KnowledgeDomain

Le package KnowledgeDomain décrit la structure de capitalisation de plusieurs concepts KCModel (Figure 77).

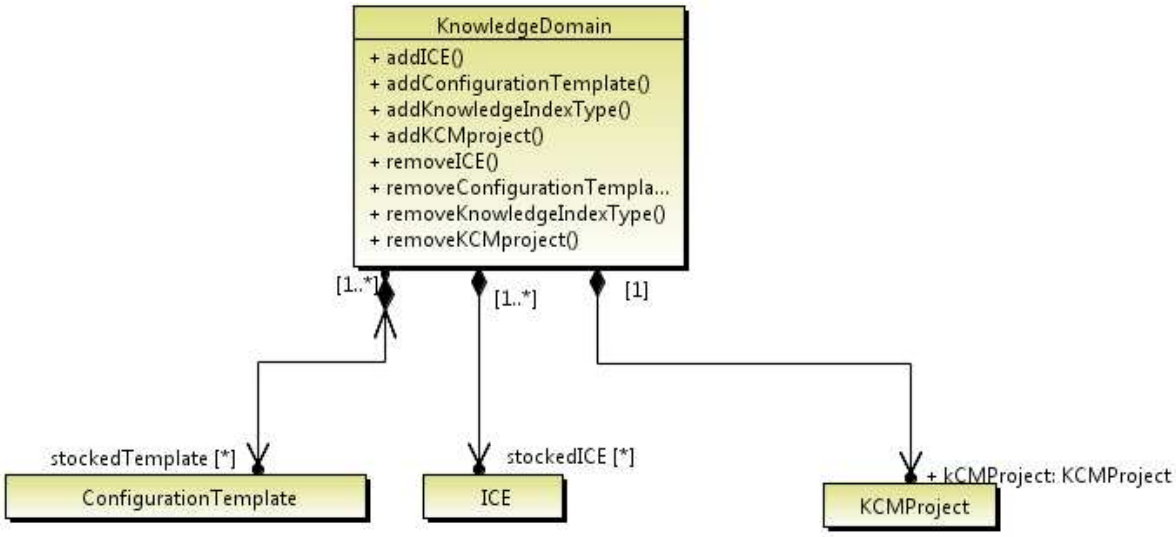


Figure 77 : package KnowledgeDomain

La classe KnowledgeDomain

Le KnowledgeDomain définit un ensemble de connaissances ou d'informations réutilisables et stables dans une certaine durée pour un domaine global.

C'est dans le KnowledgeDomain que l'on stocke la base d'ICEs, la base de Template de configurations et les KnowledgeIndex. C'est également à ce niveau que l'on stocke les différents projets nécessitant des connaissances en relation avec le Knowledge Domain. C'est-à-dire, quand un utilisateur (disposant de droits suffisants) crée un projet, il doit le placer dans un Knowledge Domain qui regroupe un ensemble de connaissances et d'informations dédiées (cf. chapitre 3 section 3.4.8).
Toutes les ICEs et les ConfigurationTemplate créées dans ce KnowledgeDomain restent dans ce KnowledgeDomain. Néanmoins, dans certain cas, il est possible d'associer à plusieurs KnowledgeDomains une ICE ou une ConfigurationTemplate. Ex : pour un Knowledge Domain "Moteur" et un autre "Boite de vitesse" regroupant respectivement les connaissances et les informations nécessaires à la conception de ces organes automobiles. L'ICE contenant le paramètre "régime normal en charge" appartiendra aux deux KnowledgeDomain. C'est-à-dire que l'ICE sera copiée avec lien dans les deux bases d'informations.

4.3.2.4 Le package KCMProject

Le package KCMProject présente la structure d'organisation des configurations de connaissances par rapport à l'organisation d'un projet de conception, c'est-à-dire, l'articulation des activités du processus de conception en lien avec les configurations de connaissances, comme illustré Figure 53.
Le diagramme présente une structure symétrique dans laquelle une configuration squelette correspond au jalon d'un projet. Les configurations squelettes sont elles mêmes composées de configurations utilisateurs correspondant aux activités du jalon. Le tout est stocké dans un projet, lui-même stocké dans un KnowledgeDomain.

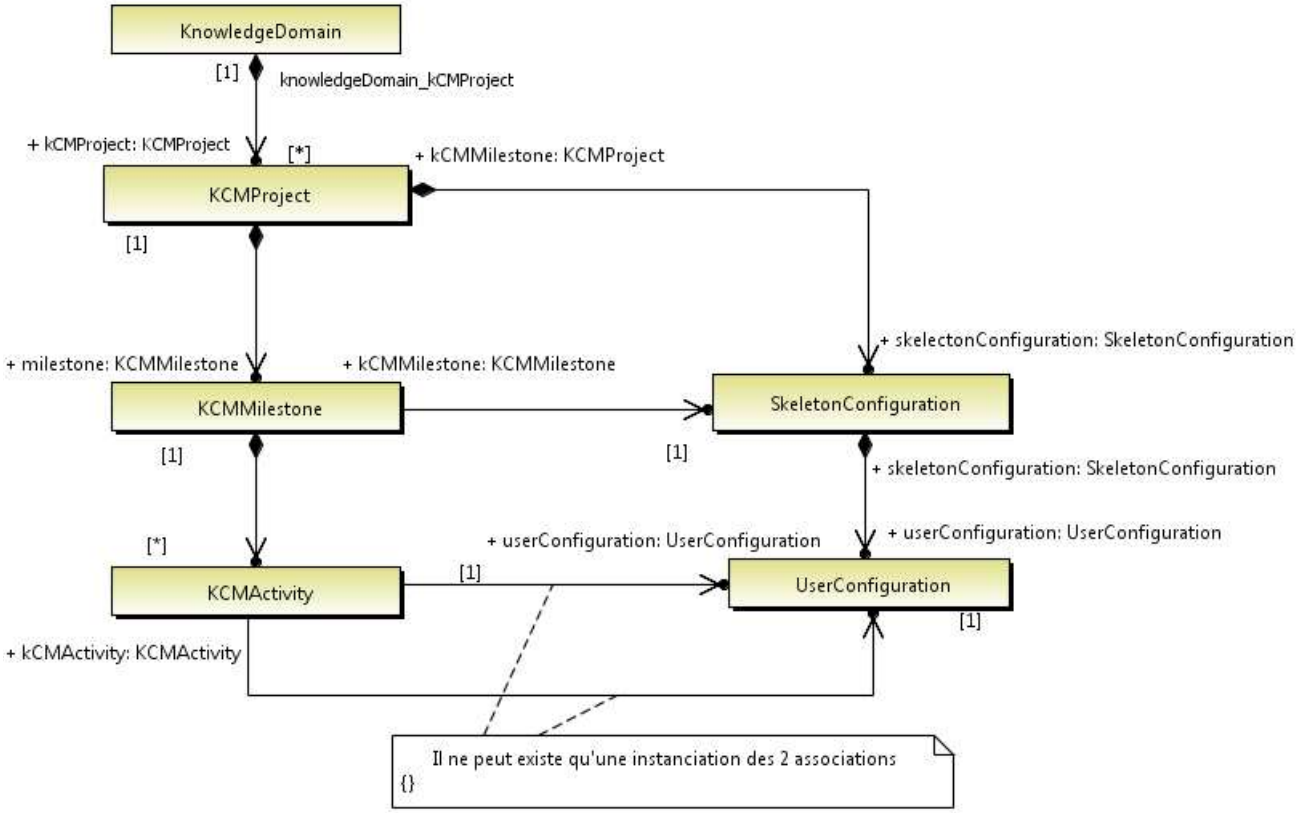


Figure 78 : package KCMProject

La classe KCMProject

KCMProject est la classe correspondant à un projet de conception.

Dans le KCModel, les projets sont stockés dans un KnowledgeDomain.
Un projet stocke les configurations squelettes et utilisateurs créées dans le cadre de ce projet.
Enfin, un projet est composé de jalons (ou phases), eux-mêmes composés d'activités de conception en lien avec les configurations.

La classe KCMMilestone

KCMMilestone est la classe correspondant à un jalon (ou phase) du processus de conception.

KCMMilestone est stocké dans KCMProject. Les configurations squelettes sont liées aux jalons.

La classe KCMActivity

KCMActivity est la classe correspondant à une activité du processus de conception.

Les KCMActivity sont associées aux configurations utilisateurs. Une configuration utilisateur peut être associée à plusieurs activités, OU plusieurs configurations utilisateurs peuvent être associées à une activité. Les deux types de liens en même temps sont impossibles. Dans le meilleur des cas, il faut privilégier **une** configuration pour **une** activité.

Exemple d'organisation des activités du processus de conception selon KCMethod :
Une entreprise fabrique des structures métalliques comme des charpentes et des supports de fixations. Cette entreprise utilise KCMethod et a donc défini plusieurs Knowledge Domain regroupant l'ensemble des connaissances pour chaque famille de produits qu'elle développe.
Dans un Knowledge Domain "Support de fixation" l'entreprise crée un projet "Support Alpha". Ce projet ne comporte qu'une phase appelée, "Réaliser le support", qui est composée de quatre activités :

- Etude marketing
- Conception initiale
- Simulation et optimisation
- Mise en plan (conception finale)

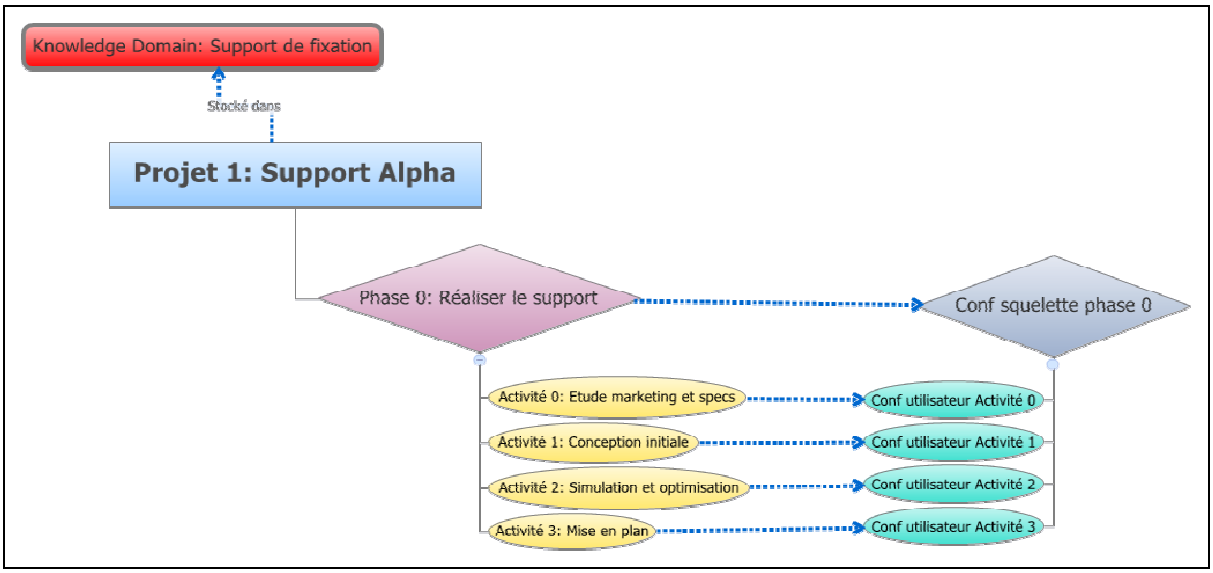


Figure 79 : exemple d'organisation du processus de conception et d'utilisation des configurations de connaissances

Les configurations de connaissances peuvent maintenant être associées à la phase et aux activités. Ainsi, la "Conf squelette phase 0" est associée à la "phase 0 : Réaliser le support" et les Conf utilisateur Activité 0 à 3 sont associées à chaque phases (Figure 79).

4.3.2.5 Le package KCMType

Le package KCMType définit un ensemble de primitives types proposées par défaut dans le KCModel.

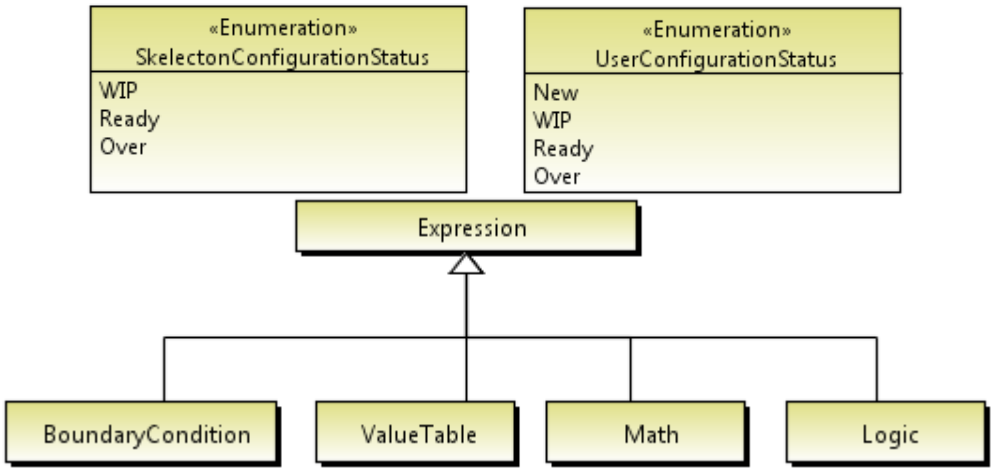


Figure 80 : diagrammes ExpressionType sur les différents types de contraintes et les statuts pris par les configurations de connaissances

Il regroupe plusieurs digrammes décrivant par exemple, les différents statuts pouvant être pris par les configurations de connaissances, les différents types de contraintes (Figure 80), ou encore les différents opérateurs pouvant être utilisés dans l'expression des contraintes.
Certaines de ces primitives types sont proposées en simple utilisation (sans modification), d'autres seront modifiables (paramétrables), ou l'utilisateur pourra en créer de nouvelles.

Les primitives types proposées dans le KCModel sont issues des constats réalisés suite à l'analyse de l'environnement industriel (cf. chapitre 3 section 3.2.1). Ainsi, par défaut, KCModel proposera de "typer" les paramètres et les contraintes selon la typologie établie au chapitre précédent, représentée Figure 80. On retrouve également, sur cette même figure, les différents statuts que nous avons définis dans le cadre du processus de validation des configurations de connaissances (section 3.4.6).

4.3.3 KCMTool

La partie KCMTool contient les packages définissant les concepts ou les fonctionnalités génériques dont nous avons besoin pour mettre en place la méthodologie KCMethod. En effet, si les concepts forts de la méthodologie KCMethod sont exprimés dans KCMCore, KCMTool définit les modèles permettant d'assurer les fonctionnalités ou les services attendus dans KCMethod comme les mécanismes de gestion en configuration, la gestion des droits ou encore la multi-représentation. Ces modèles sont séparés des concepts fondamentaux ICE et KC, dans la mesure où ils constituent une couche d'outillage transverse qui n'a pas de réelle signification métier (car plus orienté service).
Ainsi, les diagrammes de cette section présentent des solutions qui ne sont pas fondamentalement liées aux KCModel et qui peuvent servir à d'autres modèles ou méta-modèles. Nous décrivons plus succinctement ces diagrammes.

Six packages sont définis dans le KCMTool :

- Le package *Representability* qui décrit les mécanismes de multi-représentations d'un élément (ICE, paramètre, contrainte, configuration).
- Le package *ConfigurationManagement* qui décrit les mécanismes globaux de gestion de configuration, plus particulièrement le mécanisme de versionnement.
- Le package *IndexDefinition* qui décrit les mécanismes permettant d'attribuer des KnowledgeIndex pour contextualiser les ICEs et les configurations.
- Le package *AccesControlDefinition* qui définit les mécanismes de gestion des rôles. Ce package n'est pas décrit dans ce document ; il est directement repris des modèles RBAC[11] [Ferraiolo et al., 1999] (Annexe 18).
- Le package *ProjectDomain* qui définit une structure de projet générique.
- Le package *Review* qui définit les mécanismes de création de tâches pour permettre aux acteurs d'un projet de collaborer (collaboration statique).

[11] RBAC : Role Based Access Control

4.3.3.1 Le package Representability

Ce package (Figure 81) définit les mécanismes de multi-représentations d'un élément. Par exemple, il permet d'attribuer plusieurs noms à un paramètre afin qu'il soit compréhensible et utilisable par des utilisateurs possédant des langages métiers différents.

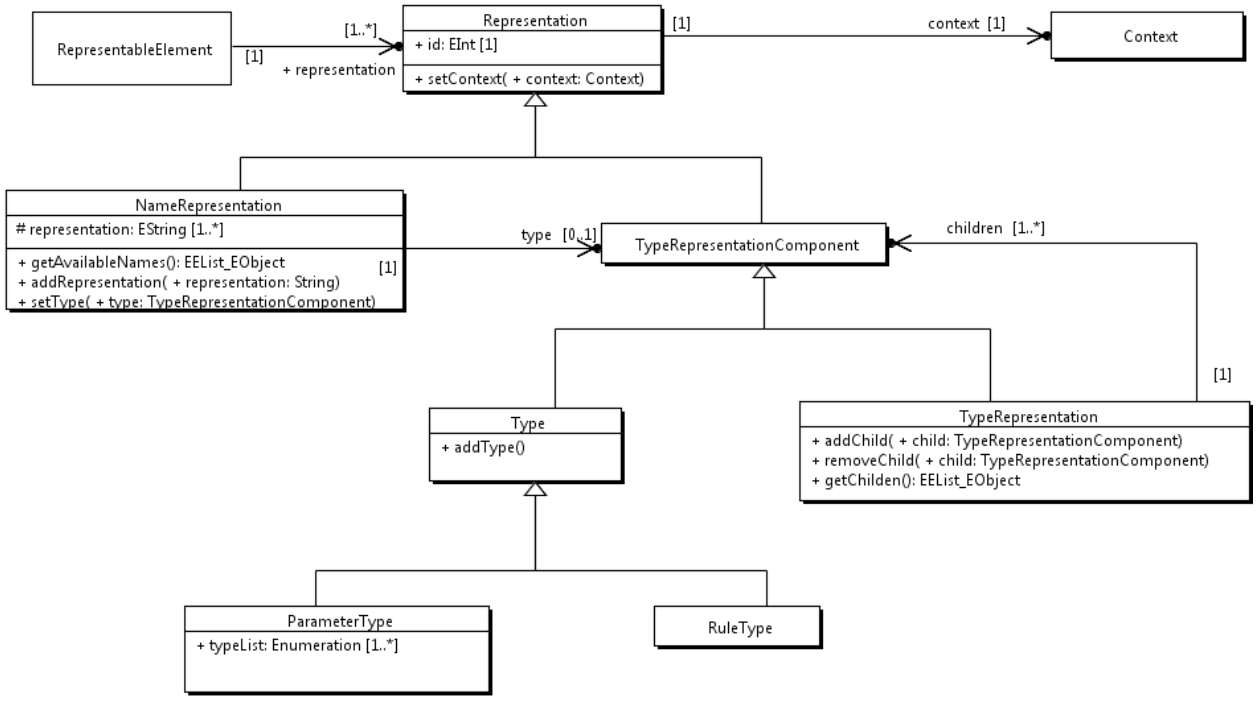


Figure 81 : diagramme illustrant les mécanismes de multi-représentation

La classe RepresentableElement

RepresentableElement caractérise un élément pouvant avoir plusieurs représentations.

Les ICEs, les paramètres, les contraintes sont les principaux éléments du KCModel qui peuvent avoir plusieurs représentations au niveau de leurs noms ou de leur type.

La classe Representation

Representation est la classe principale du modèle qui associe un identifiant unique à un RepresentableElement en fonction d'un contexte.

Deux héritages partent de cette classe permettant de définir la représentation selon un nom et un type.

La classe Contexte

Une représentation est définie à partir d'un contexte qui dépend du langage et de spécialisations.

La classe contexte est décrite via un autre diagramme (Figure 82). Une représentation est exprimée dans un langage donné et au travers de méta-données qui la caractérise. Par exemple, un nom est défini dans une ou plusieurs langues (français, anglais, allemand, etc.) et selon un contexte (qui est un agrégat de spécialisations) de conception comme "Conception bloc moteur".

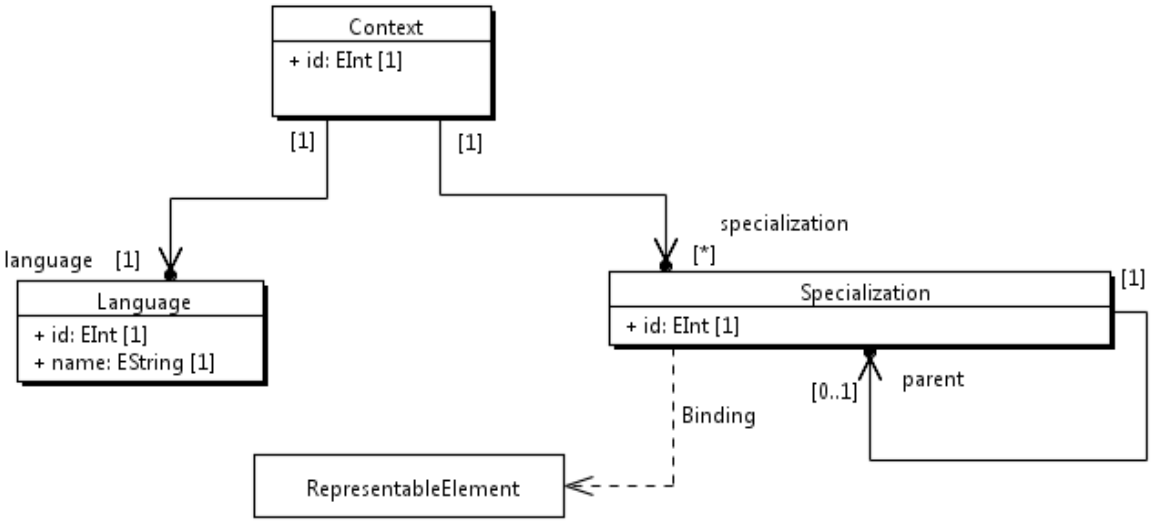


Figure 82 : diagramme de définition du contexte pour la multi-représentation

La classe language

«Language» désigne un système de signes linguistiques, vocaux, graphiques ou gestuels, qui permet la communication entre les individus [définition Larousse].

Par exemple, l'anglais, le français, l'allemand, etc.

La classe Specialization

Contextualisation sémantique décrivant une activité ou une habitude particulière.

La combinaison de plusieurs spécialisations offre une description détaillée et suffisante pour comprendre le rôle d'un acteur dans son métier.

4.3.3.2 Le package ConfigurationManagement

Le package ConfigurationManagement définit les principaux mécanismes de gestion en configuration tels que le versionnement ou la traçabilité. Attention, le terme "configuration" n'est pas lié au terme "configuration de connaissances" de la partie Core. Ici, "configuration" désigne une entité soumise aux mécanismes de gestion de configurations. Par exemple, si un paramètre était géré directement en version, il s'apparenterait à une configuration.

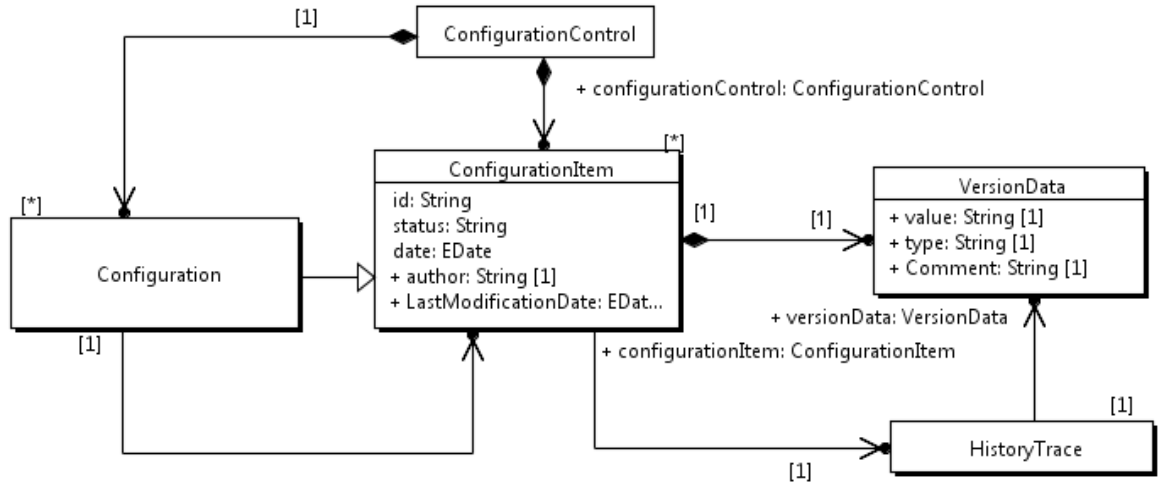


Figure 83 : diagramme de gestion en configuration

La classe Configuration

Ensemble d'items de configurations organisés selon un point de vue spécifique permettant de les réutiliser, les réorganiser ou les modifier afin de satisfaire une finalité ou un objectif.

Une configuration peut être vue comme une configuration de configurations (comme un groupe de configurations) gérées en version et qui sont assignées à un point de vue spécifique.
Dans le cadre du KCModel, la classe configuration peut correspondre à une KnowledgeConfiguration qui contient des ICEInstances (qui sont des Items de configuration).

La classe ConfigurationItem

ConfigurationItem est la plus petite unité gérée et contrôlée par système de gestion de configuration.

Une ConfigurationItem est modifiable et gérée en version. Chaque item de configuration connaît son évolution via l'enregistrement de son historique (Voir HistoryTrace).
Dans le cadre du KCModel, une configuration Item correspond, entre autre, aux ICEs et aux ICEInstances qui sont les plus petits éléments soumis à la gestion en configuration.

La classe ConfigurationControl

ConfigurationControl est la classe qui stocke les configurations.

Par exemple, dans le KCModel, les configurations squelette et utilisateur sont stockées au niveau du projet. Dans ce cas ConfigurationControl est tissée avec la classe KCMProject, ce qui signifie que les configurations sont stockées dans l'objet KCMProjet = projet de conception.
Dans le cas des Templates de configuration, on souhaite quelles soient stockées dans le domaine de connaissance (domaine qui stocke les projets), ConfigurationControl est alors tissée avec la classe KnowledgeDomain.

La classe VersionData

VersionData stocke les données permettant d'identifier une version.

Elle contient les propriétés suivantes :

- value : valeur de version. Ex : numéro de version
- type : type de version. Ex : build (les versions utilisées pendant le développement d'un produit), release (les versions utilisées pour les livraisons commerciales, etc.)

La classe HistoryTrace

Enregistrement de l'historique de versionnement.

Chaque modification de version est sauvegardée avec différents attributs, comme des commentaires ou une justification.

4.3.3.3 Le package IndexDefinition

Le package IndexDefinition permet de définir les Knowledge Index (cf. chapitre 3 section 3.4.3), avec pour objectif de donner un contexte et de la sémantique à un objet.

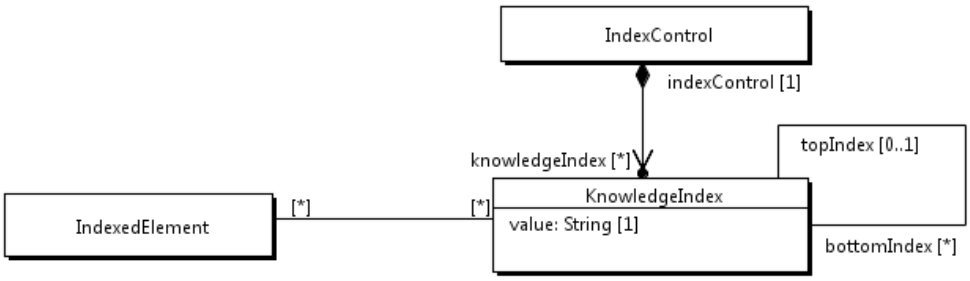


Figure 84 : diagramme représentant le package IndexDefinition

La classe KnowledgeIndex

KnowledgeIndex permet de créer des méta-données et des données organisées dans le but de définir un contexte et de donner de la sémantique à un objet.

Les KnowledgeIndex peuvent être définis selon une structure arborescente ou sous forme de graphe. Cette classe peut être assimilée à la classe Specialization du modèle de représentation. En effet, dans les deux diagrammes, les données utilisées sont similaires, ce sont les fonctionnalités qui diffèrent (Figure 85). Ainsi, en terme d'implémentation, la base de données est créée dans le KnowledgeDomain, la table d'utilisation des données est créée dans la Specialization, enfin, la classe KnowledgeIndex vient utiliser cette table mais avec des fonctionnalités différentes (car c'est la classe la plus riche en fonctionnalités, elle utilise donc toutes celles de la classe Specialization plus d'autres fonctionnalités spécifiques).

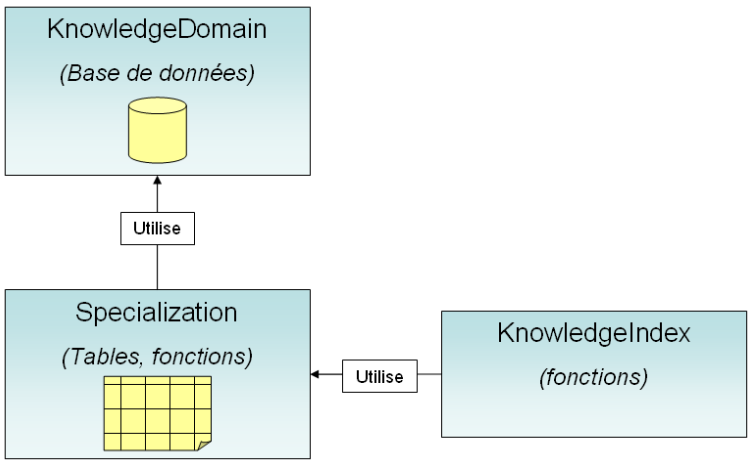


Figure 85 : lien entre la Specialization et le KnowledgeIndex lors de l'implémentation

La classe IndexedElement

IndexedElement caractérise un élément pouvant se voir attribuer des KnowledgeIndex.

Dans le cadre du KCModel, IndexedElement correspond à une ICE, ou une configuration de connaissances. Ce sont en effet les deux éléments qui peuvent être indexés.

La classe IndexControl

IndexControl est la classe qui stocke les KnowledgeIndex.

Dans le cadre du KCModel, IndexControl est tissé avec le KnowledgeDomain, c'est-à-dire que les KnowledgeIndex sont stockés au niveau du KnowledgeDomain, et non d'un projet. Néanmoins, les KnowledgeIndex peuvent être stockés ailleurs, par exemple dans des bases de données spécifiques, comme dans l'exemple suivant.

Exemple d'utilisation du Knowledge Index :
La Figure 86 illustre l'indexation d'ICEs via l'utilisation du Knowledge Index. Dans ce contexte, les ICEs "Géométrie Tourillon", "Chargement Polaire", et "Course Piston" sont stockées dans un KnowledgeDomain "GMP". Les Knowledge Index sont stockés dans un IndexControl nommé "domainIndexControl" qui est une base de données spécifique des KnowledgeIndex. Dans le "domainIndexControl", on observe trois structures arborescentes qui sont en réalité trois Knowledge Index différents. Ils correspondent à des contextes métiers, des éléments de l'organisation de l'entreprise, ou tout autre élément nécessaire permettant d'indexer des ICEs ou des configurations, afin de les réutiliser par la suite.

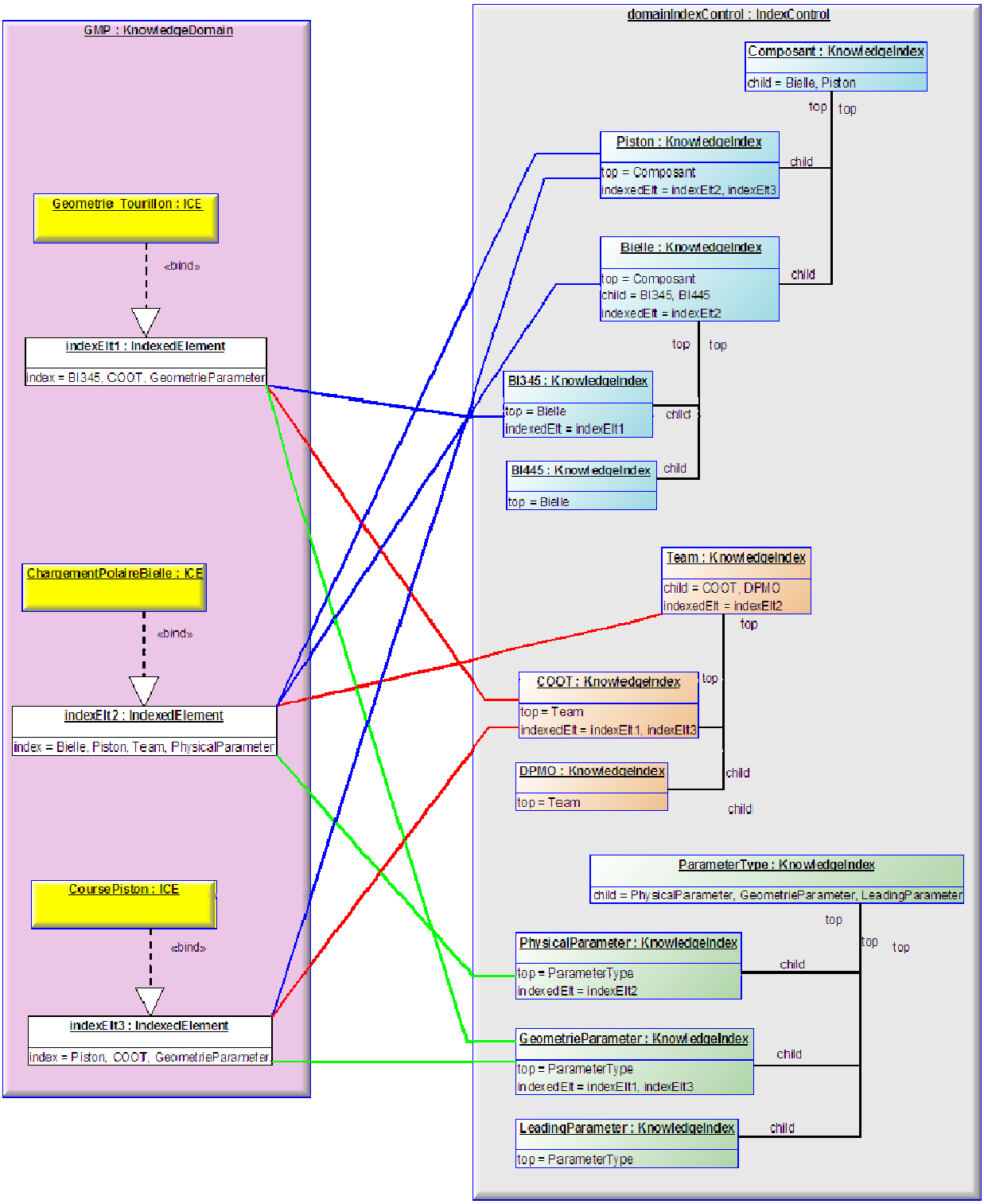


Figure 86 : diagramme de structure illustrant l'utilisation du Knowledge Index

Lors de la création d'une ICE, l'utilisateur lui affecte les différents Knowledge Index permettant de décrire de façon détaillée et suffisante ses contextes d'utilisations. Par exemple, l'ICE "Géométrie Tourillon" est indexée par le Knowledge Index composant/Bielle/BI345, mais aussi Team/DPMO et enfin ParameterType/GeometricParameter. Les Knowledge Index sont affectés aux ICEs à la manière d'un "Tag".

Ainsi, quand un utilisateur effectue une recherche sur l'un de ces Knowledge Index (recherche par filtre), il retrouve l'ICE "Géométrie Tourillon". De cette façon les utilisateurs peuvent faire plusieurs recherches avec des critères différents, en obtenant comme résultat une liste d'ICE correspondante.

Ensuite, ils sélectionnent les ICEs qui les intéressent particulièrement et qu'ils souhaitent utiliser dans une configuration. Ainsi, au fur et à mesure des recherches, les utilisateurs sélectionnent des ICEs et créent un panier d'ICEs à instancier dans une configuration.

4.3.3.4 Le package ProjectDomain

Ce package définit la structure d'un projet dans le cadre du KCModel. Un KCMProject est composé de ProjectElements (jalon, activités). Une structure de projet type peut être sauvegardée dans la classe ProjectTemplate permettant de gagner du temps lors de la mise en place de projets futurs (Figure 87).

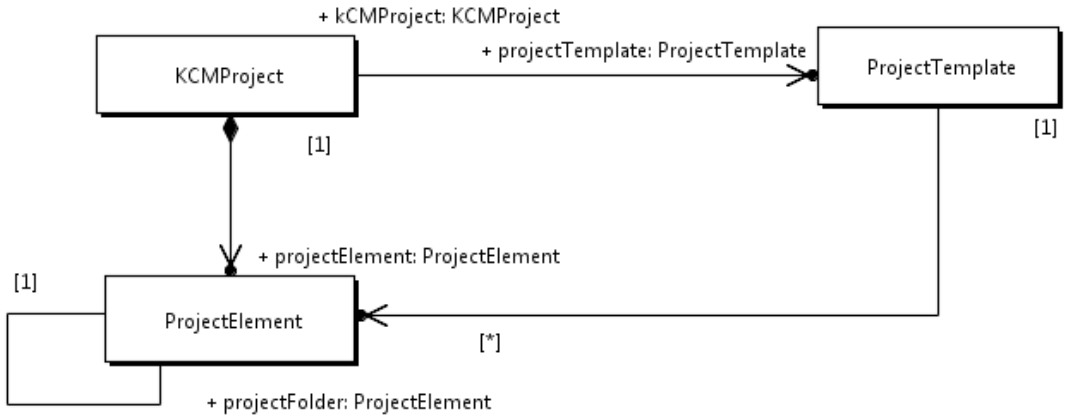


Figure 87 : digramme illustrant le ProjectDomain

4.3.3.5 Le package Review

Ce package définit un principe de collaboration statique dans le KCModel (Figure 88). Une Review est une tâche particulière assignée à un utilisateur dans un objectif donné, par exemple, résoudre un conflit dans une configuration, ou bien, demander l'ajout d'un paramètre dans une ICE générique.
Une Review est donc un item de configuration dans la mesure où elle va pouvoir avoir plusieurs versions et disposer d'un suivi de modifications. Une Review est composée d'une Target (cible) qui définit l'origine de son existence et son objectif.
Les classes Evolution (demander une évolution d'un objet), Mistake (déclarer un problème), Create (créer un nouvel objet), ToRealise (réaliser une tâche), sont des types de Review par défaut, définies dans le KCModel.

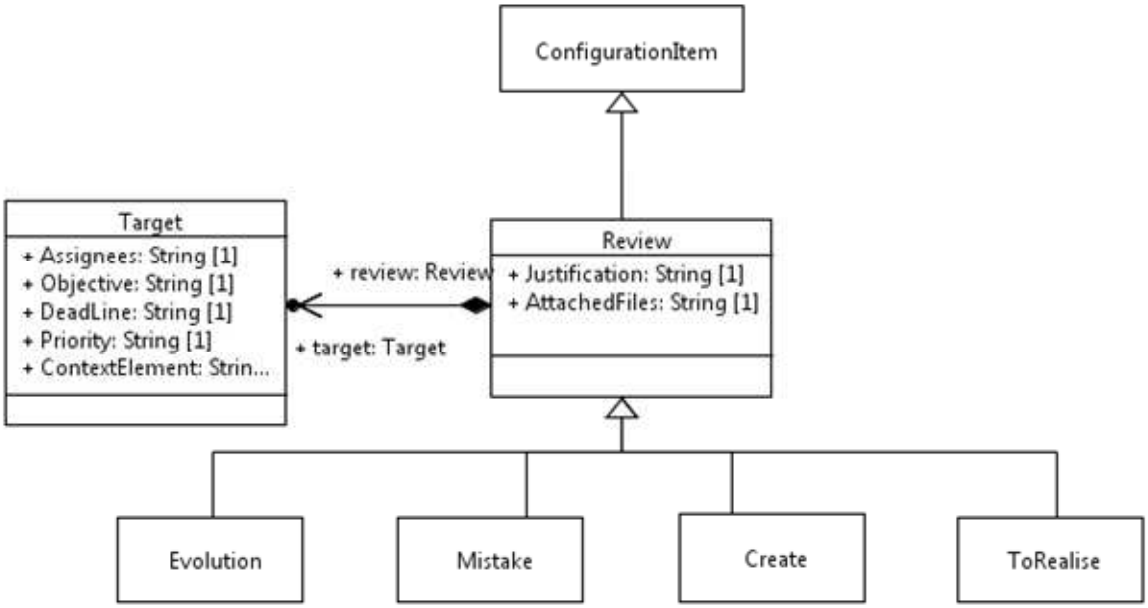


Figure 88 : diagramme illustrant les Review

Exemple de Review :

KCMethod permet à un utilisateur de créer une Review, c'est-à-dire une tâche à réaliser dans le cadre d'un objectif donné (cf. chapitre 3 section 3.4.9.1). Par exemple, un concepteur demande à un expert de modifier une ICE générique pour ajouter une contrainte de type condition limite sur un paramètre.

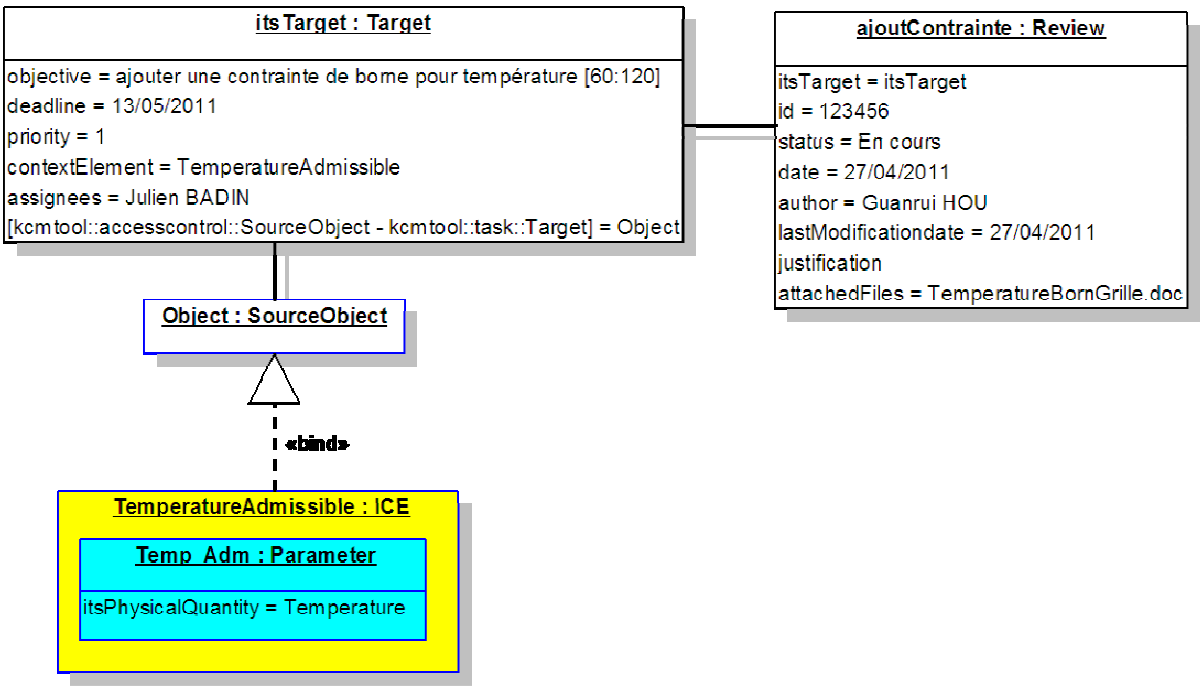


Figure 89 : exemple de Review

La Figure 89 illustre la Review crée par ce concepteur "Guanrui HOU" qui demande à l'expert "Julien BADIN" d'ajouter des bornes [60 ; 120] pour le paramètre "Temp_Adm" contenu dans l'ICE "Température admissible". Une Review est un élément géré en configuration, elle possède une version, un statut et une date finale pour laquelle la tâche doit être terminée.

4.3.4 KCMImplementation

La partie KCMImplementation permet d'illustrer le fonctionnement du KCModel.

Dans un premier temps, on montre les liens existants entre le KCMCore et le KCMTool, de manière à obtenir les fonctionnalités désirées. Ces liens sont symbolisés par des relations de "binding" (ou tissage), c'est-à-dire que l'on associe les propriétés d'une classe à une autre. Par exemple, on souhaite qu'une ICE, issue du KCMCore, soit gérée en configuration. Or, les mécanismes de gestion de configuration sont assurés par le package ConfigurationManagement du KCMTool, et plus particulièrement les classes ItemConfiguration et Configuration. Alors, on créer une relation de type "bind" de l'ICE vers ItemConfiguration, de sorte à pouvoir lui appliquer les mécanismes de gestion de configuration.
Cette méthode permet de créer des modèles très indépendants, correspondants à des aspects conceptuels ou des préoccupations particulières. Ainsi, on utilise ensuite les relations de binding pour les associer et définir un tout cohérent pour un but donné ; ici, la gestion des configurations de connaissances.

Ensuite, des diagrammes d'activités permettent d'illustrer certains processus caractéristiques du KCModel, comme l'instanciation d'ICEInstance dans une configuration de connaissances, le processus de validation de la configuration squelette par rapport aux configurations utilisateurs, etc. Ces diagrammes sont disponibles en annexes (Annexe 19-23).

4.3.4.1 Le package RepresentabilityBinding

Ce package illustre les différents types d'éléments du KCModel qui sont multi-représentables ; c'est-à-dire qu'ils correspondent à la classe RepresentableElement.

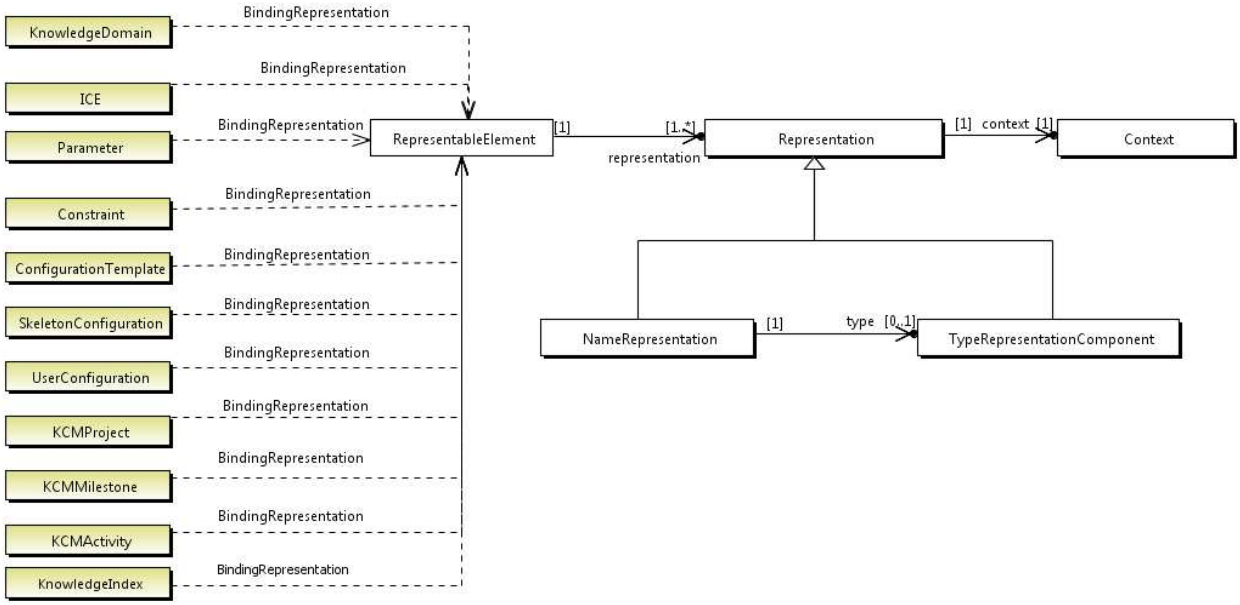


Figure 90 : diagramme représentant le RepresentabilityBinding

Par exemple, la classe Parameter issue du KCMCore, est liée à la classe RepresentableElement du KCMTool par le lien BindingRepresentation. Cela signifie que le paramètre pourra bénéficier des mécanismes de multi-représentations. Ainsi, toutes les classes liées à RepresentableElement, sont multi-représentables en noms et en types quand c'est possible (Figure 90).

4.3.4.2 Le package ConfigurationBinding et ProjectBinding

Ce package illustre les différents types d'éléments du KCModel qui sont soumis aux mécanismes de gestion en configuration (Figure 91).

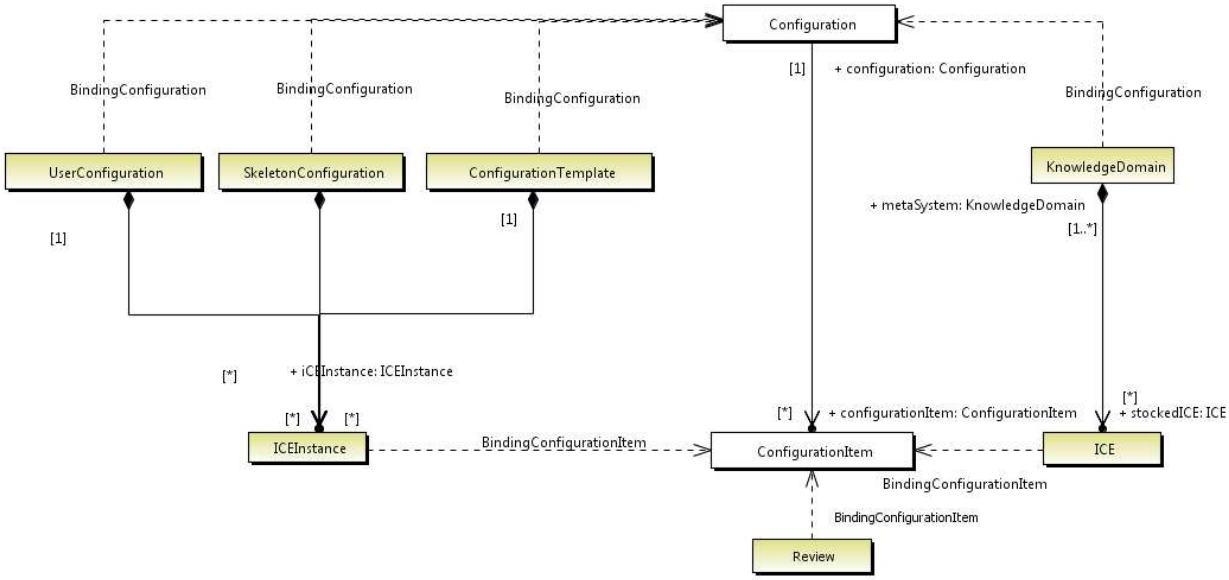


Figure 91 : diagramme ConfigurationBinding

Les ICEs, les ICEInstances, ainsi que les Reviews, sont les plus petites entités gérées en configurations et sont donc tissées avec la classe ConfigurationItem.
Les UserConfiguration, les SkeletonConfiguration, les ConfigurationTemplate, ainsi que les KnowledgeDomain, sont tissées avec la classe Configuration dans la mesure où elles sont composées d'un ensemble de ConfigurationItem.

Dans la continuité des ConfigurationBinding, le diagramme suivant (Figure 92) illustre comment les classes relatives à la structure d'un projet dans le KCModel sont également soumises aux mécanismes de gestion de configuration.

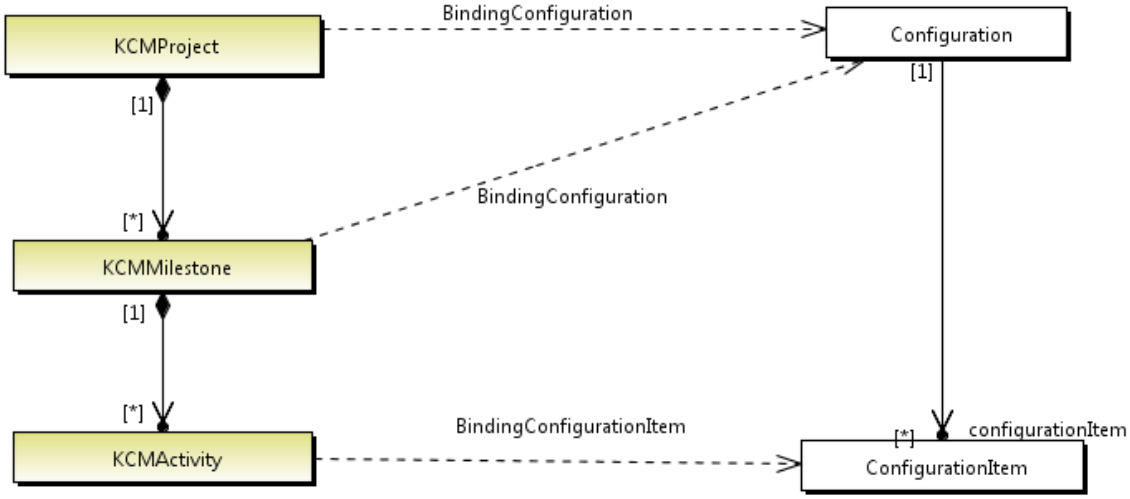


Figure 92 : diagramme ProjectBinding

Ainsi, selon les mêmes principes, KCMActivity est tissée avec la classe ConfigurationItem, et KCMProject et KCMMilestone sont tissées avec la classe Configuration.

4.3.4.3 Le package KnowledgeIndexBinding

Ce package illustre les différents types d'éléments du KCModel qui peuvent être indexés par des KnowledgeIndex (Figure 93).

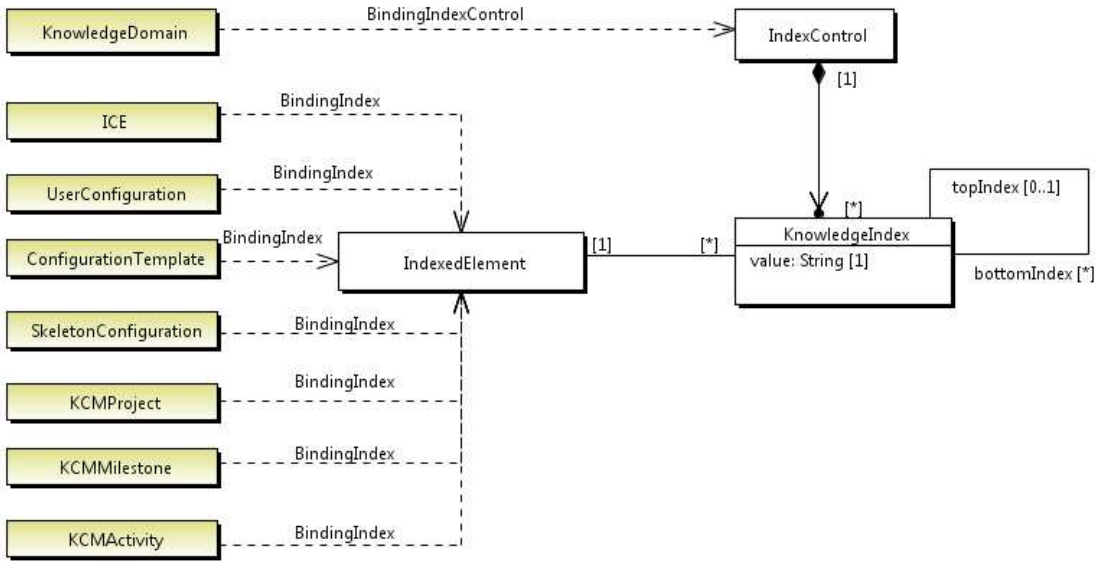


Figure 93 : diagramme KnowledgeIndexBinding

Toutes les classes du KCMCore qui sont tissées avec l'IndexedElement et peuvent posséder plusieurs KnowledgeIndex. Les Knowledge Index sont stockés dans l'IndexControl. Dans la majorité des cas, lors de l'utilisation du KCModel, IndexControl sera tissé avec le KnowledgeDomain.

4.3.5 Exemple d'application du KCModel sur un GMP – Groupe Moto-Propulseur

Dans cette section, nous proposons un exemple simple d'utilisation du KCModel reprenant les principaux éléments de KCMethod énoncés chapitre 3. Cet exemple constitue une implémentation du KCModel pour le pré-dimensionnements de composants moteur par un motoriste automobile : "l'Entreprise MotorX".

Premièrement, supposons que dans un Knowledge Domain "GMP", il existe cinq ICEs génériques préalablement créées par des experts (Figure 94).

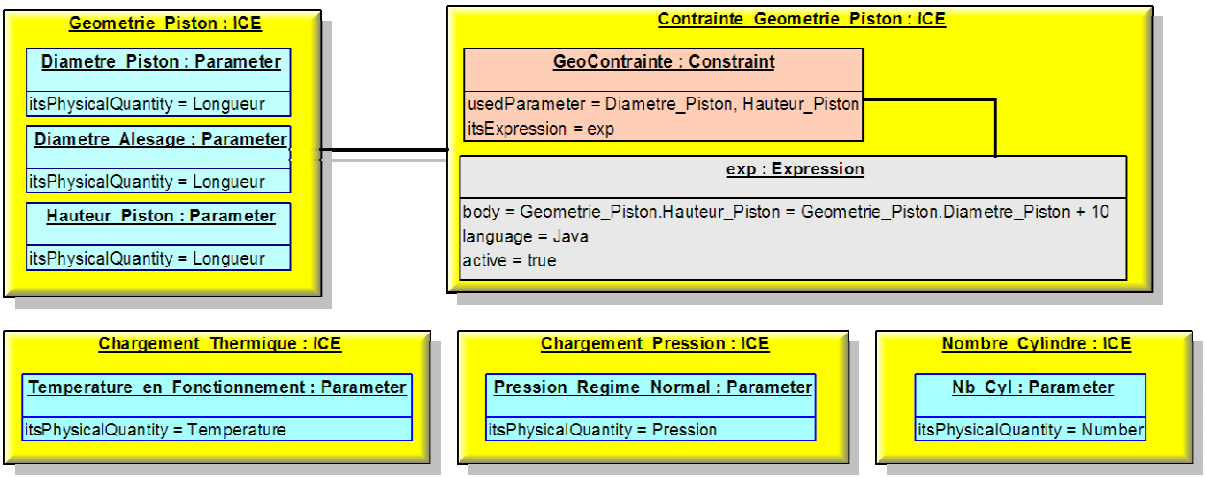


Figure 94 : les cinq ICEs existantes dans le Knowledge Domain GMP

L'ICE "Géométrie Piston" contient trois paramètres, dont deux sont utilisés dans une contrainte capitalisée dans l'ICE "Contrainte Géométrie Piston". Ces deux ICEs sont donc liées par une relation de dépendance de façon similaire à notre exemple Figure 73. Les trois autres ICEs contiennent chacune un seul paramètre et ne sont pas liées à d'autres ICEs.
Ces cinq ICEs constituent la base d'information technique pour l'ensemble des GMP conçus par l'entreprise MotorX.

MotorX veut concevoir un nouveau moteur et créer un projet "Moteur Alpha" dans le Knowledge Domain "GMP". Dans ce projet, nous ne nous intéressons qu'à la première phase qui concerne le pré-dimensionnement du piston et du carter cylindres, qui sont deux activités de conception-simulation.
Dans ce contexte, le responsable de cette phase, créé une configuration squelette en instanciant deux ICEInstances depuis la base d'informations technique, et value les paramètres (Figure 95).

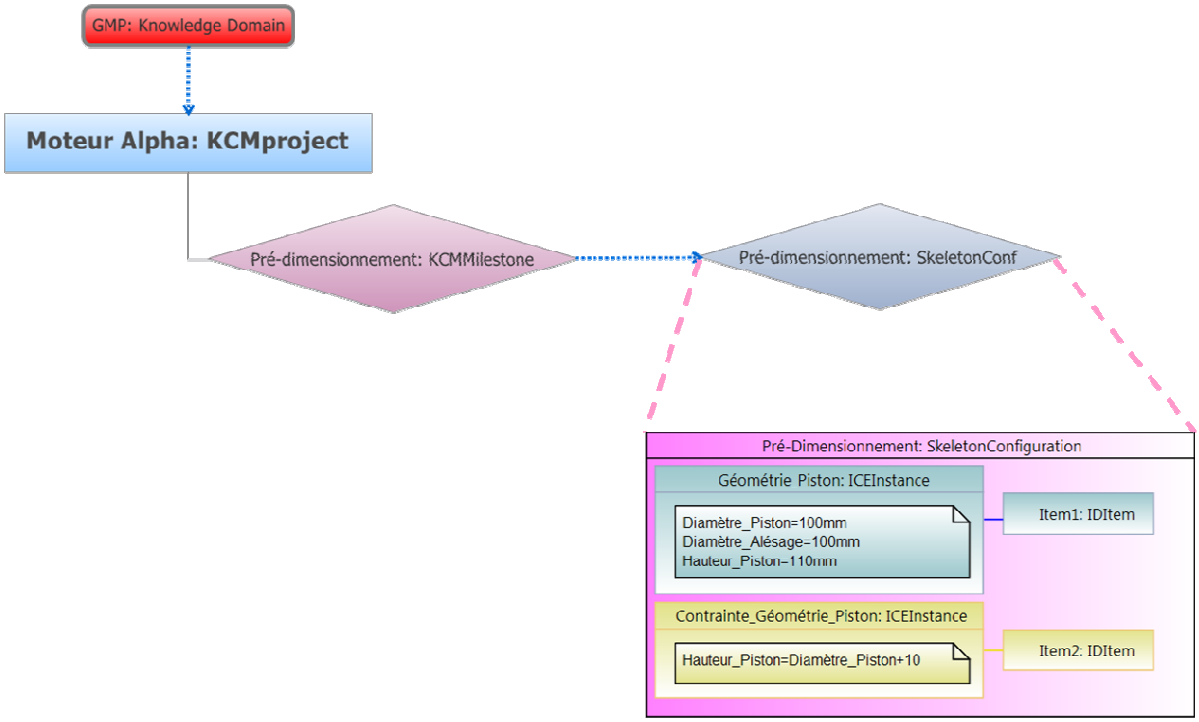


Figure 95 : création de la configuration squelette associée à la phase du projet

Une fois que la phase et la configuration squelette existent, le premier concepteur crée son activité "Etude Piston" et sa configuration utilisateur associée (Figure 96).

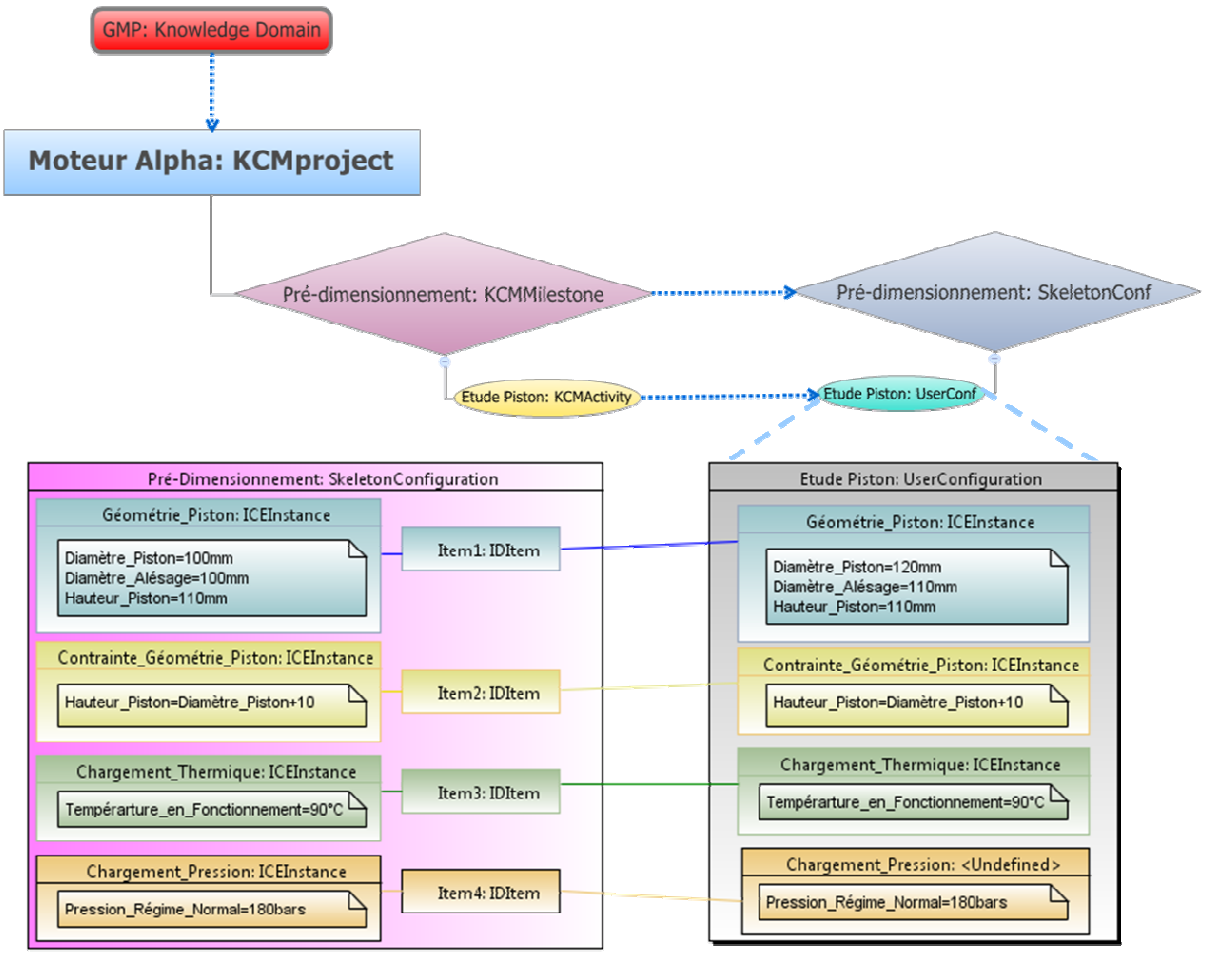


Figure 96 : création de la première configuration utilisateur

Pour créer cette configuration utilisateur, il sélectionne les ICEInstances de référence, préparées dans la configuration squelette par le responsable de la phase. Néanmoins, il à besoin de plus d'ICEs que celles initialement préparées, il va alors directement dans la base d'informations chercher celles dont il à besoin et les instancie dans sa configuration utilisateur. Les nouvelles ICEInstances ajoutées par cet utilisateur s'ajoutent également automatiquement dans la configuration squelette. Les ICEInstances de chaque configuration sont liées par les IDItem afin d'assurer la traçabilité (mais elles sont différentes). Le concepteur peut maintenant travailler sur son modèle piston et donc modifier les paramètres contenus dans sa configuration utilisateur.
Les utilisateurs peuvent créer plusieurs versions des configurations au fur et à mesure de l'évolution de leur travail, mais pour ne pas complexifier l'exemple, nous ne tiendrons pas compte du versionnement.

Le deuxième concepteur crée à son tour son activité "Etude carter cylindres" et la configuration utilisateur associée. Comme le premier concepteur, il réutilise les instances de la configuration squelette, plus une supplémentaire qu'il va chercher dans la base d'informations. Il travaille sur son modèle métier carter cylindres et modifie les valeurs des paramètres de sa configuration utilisateur. Une fois que les concepteurs sont satisfaits d'une version de leur configuration utilisateur, ils la publient vers la configuration squelette afin d'analyser les conflits (Figure 97). Cette figure permet de mettre en évidence l'existence de trois conflits.

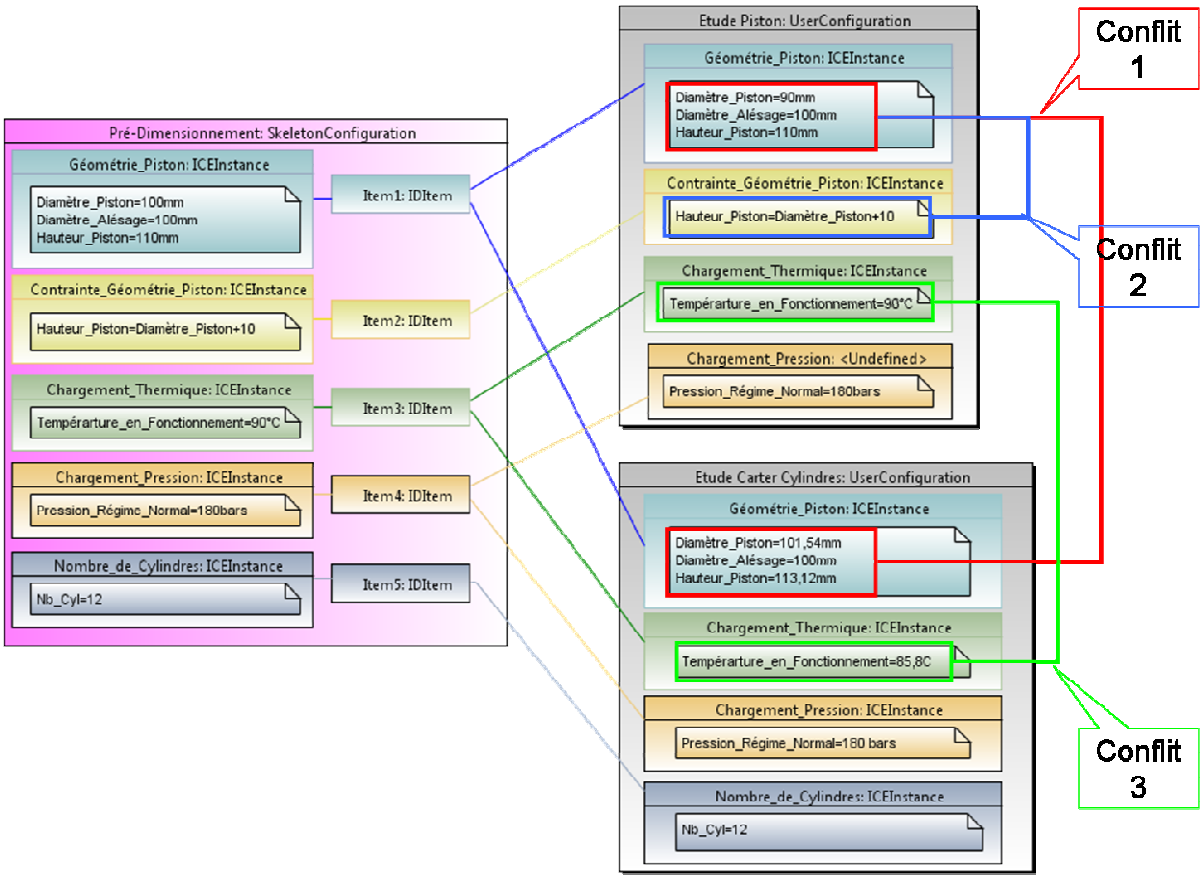


Figure 97 : conflits entre les configurations utilisateur

Le premier et le troisième conflit mettent en évidence des différences de valeurs entre les paramètres partagés dans les deux configurations utilisateur.
Le second conflit met en évidence le non-respect d'une contrainte dans la première configuration utilisateur. En effet, la valeur des paramètres "Hauteur Piston" et "Diamètre Piston" ne permet pas de vérifier que "hauteur piston = diamètre piston + 10".

La comparaison des ICEInstances entre les configurations utilisateur est rendue possible grâce aux IDItem qui permettent de constamment garder la traçabilité.
A ce stade, les utilisateurs peuvent visualiser ces incohérences et effectuer des modifications, de sorte à converger vers un compromis.

Au travers de cet exemple, nous mettons en évidence un type d'utilisation du KCModel correspondant à notre méthodologie KCMethod. Si ce simple scénario ne permet pas d'aborder l'ensemble des concepts, il illustre les principaux mécanismes d'utilisation des ICEs et des configurations de connaissances. Par ailleurs, il permet de constater l'intérêt de cette approche pour la traçabilité des connaissances et la mise en évidence des incohérences dans les activités du processus de conception, ainsi que lors de la collaboration entre acteurs dans d'un projet.
Dans le chapitre suivant, nous présentons des expérimentations plus complexes au travers de l'utilisation de démonstrateurs spécifiquement développés pour éprouver notre approche de gestion des configurations de connaissances.

4.4 Positionnement du KCModel par rapport aux modèles produits

Après avoir détaillé la méthodologie et le méta-modèle, nous proposons un positionnement de notre approche par rapport aux travaux de recherche issus de la littérature scientifique, ou d'outils existants. Ce positionnement s'appuie en partie sur les observations et les analyses réalisées lors des deux premiers chapitres de ce manuscrit.
Le spectre couvert par les modèles produits traditionnels est large puisqu'ils sont capables de considérer aussi bien les informations liées à l'organisation, aux processus (à l'image de l'approche PPO - Produit Process Organisation), au produit (structure des composants du produit), aux fonctionnalités et exigences, etc. De cette façon, ils offrent un support efficace aux applications de gestion de données et d'informations, telles que les PDM et les SDM, en permettant une meilleure organisation du travail des acteurs d'un projet et en limitant la dépendance aux outils et aux modèles métiers.
Par exemple, sans utiliser un modèle produit ou un système de gestion de données, il est courant que le début de la modélisation (généralement géométrique) soit la base de la structuration d'un projet et du premier découpage des composants d'un produit. Cela signifie que ces modèles métiers sont porteurs d'informations à partir desquelles ils sont créés, et que ces informations ne sont présentes nulle part ailleurs en dehors de ces modèles.
L'utilisation de modèles produits permet de capitaliser ces informations et de les mettre à disposition des concepteurs avant même qu'aucun modèle métier n'existe. De cette façon, un projet peut être correctement structuré en fonction de plusieurs points de vue. C'est ensuite à partir de ces informations que les activités de conception débutent et que les modèles métiers sont réalisés.
Dans ce contexte, les modèles produits de la littérature suivent une approche plutôt de type top down, c'est à dire que l'organisation des données, information et des connaissances s'effectue par rapport à une décomposition généralement structurelle, fonctionnelle et géométrique, définie à l'avance. L'expérimentation du modèle CPM - Core Product Model, sur un planétaire de boite de vitesse, illustre parfaitement cette approche [Rachuri et al., 2003]. Elle propose, dans un premier temps, un découpage structurel du système mécanique "planétaire", puis un découpage fonctionnel basé sur le premier découpage. Enfin, elle associe les données et les informations telles que les paramètres, les tolérances, etc. Au travers de cette expérimentation, on constate que ces données sont organisées spécifiquement pour la conception du planétaire, les rendant difficilement réutilisables lors d'autres contextes ou projets. De cette manière, les connaissances sont contraintes par cette structuration et il est difficile de leur donner une organisation propre et générer à la volée de nouvelles vues métiers.

Ainsi, KCModel est un modèle de connaissances du produit qui permet une gestion fine et dynamique des paramètres et des contraintes pour favoriser la collaboration et l'interopérabilité entre les modèles métiers utilisés dans les activités du processus de conception.
Nous proposons de suivre une approche de type bottom up dans laquelle les connaissances possèdent une structuration spécifique, indépendamment de toute autre considération extérieure (cf. chapitre 3 section 3.4.2.5), les rendant plus facilement réutilisables et accessibles. Une fois capitalisées, les connaissances sont ensuite contextualisées et spécifiées pour être utilisées dans un cadre particulier. Il est ainsi possible de suivre pas à pas l'utilisation des paramètres et des contraintes dans les activités du processus de conception, de suivre les modifications de valeurs ou de règles, et vérifier avec assurance la cohérence des connaissances manipulées. La notion de cohérence est cruciale et la contribution de nos travaux se situe essentiellement dans l'utilisation de concepts déjà existants (objets de connaissances, gestion de configuration), de manière à proposer une solution originale. Basée sur la collaboration en temps réel des acteurs du processus de conception et de la cohérence des connaissances, KCModel permet de comparer les différentes vues métiers en tenant compte de plusieurs niveaux de représentations des connaissances, de manière à aider les concepteurs dans leurs choix et limiter les erreurs. De plus, KCModel, de part sa structuration, permet une réutilisation transversale des connaissances et n'est pas limité à un seul projet, ce qui est souvent le cas des modèles produits plus classiques [Badin et al., 2010].

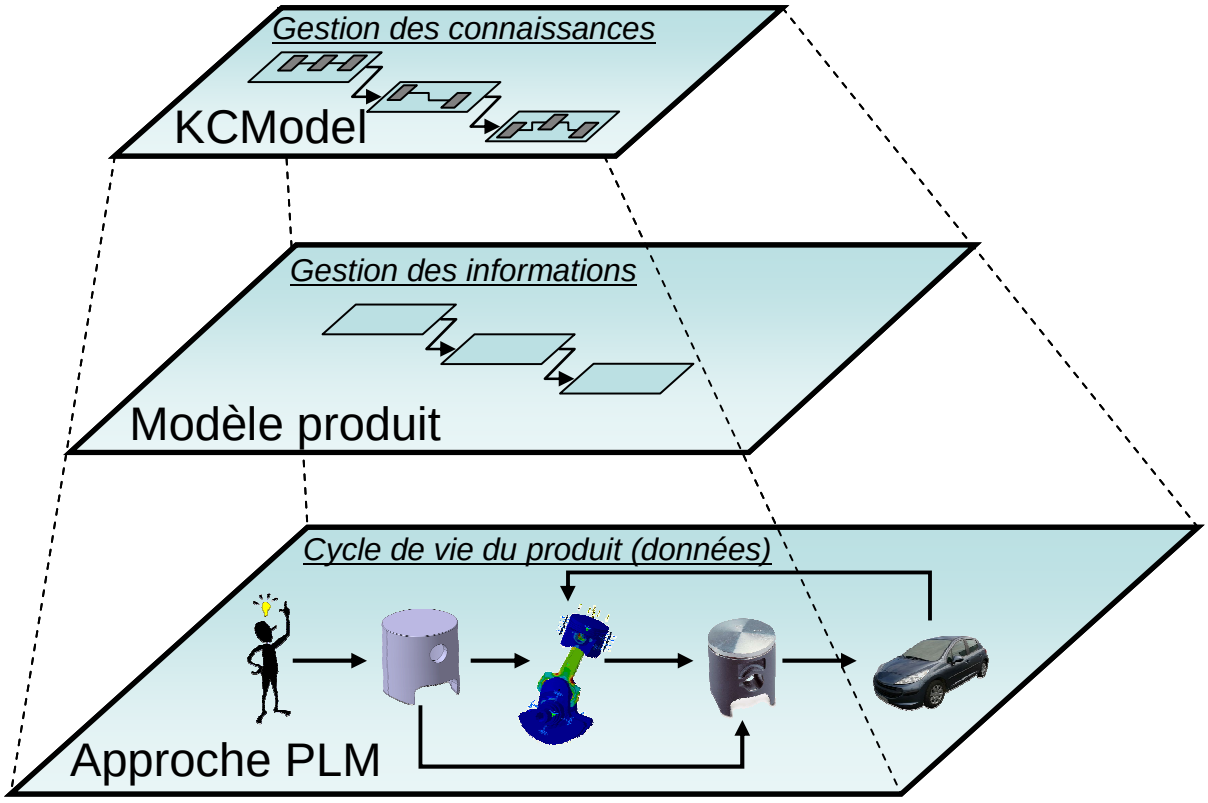


Figure 98 : positionnement du KCModel par rapport aux modèles produits et à l'approche PLM

Dans ce contexte, il nous semble que le KCModel constitue une approche, non pas alternative, mais complémentaire aux modèles produits existants (Figure 98). Il peut être considéré comme une brique ou un plug-in qui peut être ajouté pour gérer plus finement les connaissances en configurations, afin d'assurer la cohérence dans le processus de conception. Les données transitent alors du PLM vers le modèle produit chargé d'organiser et de gérer les informations selon la structure du produit, l'organisation, le process, etc. Les informations de types paramètres et contraintes, transitent ensuite vers le KCModel qui gère les connaissances en configurations afin d'aider les utilisateurs à converger au travers de processus itératifs. Enfin, ces connaissances redescendent vers les modèles produits pour être réintégrées dans la structure produit, puis vers le PLM assurant la gestion de l'ensemble des données tout au long du cycle de vie.

4.5 Vers un nouvel outil KCManager pour la gestion des connaissances de granularité fines en configurations

La définition et l'implémentation du KCModel est de nature à développer un outil d'ingénierie qualifiée de hautement productive [Gomes, 2008], à base de connaissances métier. Cet outil autoriserait l'interconnexion des différents paramètres de conception et l'intégration des contraintes multi-métier liées à la conception d'un système mécanique ou mécatronique complexe, en considérant (dans un premier temps) les métiers de la simulation numérique et de la conception (CAO) paramétrée. Cette solution sera basée sur le concept de gestion centralisée de configurations de connaissances constituant une nouvelle génération d'outil d'ingénierie qualifiée de KCManager - Knowledge Configuration Manager. Il permettra la coordination et la traçabilité des échanges de paramètres, de règles expertes et des instances de paramètres, et de règles entre les différents systèmes inhérents à la chaîne d'ingénierie numérique. C'est sur ce point que ces travaux de recherches rejoignent le projet ADN qui vise au développement d'un outil logiciel.

L'architecture envisagée pour cet outil se décompose en trois couches interconnectées.
Premièrement, le cœur, KCMCore, qui contient l'ensemble des concepts métiers de gestion des connaissances. C'est ici que "l'intelligence", ou plutôt les principes d'ingénierie visant à créer les ICEs et les configurations, gérer les conflits, etc., se positionnent.

Ensuite, une seconde couche sous forme de framework logiciel fournira un ensemble de services génériques tels que la gestion des droits, des accès (sessions), des rôles, les mécanismes de gestion de versions, etc. Les modèles décrits dans la version KCMTool peuvent être parfaitement supportés par cette plateforme logicielle générique. La plateforme envisagée dans le cadre du projet ADN est la plateforme open source TSC décrite en Annexe 24.

Enfin, une surcouche de connecteurs serviront de passerelles entre le cœur et les applications métiers quelles qu'elles soient : CAO, simulation, gestion de données, gestion des exigences. Un connecteur spécifique par application sera nécessaire et comportera des fonctionnalités particulières. Dans le cadre du projet ADN, il est prévu de fournir des connecteurs par défaut pour les grands standards de la modélisation CAO et la simulation du marché, mais également un "toolkit", c'est-à-dire une ouverture d'API (Application Programming Interface) permettant à quiconque de créer ses propres connecteurs.

Les fonctionnalités d'un outil de KCManager, basé sur le méta-modèle KCModel, peuvent être décrites de la façon suivante :

- stocker tous les paramètres et règles métiers en un point unique assurant ainsi la capitalisation et la réutilisation des connaissances et des savoir-faire.
- assurer la gestion de configuration, le "versionnement" et la traçabilité, de ces paramètres et règles métiers pour les projets d'ingénierie, mais aussi une gestion du workflow, etc.
- garantir un accès simultané et collaboratif par les applications métiers (PLM, KM, CAD, CAE, etc.) et par les différents acteurs où qu'ils se situent.
- lancer des processus de raisonnement en appliquant des techniques issues de l'intelligence artificielle (propagation de contraintes, systèmes multi-agents, etc.) afin de rendre la base "intelligente et proactive"

Les méthodes d'architecture et de conception développées permettront :

- d'assurer la cohérence de l'ensemble des travaux d'ingénierie, notamment ceux de l'ingénierie des systèmes complexes ;
- de garantir la continuité et la cohérence nécessaires à la gestion de compromis techniques entre maîtres d'œuvre et fournisseurs de différents rangs dans la chaîne de valeur ;
- de définir des architectures système réutilisables et de maîtriser la réutilisation sur plusieurs niveaux de la chaîne de valeur ;

- de réduire les coûts de reprise lors de l'intégration effective en articulant conception et intégration virtuelle.

Au même titre que l'utilisation des PDM ou des SDM, l'utilisation du KCManager modifiera le comportement des concepteurs et des responsables dans un projet de conception. Les récentes approches sur la gestion des connaissances dans les entreprises visent à les identifier pour les extraire et les capitaliser dans des outils adaptés visant à mieux les réutiliser.
Dans le cadre du KCManager, cela peut aboutir à l'apparition d'un nouveau métier dans la chaîne de conception. Ce métier serait dédié à la gestion et l'administration de ce patrimoine au niveau des paramètres et des contraintes pour une organisation.

Dans le cadre de ce projet, KCModel n'est pas suffisant pour satisfaire l'ensemble des objectifs du KCManager, mais il est considéré comme la base définissant les principaux concepts et mécanismes. Ainsi, avant d'implémenter avec certitude le KCModel dans le cadre du développement d'un l'outil pour le projet ADN, il a été nécessaire de tester et valider notre approche KCMethod/KCModel au travers de maquettes ou de démonstrateurs réalisés sur des cas industriels, principalement orientés automobile. Le résultat de ces expérimentations, nous ont permis d'adapter notre approche aux besoins des utilisateurs et d'affiner nos concepts de gestion des ICEs et des configurations de connaissances. Le chapitre suivant vous présente plusieurs de ces expérimentations.

4.5.1 Conclusion

Au cours de ce chapitre, nous avons présenté la seconde partie de notre contribution sous forme d'un méta-modèle KCModel basé sur la méthodologie KCMethod. Ainsi, KCModel a été réalisé en langage UML selon les approches de méta-modélisation, nous permettant de définir un méta-modèle générique de connaissances du produit. Il constitue une solution originale pour assurer la collaboration des acteurs métiers et la cohérence des connaissances utilisées tout au long de la chaine numérique. Néanmoins, dans un premier temps, nous nous focalisons uniquement sur les phases amont du processus de conception et sur les interactions du couple produit-simulation.

L'interaction des concepts présents dans les packages KCMCore et KCMTool semble garantir la mise en place d'un premier ensemble de fonctionnalités nécessaires à la collaboration entre acteurs et modèles métiers au niveau des paramètres et des contraintes.

Dès lors, dans l'objectif de valider l'approche générale KCM, nous avons réalisé des maquettes et des démonstrateurs nous permettant de valider avec confiance la pertinence de la solution. Ainsi, un prototype ADES (Alliance des Données Elémentaires de Simulation) a été réalisé, implémentant les principaux mécanismes de capitalisation et de gestion de configurations de connaissances. La présentation d'ADES et l'expérimentation KCMethod/KCModel sont présentées au chapitre suivant.

Chapitre 5. Gestion des configurations de connaissances : application au cas de la conception de structures et de l'automobile

Ce cinquième chapitre illustre l'expérimentation de notre approche de gestion des configurations de connaissances (KCMethod/KCModel).
Nous présentons le démonstrateur ADES – Alliance des Données Elémentaires de Simulation. ADES est une maquette de faisabilité permettant d'éprouver et valider notre démarche sur différents cas d'études.
Deux cas d'études sont abordés. Le premier concerne la conception d'un support de fixation. Assez simple, il illustre les fonctions de base développées dans KCMethod/KCModel.
Le deuxième concerne une application au cas de l'automobile. Plus complexe, il donne une meilleure vision d'une réelle implémentation de notre approche dans l'environnement industriel.

5.1 Introduction

Dans ce chapitre, nous présentons deux applications de notre proposition de gestion des connaissances en configurations sur des cas industriels. Ces applications nous permettent d'illustrer le fonctionnement de la méthodologie, mais aussi de valider nos concepts.
Pour réaliser ces expérimentations, nous avons développé une maquette appelée ADES – Alliance des Données Elémentaires de Simulation dédiée à l'évaluation de KCMethod et KCModel pour la conception-simulation de produits manufacturés. Le chapitre s'articule en trois parties illustrées Figure 99.

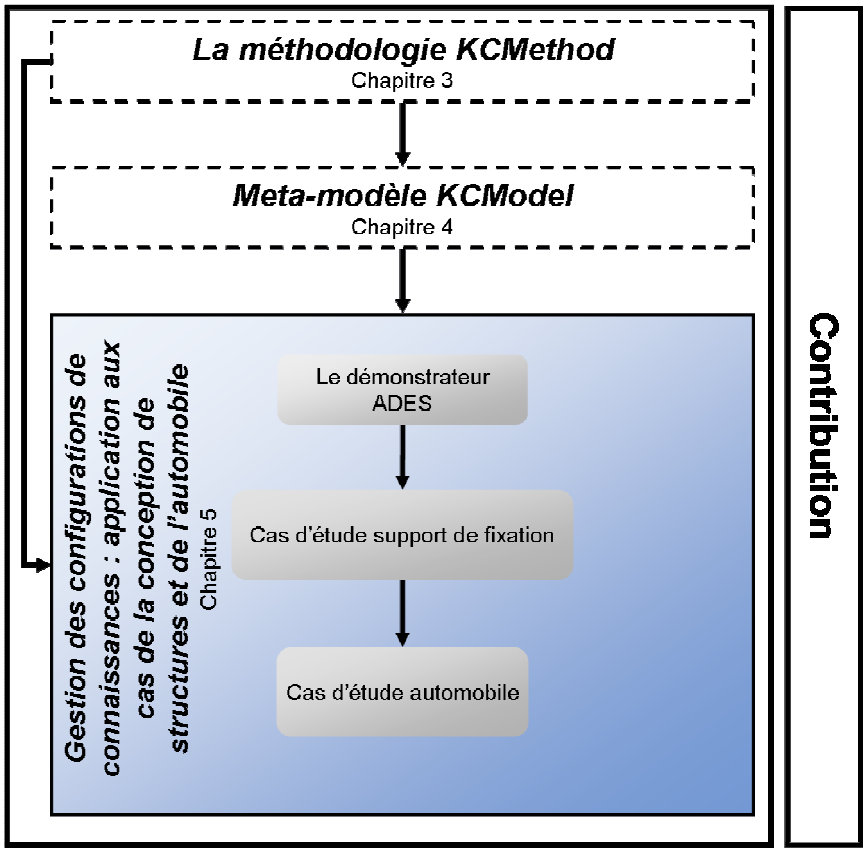


Figure 99 : organisation de la contribution pour l'expérimentation KCM

Dans une première partie, nous présentons ADES ainsi que les principales fonctionnalités et objectifs attendus.

Dans la deuxième partie, nous présenterons un premier cas d'étude portant sur la conception d'un support de fixation. Ce cas simple permet de bien comprendre comment les principaux concepts sont mis à contribution, et quels types de service l'approche KCM permet d'atteindre.

Dans la troisième partie, nous utilisons ADES sur un cas d'étude réalisé avec la collaboration de PSA – Peugeot Citroën portant sur la conception de composants du Groupe Moto-Propulseur. Ce cas, plus complexe, permet de mettre en évidence l'implémentation d'une solution de gestion des configurations de connaissances dans l'environnement industriel.

5.2 Le démonstrateur

5.2.1 ADES (Alliance des Données Elémentaires de Simulation)

ADES est un démonstrateur, ou plutôt une maquette de faisabilité permettant de tester et valider notre approche de gestion des configurations de connaissances. ADES est basée sur KCMethod et le méta-modèle KCModel, elle est dédiée au domaine de la conception et de la simulation de produits manufacturés. ADES est une application que nous qualifions de KCManager.
Le nom ADES a été choisi pour permettre à un concepteur de bien se représenter quels types de fonctionnalités il est en droit d'attendre. Ainsi, nous parlons de données plutôt que de connaissances. En effet, les données sont des entités qu'un concepteur se représente facilement dans les modèles métiers, alors que les termes information et encore plus connaissance peuvent prêter à confusion. Le terme "Elémentaire" fait référence aux données, informations et connaissances cruciales à manipuler, c'est-à-dire les paramètres et les contraintes ayant réellement un rôle dans la collaboration entre les acteurs. Le terme "Simulation" fait référence aux phases amont du processus de conception, où l'utilisation de modèles d'intégration CAO/Calcul est importante. Enfin, "Alliance" fait référence à une centralisation, une unité des données, informations et connaissances manipulées, ainsi qu'à la notion de cohérence.

Suite à de premières phase d'expérimentations, via Excel, intégrant des commandes Visual Basic, des développements informatiques ont permis de mettre au point une maquette plus évoluée baptisée ADESv1. Cette dernière intègre les principaux concepts KCM (KCMethod+KCModel) et permet à un utilisateur non initié au modèle de tester son application sur des cas d'études industriels. Ainsi, l'approche a pu être testée grandeur nature, notamment par des utilisateurs PSA, avec comme objectif d'avoir un retour quand à la crédibilité de notre démarche dans le cadre de leurs problématiques. Les résultats de ces tests nous on permit d'effectuer des ajustements, à la fois sur la méthodologie et sur le méta-modèle. Dans l'application au cas du support de fixation, décrite dans la suite de ce chapitre, nous utilisons ADESv1, et dans l'application au cas automobile GMP avec PSA, nous utilisons la dernière version ADESv2.

5.2.2 Objectifs et fonctionnalités attendues

Ces travaux de recherche sont menés dans le cadre du projet national ADN [Systematic, 2010]. En conséquence, ADES fait office d'application d'aide à la spécification du besoin pour ADN. Ainsi, nous détaillons les fonctionnalités disponibles dans la maquette que nous utiliserons dans les expérimentations.

5.2.2.1 Contexte de développement : le cadre du projet ADN

ADES est une application WEB qui implémente les principaux concepts du KCModel : ICE et KnowledgeConfiguration avec gestion des conflits via l'utilisation d'un moteur de résolution de contrainte basé sur le langage PROLOG [Blackburn et al., 2007]. Son développement est basé sur l'utilisation de certains outils Microsoft : SQLserver2005 pour la base de données ; C# avec ASP.NET pour le langage de programmation côté serveur ; HTML, CSS, Javascript pour les langages côté client.

Pour faciliter l'utilisation d'ADES dans un contexte industriel, un intérêt tout particulier a été porté à la compatibilité de l'application ADES avec d'autres outils, tels Excel et CATIA. Ainsi, deux connecteurs on été réalisés permettant à ADES de communiquer avec le tableur Excel et le modeleur CATIA V5 R19. Les connecteurs permettent de faire des exports ou de synchroniser les configurations

de connaissances avec les paramètres et les contraintes contenus dans les modèles métiers de façon bidirectionnelle (Figure 100).

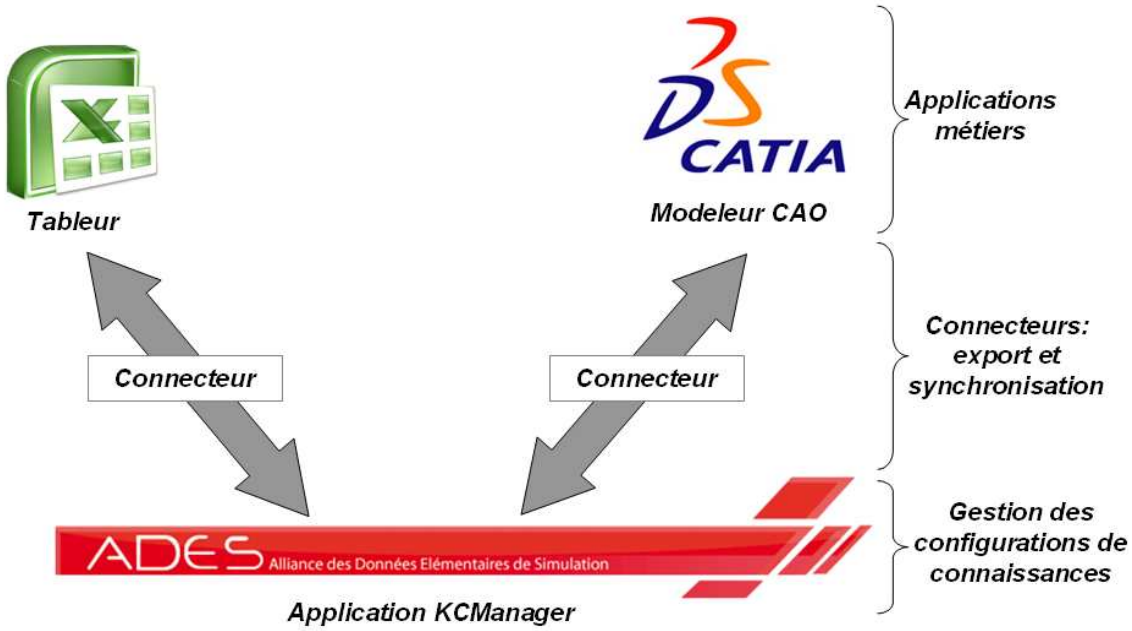


Figure 100 : développements ADES

5.2.2.2 Utilisation

L'IHM est divisé en quatre parties (Figure 101).

1. L'arbre des tâches a pour principale fonction d'organiser les configurations.
2. Le menu permet de réaliser certaines actions comme créer une ICE, des instances, gérer les tâches.
3. Le panier d'ICEs est utile lors de la création d'instances d'ICEs à associer à une configuration.
4. L'écran principal changera en fonction de ce que l'utilisateur aura sélectionné dans le menu.

Plusieurs utilisateurs peuvent accéder en même temps à l'application et modifier les données de façon collaborative. Les modifications s'effectuent en temps réel ainsi, elles sont instantanément visibles pour l'ensemble des utilisateurs.

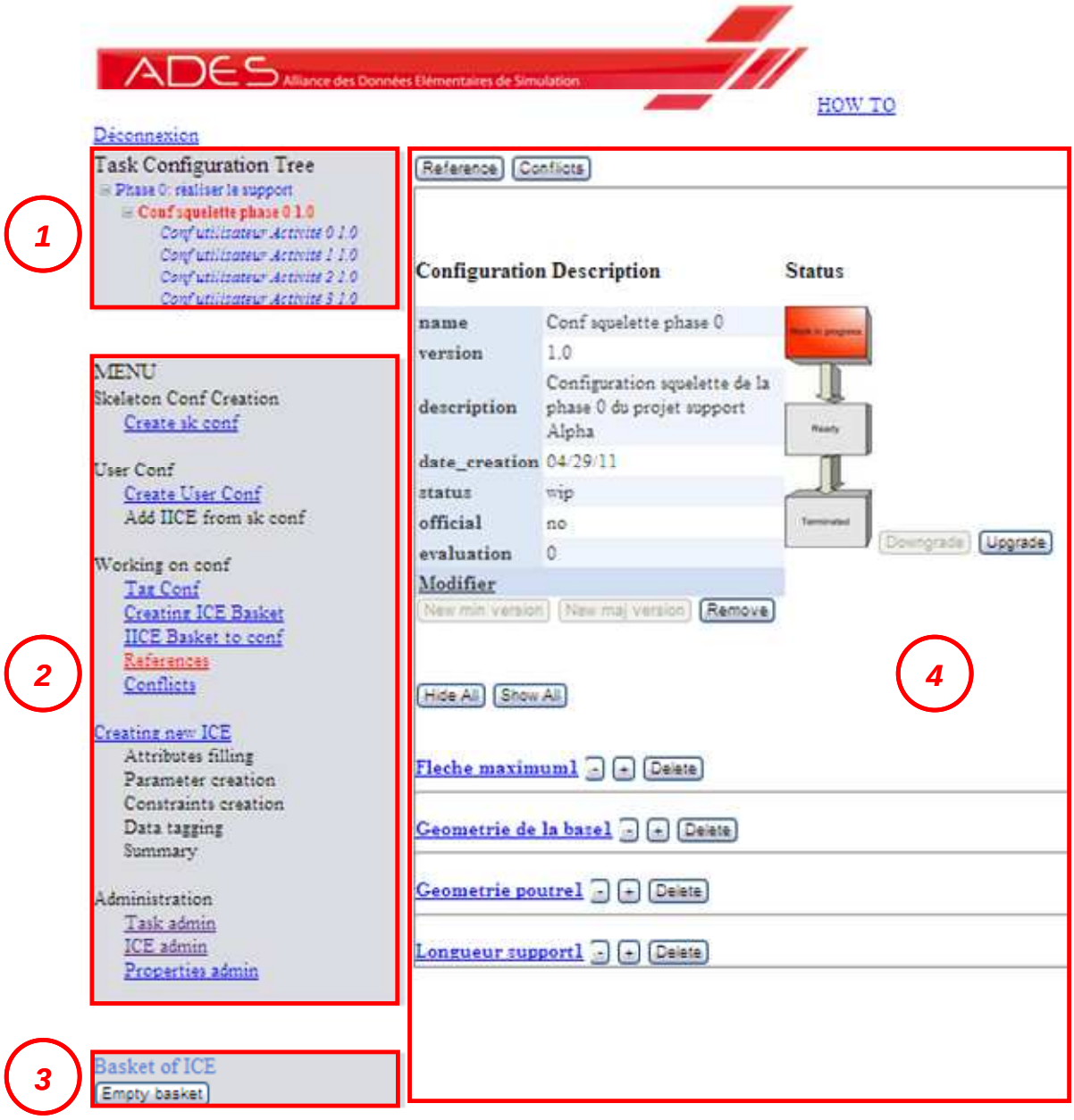


Figure 101 : organisation d'ADES

Un autre exemple d'IHM est proposé en Annexe 25.

5.2.2.3 Fonctionnalités principales implémentées

Toutes les fonctionnalités définies dans KCMethod ne sont pas encore accessibles dans ADES.
En effet, cette application est encore en cours de développement et donc en perpétuelle évolution.
A cet instant, Un utilisateur peut accéder aux fonctionnalités suivantes :

- Connexion à ADES via un identifiant et un mot de passe
- Créer, modifier, supprimer et visionner une ICE (avec gestion simple des impacts).
- Créer, modifier, supprimer, visionner et accéder aux instances d'une configuration (squelette ou utilisateur)
- Ajouter ou enlever des instances d'ICEs aux configurations
- Tagger des ICEs et configurations avec des Knowledge Index
- Rechercher des ICEs grâce aux attributs ou aux Knowledge Index
- Créer, modifier, supprimer des tâches et des Knowledge Index

- Organiser l'arbre des tâches et des configurations
- Modifier les valeurs des instances d'ICEs des configurations
- Voir les conflits sur les paramètres et sur les règles (bien que cette dernière fonctionnalité ne soit pas finalisée)

Concernant la gestion de conflits, seule la comparaison des paramètres et la vérification des contraintes n'est disponible. Aucune fonctionnalité de propagation ou de détection des redondances cycliques n'est implémentée.

Il n'existe pas réellement de notion de rôle ou de droits dans ADES, tous les utilisateurs sont considérés comme étant administrateurs, c'est-à-dire sans restriction de droits.

5.3 Cas d'étude support de fixation

Ce premier cas d'étude permet d'illustrer simplement le fonctionnement des configurations de connaissances. L'accent est porté sur l'utilisation des ICEs dans les configurations de connaissances permettant d'identifier les conflits et ainsi favoriser la collaboration. Ce cas d'étude reprend l'exemple qui illustre Le package KCMProject au chapitre 4 section 4.3.2.4.
Il est à noter que la résolution du scénario n'est qu'une proposition. En effet, plusieurs stratégies sont envisageables.

5.3.1 Contexte du cas d'étude et objectifs

L'entreprise "MetaliX"[12] fabrique des structures métalliques comme des charpentes, des supports de fixation et des châssis pour différents équipements, comme des panneaux d'affichage ou des remorques. C'est une entreprise de taille moyenne (environ 100 salariés), elle dispose d'un service de marketing/communication, d'un bureau d'études et d'un service d'installation et de maintenance des produits. MetaliX utilise ADES depuis déjà plusieurs mois et a défini plusieurs KnowledgeDomain relatifs à chaque famille de produits de l'entreprise. Chaque KnowledgeDomain regroupe l'ensemble des connaissances nécessaires à la conception des charpentes, des châssis et des supports de fixations. Dans les KnowledgeDomain on trouve les ICEs génériques, des Templates de configuration de connaissances, des canevas de projets et tous les projets déjà réalisés.

MetaliX souhaite se développer sur un nouveau marché : le loisir d'été à la montagne. En effet, l'été les télésièges sont transformés pour pouvoir y fixer des VTT. Cette transformation implique l'utilisation de supports démontables. Dans ce contexte, MetaliX souhaite créer un nouveau projet appelé "Support Alpha". L'objectif est de concevoir le support le plus rapidement possible en limitant les incohérences liées aux difficultés de collaboration. MetaliX utilise ADES et un connecteur par application extérieure à relier. L'entreprise connaît bien le processus de conception de ce type de produit et estime à environ 10 jours ouvrés (avec ADES) le temps de réalisation d'un modèle (Phase 0) prêt pour la fabrication (Phase 1).

Cette première "phase 0" de réalisation du support est composée de quatre activités (chaque activité est réalisée par un acteur différent) dont l'articulation est présentée Figure 102 :
- Etude marketing et spécifications (Fichier Excel)
- Conception initiale (CAO CATIA V5)
- Simulation et optimisation (Modèle éléments finis volumique avec le code CASTEM)
- Mise en plan (conception finale : outil développé en interne)

[12] Dans le cadre de notre expérimentation « MetaliX » est une société fictive

ID	Phase 0: Réaliser le support	Début	Terminer	Durée
1	Activité 0: Etude marketing et specs	11/05/2011	13/05/2011	3j
2	Activité 1: Conception initiale	16/05/2011	19/05/2011	4j
3	Activité 2: Simulation et optimisation	17/05/2011	20/05/2011	4j
4	Activité 3: Mise en plan	23/05/2011	26/05/2011	3,5j

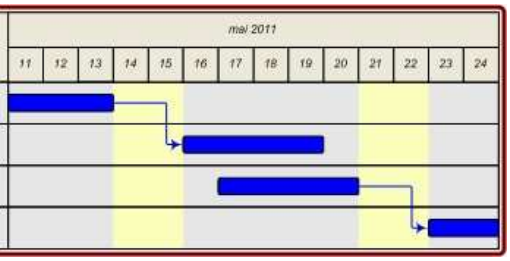


Figure 102 : planning de la phase 0

5.3.2 La démarche et la réalisation

Dans le KnowledgeDomain "Support de fixation", le responsable projet crée le projet "Support Alpha", crée la "phase 0 : Réaliser le support" ainsi que la configuration squelette associée qu'il laisse vide. C'est-à-dire qu'il n'instancie pas d'ICEs dans la partie référence. Connaissant parfaitement le déroulement du processus de conception d'un support de fixation, il crée également les activités 0 à 3 avec leurs noms, de sorte à ce que les différents acteurs n'aient plus qu'à y associer leurs configurations utilisateur.

5.3.2.1 Activité 0

Le premier utilisateur doit effectuer une étude du besoin afin de réaliser un cahier des charges fonctionnel et des spécifications. Toutes les règles permettant de définir des exigences fonctionnelles en fonction de critères de coûts, de tendance sur les marchés, sont stockées dans un fichier Excel. Il doit donc créer sa configuration utilisateur associée à l'Activité 0 et recherche les ICEs génériques, à utiliser avec son fichier, dans la base d'information du KnowledgeDomain. Il sélectionne trois ICEs et les instancie dans sa configuration utilisateur :

- ***Encombrement max*** ("Long_max" (mm), "Larg_max" (mm) et "Haut_max" (mm))
- ***Chargement maximum*** ("chargement_max_a_supporter" (Kg))
- ***Fleche maximum*** ("Fleche_maxi" (mm).)

Les instances d'ICEs, ajoutées et valuées par un utilisateur dans sa configuration, s'ajoutent automatiquement dans la partie référence de la configuration squelette puisque cette dernière n'avait pas été remplie par le responsable de projet.

A ce stade, le chef de projet valide la configuration utilisateur, modifiant ainsi son statut en "WIP". L'utilisateur, quant à lui, peut travailler sur son modèle Excel, et l'actualiser avec la configuration utilisateur. Les paramètres contenus dans les instances d'ICEs sont maintenant validés et synchrones avec le modèle Excel (Figure 103).

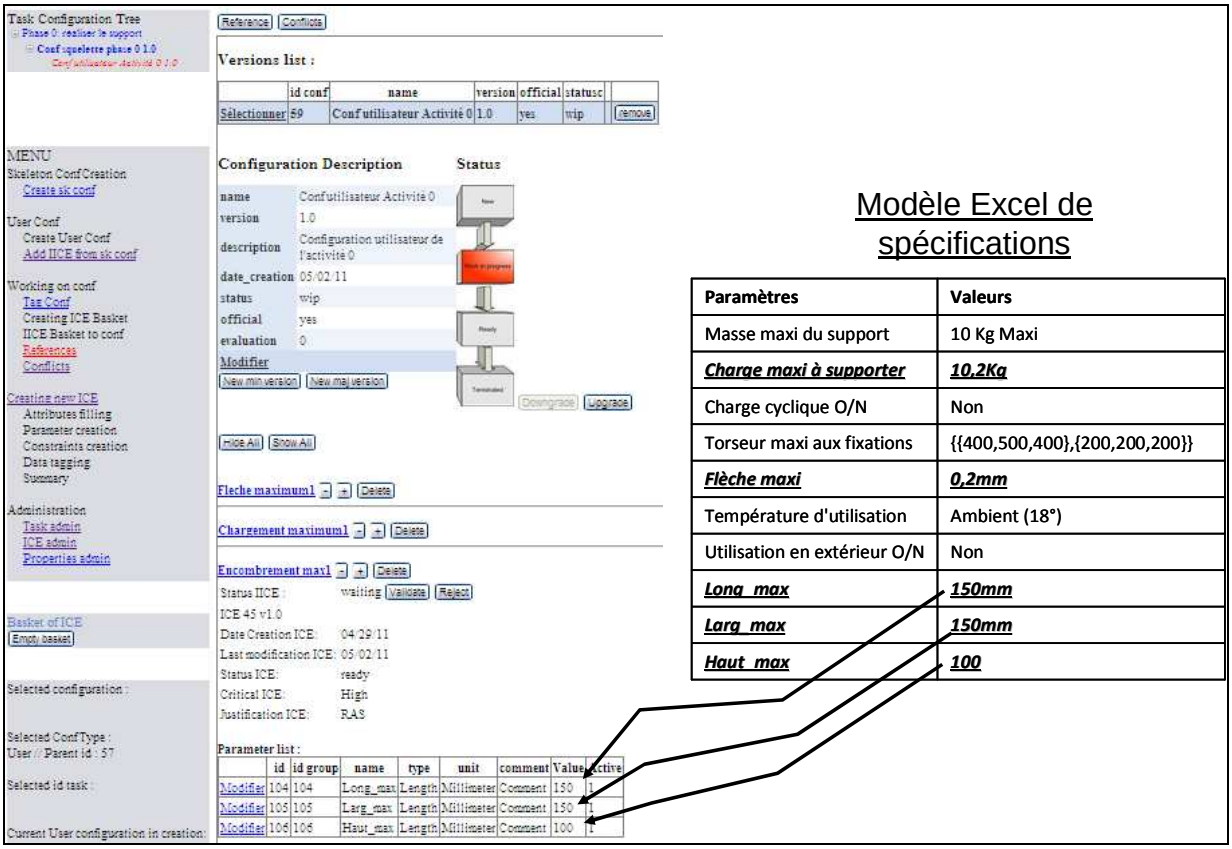


Figure 103 : configuration utilisateur synchrone avec le modèle Excel de l'activité 0

Dans le fichier Excel, seuls cinq paramètres ont été identifiés comme cruciaux (en gras) et sont présents dans ADES. La figure 94 illustre trois paramètres cruciaux ; "Long_max", "Larg_max" et "Haut_max" appartenant à l'ICE "Encombrement max".
Cet utilisateur peut faire plusieurs versions de sa configuration en fonction des valeurs des paramètres testées. Une fois satisfait d'une version, il la publie vers la configuration squelette et fait évoluer le statut de sa configuration utilisateur en "Ready" signifiant qu'il à terminé son activité.

5.3.2.2 Activité 1

Une fois les spécifications et le cahier des charges établi, le concepteur crée sa configuration utilisateur Activité 1 qui va être associée à son modèle CAO. Cet utilisateur souhaite utiliser deux instances d'ICEs pas encore présentes dans la configuration squelette (partie référence). Il va donc les chercher directement dans la base d'ICEs génériques :

- ***Géométrie de la base*** ("Epaisseur base" (mm), "L inter trou" (mm), "H inter trou" (mm))
- ***Géométrie poutre*** ("Longueur support" (mm), "Largeur support" (mm), "Hauteur support" (mm))

Cet utilisateur commence à concevoir le support de fixation et tout au long de la conception, il synchronise le modèle avec sa configuration utilisateur (Figure 104). Il peut créer plusieurs versions de la configuration utilisateur, mais une seule n'est prise en compte par la configuration squelette.

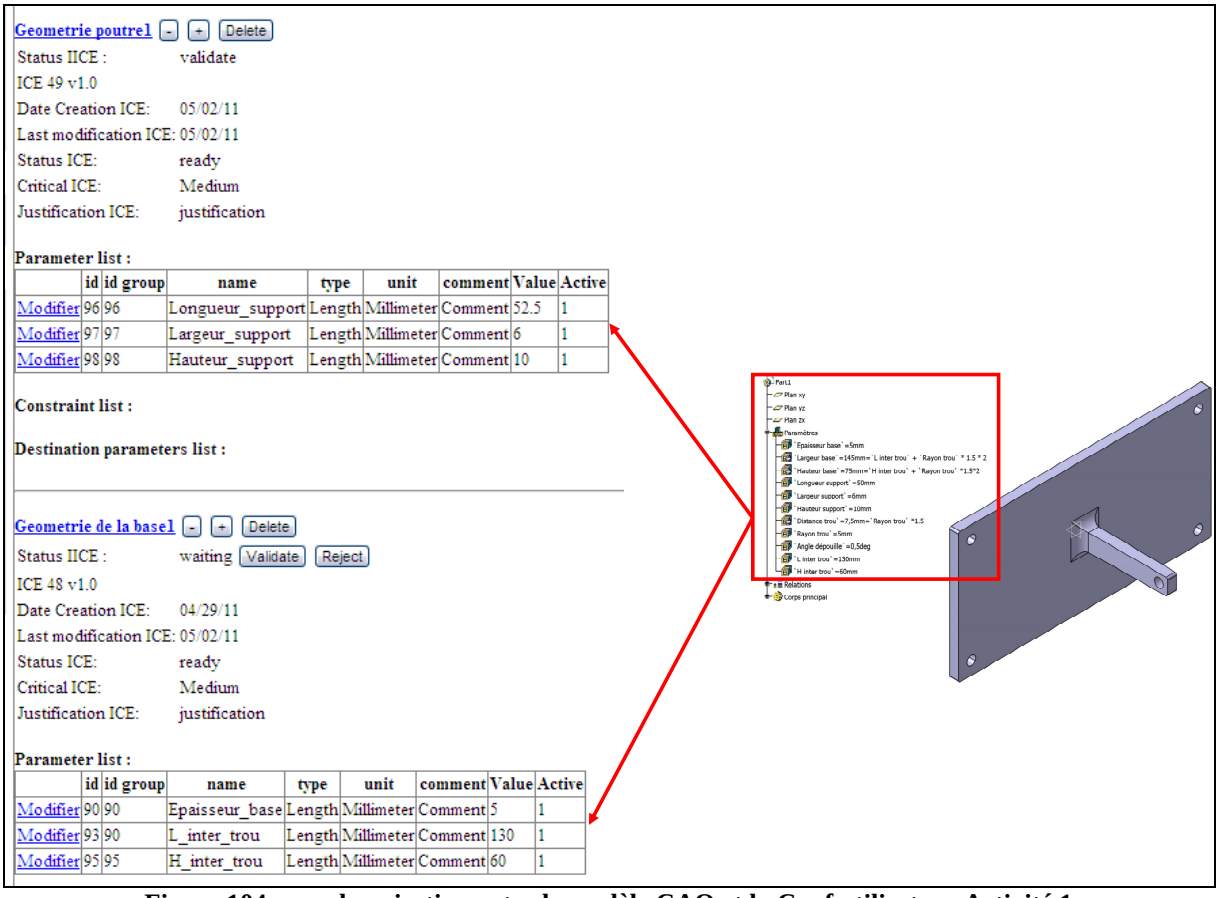


Figure 104 : synchronisation entre le modèle CAO et la Conf utilisateur Activité 1

5.3.2.3 Activité 2

L'activité 2 démarre peu après l'activité 1 et le deuxième concepteur crée la configuration utilisateur Activité 2 en récupérant les instances d'ICEs de la partie référence de la configuration squelette (valuées par les autres utilisateurs), excepté "Encombrement max". En plus, il instancie depuis la base les ICEs suivantes :

- ***Longueur support*** qui contient le paramètres "LO_SUP" (mm), et la contrainte LO_SUP=ICE_5.longueur support - (ICE_4.Epaisseur base)/2,
- ***Chargement*** qui contient le paramètre "Force" (mm) et la contrainte Force = 9,81 * ICE_2.Charge max à supporter,
- ***Déplacement maximum*** qui contient le paramètre "Deplacement maxi" (mm) et la contrainte Deplacement maxi < ICE_3.Fleche maxi.

Cette configuration utilisateur contient des instances d'ICEs partagées par les deux autres configurations utilisateurs.
Comme les utilisateurs précédents, il synchronise sa configuration avec son modèle éléments finis (Annexe 26) et lance une simulation sur le support avec les valeurs de paramètres récupérées de l'activité 1.
A la vue des premiers résultats, un conflit apparaît en interne à sa configuration car le Deplacement maxi (=0,24mm) dépasse la Fleche maxi (=0,2mm), violant la contrainte de déplacement maximum. L'utilisateur décide alors de modifier la Fleche maxi pour remonter la valeur à 0,22 mm et demande au concepteur CAO de modifier la géométrie pour avoir moins de flexion.
La modification de la Fleche maxi engendre l'apparition d'un nouveau conflit, entre la valeur de la Fleche maxi définie dans l'activité de spécification et celle de simulation (Figure 105).

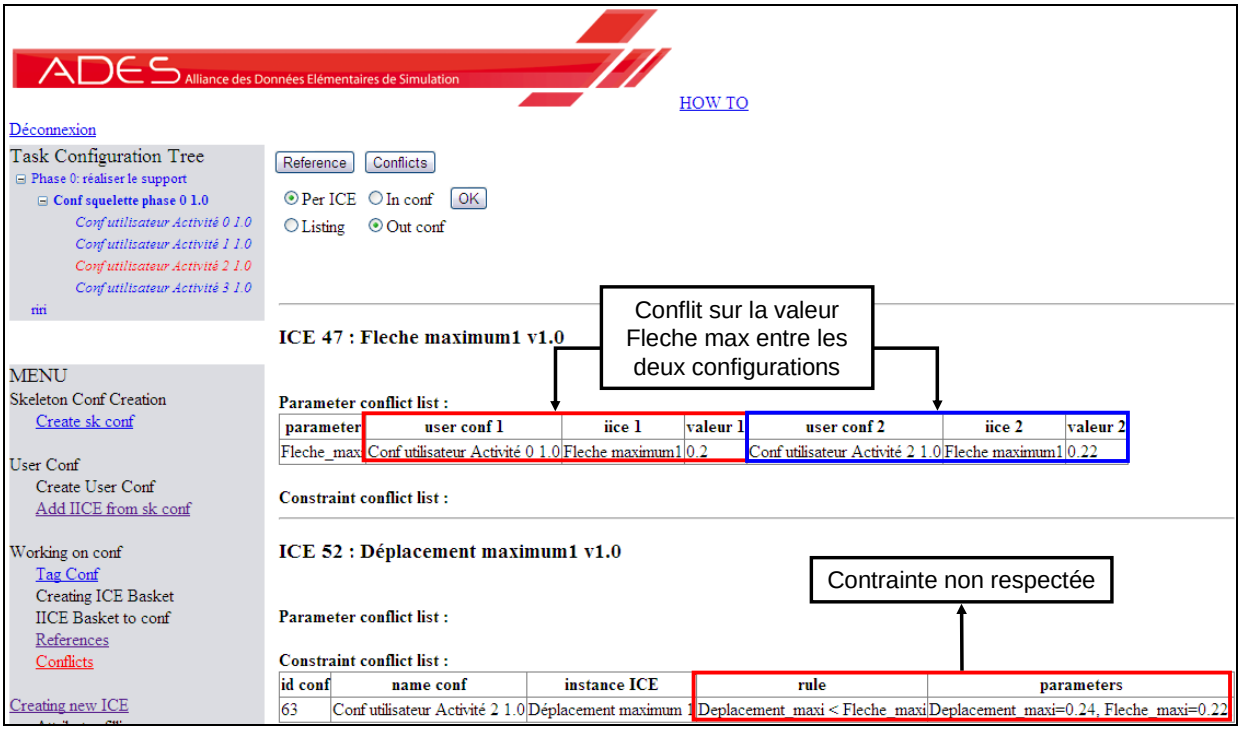


Figure 105 : affichage des conflits

En effet, la figure ci-dessus illustre l'affichage des conflits via la configuration squelette (partie conflit). Elle affiche le conflit interne à la configuration utilisateur Activité 2 sur la contrainte et le conflit intervenant entre les instances d'ICEs utilisées entre la configuration utilisateur Activité 0 et configuration utilisateur Activité 2.

Le responsable de l'activité 2 demande alors une réévaluation du paramètre afin que le responsable du cahier des charges en tienne compte dans ses spécifications, ce qui est accepté.

Le concepteur propose alors une nouvelle version de modélisation CAO (donc de la configuration utilisateur Activité 1) en modifiant les paramètres du support pour le rigidifier. Le responsable calcul, après avoir récupéré les nouvelles valeurs, lance une nouvelle simulation et obtient un résultat concluant. Les configurations utilisateur Activité 1 et 2 passent alors en statut "Ready".

Ayant validé la conception par le calcul, la dernière activité peut alors démarrer. Le dernier utilisateur crée sa configuration en récupérant les instances d'ICEs issues des résultats des activités précédentes, et réalise sa mise en plan en étant sûr d'utiliser les bonnes valeurs de paramètres, et en respectant les contraintes. De plus, si une modification imprévue devait être effectuée sur le modèle CAO, il serait prévenu très tôt et pourrait en tenir compte. Une fois terminé, il fait évoluer le statut de sa configuration.

Les quatre configurations utilisateur en statut "Ready" et les conflits résolus, le chef de projet fige l'ensemble des configurations de connaissances. Il peut alors versionner la configuration squelette afin quelle soit réutilisée dans la phase suivante de fabrication.

5.3.3 Constat support de fixation

Ce scénario est simpliste de part le nombre de modèles utilisés, les connaissances manipulées, les faibles itérations entre les utilisateurs, mais il offre l'avantage de visualiser rapidement et simplement les principaux mécanismes de l'approche KCM.
En outre, ADES permet de consulter en temps réel les connaissances manipulées par un utilisateur en dehors des applications métiers, de suivre les modifications sur les paramètres et sur les versions de

configurations, et surtout de visualiser très rapidement les conflits. Cette information est extrêmement importante et permet aux utilisateurs de mieux collaborer durant l'ensemble du processus de conception. La réutilisation est aussi facilitée, dans la mesure où les connaissances sont centralisées et possèdent un cycle de validation.
Afin de poursuivre l'expérimentation, il est nécessaire de mettre en évidence la collaboration des acteurs autour des principaux concepts présents dans ADES, dans le cadre d'un projet plus important et représentatif du milieu industriel. Ainsi, nous avons réalisé une expérimentation appliquée au cas de l'automobile.

5.4 Cas d'étude automobile

Ce second cas d'étude appliqué au domaine de l'automobile, et plus particulièrement à la conception de GMP - Groupe Moto-Propulseur, à été réalisé avec la collaboration de PSA Peugeot Citroën. Contrairement au premier scénario, ce cas d'étude met en évidence l'utilisation d'ADESv2 sur des problématiques industrielles concrètes rencontrées lors du pré-dimensionnement des composants du GMP.
Dans cette l'expérimentation, des outils spécifiques d'intégration CAO/Calcul sont utilisés en phase amont du processus de conception, ils sont décrits en Annexe 1.
Attention, les données et les processus de ce cas d'étude sont scénarisés mais représentatifs d'une utilisation réelle et visent à mettre en avant la collaboration des acteurs.

5.4.1 Contexte du cas d'étude et objectifs

Dans le cadre des directives et des normes environnementales, le nouveau règlement introduit des exigences communes concernant les émissions des véhicules. L'évolution des normes environnementales d'Euro 5 à Euro 6 nécessite donc d'obtenir un meilleur rendement du moteur thermique.
Dans le cadre d'un projet de développement d'une nouvelle gamme de moteur, le service R&D a mis au point des culasses plus performantes permettant d'atteindre un rendement répondant à ces nouvelles normes. A partir de ces premiers travaux, une équipe est en charge de la conception de l'attelage mobile et du carter cylindre du projet de conception d'un moteur Essence 150cv.
L'attelage mobile, qui regroupe l'ensemble des pièces du moteur en mouvement avec l'axe de sortie de moteur (vilebrequin, bielle, piston), influe directement sur la performance et la consommation ; le carter cylindre servant d'interface entre l'attelage mobile et la culasse. Il est donc d'usage de modéliser les parties mobiles avant de modéliser les parties fixes.

Dans le cadre de notre étude, nous nous focalisons sur les deux premières phases du projet. Elles correspondent aux activités de pré-dimensionnement impliquant un grand nombre de compromis et d'échanges entre différents acteurs métiers (Figure 106 et Tableau 7):

- Une première phase, dont l'objectif est de définir le choix du concept moteur (pré-dimensionnement 0D-1D) ;
- Une seconde phase, plus approfondie, aura pour objectif de déterminer l'ensemble des paramètres de l'attelage mobile et du carter cylindre.

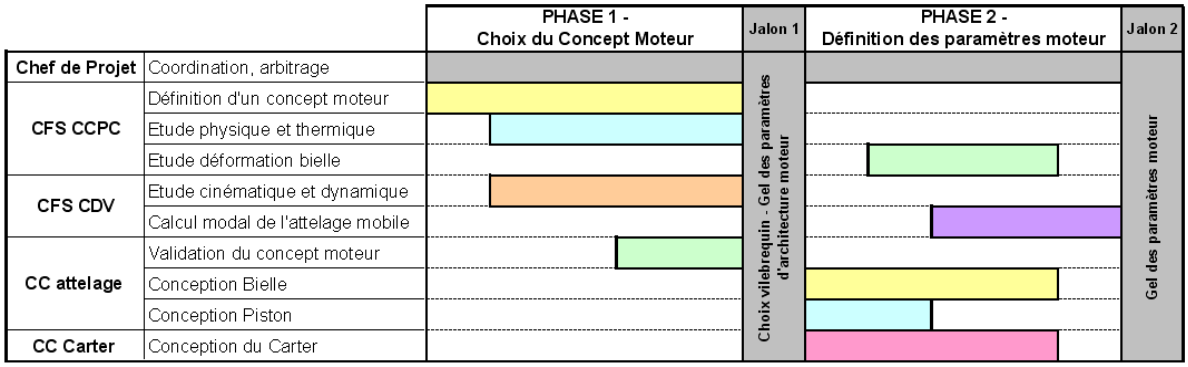


Figure 106 : rôle des acteurs dans le cadre du projet moteur

Ainsi, lors de ces phases, nous considérons donc cinq acteurs impliqués dans la conception du GMP (Tableau 7) :

• Le chef de projet (CP)	CP coordonne l'ensemble des activités de conception et de simulation.
• Le chargé de projet Fonction Système – Combustion et Conversion Pression Couple (CCPC)	CCPC est chargé, dans un premier temps, de proposer des concepts moteurs à partir d'un fichier Excel regroupant tous les paramètres et règles de base pour la définition d'un GMP (activité appelée pré-dimensionnement moteur). Ensuite, il devra simuler les sollicitations et les contraintes engendrées par le système en utilisant le modèle GMP Simulation système 0D-1D. Enfin, dans la seconde phase, il devra vérifier la tenue en fatigue de la bielle
• Le chargé de projet Fonction Système – Cinématique, Dynamique et Vibratoire (CDV)	CDV est responsable de l'étude cinématique et dynamique du concept moteur, qu'il effectuera sur le même modèle GMP Simulation système 0D-1D. En fin de phase de pré-dimensionnement, CDV devra mener un calcul modal de l'attelage mobile sous l'A² GMP (CAO/Calcul) afin de valider la préconception.
• Le chargé des composants de l'attelage mobile (CC attelage)	CC attelage devra valider le concept moteur par rapport au référentiel Bielle et aux vilebrequins disponibles. Suite à la validation du concept moteur, il sera amené à concevoir le piston et la bielle.
• Le chargé de composant Carter Cylindre (CC Carter)	CC Carter est chargé de la conception du carter cylindre.

Tableau 7 : tableau de répartition des acteurs en fonction des activités du processus de conception

Les filières de calcul du modèle GMP Simulation système 0D-1D et de l'A² GMP sont illustrées en Annexe 27.

La conception de ce GMP doit répondre au Cahier des Charges moteur, qui regroupe l'ensemble des données d'entrées suivantes (Tableau 8):

Type Moteur	Essence
Cylindrée	1600 cm3
Puissance	150 cv
Vitesse rotation Max	6500 tr/min
Encombrement	VF.CatPart

Tableau 8 : données du cahier des charges moteur

Phase 1 : Choix du concept moteur

Cette première phase a pour but de définir les principaux paramètres de l'architecture moteur (Tableau 9), et concerne uniquement la conception de l'attelage mobile (le choix du vilebrequin en fin de phase 1).

Nombre Cylindres
Diamètre alésage carter (Diamètre piston)
Course
Entraxe Bielle
Hauteur Compression piston

Tableau 9 : paramètres principaux de l'architecture moteur

Cette première phase implique une collaboration de trois acteurs autour de la définition du concept moteur (Figure 107) :

- CCPC devra dans un premier temps proposer des choix de concepts moteurs
- CCPC et CDV collaboreront pour optimiser le choix du concept moteur
- CC Attelage validera le concept en vérifiant le bon respect de son référentiel métier et en évaluant la compatibilité du concept avec les vilebrequins disponibles en base (sous forme de Templates).

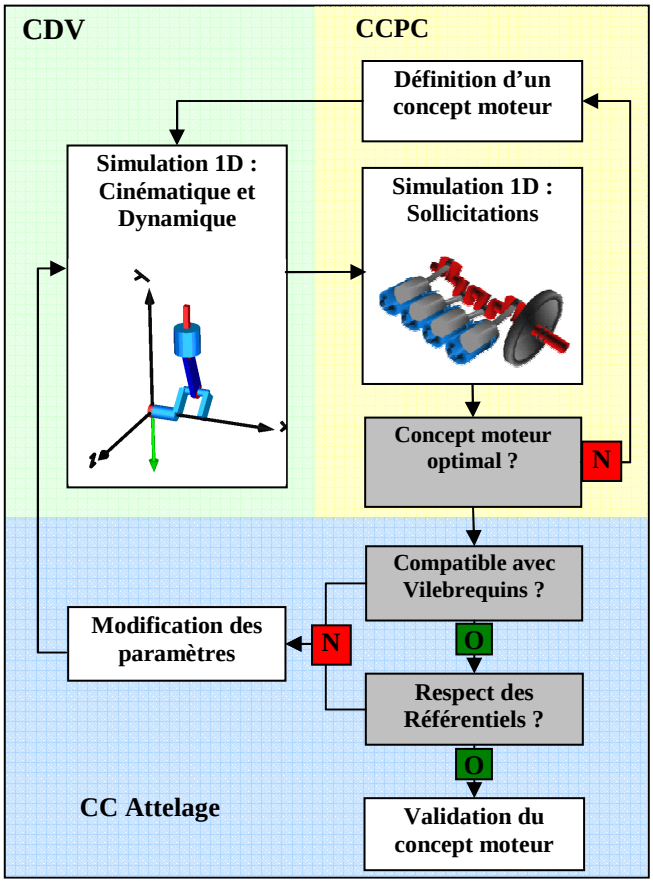


Figure 107 : processus phase 1

Dans un premier temps, une étude préliminaire sous Excel permet à CCPC d'évaluer l'impact des données du cahier des charges et du choix d'un concept sur les paramètres moteurs. Ce fichier regroupe l'ensemble des « bonnes pratiques » sous forme de paramètres du moteur, ainsi que les règles qui les régissent, permettant aux concepteurs, en début de projet, de définir les grands axes d'architectures.

Exemple de Règle : ‘Cylindrée’ = ‘Nombre Cylindres’ x ‘Course’ x ‘Diamètre alésage’^2 x Pi / 4. Le choix d’un moteur 3 ou 4 cylindres impactera les dimensions du carter cylindre
Dans un second temps, à partir des valeurs transmises par CCPC, CDV est amené à étudier le comportement cinématique et dynamique de l’attelage mobile via le modèle GMP simulation système 0D-1D. Il fournit à son tour à CCPC les résultats issus de son étude, qui se chargera d’étudier les sollicitations engendrées sur le système.
Après plusieurs boucles itératives de conception et de simulation, CCPC fait un compromis entre tous les critères évalués et transmet les données correspondantes au concept optimal à CC Attelage.
Enfin, CC Attelage vérifie la cohérence du modèle par rapport à son référentiel métier et la réutilisation d’un vilebrequin instancié depuis un Template ; auquel cas il modifie certaines données de sorte à respecter toutes les contraintes. A l’issue de cette modélisation, les chargés de projet Fonction Système testent la nouvelle configuration sur le modèle GMP simulation 0D-1D avant la validation de la phase 1.

Phase 2 : Détermination des paramètres moteurs

L'objectif de la phase 2 consiste à définir l'ensemble des paramètres moteurs afin de concevoir les pièces de l'attelage mobile et du carter cylindre, et de simuler différentes sollicitations. A ce stade du projet, les dimensions du vilebrequin sont connues et les paramètres d'architecture du moteur énoncés en phase 1 sont définis.

Tous les acteurs sont impliqués dans cette phase, constituée de 5 tâches (Figure 108) :

- La conception de la bielle, du piston et du carter cylindre se fait en parallèle ;
- La bielle doit être simulée en traction et en compression ;
- Le GMP doit être simulé (calcul modal) pour valider certaines règles de vérifications.

Toutefois, au cours de la modélisation des pièces, les valeurs de certains paramètres partagés seront amenées à évoluer et impacteront les CAO des pièces voisines. Ainsi, les acteurs devront collaborer autour de certains paramètres afin de maintenir à jour leurs modèles CAO.

Par ailleurs, CC attelage et CCPC devront effectuer des itérations (conception-simulation) afin d'obtenir une bielle qui respecte le référentiel "Allongement et Compression maxi admissible".

Pour conclure, CDV devra simuler le comportement vibratoire de l'assemblage afin de vérifier le bon respect de son référentiel GMP. Si la simulation n'est pas concluante, les pièces concernées devront être modifiées.

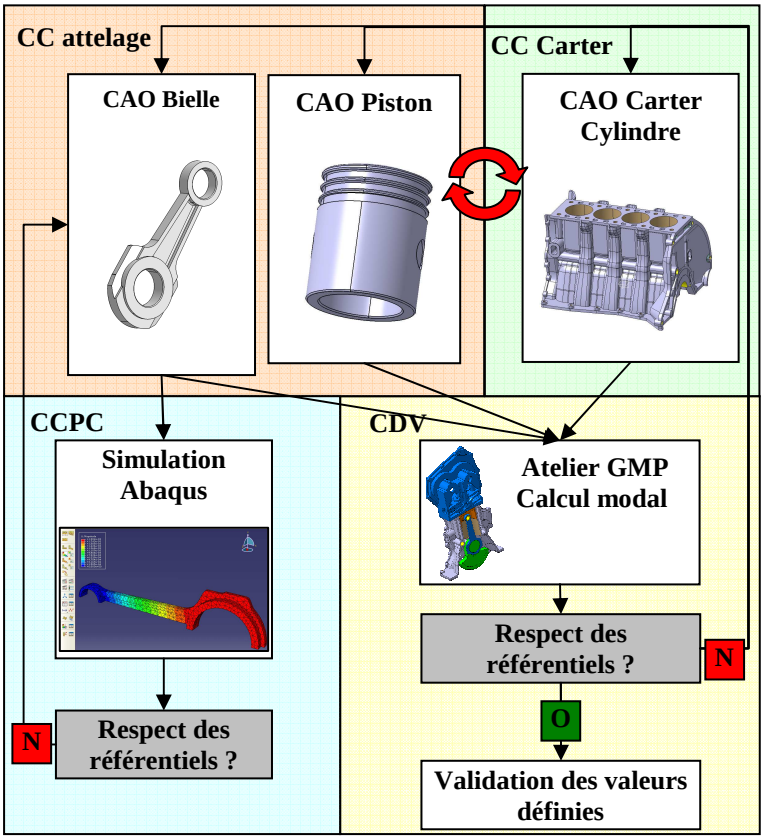


Figure 108 : processus phase 2

A la fin de la phase 2, tous les principaux paramètres du moteur sont figés.

5.4.2 La démarche et la réalisation

Dans le KnowledgeDomain "GMP" (le domaine regroupant l'ensemble des connaissances pour la conception de GMP), le chef de projet crée un nouveau projet moteur en récupérant un canevas projet contenant déjà une arborescence de phases et d'activités.
Pour piloter la première phase du projet, il crée une configuration squelette associée à la première phase et sélectionne les ICEs à utilisées dans le cadre de ce jalon. Ces ICEs sont instanciées dans la partie référence de la configuration squelette et le chef de projet value certain paramètres (Figure 109).

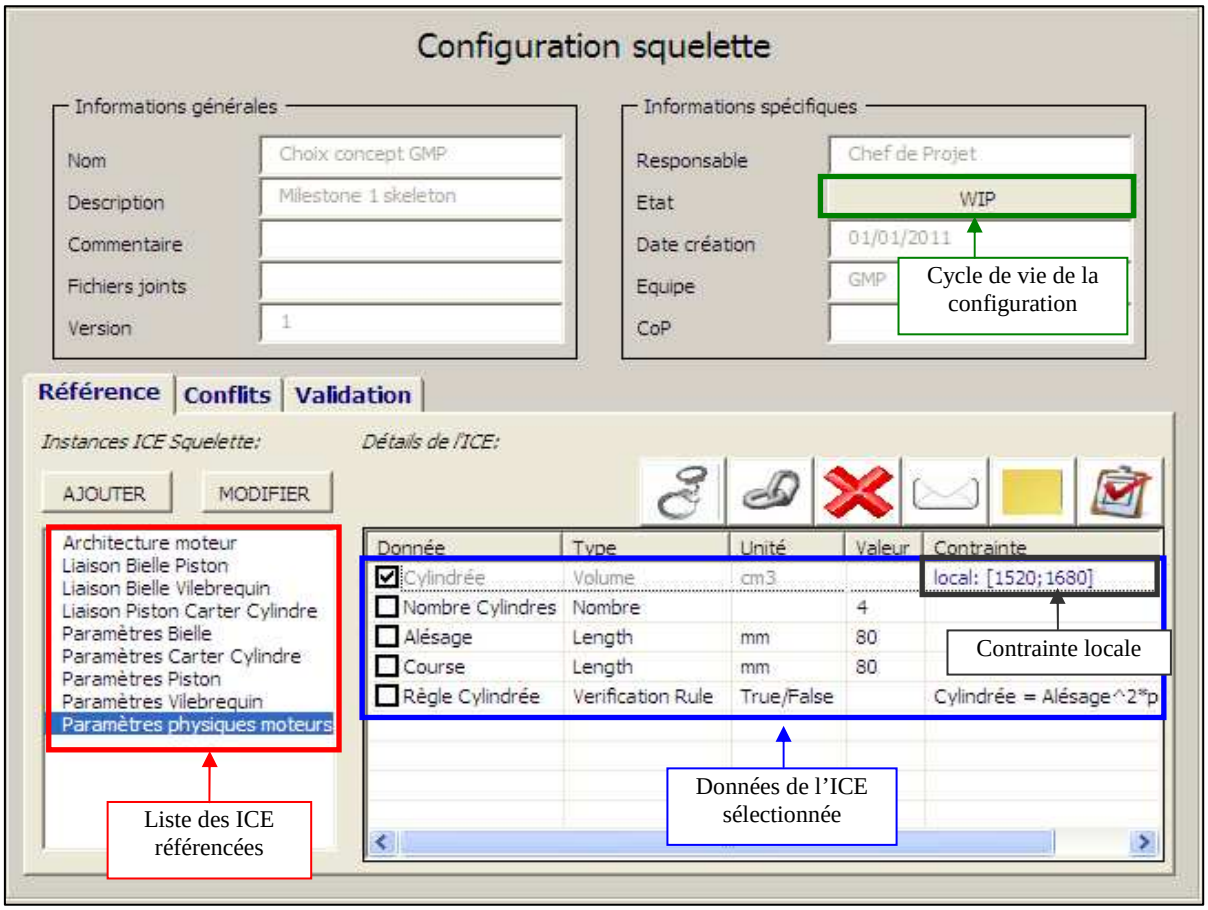


Figure 109 : configuration squelette, section « Références »

Ce nouveau projet moteur est assez similaire à un autre projet sur lequel le chef de projet à déjà travaillé. Fort de cette expérience, il ajoute des contraintes locales sur certain paramètres, comme par exemple, sur le paramètre "Cylindrée" appartenant à l'ICE "Architecture moteur" (Figure 109 : contrainte locale de type condition limite [1520 ; 1680]).

Ensuite, sur l'ICE "Paramètres physiques moteurs", il désactive le paramètre "couple maxi" (Figure 110). En effet, "couple maxi" ne peut pas être défini à ce stade du processus de conception, il est donc désactivé et sera réutilisé ultérieurement.
Enfin, le chef de projet fige les valeurs des paramètres issus du cahier des charges, de sorte à ce que les utilisateurs ne puissent plus modifier les valeurs lors de leurs utilisations dans les configurations utilisateur.

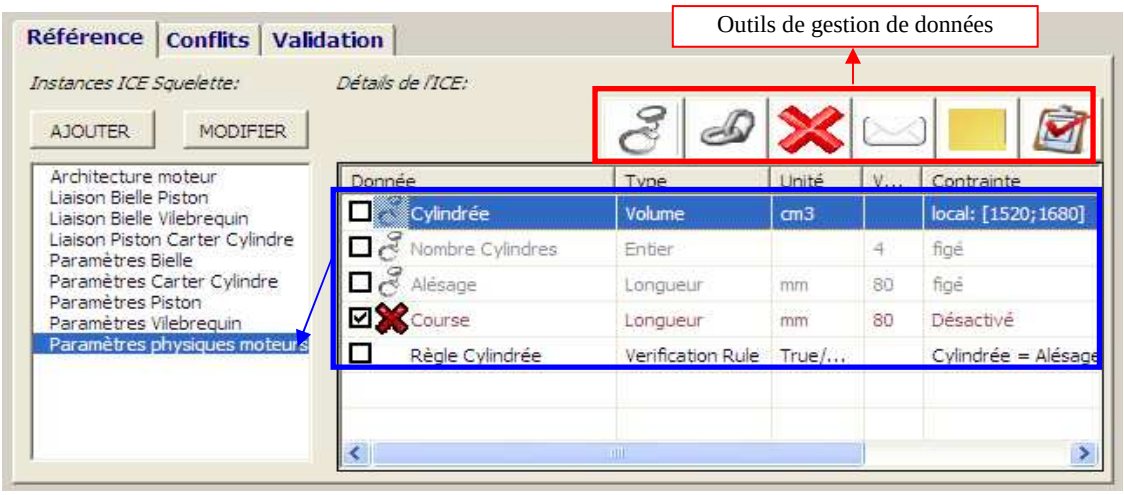


Figure 110 : désactivation du paramètre couple maxi de l'ICE Paramètre physiques moteurs

La configuration squelette ainsi préparée, les acteurs de la phase 1 se créent une configuration utilisateur associée à chaque activité du jalon 1 (Figure 111) et récupèrent les ICEs instanciées dans la partie référence de la configuration squelette. D'autres ICEs sont ajoutées depuis la base générique, ce qui à pour conséquence d'enrichir automatiquement la configuration squelette. Enfin, les utilisateurs synchronisent leurs configurations à leurs modèles métiers et commencent à travailler.

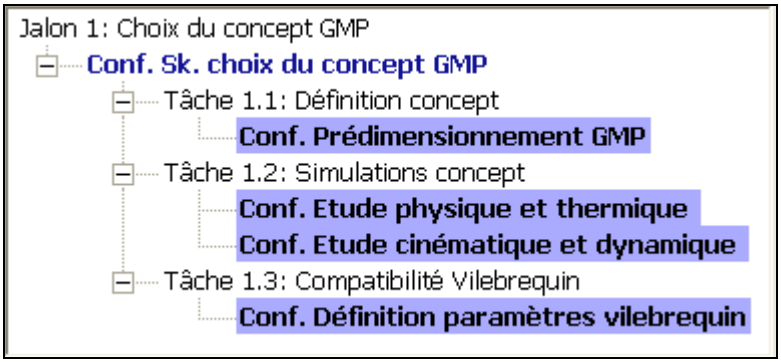


Figure 111 : organisation des configurations de la phase 1

Première phase :
CCPC travaille sur son fichier Excel pour déterminer les grandes tendances du moteur, et synchronise les paramètres résultants de cette étude avec la "Conf.Prédimensionnement GMP".
Ensuite, il doit réaliser une étude physique et thermique à l'aide d'un modèle métier 0D-1D. Pour ce faire, il réutilise dans sa "Conf.Etude physique et thermique", les instances d'ICEs avec les valeurs de paramètres définis dans son autre configuration utilisateu,r et synchronise le modèle 0D-1D. En parallèle, CDV débute une étude cinématique en utilisant le même modèle métier 0D-1D GMP que CCPC.
Des résultats issus des deux simulations sont ensuite réinjectées (via les configurations) dans le fichier Excel pour affiner certaines valeurs de paramètres.

En travaillant sur leur modèle métiers, les utilisateurs créent plusieurs versions de configurations correspondant à différentes architectures, concepts ou simulations. Quand il le souhaite, un acteur "publie" une version de sa configuration utilisateur dans la configuration squelette pour analyser la cohérence des connaissances avec les autres configurations utilisateurs publiées (e.g. bouton "Submit", Figure 112).

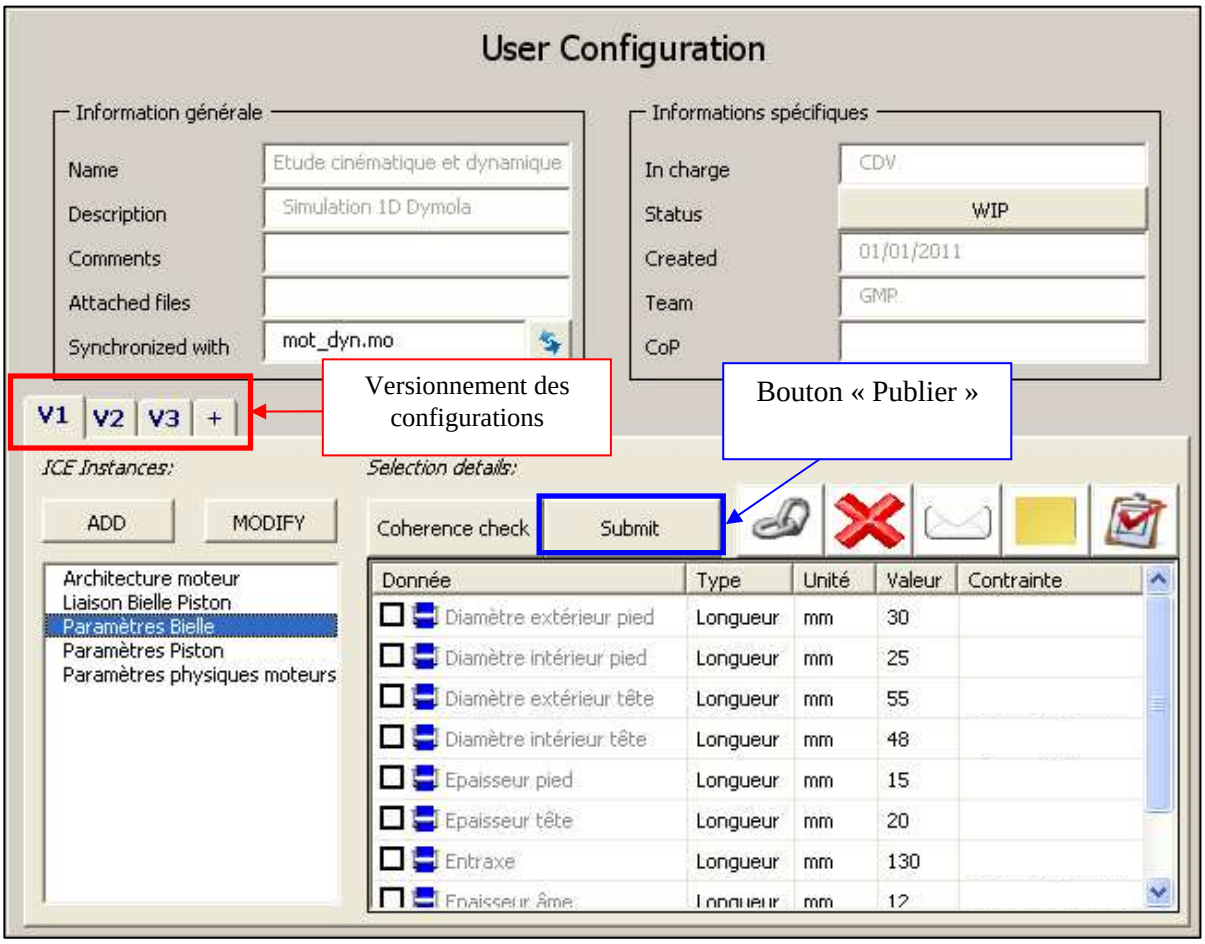


Figure 112 : version 1, Configuration Utilisateur de CDV, tâche 1.2

La configuration squelette affiche alors des conflits ente instances d'ICEs partagées dans plusieurs configurations. Afin de faciliter la convergence des utilisateurs vers une solution, des fonctionnalités, comme les Reviews ou encore la création de liens maitres/esclaves ou l'abonnement, leurs permettent une meilleure collaboration.

Ainsi, suite à des conflits détectés entre CDV et CCPC dans la phase 1 entre les deux tâches de simulation, CDV s'abonne à certaines ICEs définies par CCPC lors de l'étude physique et thermique. Grâce au système d'abonnement, il est averti, dans sa configuration et en direct, des modifications réalisées par CCPC et peut en tenir compte pour lancer de nouvelles simulations (Figure 113).

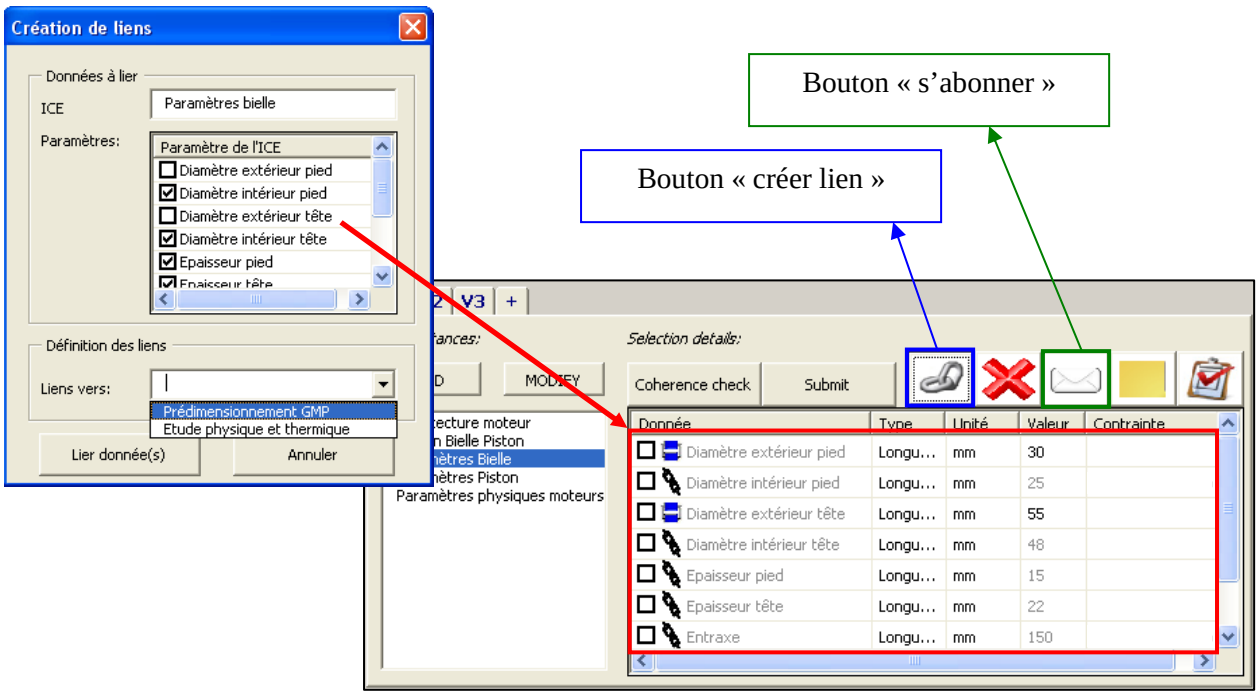


Figure 113 : utilisation du système d'abonnement entre deux configurations utilisateur

Suite à la collaboration des deux acteurs sur le choix de l'architecture moteur, CC attelage crée une configuration dans le but de vérifier si les données définies sont compatibles avec la conception d'un vilebrequin standard. Dans sa configuration, il récupère les instances d'ICEs à jour et cohérentes des deux configurations de CCPC et CVD. De plus, il instancie l'ICE "Liaison vilebrequin bielle" qui contient des règles de vérification lui permettant de valider la compatibilité du vilebrequin avec les dimensions de l'attelage mobile (Figure 114). Les trois contraintes de cette ICE utilisent les valeurs de paramètres issues des ICEs récupérées des configurations de CCPC et CVD, et sont validées. C'est-à-dire que les paramètres respectent les contraintes, validant ainsi la compatibilité de la bielle et du vilebrequin.

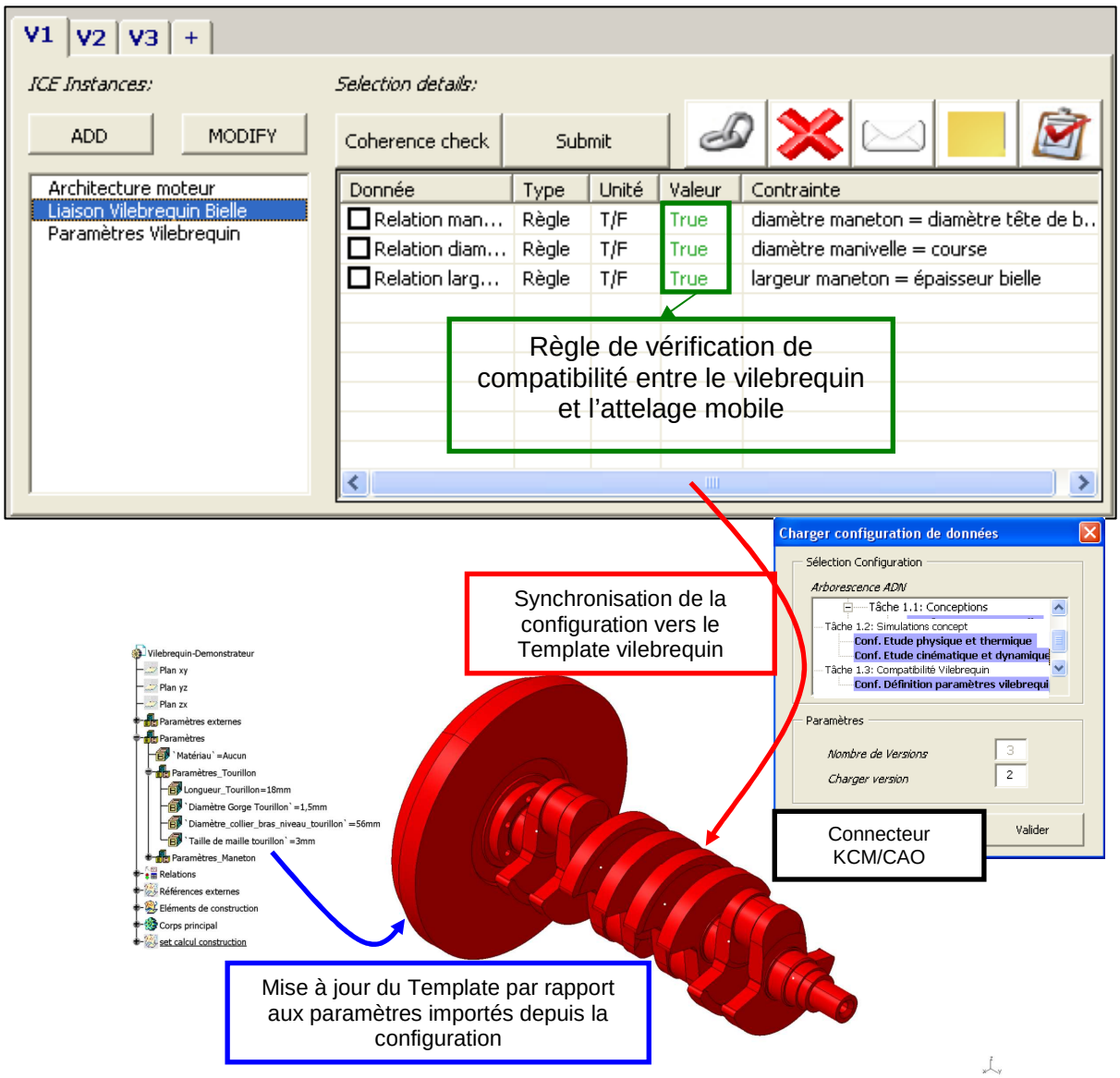


Figure 114 : configuration "Définition paramètres vilebrequin" synchronisée avec le Template vilebrequin

Ensuite, CC attelage synchronise sa configuration avec un modèle CAO dans lequel, il importe un Template de vilebrequin (UDF ou power-copy) standard dont la géométrie se met à jour automatiquement par rapport aux valeurs issues de la configuration utilisateur (Figure 114). Le Template se mettant correctement à jour, CC attelage en conclut que les principaux paramètres moteurs définis sont cohérents par rapport au référentiel de conception du vilebrequin sur lequel se base le Template géométrique.
L'architecture du moteur et les dimensions du vilebrequin étant définies, les acteurs métiers valident leur travail en modifiant le statut de leur configuration vers l'état "A valider", signifiant que leur activité est terminée (Figure 116).
Le chef de projet peut alors valider la configuration squelette et ainsi figer l'ensemble des configurations utilisateur de la phase, et donc l'ensemble des connaissances utilisées.

Deux conditions sont nécessaires pour valider la phase et satisfaire au jalon :

- Toutes les configurations utilisateurs doivent être "A Valider" ;
- La configuration squelette ne doit présenter aucun conflit (ou qu'ils soient négligés).

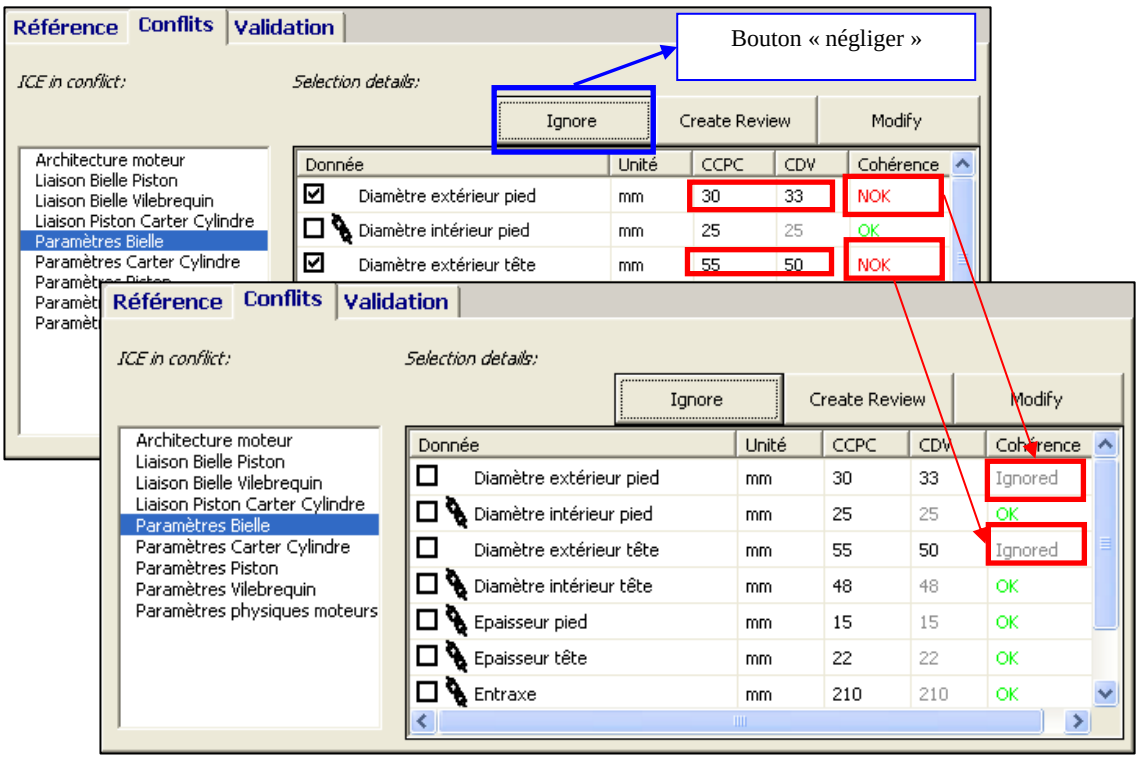


Figure 115 : conflits négligés, Section Conflit, Configuration Squelette

Or, la partie conflit de la configuration squelette met encore en évidence l'existence d'incohérences sur la valeur de deux paramètres utilisés par CCPC et CDV dans leur configuration respective, empêchant le chef de projet de valider la configuration squelette (Figure 115). Le chef de projet décide alors de négliger ces conflits dans le cadre de la phase 1 du projet ; les conflits concernent des données qui seront traitées ultérieurement en phase 2. Les ICEs contenant les conflits négligés passent alors dans la partie validation de la configuration squelette.

Enfin, le chef de projet valide la configuration squelette et crée une nouvelle version pour l'utiliser dans la seconde phase du projet (Figure 116). Sur cette figure, la nouvelle version de la configuration squelette est associée à la phase 2. Dans la version 2 de la configuration squelette, la partie validation (équivalent à conformité) de la configuration en version 1 est devenue la partie référence de la version 2. Ainsi, dans la seconde phase, les utilisateurs partent des connaissances utilisées et validées dans la phase précédente.

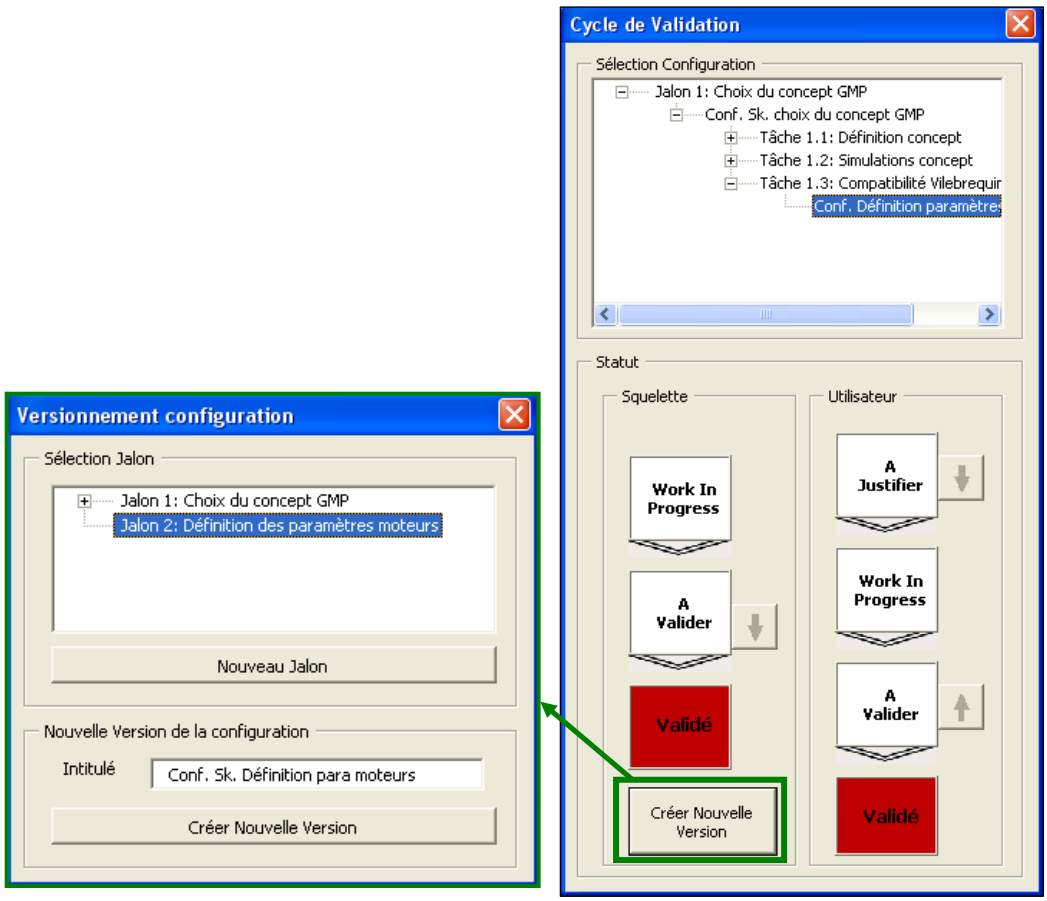


Figure 116 : cycle de validation des configurations

Seconde phase :

Au lancement de la deuxième phase, les acteurs du projet disposent de la configuration squelette en version 2 avec la partie référence construite à partir des résultats de la phase précédente. Par ailleurs, d'autres ICEs sont ajoutées relativement à la conception des composants du moteur. En effet, dans cette phase, les acteurs vont concevoir la bielle, le carter et le piston en s'appuyant sur des simulations (EF volumique) qu'ils vont réaliser itérativement, de sorte à orienter leur choix d'architectures (Figure 117).

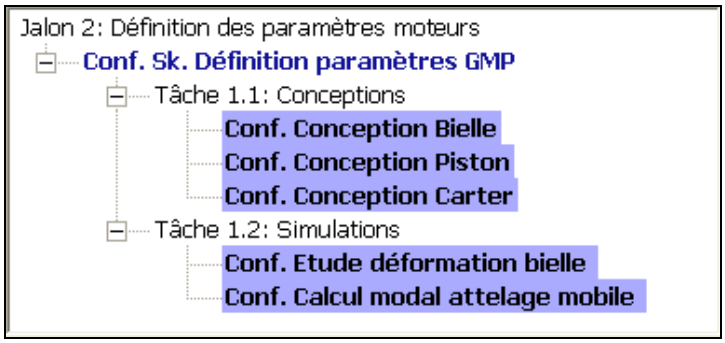


Figure 117 : organisation des configurations de la phase 1

Dans un premier temps, le chef de projet fige dans la configuration squelette certaines valeurs de paramètres préalablement définis dans la phase 1. Ces paramètres pourront être utilisés mais pas modifiés par les utilisateurs des configurations utilisateur.

Dans un deuxième temps, CC Attelage et CC Carter créent leurs propres configurations utilisateur à partir de la partie référence de la configuration squelette.

CC Attelage conçoit un modèle préliminaire de la bielle et du piston en CAO, en parallèle CC Carter s'abonne à certaines ICEs de CCAttelage (dans ses deux configurations) et définit également une conception préliminaire d'un carter cylindres.

Tous deux créent plusieurs versions de leur configuration et publient régulièrement dans la configuration squelette afin d'analyser les conflits, de manière à converger plus rapidement (le chef de projet guide ou impose aux concepteurs certains choix d'architectures).

Une fois une solution préliminaire définie et cohérente, CCPC et CVD vont, l'un après l'autre, créer leur configuration et utiliser des ateliers d'architectures (Annexe 1) afin d'optimiser la conception des composants du moteur (étude vibratoire de l'attelage mobile et étude en traction de la bielle). Ainsi, sur la base des conceptions préliminaires (récupération des instances d'ICEs), ils vont tester plusieurs variations des architectures afin d'optimiser les composants (export des valeurs de paramètres de la configuration dans leur modèles et mise à jour de la CAO simplifiée).
Suite aux itérations de simulation, CVD demande à CCAttelage de modifier la bielle car elle ne supporte pas les sollicitations en traction. Il doit alors modifier le diamètre extérieur tête et pied (conflits négligés en phase 1 mais traité en phase 2), ainsi que la longueur de la bielle.
Ces modifications de la "Conf.Conception bielle" entrainent l'apparition de conflits lors de sa publication dans la configuration squelette.
En effet, ces modifications doivent maintenant être prises en compte et propagées dans la "Conf.Conception piston" et dans la "Conf.Conception carter". Ainsi, ces configurations sont modifiées et les modèles CAO sont synchronisés et mis à jour (Figure 118).

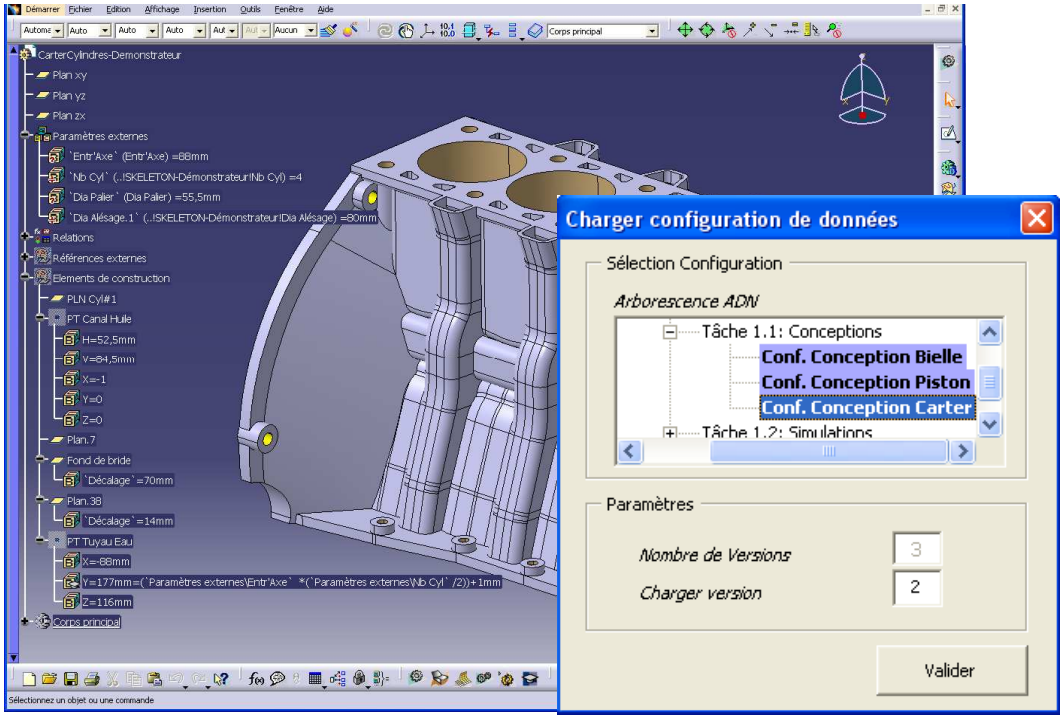


Figure 118 : mise à jour du modèle de conception du carter cylindres par rapport aux modifications de la configuration utilisateur

Après ces itérations, les concepteurs ont tous publié leur configurations et les connaissances sont cohérentes. CCPC et CVD récupèrent les nouvelles valeurs de paramètres via le système d'abonnement et lancent de nouvelles simulations pour valider la nouvelle conception.
Les simulations étant concluantes, tous les utilisateurs valident leurs configurations utilisateur. Le chef de projet s'assure que l'ensemble des conflits sont résolus (notamment les conflits négligés lors de la première phase) et que toutes les configurations utilisateurs sont en statut "A Valider". Ensuite, il valide la configuration squelette.

Dans la phase 3, les concepteurs devront reprendre leurs modèles de composants du moteur pour modéliser finement la bielle, le carter, les pistons et le vilebrequin (avec réutilisation de tout ou partie de la configuration squelette Version 2), et ainsi poursuivre le processus de conception qui finalement, les mènera à la fabrication d'un prototype.

5.4.3 Constat GMP

Cette expérimentation met en évidence l'utilisation d'ADES dans le cadre d'un projet impliquant plusieurs utilisateurs et de nombreux modèles hétérogènes pour la conception d'un GMP.
L'expérimentation met l'accent sur la collaboration entre les acteurs autour des principaux concepts issus de l'approche KCM.
Le constat est satisfaisant et met en évidence la facilité d'échange des connaissances au travers de différents modèles hétérogènes. Par exemple, dans la première phase, CCPV met en corrélation son fichier Excel et son modèle de simulation 0D-1D via les configurations de connaissances. Cette collaboration est rendue possible via l'expression unifiée et centralisée des connaissances dans ADES.

De plus, l'utilisation des fonctionnalités visant à favoriser la collaboration de type abonnement, lien maitre/esclave, etc., permet de mettre en évidence le fait que les modèles métiers ne sont plus les "éléments centraux", siège des décisions et des discussions entre les acteurs. Au contraire, les modèles ne sont que des outils, et les décisions et les compromis sont discutés à partir des connaissances centralisées dans ADES.

5.5 Conclusion

Dans ce chapitre, nous avons présenté deux expérimentations de l'approche KCM mettant en scène la maquette ADES pour la conception d'un support de fixation, puis ensuite pour la conception d'un GMP en collaboration avec PSA Peugeot Citroën. Ainsi, nous pouvons conclure quant à la pertinence de notre approche et élaborer une critique qui nous permettra de définir les perspectives à envisager.

Les résultats des expérimentations démontrent l'intérêt du concept d'ICE et de KnowledgeConfiguration, permettant d'assurer la centralisation, la traçabilité, l'affichage des conflits et une meilleure réutilisation des connaissances dans un projet de conception.
Par rapport à un processus classique, ADES permet d'améliorer significativement la collaboration entre les acteurs et l'interopérabilité des modèles métiers. Ainsi, le KCM devient le centre névralgique de la collaboration permettant aux utilisateurs d'échanger en temps réel, de prendre des décisions et d'appliquer ces choix directement sur leurs modèles métiers. Ainsi, on souligne l'intérêt de l'aspect dynamique dans l'utilisation d'ADES et son importance lors des phases amont dans le processus de conception.
Cet apport est très important, dans la mesure ou le nombre d'incohérences non identifiées ou identifiées tardivement dans le processus de conception, est considérablement réduit. Cela permet aux utilisateurs de cibler avec plus d'efficacité l'origine des problèmes et potentiellement de converger plus vite vers de nouvelles solutions.

Parmi les difficultés rencontrées, la première vient de l'appréhension et de l'appropriation des concepts par les utilisateurs. Ainsi, il est apparu comme essentiel de bien séparer (dans l'IHM) les différents concepts par rapport à leurs utilisations et leurs objectifs.
Par exemple, séparer les ICEs dont l'objectif est de capitaliser de façon générique l'ensemble des paramètres et des contraintes, de l'utilisation des configurations utilisant des instances d'ICEs qui sont représentatives des connaissances utilisées dans les modèles métiers (la notion d'ICEInstance n'est pas visible pour l'utilisateur). De la même manière, il est important de bien expliquer les différences et le processus d'utilisation de la configuration squelette par rapport aux configurations utilisateur.
En somme, bien veiller à créer un "espace capitalisation et base d'information" (pour les ICEs), un "espace collaboration, gestion et suivi" (configurations squelette), ainsi qu'un "espace opérationnel" (configurations utilisateur). Par ailleurs, il semble indispensable de renomer les concepts avec des noms plus familiers des concepteurs que les termes d'ICE ou de KC.
Liés à cette première difficulté, viennent s'ajouter les notions de droits et de rôles très peu implémentés dans ADES qui perturbent les utilisateurs. En effet, savoir qui a le droit de faire quoi et comment permettre aux utilisateurs de structurer leurs actions et de se positionner les uns par rapport aux autres.
De plus, il n'existe pas aujourd'hui de processus de collaboration clairement définis. Les utilisateurs ont une liberté totale quant à la publication de leur configuration vers la configuration squelette par

exemple. Un léger sentiment de flou a été ressenti par les utilisateurs souhaitant être plus encadrés par un processus commun de collaboration, les incitant à publier, synchroniser, communiquer, décider, etc. à différents moment identifiés.

La gestion des conflits sur les contraintes a soulevé quelques interrogations. En effet, dans ADES la gestion des conflits est limitée et porte sur la comparaison de paramètres entre configurations utilisateur, ainsi que sur la vérification des contraintes en fonction des valeurs de paramètres entrées par les utilisateurs.
Aucune fonction de propagation n'est implémentée, or l'utilisation de mécanismes de propagation aurait pu permettre aux utilisateurs de converger plus vite sur certaines solutions techniques tout en assurant le respect des contraintes.
En effet, face aux possibilités de collaboration offertes par le KCM, le souhait logique des utilisateurs serait d'avoir des outils leur permettant de les orienter dans les décisions à prendre face à un conflit, voire détecter le conflit avant même qu'il ne survienne. Ces thématiques touchent directement les domaines de recherche issus de l'intelligence artificielle.

Enfin, si les scénarios d'expérimentation mettent davantage l'accent sur la collaboration et l'utilisation des configurations de connaissances, il a fallu préalablement construire la base d'ICEs génériques. Cette étape délicate a demandé un temps de discussion et de réflexion avec plusieurs retouches sur les ICEs créées.
Afin d'optimiser le temps nécessaire à la mise en place d'une base préliminaire, il sera intéressant de définir une méthodologie afin d'aider les organisations à identifier et structurer leurs connaissances.

A partir des résultats obtenus et des retours recueillis, suite à ces expérimentations, nous proposons, dans le chapitre suivant, un bilan et une critique de l'approche KCM pour la gestion des connaissances du produit en configurations. En effet, si nos expérimentations semblent concluantes et encourageantes, dans l'optique du développement d'une solution logicielle complète et de son déploiement, des incertitudes persistent et demandent de plus amples investigations. Ainsi, nous proposons également dans le dernier chapitre de ce manuscrit, une discussion sur les perspectives à donner à ces travaux.

Chapitre 6. Conclusion et perspectives

6.1 Conclusion générale

Dans le contexte économique difficile que nous connaissons, il est aujourd'hui admis que la collaboration des acteurs autour des technologies logicielles, exploitant des données, information et connaissances, apparaît comme un des aspects fondamentaux du processus de conception de produits, abordé dans le cadre de notre problématique de recherche.

Ainsi, les principaux résultats de notre travail de recherche se structurent autour de trois axes :

- Une méthodologie de gestion des connaissances en configurations qualifiée de KCMethod.
- Un méta-modèle, baptisé KCModel, de structuration des concepts manipulés par KCMethod.
- Une maquette de faisabilité sous forme d'outil logiciel ADES, permettant d'expérimenter et valider notre approche.

L'ensemble des résultats obtenus s'articule autour d'une solution logicielle de nouvelle génération, qualifiée de KCManager, permettant de déployer en entreprise l'ensemble de la démarche proposée (Figure 119).

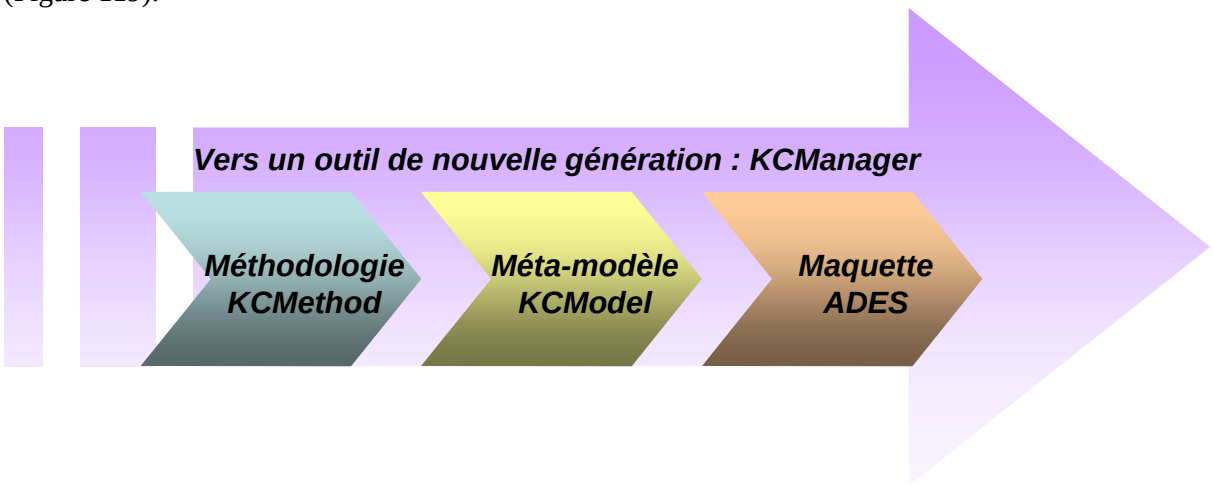


Figure 119 : les principaux résultats de ces travaux de recherche

Ainsi, la méthodologie KCMethod définit un ensemble de concepts permettant la manipulation et la gestion des paramètres et des contraintes, considérés comme des connaissances d'un niveau granulaire fin, et ce, indépendamment des outils métiers. Sa formalisation en méta-modèle, tenant compte des particularités de structuration des connaissances et du formalisme MDA, permet de définir une solution générique et flexible. Enfin, les expérimentations réalisées au sein de la société DPS, en collaboration étroite avec de grands groupes industriels, tel que PSA – Peugeot Citroën ou encore EADS, ont permis de valider les choix et les orientations prises, ainsi que le positionnement de cette approche par rapport aux outils et méthodes traditionnels.
Notre contribution de recherche apporte donc une réponse à la problématique de collaboration au niveau des connaissances de granularité fine, partagées simultanément entre modèles métiers au niveau du couple produit-simulation, en offrant des solutions de capitalisation, de traçabilité, de réutilisation et de vérification de la cohérence.

En particulier, notre approche est de nature à favoriser, la collaboration et l'interopérabilité, dans la mesure où les connaissances sont gérées indépendamment des applications logicielles qui les utilisent. Ainsi, au travers du concept d'ICE et de KnowledgeConfiguration, les connaissances disposent d'une organisation unique compréhensible par n'importe quel acteur, quel que soit son point de vue. Cette gestion autorise la mise en place de processus de gestion de cycles de vie, de validation et de contrôle spécifiquement adaptés. Ces concepts favorisent la réutilisation des connaissances au sein d'un projet de conception ou entre plusieurs projets, tout en tenant compte de la notion de traçabilité et de gestion

de configurations. La détection des conflits constitue également un élément essentiel de notre démarche, dans la mesure où elle évite de nombreuses erreurs conduisant souvent à des problèmes de qualité et à des pertes de temps au cours du processus de conception.

La méthodologie et le méta-modèle proposés sont donc complémentaires des outils de gestion de données et des modèles produits classiques de la littérature car ils permettent une meilleure circulation des informations dans le processus de conception. Le couplage du KCModel et d'un modèle produit issu de la littérature scientifique, tel que par exemple le modèle produit MD-MV, constitue une perspective de recherche intéressante parmi les différents axes potentiels que nous proposons dans la section suivante.

6.2 Perspectives de recherche

De nombreuses perspectives sont envisageables suite à la proposition formulée dans le cadre de ces travaux de recherche. Ainsi, afin d'offrir un maximum d'efficacité dans le processus de conception, nous proposons quatre axes, qui selon nous, constituent les principaux efforts à fournir dans le cadre du développement d'une solution logicielle de type KCManager :

- Quantifier les gains réalisés suite à l'application de notre approche dans le processus de conception et identifier les difficultés d'utilisation.
- Intégrer des approches d'aide à la décision pour guider les concepteurs dans leurs choix face aux conflits.
- Réaliser un couplage avec un modèle produit traditionnel.
- Définir s'il est intéressant de travailler à l'élaboration d'un modèle d'un niveau d'abstraction supérieur au KCModel, qui serait capable de générer plusieurs modèles de connaissances.

Concernant le premier axe, il s'agit de quantifier les gains réalisés dans le processus de conception, suite à la mise en place de notre démarche dans une entreprise. Cette tâche nécessite le déploiement de l'approche KCM dans le processus de conception à une échelle suffisante, pour observer ses effets sur le rendement des utilisateurs. Il faudra alors définir des critères d'observation, des indicateurs, voire des procédures de mesures, capables de quantifier les gains sur un ou plusieurs projets complets. De plus, de part les spécificités de chaque entreprise, cette étude devra être menée dans plusieurs structures et dans des domaines différents (par exemple, automobile et aéronautique).
De plus, il serait intéressant de déployer l'approche KCM à plus grande envergure pour identifier des problèmes encore non déclarés. En effet, même si les expérimentations menées dans le cadre de nos travaux de recherche, semblent prometteuses et permettent de valider notre démarche, nous avons néanmoins identifié certains points pouvant potentiellement poser problèmes lors d'un déploiement réel en entreprise :

- la stabilité de l'application à long terme,
- les difficultés de mise en place de l'outil,
- la réaction des utilisateurs vis-à-vis de l'outil.

Le nombre relativement faible d'ICEs et KnowledgeConfigurations manipulés lors de nos expérimentations n'a pas permis d'exploiter au maximum les possibilités de réutilisation des connaissances. De fait, se pose alors la question de la stabilité de l'application KCManager, dans le cas ou elle manipulerait plusieurs milliers de paramètres, de contraintes (donc d'ICEs) et plusieurs centaines de configurations.
Un autre point bloquant concerne les difficultés de mise en place de l'approche, et plus particulièrement la première définition de la base d'ICEs. En effet, nous avons constaté que la définition de cette base nécessite une réflexion profonde avec les utilisateurs, afin d'identifier les informations jugées cruciales et la façon de les structurées dans les ICEs. Il serait alors préférable de définir une méthodologie spécifique permettant de faciliter cette tâche essentielle.
Enfin, la réaction des utilisateurs face à notre proposition est un élément essentiel, pour optimiser les gains dans le processus de conception. Dans le cadre de nos expérimentations, les utilisateurs avaient été sensibilisés aux problèmes de collaborations. Qu'en sera-t-il en contexte projet réel, comment serait accueilli l'outil KCManager ?

La quantification des gains, relative à l'utilisation de la démarche KCM, permettra alors d'affiner notre proposition et d'inclure de nouvelles fonctionnalités, comme la gestion des hypothèses de configurations de connaissances, sur laquelle nous travaillons déjà.
Une fois ces investigations menées, nous pourront également statuer quant à la possibilité de déployer notre approche à d'autres processus d'ingénierie, comme la fabrication, ou d'autres secteurs, comme l'énergie, le bâtiment, voire à d'autres domaines que celui de l'industrie, comme la finance.

Le second axe concerne l'intégration et le développement de systèmes d'aide à la décision dans l'approche KCM.
En effet, pour cette première étape que constitue ces travaux de recherche, nous nous sommes focalisés sur la structure d'organisation et de gestion des donnés, informations et connaissances en configurations. Ainsi, nous proposons un moyen de visualiser les conflits résultant d'incohérences entre plusieurs activités de conception et de simulation dans un contexte collaboratif. En plus de cette information sur les conflits, nous proposons également un cadre d'utilisation très applicatif et générique des CSP (Constraint Satisfaction Problem) pour permettre aux utilisateurs de mieux prendre en compte les contraintes et de les propager sur les valeurs de paramètres impliqués. L'application d'un CSP n'est d'ailleurs pas complètement opérationnelle et demande de plus amples investigations.
Il s'agira par la suite de proposer des mécanismes d'aide à la décision poussés, qui permettront aux utilisateurs de résoudre les conflits en se basant sur des expériences passées, ou d'éviter les conflits avant qu'ils ne surviennent. Ainsi, nous envisageons le recours à des approches telles que la logique floue, le retour d'expérience, ou encore l'utilisation de systèmes multi-agents.

Le troisième axe concerne le couplage avec un modèle produit. En effet, les approches classiques et le KCModel semblent complémentaires. Ainsi, il faudrait définir les protocoles et les processus d'échanges entre les modèles et mesurer l'impact sur la collaboration des acteurs et l'interopérabilité entre les modèles métiers.
Ceci permettrait notamment, de combler un manque dans notre démarche, dans la mesure où il nous semble que l'utilisation d'une approche seule, basée sur la gestion des configurations de connaissances, traduit un defaut d'intention de conception. Le couplage avec un modèle produit tel que le MD-MV pourrait compléter cette absence.

Enfin, le quatrième axe envisagé, concerne des réflexions portant sur la définition d'un modèle à un niveau d'abstraction supérieur au KCModel (un méta-méta-modèle). Un tel modèle serait en mesure de générer le KCModel, ou tout autre modèle de connaissances du produit pour l'amélioration de la collaboration. Ce modèle serait constitué de concepts génériques et abstraits, définissant un langage capable de tenir compte de n'importe quel type de connaissances lors de processus collaboratifs. Ainsi, avec une bonne maitrise de ce langage, il serait aisé d'explorer de nouvelles façons de manipuler les connaissances.

Certains de ces axes de recherche sont d'ors et déjà en cours d'études, notamment au sein du consortium du projet ADN.

Bibliographie

A

[Adeli, 2001 a] Lettre ADELI (2001), Gestion de Configuration Logicielle, Guide méthodologique partie 1, pp. 13-34, Juillet 2001.

[Adeli, 2001 b] Lettre ADELI (2001), Gestion de Configuration Logicielle, Guide méthodologique partie 2, pp. 13-34, Octobre 2001.

[AFNOR, 1994] AFNOR (Association Française de NORmalisation) : http://www.afnor.fr/.

[AFROR 2002] AFNOR (Association Française de NORmalisation), (2002), Outils de management - Maîtrise du processus de conception et développement, indice de classement, X50-127, Fascicule de documentation, http://www.afnor.fr/.

[Agard, 2002] Agard B., (2002), Contribution à une méthodologie de conception de produits à forte diversité, Thèse de doctorat, Institut National Polytechnique de Grenoble.

[Aidi, 2007] Aidi M., (2007), Vers la planification des buts de simulation en conception dans une démarche d'ingénierie système, Thèse de doctorat, LASEM (Laboratoire des Systèmes Électromécaniques) ENIS –US- Tunisie & GILCO (Gestion Industrielle, Logistique et Conception) ENSGI–INPG –France.

[Aifaoui et al. 2004] Aifaoui N., Benamara A., Dogui A., (2004), Interopérabilité des processus de conception et de calcul : une approche basée sur les features de calcul, Journal Européen des Systèmes Automatisés – JESA, Vol. 38, No. 3-4, pp. 443-464.

[Aldanondo et al., 2010] Aldanondo M., Vareilles E., Djefel M. (2010), Towards an association of product configuration with production planning, INTERNATIONAL JOURNAL OF MASS CUSTOMISATION, Vol. 3, No. 4, pp. 316-332.

[Amann, 2002] Amann K., (2002), Product lifecycle management: empowering the future of business, CIM Data, Inc.

[Ammar-Khodja et al.,, 2005] Ammar-Khodja S., Bernard A., (2005), Déploiement d'une démarche de Knowledge Management dans le cadre d'un projet de spécification d'un outil KBE CPI'2005 – Casablanca, Morocco.

[Amann, 2010] Amann K., (2010), The Growing Importance of Simulation & Analysis in Product Development, CIMDATA, FR2010 – NAFEMS, Paris, France.

[Anderl et al., 2002] Anderl R., Kleiner S. et Krastel M., (2002), Product Data Management in Simulation and Calculation, Product Data Journal, No.1/2002, pp. 37-41.

[Andreasen et al., 1985] Andreasen M., Hein L., (1985), Integrated Product Development, IFS Springer-Verlag.

[Aoussat, 1990] Aoussat A., Le Coq M., (1998), Contraintes d'assemblage, In Tollenaere M., Conception de produits mécaniques : méthodes, modèles et outils, Paris, Hermès, ISBN 2-86601-694-7, pp. 185-200.

[Arana et al., 2000] Arana I., Ahriz H., Fothergill P., (2000), Redesign knowledge analysis, representation and reuse Industrial knowledge management: a micro-level approach Springer-Verlag, London pp. 139-146.

[Arbouy et al., 1994] Arbouy S., Bezos A., Cutting-Decelle F., Diakonoff P., Germain Lacour P., Letouzey J-P, Viel C., (1994), STEP, published by AFNOR.

[Assouroko et al., 2009] Assouroko I., Ducellier G., Eynard B., Boutinaud P., (2009), Processus d'ingénierie numérique et intégration Conception-Calcul, 19ème congrès français de mécanique, Marseille.

B

[Badin et al., 2009] Badin J., Troussier N., Gomes S., (2009), Vers de nouvelles méthodes et outils pour la simulation multi-physiques à base de connaissances métiers, Virtual PLM09 Micado, Charleville Mézières.

[Badin et al., 2010] Badin J., Monticolo D., Chamoret D., Gomes S., (2010), Moving toward a product model to manage data, information and knowledge for mechanical design and simulation, Knowledge Acquisition, Reuse and Evaluation (KARE 2010) December 15th - December 18th, 2010, Kuala Lumpur, Malaysia.

[Badin et al., 2011] Badin J., Monticolo D., Chamoret D., Gomes S., (2011), Knowledge Configuration Management for product Design and Numerical Simulation, International Conference on Engineering Design, ICED11, 15 - 18 august 2011, Technical University of Denmark.

[Baizet, 2004] Baizet Y., (2004), La gestion des connaissances en conception-Application à la simulation numérique chez Renault-DIEC, Thèse de doctorat du laboratoire 3S de Grenoble, France.

[Barthès, 1997] Barthès J-P. A., (1997), Capitalisation des connaissances et intelligence artificielle, Journées Franco Finlandaises de Tampere 9-10 juin.

[Bathe, 1996] Bathe K., (1996), Finite element procedure, Prentice-Hall.

[Belytschko, 2000] Belytschko, T., Liu W. K., Moran B., (2000), Nonlinear Finite Elements for Continua and Structures, John Wiley & Sons.

[Bender et al., 2000] Bender S., Fish A., (2000), The transfer of knowledge and the retention of expertise: the continuing need for global assignments, Journal of Knowledge Management, Vol.4, No.2, pp.125-137.

[Benhafid, 2005] Benhafid Y., Troussier N., Boudaoud N., Cherfi Z., (2005), Méthode d'aide à l'idéalisation de modèles issus de la CAO pour le calcul de structures; AFM, EDP Sciences 2005; DOI: 10.1051/meca:2005032; Mécanique & Industries 6, pp. 289–295.

[Benhabib, 2003] Benhabib B., (2003), Manufacturing: Design, Production, Automation, and Integration, Marcel Dekker, New York, USA.

[Bergsjö et al., 2007] Bergsjö D., Burr H., Malvius D., Mueller M., Vielhaber M., (2007), Product Lifecycle Management for cross-x engineering design, ICED'07, Paris.

[Blackburn et al., 2007] Blackburn P., Bos J., Streignitz K., (2007), PROLOG tout de suite, College Publications.

[Blessing 1994] Blessing L.T.M., (1994), A process-based approached to computer-supported engineering system, Thèse de doctorat, University de Twente, Enschede, The Netherlands.

[Bekhti et Matta, 2003] Bekhti, S., et Matta, N., (2003), A Formal Approach to Model and Reuse the Project Memory, Journal of Universal Computer Science (J.UCS), Vol. 6, No. special, pp. 12-22.

[Blessing, 1996] Blessing L. T., (1996), Design process capture and support, in 2nd WDK Workshop on Product Structuring, Delft, pp. 109-121.

[Beylier, 2007] Beylier C., (2007), Une approche collaborative de gestion des Connaissances, Thèse de doctorat, Université Joseph Fourier Grenoble, France.

[Bidal et al., 2002] Bidal C., Bordier P.E., Brethous G., Caumes S., (2002), knowledge management, Dossier sur la Capitalisation de Connaissance, EPITA.

[Bocquet, 1998] Bocquet J-C., (1998), Ingénierie simultanée, conception intégrée, Conception de produits mécaniques : méthodes, modèles et outils, sous la direction de M. Tollenaere, Editions Hermès, pp. 29-52.

[Boudichon, 1994] Boudichon P., (1994), L'Ingénierie Simultanée et la gestion d'Informations, Hermès, Paris.

[Boujut et al, 2003] Boujut J.F., Blanco E., (2003), Intermediary Objects as a Means to Foster Co-operation in Engineering Design, Journal of Computer Supported Collaborative Work, Vol. 12, No. 2.

[Bouras et al., 2005] Bouras A., Gurmoorthy B., Sudarsan R., (2005), Product Lifecycle Management: Emerging solutions and challenges for Global Networked Entreprise, PLM-SP1, and ISBN 0-907776-18-3.

[Briand, 2001] Briand H., Guillet F., (2001), Extraction des Connaissances et Apprentissage, Vol. 1 : extraction et gestion des connaissances, Hermès Science Publications.

[Brissaud et al., 1996] Brissaud D., Garro O., (1996), An Approach to Concurrent Engineering using Distributed Design Methodology, Concurrent Engineering: Research and Applications, Vol. 4, No. 3, pp. 303-311.

[Brown et al., 1985] Brown D. C., Chandrasekaran B., (1985) Expert systems for a class of mechanical design activity, in J.S. Gero (ed.), Knowledge Engineering in Computer-Aided Design, North-Holland, Amsterdam.

[Budinsky et al., 2003] Bundinsky F., Steinberg D., Ellersick R. (2003), Eclipse Modeling Framework : A Developer's Guide. Addison-Wesley Professional, pp 18, 36, 39, 43, 90, 154.

[Burman, 1992] Burman R., (1992), Designing for minimum lead times, Automotive Engineering, Vol. 17, No. 2, pp. 61-65.

[Burr et al., 2005] Burr H., Vielhaber M., Deubel T., Weber C., Haasis S., (2005), CAx/engineering data management integration: enabler for methodical benefits in the design process, Journal of Engineering Design Vol. 16, No. 4, August 2005, pp. 385–398.

C

[CIMDATA, 2001] CIMDATA, (2001), Collaborative Product Definition management (cPDm): An Overview of a Collaborative Approach to Managing the Product Definition Lifecycle », CIMdata Inc, Ann Arbor, http://www.cimdata.com.

[Chapa, 1997] Chapa E., (1997), Outils et structure pour la coopération formelle et informelle dans un contexte de conception holonique, thèse de doctorat de l'Institut National Polytechnique de Grenoble, France.

[Chapman et al., 2001] Chapman C. B., Pinfold M., (2001), The application of a knowledge based engineering approach to the rapid design and analysis of an automotive structure, Original Research Article Advances in Engineering Software, Vol. 32, Issue 12, pp. 903-912.

[Charles et al., 2005] Charles S., Ducellier G., Li L., Eynard B., (2005), Improvement of 3D Data Exchanges in the Product Lifecycle Management, International Conference on Product Lifecycle Management – PLM'05, Lyon, France, July 11-13.

[Charles, 2005] Charles S., (2005), Gestion Intégrée des données CAO et EF - Contribution à la liaison entre conception mécanique et calcul de structures, Thèse de doctorat, Université de Technologie de Troyes.

[Christmas, 2001] Christmas A., (2001), CAD/CAM and PLM, Modern Machine Shop 2001.

[Clautrier, 1991] Clautrier M., (1991), Difficultés de nouvelles approches de conception dans le spatial, Séminaire GSIP, Méthode et outils pour la conduite de projet, Grenoble.

[Combemale, 2008] Combemale B., (2008), Approche de méta-modélisation pour la simulation et la vérification de modèle, Thèse de doctorat, Université de Toulouse, Institut National Polytechnique de Toulouse, France.

[Comsol, 2004] (2004), Couplage Multi Physique: Simuler le monde réel, source COMSOL, CAD. Magazine, No. 121, pp. 47 – 52

D

[Davenport et al., 1998] Davenport T., Prusak L., (1998), Working Knowledge: How organizations manage what they know, Harvard Business School Press.

[DEKLARE, 1995] Consortium DEKLARE (1995), DEKLARE Small Book – ESPRIT Project 6522, Final project report, CEE.

[Demoly, 2010] Demoly F., (2010), Conception intégrée et gestion d'informations techniques : application à l'ingénierie du produit et de sa séquence d'assemblage, Thèse de doctorat, Université de Technologie de Belfort-Montbéliard.

[Deneux, 2002] Deneux D., (2002), Méthodes et modèles pour la conception concourante, Habilitation à Diriger des Recherches, Université de Valenciennes.

[Devalan, 2009] Devalan P., (2009), Simulation numérique dans le processus de conception de systèmes mécaniques, Techniques de l'ingénieur, Référence BM5013.

[Dorville et al., 1996] Dorville G., Lecompte G., Serrafero P., Bourne C., (1996), Flange : assistant métier pour la conception optimale de liaisons à brides, Conférence Internationale IDMME'96 sur la Conception et la Fabrication Intégrées en Mécanique, pp. 335-343, Edition ECN, Nantes.

[DPS, 2010 a] Collard E., (2010), Dimensionnement de pièces de conduite d'hydrocarbures par une approche "Design by Analysis", FR2010 – NAFEMS, Paris, France.

[DPS, 2010 b] DPS : Digital Product Simulation (2010), plaquette de présentation de l'entreprise.

[Dreisbach, 2010] Dreisbach R., (2010), Leveraging Simulation for Competitive Advantage, BOEING Company, FR2010 – NAFEMS, Paris, France.

[Ducellier et al., 2006] Ducellier G., Charles S., Eynard B., Caillaud E., (2006), Rule based Specification for Collaborative Design, 5th International Conference on Advanced Engineering Design - AED 2006, Prague, Czech Republic, June 11-14.

[Ducellier, 2008] Ducellier G., (2008), gestion des règles expertes en ingénierie collaborative : application aux plateformes PLM, Thèse de doctorat, Université de Technologie de Troyes, Troyes, France.

[Durruvu, 1989] Durruvu S., et al., (1989), Knowledge based systems applications in engineering design: research at MIT. AI Magazine, Vol. 10, No. 3, pp. 79-96.

[Dutant et Engel, 2005] Dutant, J., Engel P., (textes réunis par), (2005), Philosophie de la connaissance : Croyances, connaissance, justification. Paris: Vrin, pp. 448.

E

[Eckard, 2000] Eckard C., (2000), Advantages and Disavantadges of FEM Analysis in an early state of the Design Process, Proceedings of the 2nd Worldwide Automotive Conference, MSC.Software Corporation, Dearborn, Michigan, USA.

[El Khalkhali , 2002] El Khalkhali, I., (2002), Système intégré pour la modélisation, l'échange et le partage des données de produits, Thèse de doctorat, Institut National des Sciences Appliquées de Lyon, France.

[Engel, 2000] Engel P., (2000), La Philosophie de la connaissance dans Précis de philosophie analytique, Paris: PUF, pp. 63-89.

[Ermine et al. 1996] Ermine J.L., Chaillot M., Bigeon P., (1996), Charreton B. et Malavieille D., MKSM : Méthode pour la gestion des connaissances, Ingénierie des systèmes d'information, AFCETHermès, Vol. 4, No. 4, pp. 541-575.

[Ermine, 2000] Ermine J.L., (2000), La gestion des connaissances, un levier stratégique pour les entreprises, IC'00, Toulouse.

[Ermine, 2001] Ermine J.L., (2001), Les processus de la gestion des connaissances, dans [Briand, 2001], pp. 17-30.

[Eynard et al., 2004] Eynard B., Gallet T., Nowak P., Roucoules L., (2004), UML based specifications of PDM product structure and workflow, Computers in Industry, Vol.55, pp.301-316.

[Eynard, 2005] Eynard B., (2005), Gestion du cycle de vie des produits et dynamique des connaissances industrielles en conception intégrée, Habilitation à Diriger des Recherches, Université de Technologie de Compiègne.

[Eynard et al., 2005] Eynard B., Liénard, S., Charles, S. and Odinot, A., (2005), Web-based Collaborative Engineering Support System: Applications in Mechanical Design and Structural Analysis, Concurrent Engineering: Research and Applications, Vol. 13, pp. 145-153.

F

[Ferraiolo et al., 1999] Ferraiolo D.F., Barkley J.F., Kuhn D.R., (1999), A Role Based Access Control Model and Reference Implementation within a Corporate Intranet ACM Transactions on Information Systems Security, Volume 1, Number 2.

[Fenves, 2001] Fenves S.J. (2002), A core product model for representing design information, National institute of Standards and Technology, NISTIR 6736, Gaithersburg, MD 20899.

[Fenves et al., 2004] Fenves S.J., Foufou S., Bock C., Bouillon N., Sriram R.D., (2004), CPM2 : A Revised Core Product Model for Representing Design Information, National Institute of Standards and Technology, Report No. NISTIR 7185, Gaithersburg, MD.

[Fenves et al., 2008] Fenves S.J., Foufou S., Bock C., Sriram R.D., (2008), CPM2: A Core Model for Product Data, Journal of Computing and Information Science in Engineering, Vol. 8.

[Fine et al., 2002] Fine L., Remondini L. et Léon J.C. (2000), Automated generation of FEA models through idealization operators, International Journal for Numerical Methods in Engineering, Vol. 49, pp. 43 - 108.

[Frey, 2010] Frey E., (2010), Contribution à une méthode de chaînage numérique de données pour la gestion des modifications lors de la conception de produits multi-métiers, Thèse de doctorat, Université de Technologie de Belfort-Montbéliard.

[Fukuda, 1995] Fukuda Y., (1995), Variations of Knowledge in Information Society, In Proceedings ISMICK 95, pp. 3-8.

G

[Gardan et al., 2003] Gardan N., Gardan Y., (2003), An application of knowledge based modelling using scripts, CMCAO team (CAD computer science lab.), IFTS, Pôle du moulin le blanc, Charlevilles-Mézières, France.

[Garetti et al., 2007] Garetti M., Terzi E.b.S, (2007), A new point of view on Product Lifecycle Management, University of Bergamo, Italy.

[Gero, 1990] Gero JS. (1990), Design prototypes: A knowledge representation scheme for design, AI Magazine, Vol. 11, pp. 26-36.

[Gomes et al., 2002] Gomes S., Sagot J-C., (2002), A concurrent engineering experience based on a cooperative and object oriented design methodology. In Best papers book from 3rd International Conference on Integrated Design and Manufacturing in Mechanical Engineering, Kluwer Publishers.

[Gomes et al., 2004] Gomes S., Serrafero P. (2004), Ingénierie collaborative et management des connaissances industrielles : du modèle systémique multi-vues au modèle positiviste Knova, Gestion dynamique des connaissances industrielles, traité IC2, série et systèmes d'information, chap. Systèmes d'information et capitalisation des connaissances, Eds B.Eynard, M.Lombard, Nada Matta, J.Renaud. Hermès, ch. 9, pp. 181-196.

[Gomes, 2008] Gomes S., (2008), Ingénierie à base de connaissances pour une conception, productive, optimisée, collaborative et innovante du système Projet-Produit-Process-Usage, Habilitation à Diriger des Recherches, Université de Technologie de Belfort-Montbéliard.

[Grebici, 2007] Grebici K., (2007) La Maturité et le Processus de Conception Collaborative, Thèse de Doctorat, INP de Grenoble.

[Grundstein, 1995] Grundstein M., (1995), La capitalisation des connaissances de l'entreprise, système de production de connaissances, l'entreprise apprenant et les sciences de la complexité, Aix en Provence.

[Grundstein, 2000] Grundstein, M. (2000), From capitalizing on Company Knowledge to Knowledge Management. In Daryl Morey, Mark Maybury, Bhavani Thuraisingham (Eds) Knowledge Management, Classic and Contemporary Works, ch. 12, pp. 261-287. Cambridge, Massachusetts: The MIT Press.

[Grundstein, 2002] Grundstein M., (2002), De la capitalisation des connaissances au renforcement des compétences dans l'entreprise étendue, 1er Colloque du groupe de travail Gestion des Compétences et des Connaissances en Génie Industriel, GCC-GI02, Nantes, France.

[Grundstein, 2007] Grundstein M., (2007), GAMETH: Un cadre directeur pour repérer les connaissances cruciales pour l'entreprise, rapport de recherche, #RR09, Nogent sur Marne, France.

[Grundstein, 2009] Grundstein M., (2009) GAMETH: a constructivist and learning approach to identify and locate crucial knowledge, International Journal of Knowledge and Learning, Vol. 5, No. 3-4, pp. 289-305.

H

[Hamdi, 2007 (1)] Hamdi M., Aifaoui N., BenAmara A., (2007), Intégration CAO Calcul par idéalisation des modèles CAO, 5ème Conférence Internationale Conception et Production Intégrées - CPI 2007, Vol. 2007, No. 37.

[Hamdi, 2007 (2)] Hamdi M, Aifaoui N, BenAmara A, (2007), Etat de l'art des méthodes d'idéalisation en CAO, CMSM'07, Monastir.

[Harani, 1997] Harani Y., (1997), Une approche multi-modèles pour la capitalisation des connaissances dans le domaine de la conception, Thèse Institut National Polytechnique Grenoble, France.

[Hatchuel et al., 2004] Hatchuel A., Le Masson P., Weil B., (2004), C-K Theory in Practice: Lessons from Industrial Applications, on international design conference-DESIGN 2004 Dubrovnik, May 18 – 21.

[Helms, 2002] Helms R.W., (2002), Product Data Management as enabler for Concurrent Engineering, PhD Thesis, Eindhoven University of Technology.

[Holt et al., 2009] Holt R., Barnes C., (2009), Towards an integrated approach to "Design for X": an agenda for decision-based DFX research, Research in Engineering Design, Original Paper, Springer London.

[Huynh, 2006] Huynh Q. H., (2006), Gestion de la complexité dans un logiciel destiné à la simulation numérique Multiphysique, Thèse de doctorat, INP Grenoble, France.

I

[IGES, 1988] IGES, (1988), Initial Graphics Exchange Specification (IGES) Version 4.0, NBSIR 88-3813, pp. 1-3.

[ISO 10007: 1995] Standard ISO 10007:1995, Quality management -- Guidelines for configuration management, 1995.

[ISO 10007: 2003] Standard ISO 10007:2003, Quality management -- Guidelines for configuration management, 2003.

[ISO 10303-209:2001] Standard ISO 10303-209:2001, Composite and metallic structural analysis and related design 2001.

[ISO/CEI 11179 : 2007] Standard ISO/CEI 11179 : 2007 Metadata Registry (MDR) standard, 2007

J

[Jagou, 1993] Jagou P., (1993), Concurrent Engineering – la maîtrise des coûts, des délais et de la qualité, Hermès.

K

[Karcher et al., 2003] Karcher A., Glander Matthias, (2003), Global distributed engineering – integrating different process paradigms, Journal of Materials Processing Technology, Vol. 138, pp. 131-137.

[Kant, 1781] Kant E. (1781), Critique de la raison pure, Paris: Quadrige.

[Kim et al., 2005] Kim H., Kim H-S., Lee J-H., Jung J-M., Lee J-Y. Chul N., (2005), Do A framework for sharing product information across enterprises, The International Journal of Advanced Manufacturing Technologies Vol. 27, No. 5-6, pp. 610-618.

[Kim et al., 2008] Kim D.K., Xirouchakis P., (2008), CO2DE: a decision support system for collaborative design, Journal of Engineering Design Vol. 00, No.0, pp. 1–18.

[Kleiner, 2003] Kleiner S., (2003), A collaborative design system for product data integration, Journal of Engineering Design, Vol.14, No. 4, pp. 421-428.

[Krastel et al. 02] Krastel M., Merkt T., (2002), Integration of Simulation and Calculation in a PDM Environment, Product Data Journal, Vol. 9, No. 2, pp. 7-9.

[Krause et al., 1995] Krause F.L., Cielsa M., Rieger E., Stephan M., Ulbrich A., (1995), "Features - semantic objects for the integration of tasks in the product development process", Computers in Engineering and Engineering Symposium, ASME 1995, Boston, pp. 667-685.

[Kurowski, 1995] Kurowski P.M. (1995), When good engineers deliver bad FEA, Machine Design, Edition Dvorak P.

L

[Labrousse et al., 2008] Labrousse M., Bernard A., (2008), FBS-PPRE, an Enterprise Knowledge Lifecycle Model, Methods and Tools for Effective Knowledge Life-Cycle-Management, Eds. A. Bernard and S. Tichkiewitch, Springer Verlag, pp. 285-305.

[Laidlaw et al., 1986] Laidlaw D.H., Trumbore W.B., Hughes J.F., (1986), Constructive solid geometry for polyhedral objects, ACM SIGGRAPH Computer Graphics, SIGGRAPH '86 Proceedings of the 13th annual conference on Computer graphics and interactive techniques, Vol. 20, Issue 4.

[Large et al., 2009] Large D., et al., (2009), Travail collaboratif : concepts et outils, Les guides de Micado, ISBN 978-2-9532336-2-9, édition Harigué.

[Le Duigou, 2010] Le Duigou J., (2010), Cadre de modélisation pour les systèmes PLM en entreprise étendue: application aux PME mécaniciennes, Thèse de doctorat, Ecole Centrale de Nantes.

[Lee, 2005] Lee, S. H., (2004), A CAD–CAE integration approach using feature-based multi-resolution and multi-abstraction modelling techniques, School of Mechanical and Automotive Engineering, Kookmin University, Jeongneung-Dong, Seongbuk-Gu, Seoul, pp. 136-702, South Korea.

[Lesage, 2003] Lesage D., (2003), Un modèle dynamique de spécifications d'ingénierie basée sur une approche de géométrie variationnelle, Thèse de doctorat, INPG-Institut National Polytechnique De Grenoble.

[Le Moigne, 1999] Le Moigne J.L., (1999), Les épistémologies constructivistes, Collection Que sais-je ?, Edition Puf, 1999, pp. 12.

[Le Moine 1995] Le Moine J.L., (1995), Les épistémologies constructivistes. PUF.

[Letemplier et al., 2006] Letemplier W., Beaune L., (Société PSA - Peugeot-Citroën) (2006), La simulation au cœur de la conception collaborative, Les Etats Généraux de Micado, sept. 2006,

[Li et al., 2005] Li W.D., Lu W.F., Fuh J.Y.H., Wong Y.S., (2005), Collaborative computer-aided design – research and development status, Computer-Aided Design, Vol. 37, pp. 931-940.

[Li et al., 2006] Li W.D., Qiu Z.M., (2006), State-of-the-art technologies and methodologies for collaborative product development systems, International Journal of Production Research, Vol. 44, No. 13, pp. 2525-2559.

[Louhichi et al. 2005] Louhichi B., Aifaoui N., BenAmara A., François V., Deneux D., (2005), Intégration numérique et interopérabilité des processus de CAO et de Calcul; 17ieme Congrès Français de Mécanique. Troyes, sept 2005.

[Lu et al., 2007] Lu S. C-Y., Elmaraghy W., Schuh G., Wilhelm R., (2007), A Scientific Foundation of Collaborative Engineering, CIRP Annals - Manufacturing Technology, Vol. 56 No. 2, pp. 605-634.

[Liu et al., 2001] Liu D.T., Xu X.W., (2001), A review of web-based product data management systems, Computers in Industry, Vol. 44, pp. 2551-2562.

[Liu et al., 2009] Liu W., Zeng Y., Maletz M., Brisson D., (2009), Product Lifecycle Management : A Review, Proceedings of the ASME 2009 International Design Engineering technical Conference IDETC, August 30 – September 2, San Diego, USA.

M

[Männisto et al., 1998] Männisto T., Peltonen H., Martio A., Sulonen R., (1998), Modelling generic product structures in STEP, Computer Aided Design, Vol. 30, No. 14, pp. 1111-1118.

[Massa et al., 2005] Massa F., Tison T., Lallemand B., (2005), A fuzzy procedure for the static design of imprecise structures, Computer Methods in Applied Mechanics and Engineering.

[Matta et al., 2000] Matta N., Ribiere M., Corby O., Lewkowicz M., Zaclad M., (2000), Project Memory in Design, Industrial Knowledge Management - A Micro Level Approach, Rajkumar Roy (Eds), Springer-Verlag.

[Milton, 2008] Milton N., (2008), Knowledge Technologies, Publishing Studies, POLIMETRICA.

[Menand, 2002] Menand S., (2002), Modélisation pour la réutilisation du processus de conception Multi acteurs de produits industriels, Thèse de Doctorat de l'institut National Polytechnique, Grenoble, France.

[Mer, 1998], Mer S., (1998), Les mondes et les outils de la conception. Pour une approche socio-technique de la conception du produit, Thèse de doctorat de Génie Industriel de l'Institut National Polytechnique de Grenoble, France.

[Merlo, 2003] Merlo C., (2003), Modélisation des connaissances en conduite de l'ingénierie : mise en œuvre d'un environnement d'assistance aux acteurs, Thèse de doctorat, Université de Bordeaux 1.

[Merlo et al., 2004] Merlo J.C., Girard Ph., (2004), Information system modelling for engineering design co-ordination, Computers in Industry, Vol. 55, Issue 3, pp. 317-334, Object-oriented modelling in design and production.

[Midler, 1993] Midler C., (1993), L'acteur-projet : situations, mission, moyens ; dans Ecosip, Pilotage de projet etentreprises, Economica, Paris.

[Miles et al., 1992] Miles B.L., Swift K.G., (1992), Working together, Manufacturing Breakthrough, Vol. 4, pp. 69-73.

[Monticolo et al., 2008] Monticolo D., Hilaire V., Gomes S., Koukam A., (2008), A multi-agent system for building project memories to facilitate the design process, Integrated Computer-Aided Engineering, Vol. 15, No. 1 ,pp.3-20.

[Monticolo, 2008] Monticolo D., (2008), Une approche organisationnelle pour la conception d'un système de gestion des connaissances fondé sur le paradigme agent, Thèse de doctorat, Université de Technologie de Belfort-Montbéliard.

[Maurino, 1995] Maurino M., (1995), La gestion des données techniques, Masson.

N

[Noël, 2003] Noël F., (2003), Outils dédiés à la Conception de Produits Mécaniques dans un contexte de Liaison Conception-Simulation, Habilitation à diriger des recherches, Universite Joseph Fourier Grenoble, France.

[Noël et al., 2005] Noël F., Roucoules L., Teissandier D., (2005), Specification of Product Modelling Concepts Dedicated to Information Sharing in a Collaborative Design Context, Advances in Integrated Design and Manufacturing in Mechanical Engineering, Part 1, pp. 135-146, DOI: 10.1007/1-4020-3482-2_11.

[Noël et al., 2007] F. Noël, L. Roucoules, (2007), The PPO design model with respect to digital enterprise technologies among product life cycle, International journal of computer integrated manufacturing, Taylor and Francis Ltd, Vol. 21, No. 2, pp. 139-145.

[Nonaka, 1995] Nonaka I., Takeuchi H., (1995), The Knowledge-Creating Company, Oxford University Press, Oxford.

O

[O'Bara et al., 2002] O'Bara R.M., Beall M.W. et Shephard M.S., (2002), Attribute Management System for Engineering Analysis, Engineering with Computers, Vol. 18, pp. 339-351.

[OMG, 2001] Object Management Group, (2001), Inc Model Driven Architecture.

[OMG, 2007a] Object Management Group, (2007), Inc. Unified Modeling Language (UML) 2.1.2 Infrastructure, Final Adopted Specification, pp 18, 30, 49.

[OMG, 2007b] Object Management Group, (2007), Inc. Unified Modeling Language (UML) 2.1.2 Superstructure, Final Adopted Specification, pp 18, 30, 33.

[OMG, 2008c] Object Management Group, Inc. Meta Object Facility (MOF) 2.0 Query/ View/Transformation (QVT) Specification, version 1.0, avril 2008, pp 13, 44, 45.

P

[Paviot et al., 2009] Paviot T., Cheutet V., Lamouri S., (2009), Federate design and logistic through Product Lifecycle Support standard, International Conference on Product Lifecycle Management, Bath, UK.

[Peltonen, 2000] Peltonen H., (2000), Concepts and an Implementation for Product Data Management, Doctorate Thesis. Acta Polytechnica Scandinavia, Mathematics and Computing Serie No.105, 188pp. Published by The Finnish Academics of Technology.ISBN 951-666-538-1. ISSN 1456-9418.

[Pahl et al., 1984] Pahl G., Beitz W., (1984), Engineering design, Design Council, London.

[Pahl et al., 1996] Pahl G., Beitz W., (1996), Engineering Design: a Systematic approach, 2nd Ed., London.

[Pahl et al., 2007] Pahl G., Beitz W., Feldhusen J., Grote K.H., (2007), Engineering Design: a Systematic Approach, Third Edition, Springer- Verlag, London.

[Polanyi, 1967] Polanyi M., (1967), The Tacit Dimension, ISBN 0-8446-5999-1.

[Prasad, 1996] Prasad B., (1996), Concurrent Engineering fundamentals: Integrated Product and Process Organization, Prentice Hall, pp. 502.

[Prasad, 2005] Prasad B., (2005), What distinguishes KBE from Automation COE NewsNet.

[Pratt et al., 2005] Pratt. M.J., Anderson. B.D., Ranger. T., (2005), Towards the standardized exchange of parameterized feature based CAD models, Computer-Aided Design, Volume 37, Issue 12, Pages 1251-1265.

[Prax, 2000] Prax J.Y., (2000), Le guide du Knowledge management: concepts et pratiques du management de la connaissance, Edition Dunod, Paris.

[Prudhomme et al., 2001] Prudhomme G., Boujut J.F. et Pourroy F., (2001), Activités de conception et instrumentation de la dynamique des connaissances locales, Ingénierie des Connaissances, Plateforme AFIA, Presses Universitaires de Grenoble, pp. 41-60.

[Prudhomme et al. 2007] Prudhomme G., Pourroy F., Lund K., (2007), An empirical study of engineering knowledge dynamics in a design situation, Journal of Design Research, Vol. 6, No. 3, pp. 333 – 358.

[PSA, 2010] Petit L., (2010), Présentation des ateliers d'architectures, La Garenne Colombes, France.

Q

R

[Randoing, 1995] Randoing J.M., (1995), Les SGDT, Hermès, Paris.

[Rivière, 2004] Rivière A., (2004), Gestion de configuration et des modifications lors du développement de grands produits complexes en ingénierie concourante, cas d'application aéronautique, Thèse de doctorat, Institut National Polytechnique de Grenoble, France.

[Rodriguez et al., 2002] Rodriguez K., Al-Ashaab A., (2002), A review of internet based collaborative product development systems, Proceedings of the International Conference on Concurrent Engineering, Cranfield, UK.

[Rodriguez et al., 2005] Rodriguez K., Al-Ashaab A., (2005), Knowledge-based system architecture for collaborative product development, Computers In Industry, Vol. 56, pp. 125-140.

[Roozenburg et al., 1995] Roozenburg N.F., Eeckels J., (1995), Product design: fundamentals and methods, John Wiley & Sons.

[Roucoules et al., 2006] Roucoules L., Noël F., Teissandier D., Lombard M., Débarbouillé G., Girard P., Merlo C., Eynard B., (2006), IPPOP : an open source collaborative design platform to link product, design process and industrial organisation information, 6th International Conference on Integrated Design and Manufacturing in Mechanical Engineering, IDMME'06, Grenoble : France.

[Roucoules, 2007] Roucoules L., (2007), Contribution à l'intégration des activités collaboratives et métier en conception de produit, une approche au juste besoin : des spécifications fonctionnelles du produit aux choix des procédés de fabrication (la maîtrise du produit par la maîtrise des procédés), HDR Soutenue publiquement à l'Université de Technologie de Troyes le 09 novembre 2007 p 46 à 54.]

[Rouibah et al., 2007] Rouibah K., Ould-Ali S., (2007), Dynamic data sharing and security in a collaborative product definition management system, Robotics and Computer-Integrated Manufacturing Vol. 23, Issue 2, pp. 217-233.

S

[Säaksvuori et al., 2005] Saaksvuori A., Immonen A., (2005), Product Lifecycle Management, 2nd ed., Berlin, London, Springer.

[Salome, 2005] http://www.salome-platform.org/

[Sanchez et al., 1996] Sanchez R., Heene A., Thoimas H., (1996), Dynamics of Competence- based competition, England: Elsevier Science Ltd, Pergamon, pp. 403.

[Schlenkrich et al. 2004] SCHLENKRICH M., MAYER S., MINABE S., (2004), Gestion des données de simulation : automatisation des cycles et gestion des données pour l'optimisation des simulations numériques, Journée MICADO - Organisation et rentabilité de la fonction calcul en entreprise, Clamart – France.

[Sellini, 1999] Sellini F., (1999), Contribution à la représentation et à la vérification de modèles de connaissances produit en ingéniérie d'ensembles mécaniques, Thèse de doctorat, Ecole Centrale Paris, ISMCM – GRIIEM.

[Serrafero, 2006] Serrafero P., Gomes S., Bonnivard D., Jézéquel L., (2006), De la mémoire projet à la compétence métier : vers la synthèse de connaissances métier en ingénierie robuste des produits/process, Conférence Internationale IDMME.

[Serrafero et al., 2007] Serrafero P., Gomes S., Monticolo D., (2007), Ingénierie innovante, collaborative et apprenante en développement automobile : vers une CAO en 5D au service de la mobilité congruente, Revue Française de Gestion Industrielle, Vol. 26, No. 3, pp. 7-18.

[Shephard, 2000] Shephard M.S. (2000), Meshing environment for geometry-based analysis, International Journal for Numerical Methods in Engineering, Vol. 47, Issue 1-3, pp. 169-190.

[Shephard et al., 2004] Shephard, M. S., Beall, M. W., O'Bara, R. M. and Webster, B. E., (2004), Toward simulation-based design, Finite Elements in Analysis and Design, Vol. 40, pp. 1575–1598.

[Suh, 1999] Suh N.P., (1999), Applications of Axiomatic Design, Integration of process Knowledge into Design Support, ISBN 0-7923-5655-1, Kluwer Academic Publishers.

[Simdat, 2005] http://www.ecmwf.int/services/grid/simdat/

[Skarka, 2007] Skarka W., (2007), Application of MOKA methodology in generative model creation using CATIA, Engineering Applications of Artificial Intelligence.

[Sohlenius, 1992] Sohlenius G., (1992), Concurrent engineering, Annals of the CIRP, Vol. 41, No. 2, pp. 645-655.

[Sung et Park, 2007] Sung C.S., Park S.J., (2007), A component-based product data management system, International Journal of Advanced Manufacturing Technology, Vol. 33, No. 5-6, pp. 614-626.

[Stark et al., 2004] Stark J. et al., (2004), Product Lifecycle Management – Paradigm for 21st century, Springer-Verlag, Berlin.

[Stokes, 2001] Stokes M., (2001), Managing Engineering Knowledge; MOKA: Methodology for Knowledge Based Engineering Applications. Professional Engineering Publishing, London.

[Subrahmanian et al., 2005] Subrahmanian E., Rachuri S., Fenves S.J., Foufou S., Sriram R.D., (2005), Product lifecycle management support: a challenge in supporting product design and manufacturing in a networked economy , International Journal of Product LifeCycle Management (IJPLM), Vol. 1, No. 1, pp.4-25, 2005.

[Sudarsan et al., 2005] Sudarsan R., Fenves S.J, Sriram R.D., Wang F., (2005), A product information modeling framework for product lifecycle management, Computer-Aided Design, Vol. 37, Issue 13, pp. 1399-1411.

[Rachuri et al., 2003] Rachuri S., Han Y.H., Feng S.C., Roy U., Wang F., Sriram R.D. and Lyons K.W., (2003), Object-oriented Representation of Electro-Mechanical Assemblies Using UML, National Institute of Standards and Technology, NISTIR 7057, Gaithersburg, MD 20899, USA.

[Suh, 1990] Suh N., (1990), The principles of design, Oxford series on advanced manufacturing, Oxford university press.

[Systematic, 2010] Systematic, (2010), R&D Project 2010, System Design and Development Tools WG, Modelling system simulation, AND - Alliance des Données Numériques, pp. 62.

T

[Tang et al. 04] Tang S-H, MaY. –S. and Chen G, A (2004), Feature-oriented Databased Framework for Web-based Cax Applications, Nanyang Technological University.

[Taylor et al., 1994] Taylor L., Henderson M., (1994), Validating a feature-based meta-model for mechanical products : a case study, IFIP International Conference on Feature Modeling and Recognition in Advanced CAD/CAM Systems, Valenciennes, mai 1994, pp. 21-39.

[Thomas et al., 2004] Thomas B., Crepel J.M., (2004), La gestion des données de la simulation numérique dans l'ingénierie véhicule Renault, MICADO - Organisation et rentabilité de la fonction calcul en entreprise, Clamart – France.

[Tichkiewitch et al., 1993] Tichkiewitch S., Tiger H., Jeantet A., (1993), Ingénierie simultané dans la conception de produit, Université d'été du pôle Rhône Alpes sur la modélisation en entreprise, Modane.

[Tichkiewitch et al., 1995] Tichkiewitch S., Chapa E., Belloy P., (1995), Un modèle produit multi-vues pour la conception intégrée, Productivity in world without borders conference, Montréal.

[Tichkiewitch, 1996] Tichkiewitch S. (1996), Specifications on integrated design methodology using a multi-view product model, System Design and Analysis Conference, Montpellier, pp. 101-108.

[Tollenaere, 1994] Tollenaere M., Contribution à la modélisation de connaissances pour la conception mécanique, Habilitation à diriger des recherches, Université Joseph Fourier Grenoble I, France.

[Tollenaere, 1998] Tollenaere M. (1998), Conception de produits mécaniques, Editions Hermès, Paris, ISBN 2-86601-694-7.

[Toussaint et al., 2010] Toussaint L., Demoly F., Lebaal N., Gomes S., (2010), PLM based approach for design verification and validation using manufacturing process knowledge, Journal of Systemics, Cybernetics and Informatics, Vol. 8, No. 1, pp. 1-7.

[Toussaint, 2010] Toussaint L., (2010), Modèles et méthodes pour une conception hautement productive orientée vers la fabrication : application à l'ingénierie routinière de pièces plastiques, Thèse de doctorat, Université de Technologie de Belfort-Montbéliard.

[Troussier, 1999] Troussier N., (1999), Contribution à l'intégration du calcul mécanique dans la conception de produits techniques : proposition méthodologiques pour l'utilisation et la réutilisation, Thèse de doctorat, Institut National Polytechnique de Grenoble, Grenoble.

[Tsuchiya, 1995] Tsuchiya S., (1995), Commensurability, a key concept of business reengineering, 3rd international symposium of the management of information and corporate knowledge, institut national pour l'intelligence artificielle, Compiègne.

U

[Ullman, 2002] Ullman D.G., (2002), The mechanical design process, Mc Graw Hill, 3ème edition.

V

[Van Leeuwen et al., 2003] Van Leeuwen, J.P., S. Fridqvist, (2003), Object Version Control for Collaborative Design In: Tunçer, Özsariyildiz, and Sariyildiz, E-Activities in Building Design and Construction, Proceedings of the 9th EuropIA International Conference, Istanbul, EuropIA Productions, pp. 129-139.

[Vareilles, 2005] Vareilles E., (2005), Conception et approches par propagation de contraintes : contribution à la mise en œuvre d'un outil d'aide interactif, Thèse de doctorat, Institut National polytechnique de Toulouse, France.

[Vargas, 1996] Vargas C., (1996), Modélisation du Processus de Conception en Ingénierie des Systèmes Mécaniques", Thèse de Doctorat, pp. 38-39, ENS de Cachan.

[VDA, 1987] VDA, (1987), Vda flächen schnittstelle, norme vda-vdma Version 2.0, pp. 66-301.

[Vernier et al., 2011] Vernier C., Lebaal N., Boudouh T., Gomes S., (2011), Conception intégrée à base de connaissances en environnement PLM, 12ème Colloque National AIP PRIMECA, Le Mont Dore, France.

[Veron, 2005] Véron P., (2005), Méthodes dédiées à la Construction, la Manipulation et l'Exploitation de Maquettes Numériques de Produits dans un Contexte d'Ingénierie Collaborative, Habilitation à diriger des recherches, l'Institut National Polytechnique de Grenoble, France.

[Vinck, 1997] Vinck D., (1997), La connaissance : ses objets et ses institutions, in Connaissances et savoir-faire en entreprise, intégration et capitalisation, Coordonnateur J.M. Fouet, Editions Hermes.

W

[Weggeman, 1996] Weggeman M., (1996), Knowledge Management: The Modus Operandi for a Learning Organization, In J. F. Schreinemakers ed, Knowledge Management: Organization, Competence and Methodology, Proc. Of ISMICK'96, Rotterdam, the Netherlands, Wurzburg:Ergon Verlag, Advances in Knowledge Management, Vol. 1, pp. 175- 187

[Wenger, 1999] Wenger E., (1999), Communities of practice. Learning, Meaning and Identity, Cambridge: University Press, pp. 336

[Wenger et Snyder, 2000] Wenger E. et Snyder W., (2000), Les communautés de pratique, le nouvel horizon organisationnel, Le management du savoir en pratique, Paris : Les Editions d' Organisation.

X

Y

[Yang et al., 2008] Yang D., Dong M., Miao R., (2008), Development of a product configuration system with an ontology-based approach, Department of Industrial Engineering, School of Mechanical Engineering, Shanghai Jiao Tong University, 800 Dong Chuan Road, Shanghai 200240, China.

[Yeh, 1992] Yeh R.T., (1992), Notes on concurrent engineering, IEEE Transation on Knowledge Data Engineering, Vol .4, No. 5, pp. 407-414.

[Yvars, 2001] Yvars P.A., (2001), Contribution à la représentation des connaissances en ingénierie intégrée de produits et de systèmes automatisés de production, Habilitation à Diriger des Recherches, Institut National Polytechnique de Grenoble.

Z

[Zaclad, 2003] Zaclad M., Grundstein M., (2001), Ingénierie et capitalisation des connaissances Hermes Science Publications, Paris, ISBN 2-7462-0234-4, chapitre 1, pp. 15- 22.

[Zarifian, 2002] Zarifian P., (2002), La politique de la compétence et de l'appel aux connaissances dans la stratégie d'entreprise, Actes du 1er colloque du groupe de travail Gestion des Compétences et des Connaissances en Génie Industriel "Vers l'articulation entre Compétences et Connaissances", Nantes, France.

[Zouari et al., 2002] Zouari A., Tollenaere M., Menand S., (2002), Application of a multi-actors design model to the car air-conditioning system functional design, IEEE SMC02 Conference, Hammamet, Tunisie.

[Zouari, 2007] Zouari A., (2007), Proposition de mécanismes de versionnement et d'agrégation des connaissances de domaine en conception collaborative de produits industriels, Thèse de doctorat, L'INP Grenoble et Université de Sfax.

ANNEXES

Annexe 1 : Concept des ateliers d'architectures

Le concept d'atelier a été développé, dès 2002 par le service MTAF (**M**odélisation **T**enue, **A**coustique et vibration, **F**iabilité) du groupe PSA Peugeot Citroën. L'idée générale est de fournir aux métiers de la conception et de la simulation numérique des outils intégrés à CATIA V5, leur permettant de valider certaines prestations sur un organe donné. Ces ateliers (symbolisés A²) sont donc utilisés pour guider le concepteur très tôt sur le choix de la bonne architecture, via des IHM ergonomiques qui vont lui permettre d'itérer sur plusieurs types de modèles et d'effectuer des calculs prédéfinis.
Ces ateliers ont trois objectifs principaux qui sont :

- **Objectif 1 :** Capitaliser un certain nombre d'architectures génériques de pièces.
- **Objectif 2 :** Agir sur les architectures et être guidé dans les choix à effectuer.
- **Objectif 3 :** Intégrer et fiabiliser des méthodologies de calcul.

Les ateliers se présentent sous forme d'environnements CATIA V5 ou, "Workbenchs", tout comme n'importe quel autre module du logiciel. Une barre d'icones est associée à différents comportements, ou fonctions qu'il est possible de réaliser sur les modèles qui sont en cours d'études (Figure 120).
Cela permet à des personnes qui ne possèdent pas de connaissances en calcul d'effectuer des études précises et de simuler simplement et rapidement le comportement d'une piece ou d'un ensemble mécanique.

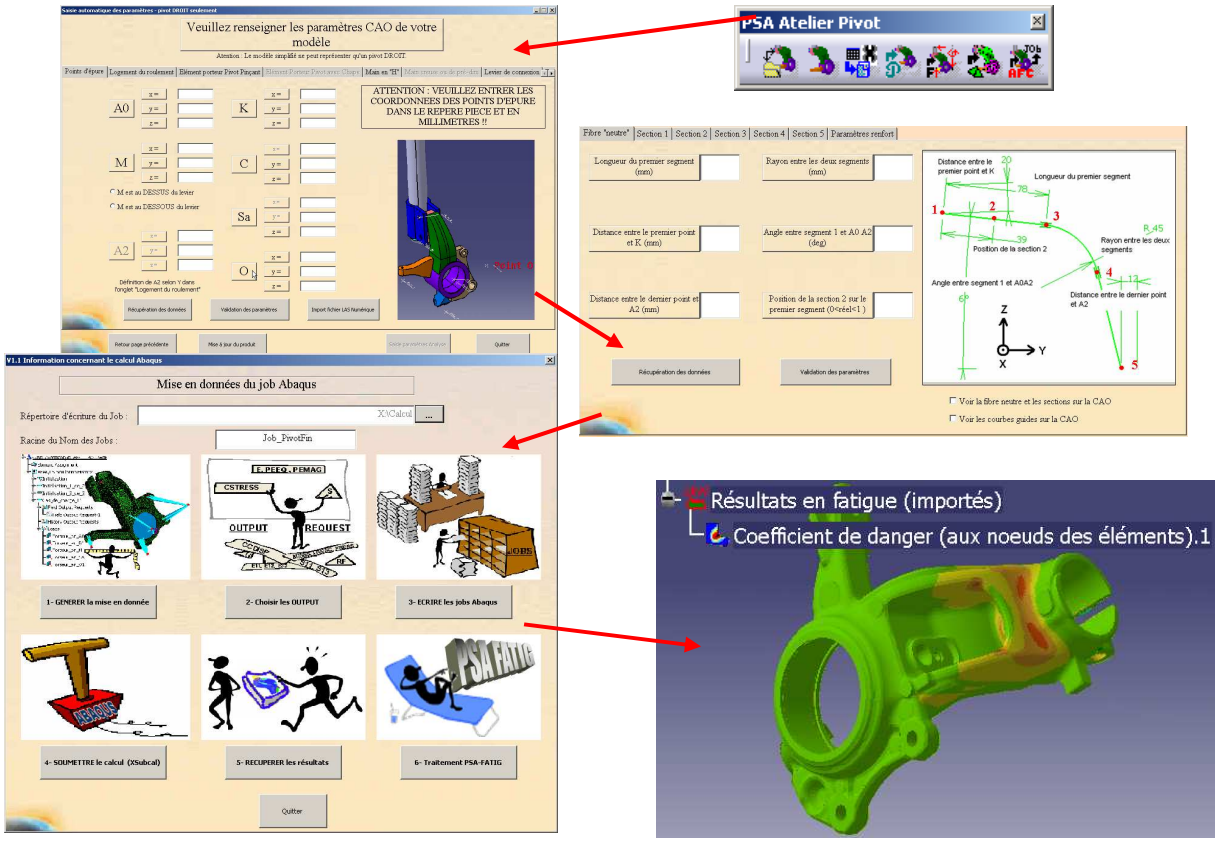


Figure 120 : exemple de fonctionnement d'un atelier

Dans le cadre du développement de ces outils, les langages de programmation utilisés sont le CAA (Component Application Architecture) pour les ateliers les plus anciens, et le Visual Basic (VB) pour les plus récents. Afin de guider au mieux les utilisateurs, les IHM (Interface Homme Machine) sont prédominantes dans le fonctionnement de ces outils. Leur mode d'emploi est le suivant : l'utilisateur (concepteur/calculateur) récupère une CAO paramétrée et générique. Il modifie les paramètres et règles métiers afin d'arriver à une architecture qu'il souhaite tester en simulation numérique. Ces paramètres peuvent être géométriques ou liés à une filière de calcul (tension de vis, température imposée, etc.). Une fois les paramètres renseignés, l'utilisateur suit un processus automatique ou semi-automatique (via des IHM) lui permettant de mailler ses composants, faire la mise en donnée, générer

les fichiers de calcul conformes aux paramètres entrés, lancer le calcul sur le serveur et le solveur approprié, et réimporter les résultats dans une application appropriée pour l'analyse (Figure 121).

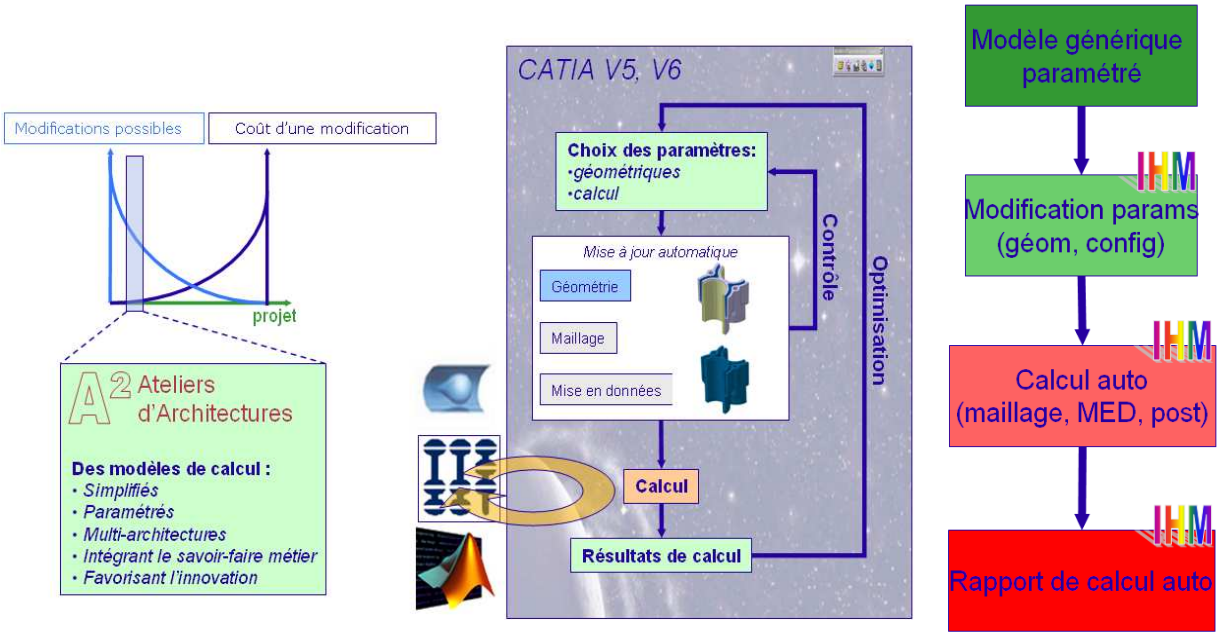


Figure 121 : principe de fonctionnement d'un atelier d'architecture [PSA, 2010]

En effet, un atelier est dédié à un composant et une prestation donnés, par exemple : un atelier piston thermique, un atelier carter acoustique, un atelier culasse thermomécanique, etc. Tous les ateliers utilisent des représentations de composants adaptées à chaque situation mais différentes d'un atelier à l'autre. Ils font intervenir plusieurs outils de simulation (différents types de solveurs par exemple). On peut considérer qu'aujourd'hui il existe environ trente ateliers différents, utilisés dans plusieurs domaines métiers en simultané et sur plusieurs sites géographiques.
Le problème auquel est confronté PSA Peugeot Citroën consiste en l'absence de cohérence entre les différents jeux de paramètres pouvant être entrés dans deux ateliers différents (Figure 122). Ainsi, l'utilisateur de l'A²CarterCylindres pourra choisir un diamètre de fûts différent de la valeur choisie par l'utilisateur de l'A²Piston, ce qui n'est pas cohérent. Aujourd'hui la cohérence est assurée par les Pilotes d'Architecture Moteur (PAM), mais les informations sont souvent échangées par mail, voire oralement, ce qui n'est pas fiable et ne garantit aucune traçabilité.

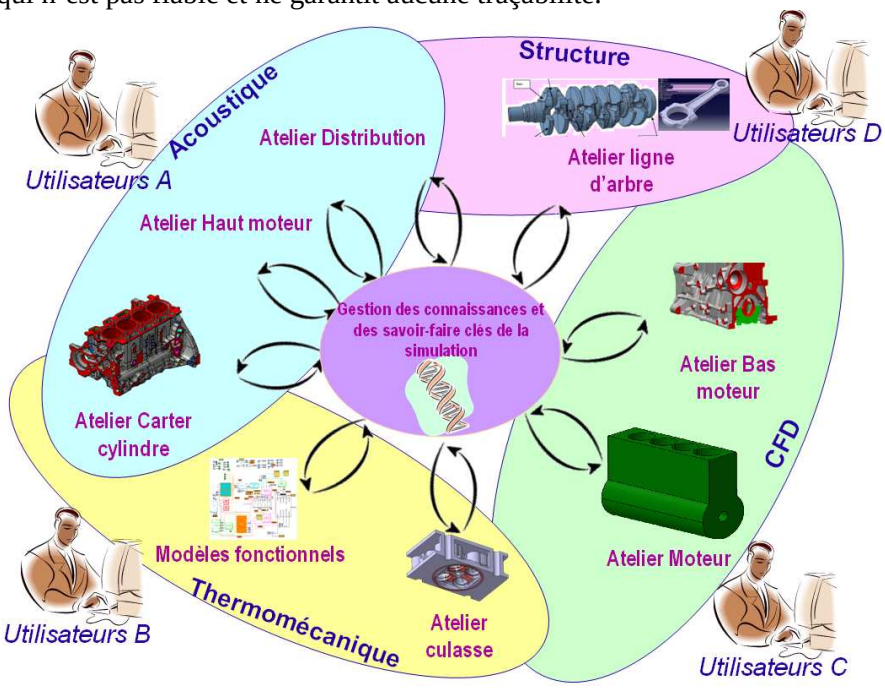


Figure 122 : communication entre les ateliers

En effet, il arrive trop souvent que des calculs soient lancés sur de mauvaises configurations et ou versions de paramètres, entre pièces connexes ou non. Les pertes de temps et financières qui en découlent dépendent du moment où l'on s'aperçoit de l'erreur. Dans le cas d'une erreur constatée tardivement, les pertes peuvent être considérables.

Annexe 2 : structure d'un calcul EF

Simuler numériquement le comportement d'une pièce ou d'un ensemble de pièces nécessite l'utilisation de modélisations géométriques et comportementales. Cela se concrétise au travers de l'utilisation de modèles métiers, c'est-à-dire des modèles CAO représentatifs de la géométrie de la pièce et des modèles de calculs (CAE Computed Aided Design) comprenant l'ensemble des conditions de simulation. Cela nécessite également un processus de préparation de la géométrie (appelé idéalisation) et de préparation au calcul, afin de mener l'étude dans de bonnes conditions. Ce processus de simulation est bien particulier et n'a pas d'équivalent dans le processus de conception.
Il n'existe pas un processus type de simulation, car il est en effet adapté à chaque contexte d'études et dépend fortement de la méthode de calcul, des outils utilisés, de la physique étudiée et de la sensibilité des concepteurs et des calculateurs. Néanmoins, nous proposons un processus global illustrant les différentes étapes permettant de conduire à l'observation numérique du comportement d'une pièce mécanique.

- **La conception** est souvent la première étape, il s'agit de modéliser géométriquement le composant que l'on veut étudier. Etant donné que les méthodes de calculs les plus utilisées aujourd'hui concerne la méthode des éléments finis, la modélisation géométrique est donc un modèle 3D (modèle paramétrique ou non).

- **La préparation de la géométrie (ou idéalisation)** [Hamdi, 2007 (1)] [Hamdi, 2007 (2)] [Benhafid, 2005] est une étape importante qui succède à l'étape de conception. Il s'agit ici de simplifier la géométrie en retirant tout ce qui n'est pas indispensable pour le type de calcul à réaliser. En effet, la mise en donnée d'un calcul est une tâche délicate, alors si l'on peut simplifier le plus tôt possible la géométrie, cela permet de faciliter les étapes suivantes. Cette simplification dépend de nombreux facteurs tels que le type de calculs à réaliser, les zones précises des pièces à étudier (les zones éloignées des zones d'études peuvent être plus grossières), la géométrie des pièces (il est souvent possible de limiter l'étude à une demie ou un quart de pièce si celle-ci est axisymétrique par exemple**).** L'objectif est de disposer d'une représentation géométrique du composant la plus simple possible, adaptée au cas d'étude en particulier, et tout de même représentative du comportement réel de la pièce. Enfin, en fonction de ce que l'on veut observer, on découpe la pièce en zones. Par exemple, si l'on souhaite étudier finement la face feu d'un piston, on découpera cette face afin de pouvoir l'isoler et le moment venu, lui appliquer plus facilement certaines caractéristiques de maillage ou des propriétés particulières.

- **La récupération des données de calculs** est une étape qui consiste à récupérer l'ensemble des paramètres, des règles, des méthodes nécessaires à la mise en place du modèle de calcul. Cette étape est délicate dans la mesure où, a priori, il n'existe pas vraiment de plateforme ou d'outils capitalisant et structurant ces données. En effet, les SGDT utilisés pour le stockage des données de type CAO sont souvent inadaptés pour la manipulation des données de calculs nécessitant une organisation différente. De plus, les paramètres et les règles de comportement nécessaires aux calculs sont rarement pris en compte par ce type d'outils [Charles, 2005] [Ducellier, 2008].

- **Le pré-traitement** est une étape qui consiste à préparer le calcul. Elle comporte de nombreuses sous-tâches qui possèdent plus ou moins de valeur ajoutée. On retrouve comme sous-tâche :
    - Le maillage (opération sans valeur ajoutée) qui permet la discrétisation du modèle géométrique en éléments et en nœuds nécessaires aux algorithmes et aux fonctions analytiques de calculs.
    - Le choix des éléments et de leurs propriétés (sans réelle valeur ajoutée). Il existe de nombreux types d'éléments correspondants à chaque contexte d'étude.
    - La création des liaisons et des contacts entre les pièces (opération avec valeur ajoutée). Cette étape consiste à appliquer les propriétés qui vont retranscrire le comportement d'une pièce envers une autre, comme par exemple la liaison avec frottement d'un

piston dans un cylindre. A ce stade, on ajoute également les conditions limites, c'est-à-dire les blocages sur le composant à étudier, ou les symétries si l'on travaille sur une demie ou un quart de pièce.

- Création d'un cas de simulation et mise en place des chargements (opération avec valeur ajoutée). A cette étape, on met en place les chargements pour le type d'études que l'on souhaite mener et les conditions de calculs qui seront nécessaires. Par exemple, chargements de type force en Newton pour un cas statique linéaire.
- Création des fichiers de calculs, génération des fichiers (suite aux étapes précédentes) qui seront envoyés sur le calculateur.

- **Le lancement du calcul (souvent appelé Job de calcul)** consiste à lancer le calcul sur un calculateur. Le temps de calcul dépend du type d'étude, de la puissance du calculateur (cela se compte en nombre de processeurs ou plus précisément en flops), de la taille du modèle de calculs (nombre d'éléments, de nœuds etc.), etc.

- **Le post-traitement** est une étape qui consiste à observer et sélectionner les résultats pertinents en fonction de l'étude menée.

L'analyse et l'exploitation des résultats est la dernière étape qui consiste à dépouiller et comprendre les résultats fournis par le calculateur. Ce résultat est issu de représentations graphiques et de données récupérées des fichiers de calculs. Ils doivent normalement être stockés afin de potentiellement être réutilisés dans un autre calcul. On peut donner comme exemple la récupération des données d'un calcul thermique nécessaires au lancement d'un calcul thermomécanique.

Annexe 3 : le calcul multi-physique

Dans la réalité, une pièce est rarement soumise à une sollicitation unique mais plutôt à un ensemble de contraintes d'origines diverses faisant intervenir plusieurs physiques. Tout objet est en permanence soumis de façon plus ou moins intense à un ensemble de phénomènes physiques, incluant mouvements, efforts, écoulements fluides, vibrations, transferts thermiques, réactions, chimiques et électromagnétisme. Ces phénomènes ne sont pas indépendants les uns des autres, et il s'avère souvent indispensable de ne pas négliger leurs influences mutuelles dans le cadre de la simulation numérique. Par conséquent, simuler numériquement le comportement d'une pièce dans une physique particulière (même dominante) biaise forcément le résultat du calcul. C'est pour cette raison que l'introduction du calcul multi-physique est considérée comme un axe majeur de progrès par les industriels car il permet aux ingénieurs d'obtenir des résultats toujours plus proches de la réalité [Comsol, 2004]. La simulation multi-physique permet de coupler plusieurs physiques afin de mettre en évidence le comportement d'une pièce plus finement qu'une étude sur une physique unique. Cela se concrétise par la mise en relation étroite de modèles métiers traduisant un enchainement d'analyses variées exécutées itérativement. La multiplicité des physiques augmente la complexité du modèle de données [Huynh, 2006]. Un volume important de données doit transiter d'un modèle métier à l'autre sans perte d'information et ce, malgré le caractère hétérogène de ces modèles. Il en résulte des difficultés de pilotage et d'erreurs, comme par exemple des paramètres perdus et non pris en compte ou des calculs lancés sur de mauvaises valeurs entre deux modèles métiers.

Annexe 4 : évolutions des pratiques de la simulation numérique

La Figure 123 illustre l'évolution des pratiques de la simulation dans le cadre du passage de l'ingénierie dite séquentielle, à l'ingénierie dite simultanée.

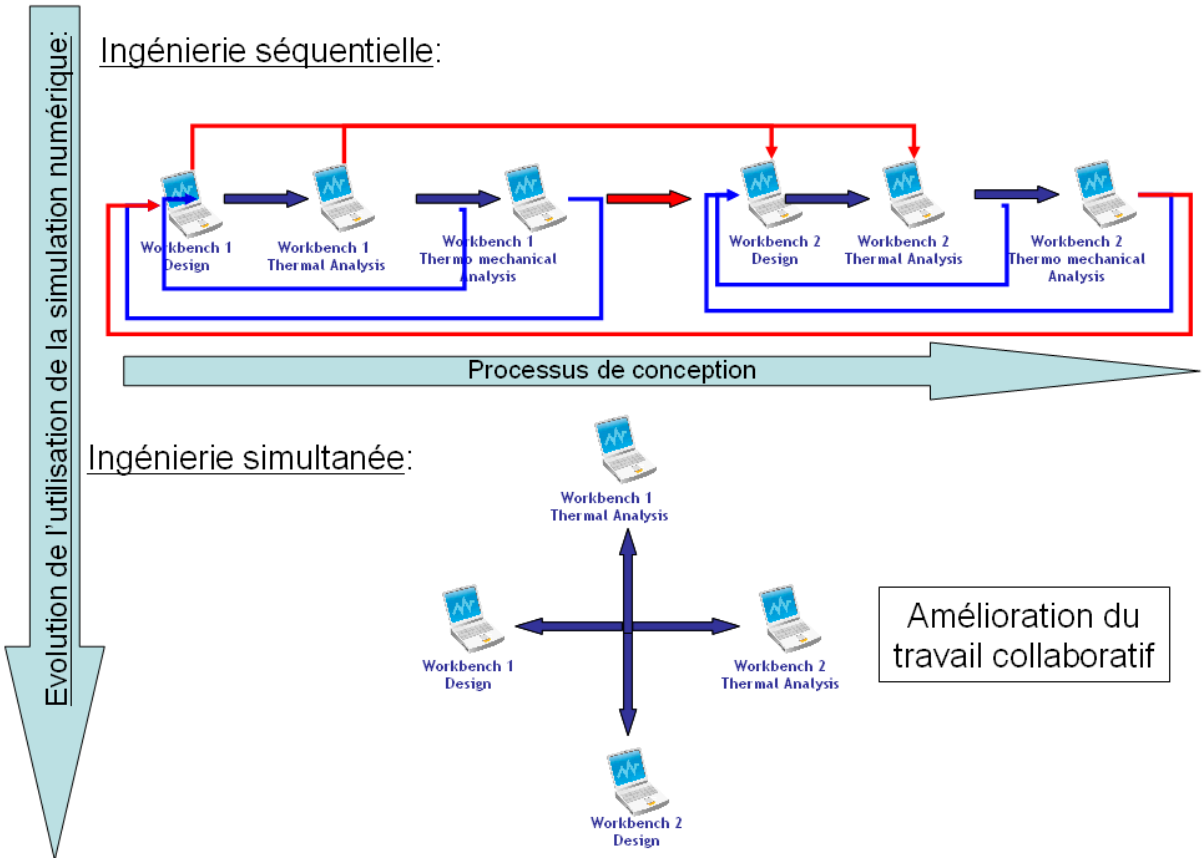


Figure 123 : évolutions des pratiques de la simulation numérique dans le processus de conception

L'ingénierie simultanée permet d'éviter les boucles de re-conception qui peuvent apparaitre suite à la découverte tardive de problèmes, entrainant la remise en cause des activités de conception en amont (entre le Workbench 1 et 2). Ainsi, l'ingénierie simultanée permet une amélioration du travail collaboratif des acteurs et des gains en termes de temps et de qualité sur les produits.

Annexe 5 : Typologie de la simulation numérique

A partir des travaux de [Charles, 2005] et [Amann, 2010], nous proposons la typologie suivante qui regroupe les différentes pratiques de la simulation. Elles s'articulent en quatre types répartis dans deux catégories.

<table>
<tr><td rowspan="2">La simulation dite passive est utilisée de façon plus séquentielle. Elle est plus indépendante du processus de conception. Elle est utilisée pour contrôler, pour valider, certifier, comprendre, sur la base d'une proposition de solution.</td><td>La simulation de validation vise à entériner les choix réalisés par les acteurs de la conception. Cette activité est historiquement la plus pratiquée par les industriels lors de processus d'ingénierie séquentiel. La simulation de validation arrive en fin de cycle de conception et valide (ou invalide) le résultat du travail des concepteurs. En cas d'invalidation de la conception, des boucles plutôt néfastes de re-conception sont nécessaires impactant des choix faits plus en amont, provoquant la perte de temps.</td></tr>
<tr><td>La simulation de compréhension a pour objectifs d'expliquer ou de déterminer l'origine de phénomènes observés et non compris jusque là. Pour cette raison, elle intervient, soit très tôt sur des problèmes constatés lors d'essais physiques ou issus de retours d'expériences d'anciens projets, soit très tard lorsqu'un produit est commercialisé et qu'un certain nombre de retours non prévus sont constatés.</td></tr>
<tr><td rowspan="2">La simulation dite active car elle est utilisée en continu et très intégrée dans le processus de conception. Dans ce contexte, elle est présentée comme un élément central et guide les concepteurs en les assistant dans leurs choix et en leur permettant de prospecter vers de nouvelles solutions.</td><td>La simulation d'optimisation vise, à partir d'une solution proposée, d'agir sur différentes grandeurs géométriques ou physiques afin de déterminer une variante répondant mieux au besoin, ou disposant de caractéristiques nettement améliorées (par exemple, pour un mousqueton dont le besoin est de tenir à une certaine charge l'optimisation de forme permet de réduire sa masse tout en conservant ses caractéristiques de mécanique de tenue). Cette approche est généralement itérative et permet aux utilisateurs d'atteindre des compromis ou d'identifier des solutions auxquelles ils n'auraient pas pensé.</td></tr>
<tr><td>La simulation indicative qui est présente tout au long du processus de conception et qui est utilisée pour soutenir l'acte de conception et aider le concepteur dans ses choix en lui permettant de tester rapidement et simplement un grand nombre d'architectures. Ces simulations sont généralement itératives et sont utilisées dans le cadre de modèles paramétrés (ou variationnels) et d'approches d'intégration CAO/Calculs. Un point particulièrement intéressant concerne l'utilisation de la simulation indicative, très tôt dans le processus de conception sur des modélisations paramétrées génériques de composants, permettant aux utilisateurs de déterminer les grandes tendances et les grandes architectures qui seront retenues pour la suite d'un projet. Ces simulations comprennent les activités de simulation à base de cas et d'innovation énoncés par Charles.</td></tr>
</table>

Annexe 6 : Les acteurs de la simulation

De façon générale, les tâches de simulation numérique sont assurées par deux types d'acteurs différents [Troussier, 1999] :

- Le non-expert qui utilisera des outils d'automatisation des processus de simulation et de traitement lui permettant de tirer des conclusion sur sa proposition de solution, sans avoir de compétence particulière dans le domaine d'expertise de simulation qu'il traite. Il se focalise sur les modifications géométriques du composant à simuler mais ne s'occupe pas (ou très peu) du maillage, du choix du type d'éléments, des chargements, de la méthode de calcul, et du solveur qui réalisera ces calculs. Généralement, pour perturber le concepteur le moins possible, on intègre au maximum les processus de simulation dans l'outil ou les interfaces du logiciel de conception qu'il à l'habitude de manipuler. La simulation se fait de façon "boite noire" et est presque totalement transparente pour le concepteur.

- L'expert sera amené à effectuer des calculs poussés et maitrise l'ensemble du processus et des données relatives à son calcul. Généralement, ses tâches sont peu automatisées et nécessitent des connaissances pointues dans le domaine physique concerné, ainsi qu'une maitrise des logiciels qui lui serviront à étudier le comportement d'un composant. A l'inverse de l'utilisateur précédent, l'expert calcul ne manipule presque pas de géométrie pour se concentrer sur le processus de simulation (maillage, mise en donnée, etc.).

Un troisième type profil d'acteur prend de plus en plus d'importance chez les industriels. Cet acteur possède à la fois les connaissances des concepteurs et aussi les connaissances de simulation (sans pour autant forcément être un expert). On peut le qualifier d'intégrateur, dans la mesure où il est chargé de la conception de composants et utilise la simulation pour valider ses choix via des processus itératifs de modifications. Ce lien, plus fort entre les activités de conception et de simulation, permet une meilleure prise en compte des contraintes inhérentes à ces deux domaines de conception et permet ainsi d'optimiser le développement d'un produit. Ce type de profil peut, soit être un concepteur ou un calculateur qui se diversifie et évolue vers l'autre domaine de conception, soit une personne qui à un profil mixte à la base de sa formation. Si l'expert en conception et en simulation reste indispensable, ce troisième profil est aujourd'hui privilégié chez les industriels.

Annexe 7 : Les types de modèles utilisés en fonction de l'utilisation de la simulation dans le processus de conception

Le Trade-off : Les types de modèles de calculs utilisés sont souvent des modèles experts 0D-1D, c'est-à-dire une représentation du système, par un schéma de type relationnel ou fonctionnel (schéma logique), mais représentatif du comportement réel. L'avantage de la simulation 0D-1D (aussi appelée simulation système) est de pouvoir simuler rapidement et simplement un système complet très tôt dans le développement d'un produit. Les résultats de ces simulations permettent aux décisionnaires et aux concepteurs de faire des choix et de définir les « grandes lignes » du produit à concevoir ainsi qu'un ensemble de spécifications à respecter. A titre d'exemple, sur un moteur automobile, ce type d'approche permet de définir l'architecture moteur globale (nombre de cylindres, cylindrée, etc.), la puissance, la consommation…

Le pré-dimensionnement : On manipule des modélisations géométriques 3D des composants peu détaillés, voire même génériques à tout types de moteur que l'on va par exemple modifier à l'aide du paramétrage des modèles pour tester de façon itérative de nombreuses solutions. Généralement, on utilise de très nombreux modèles métiers, chacun dédié à un composant en particulier pour une prestation donnée (un domaine d'expertise de la simulation), souvent présentés comme des ateliers. Par exemple, un modèle CAO/CAE générique piston pour une étude thermique, un modèle CAO/CAE carter pour une étude thermomécanique, etc. Ces approches sont aujourd'hui très utilisées dans le domaine automobile et aéronautique et conduisent à manipuler un très grand nombre de modèles métiers et experts hétérogènes, mais permettent de fiabiliser le processus de conception en identifiant les meilleurs architectures à développer.

Le dimensionnement : La simulation intervient surtout pour figer la conception et la valider au travers de modèles 3D complexes.
Chez certains industriels, le dimensionnement peut etre dissocier de la validation. Le premier concernant principalement l'étude des composants issus des fournisseurs et le second la validation finale de la conception. C'est le cas chez le constructeur PSA- Peugeot Citroën.

Annexe 8 : Exemple de la place de la simulation dans le processus de conception chez le groupe EADS

La Figure 124 illustre l'enchainement des phases du processus de conception dans lesquelles la simulation numérique est utilisée. Cette figure permet de mieux comprendre les finalités de chaque phase, détaillées ci-dessous :

- Trade-off : Spécifier le besoin et l'architecture
- Pré-dimensionnement : fixer les architectures
- Dimensionnement : conception détaillée.

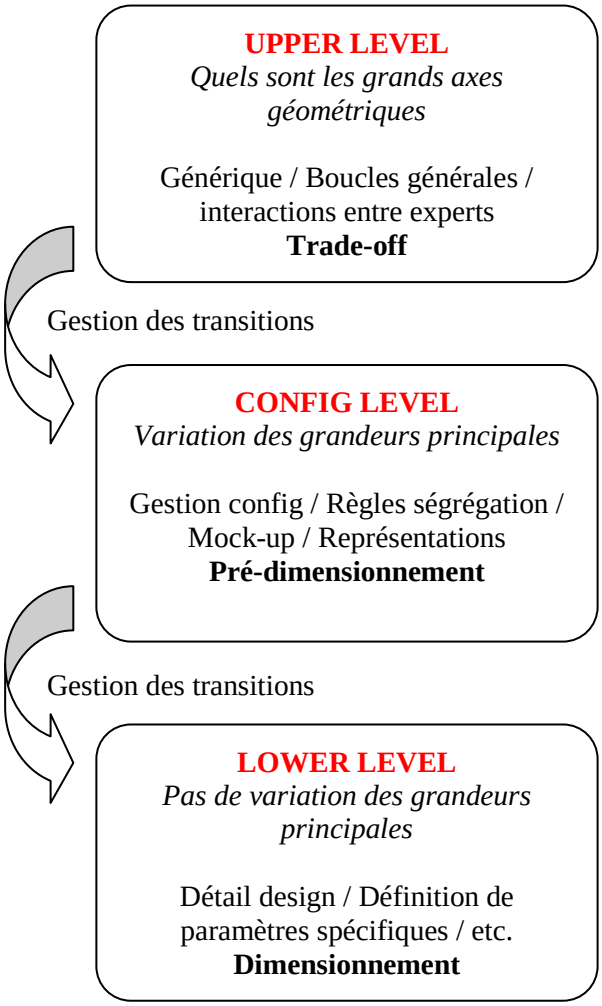


Figure 124 : interactions entre les différentes phases du processus de conception relativement à l'utilisation de la simulation et à la collaboration des acteurs

Ainsi, le constat réalisé chez EADS révèle que les interactions entre les acteurs d'un projet sont maximum dans les phases de Trade-off et le pré-dimensionnement. Dans ces phases, les concepteurs et calculateurs manipulent et échangent des paramètres et des règles métiers ; ils utilisent notamment des bornes de paramètres pour converger le plus vite possible vers une solution.

Annexe 9 : EGDS – la prise en compte des boucles de simulation dans l'analyse des résultats

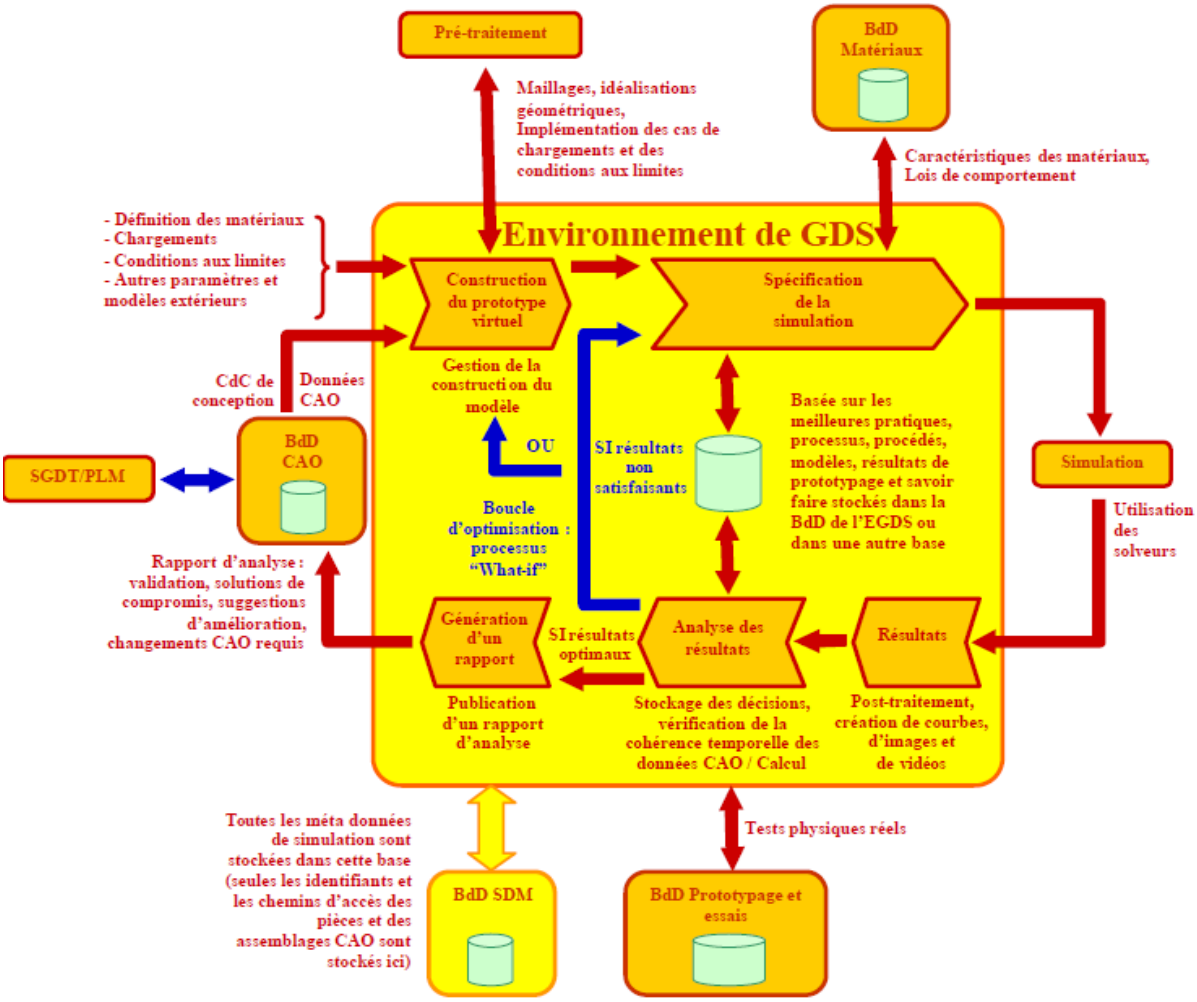


Figure 125 : EGDS [Charles, 2005]

Au terme de la simulation, l'application de traitement délivre des données brutes sous forme de fichiers au format natif. L'acteur de la simulation peut alors lier ces derniers aux autres données de la boucle de simulation sous forme de méta-données et ouvrir ces fichiers avec l'application de post-traitement, de façon à extraire les résultats pertinents dans le cadre de son projet de simulation. Tous les rapports, courbes, vidéos et fichiers qui résultent de cette activité de synthèse des résultats peuvent être ajoutés aux données de la boucle.

Une fois les données récupérées dans l'EGDS, l'environnement consulte alors le SGDT, de façon à vérifier si le modèle géométrique utilisé comme base de la simulation n'a pas subi de changement durant la durée du cycle de simulation, et le signale aux utilisateurs si le cas se présente. Cette vérification est nécessaire dans la mesure où le processus de simulation est souvent coûteux en temps du fait des délais requis par les analyses et de ce fait, il se peut que la conception ait évolué dans une autre direction, ce qui rend alors caduque les résultats obtenus. Le dépouillement automatique de résultats génériques tels que les contraintes et déplacements maximaux relevés peut être assuré par l'EGDS que nous proposons, mais requiert alors que le format des résultats soit conforme à STEP. Sans l'emploi du format STEP AP209 [ISO 10303-209:2001], le développement d'un module de lecture et d'extraction des résultats, à partir d'un format natif, est du coup nécessaire.

Annexe 10 : Le format STEP (AP 203, AP214)

Le standard pour l'échange de données de produit, STEP (STandard for the Exchange of Product model data en anglais), ou **ISO 10303,** porte sur la représentation et l'échange de données du produits et a pour objectif d'intégrer les processus de conception, de développement, de fabrication et de maintenance de ces derniers. Cette méthode permet donc de définir une représentation non ambiguë des données du produit, interprétable par tout système informatique, et couvrant tout le cycle de vie des produits. Cet objectif nécessite deux choses :

- La définition d'un format ouvert, interprétable par tout système informatique et indépendamment du système ayant généré les données.
- La couverture d'un très vaste domaine de connaissances, correspondant à l'ensemble des catégories de produits (pièces élémentaires, assemblages, mécanismes, etc.), selon le point de vue de tous les métiers (électronique, mécanique, etc.), et à toutes les phases du cycle de vie (conception, analyse, fabrication, maintenance, démantèlement).

Un système décrit dans STEP, l'est en fonction de ce qu'il est, ce qu'il fait, et de ce qu'il devient. En limitant STEP aux types de données couvertes par IGES et SET (Standard d'Echange et de Transfert), l'intérêt de STEP réside dans la gestion de l'échange des tables de nomenclatures, l'historique des modifications, l'ensemble des décompositions du produit en versions multiples, et des niveaux d'autorisation. On a donc un lien fort entre la description géométrique du produit, le contrôle de sa configuration, et la gestion de ses données techniques.

Plus précisément, STEP considère des "ressources intégrées". Elles constituent la base permettant de construire les descriptions des données d'un produit dans un domaine d'application. Ces ressources sont génériques et interprétées en fonction du contexte d'utilisation. Leur utilité est d'assurer l'unicité des définitions, afin de garantir la cohérence des interprétations, quel que soit le domaine d'application. Les ressources sont de deux types :

- génériques : définies indépendamment du contexte,
- d'application : classes de définitions pouvant être commune à plusieurs applications partageant une même partie du domaine considéré.

Ressources génériques :

- Fundamentals of product description
- Geometric and topological representation
- Representation structures
- Product structure configuration
- Materials
- Visual presentation
- Shape tolerances
- Form features
- Process structure, property and representation

Ressources d'applications :

- Dessin
- AEC
- E/E connectivity
- Modelisation par éléments finis
- Cinematique

Définir une application dans la méthodologie STEP requiert l'usage d'un "protocole d'application". Un protocole d'application donné définit le contexte, le domaine (type de produit, phase du cycle de vie prise en compte, type de données, exploitation et disciplines concernées), les besoins en information et les méthodes STEP utilisées pour exprimer ces besoins. Au fil du temps et de l'apparition de nouveaux

domaines d'activités ayant besoin de STEP, le nombre de protocoles d'application augmente. Pour chacun des produits considérés, ces derniers prennent en compte trois types de modèles :

- le modèle de référence (spécification de la structure conceptuelle et des contraintes utilisées pour décrire les informations liées au produit),
- le modèle interprété (application des contraintes afin de satisfaire le modèle de référence),
- le modèle d'activité qui décrit les activités et processus permettant de produire les données liées au produit dans un contexte spécifique).

Les protocoles d'application le plus souvent utilisés, et d'ailleurs ceux qui sont le plus supportés par les logiciels de CAO sont les AP 203 et AP 214.

Annexe 11 : Dimensions des connaissances

Dans la mesure où les connaissances peuvent être partagées par les membres d'une organisation, on peut représenter, sous une dimension ontologique, différents niveaux de connaissances. Dans les aspects sociaux, coexistent plusieurs notions : d'organisation, d'équipe, de groupes, auxquels il faut ajouter les communautés de pratiques (CoP) développées par [Wenger, 1999].
La communication au sein de l'entreprise se fait donc par le biais de relation entre les connaissances des personnes (connaissances individuelles) ainsi que les connaissances de l'entreprise (connaissances collectives).

Les connaissances individuelles :
La notion de connaissances individuelles regroupe l'ensemble des croyances, de l'expérience, de l'inné d'un individu, quel que soit le contexte dans lequel il les utilise et se référant souvent au caractère tacite de la connaissance. Néanmoins, les connaissances individuelles peuvent avoir une dimension collective, dans la mesure où les compétences mises en jeux le sont pour une organisation, une entreprise. De plus, une partie des connaissances individuelles est également explicite, elle s'exprime sous forme d'idées, de discours, de métaphores, de modèles ou schéma se matérialisant sous forme de notes personnelles, de consignes sur des feuilles volantes, etc. [Grundstein, 2007]. Cet auteur définit également les "connaissances organisationnelles" comme étant la somme des connaissances individuelles des individus, ce qui potentiellement permet d'aborder la notion de connaissances collectives, mais il manque une dimension organisationnelle pour que le transfert soit possible.

Les connaissances collectives :
Les connaissances collectives sont les connaissances partagées par un groupe dans une organisation. Ce n'est pas seulement la somme des connaissances individuelles détenues par les employés, mais cela concerne aussi la gestion et les théories de l'organisation elle-même. Il est alors normal d'avoir, comme pour les connaissances individuelles, une dualité explicite/tacite des connaissances collectives. On peut considérer que les connaissances collectives explicites se matérialisent dans des documents, des processus, des modèles partagés et exploitables par d'autres acteurs (dans le cadre de l'entreprise ce sont souvent les experts), définissant un champ de connaissances métier en particulier. Les connaissances collectives tacites sont plus difficiles à identifier et résultent d'un apprentissage collectif et mis en commun au travers de tâches particulières dans un contexte donné. Elles peuvent notamment êtres présentes au sein des CoP qui est un regroupement informel d'individus ayant en commun un domaine de spécialisation précis [Wenger et Snyder, 2000]. Ce sont souvent des experts métiers, impliqués dans la même pratique qui communiquent régulièrement entre eux au sujet de leurs activités. Ses membres partagent leur expérience et leurs connaissances avec une liberté et une créativité qui favorisent l'émergence de nouvelles connaissances et compétences.

Annexe 12 : Le problématique industrielle chez PSA et chez EADS

Dans cette annexe, nous proposons une expression de besoin de deux industriels du projet ADN, par rapport aux problèmes de collaborations entre modèles métiers, rencontrés dans le processus de conception de produit.

Les besoins PSA :

Chacun des ateliers d'architecture, dédiés à un organe du GMP en particulier, est un outil à part entière utilisant un panel de connaissances métier sous forme de paramètres et de relations clés de la modélisation et de la simulation : Paramètres géométriques, Courbes, Surfaces, Paramètres de maillage, Paramètres de MED (Mise En Donnée), Abaques, Règles métiers, Bonnes pratiques, Modèles génériques (ou Templates), etc. Aujourd'hui, il existe chez PSA plusieurs ateliers d'architecture dédiés à différentes expertises métiers (Structure, Acoustique, Thermomécanique, etc.) liées aux différents organes du Groupe Moto-Propulseur (GMP) automobile : Atelier Moteur, Atelier Bas moteur, Atelier ligne d'arbre, Atelier distribution, Atelier Haut Moteur, Atelier Carter cylindre, Atelier culasse, etc. Cependant, l'utilisation de ces différents ateliers d'architecture fait émerger de nouvelles difficultés et donc de nouvelles problématiques. En effet, les ateliers d'architecture utilisent tous des connaissances communes (paramètres, règles métier, instances de paramètres, etc.) et par conséquent dépendent tous les uns des autres. Il apparaît indispensable de pouvoir échanger et synchroniser ces connaissances entre différents domaines d'expertise et différents sites d'ingénierie, tout en suivant le processus de conception et de simulation d'un ensemble d'organes. Aujourd'hui chez PSA, les connaissances sont échangées de façon déstructurée et anarchique, lors de réunions, d'envoi de courriers électroniques, ou de discussions, ce qui ne garantit aucune capitalisation, trouble les processus d'ingénierie, car provoquant des erreurs qui se traduisent, par exemple, par des lancements de calculs longs, mais avec de mauvais paramètres, car non synchronisés entre les différents ateliers d'architecture dédiés à la modélisation, au calcul et à la simulation concernés.
Un nouveau besoin apparaît, à savoir celui de mettre en interrelation les différents outils dédiés au processus de conception et de simulation, de permettre à plusieurs personnes, à différents endroits, issus de différents métiers, de partager et de travailler de façon collaborative avec les mêmes paramètres, relations et instances de paramètres.

Les besoins EADS :

Deux évolutions majeures des technologies aéronautiques amènent à reconsidérer les façons dont les systèmes (équipements, servitudes hydrauliques, électriques, etc.) sont intégrés dans les aéronefs. D'une part, la structure des aéronefs fait maintenant largement appel aux matériaux composites. Les matériaux composites, structurellement très performants, offrent aussi des nouvelles opportunités d'intégration. Par contre, des challenges différents concernant la protection électromagnétique et l'incidence thermique des systèmes apparaissent. D'autre part, les aéronefs sont de plus en plus "électriques" avec à la fois, une génération de puissance électrique, une puissance de calcul et une instrumentation plus importantes.
Le développement de ces nouveaux concepts d'intégration système est propre à donner un avantage compétitif à EADS face à ses concurrents d'aujourd'hui (principalement américains), mais aussi ceux de demain (canadiens, brésiliens, russes, chinois, japonais).
Le développement des nouveaux concepts d'intégration envisagés doit être appuyé par de nouveaux outils d'ingénierie d'intégration car ils amènent à coupler plus finement et plus tôt dans le processus de développement, différentes physiques (structure, thermique, etc.) et différents domaines (conception, simulation, production & maintenance) correspondant à différentes phases de vie du produit. Ce couplage multi-physique et temporel fait appel à des boucles de simulation qui concernent l'intégration spatiale et la gestion des contraintes environnementales (vibratoires, thermiques, électromagnétiques, etc.). De façon à explorer de multiples concepts et à valider avec confiance des décisions d'intégration innovante, il est visé de pouvoir mettre en œuvre, de façon rapide et à la volée, des simulations multi-métiers sur la base d'hypothèses ou de propositions de concepts variées. Le challenge est double car il s'agit à la fois de construire des couplages entre des domaines distincts mais surtout de les construire dynamiquement car ils viennent en support d'un processus de conception d'architecture créatif.

Annexe 13 : Réseau sémantique[13]

Un réseau sémantique est une structure de graphe dont la fonction est l'encodage des connaissances taxonomiques concernant des objets ainsi que leurs propriétés. Les cartes heuristiques ou les arbres heuristiques sont un des moyens de représenter des réseaux sémantiques. Un réseau sémantique est un outil intéressant et similaire à une "ontologie limitée", dans la mesure où, dans une ontologie, les liens entre objets ou concepts peuvent être de n'importe quelle nature, alors que dans un réseau sémantique ils ne peuvent être que de trois types :

- les arcs d'agrégation (appelés aussi "liens isa (IS A)": par exemple, l'homme *est un* humain et la femme *est un* humain.
- les arcs de composition (appelés aussi "liens asa (HAS A)": par exemple, l'homme *à un* squelette, le chat *a un* squelette.
- les arcs d'instanciation (appelés aussi "liens iko (IS A KIND OF)": par exemple, Marc *est un type* d'homme, Olivier *est un type* d'homme.

Les réseaux sémantiques permettent d'avoir une vue d'ensemble des connaissances et de leurs relations dans le cadre d'un processus de conception. A partir de cette analyse, il est ainsi possible d'effectuer des regroupements de concepts ou d'objets permettant, par la suite, d'élaborer une méthodologie avec plus de justesse.

[13] Référence Wikipedia

Annexe 14 : L'approche "bottom-up" KCMethod

La Figure 126 illustre l'orientation plutôt "bottom-up" de KCMethod. A partir d'un nuage d'informations, caractérisées par un ensemble d'ICEs, on construit dynamiquement des configurations de connaissances correspondant à plusieurs vues métiers spécialisées. La collaboration entre ces configurations est assurée par un autre type de configuration permettant la mise en commun des instances d'ICEs partagées.

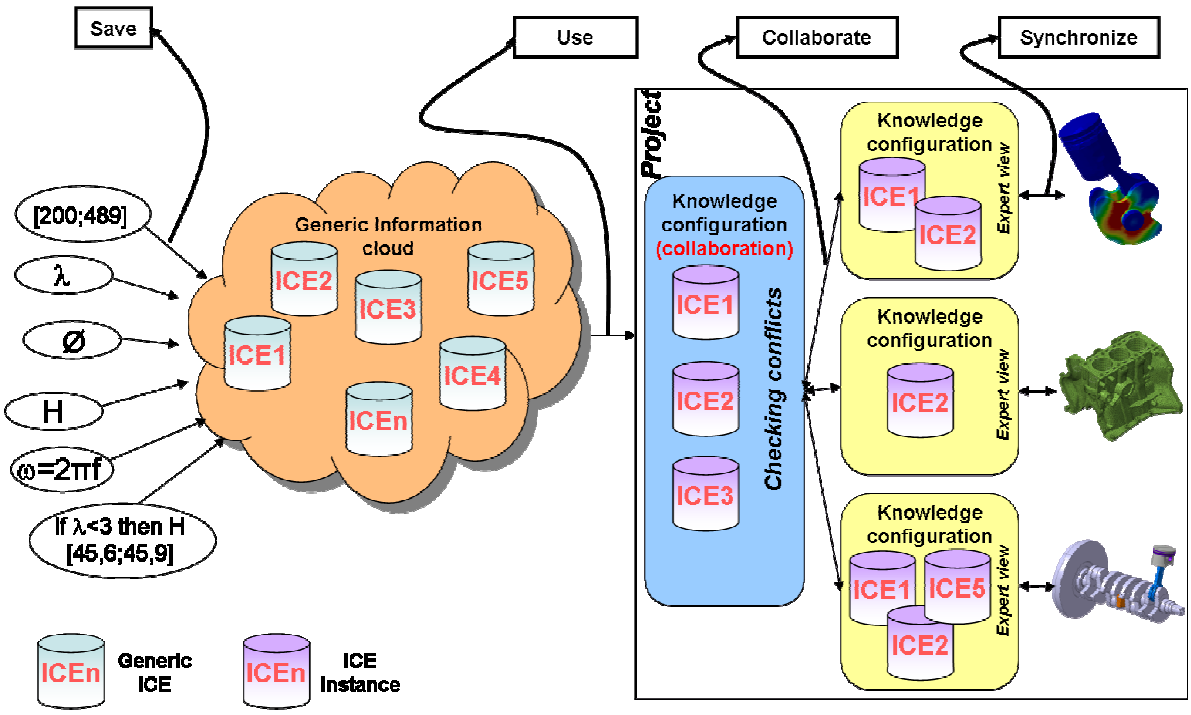


Figure 126 : approche "bottom-up" KCMethod

Annexe 15 : différentes possibilités de création des configurations

Les utilisateurs disposent de plusieurs possibilités pour créer des configurations. Nous avons vu, dans le chapitre 3, la création de configurations depuis la "feuille blanche", c'est-à-dire en partant directement d'une sélection d'ICEs dans la base d'informations.
Il est possible de créer des configurations depuis la base de configurations génériques. L'utilisateur sélectionne dans cette base le "Template" de configuration qu'il veut utiliser et l'instancie dans son projet.
Il peut également créer une configuration en réutilisant une configuration existante dans son projet. Il peut la créer en définissant sur une nouvelle version, en copiant partiellement ou entièrement une configuration ou en agrégeant des instances d'ICEs utilisées dans plusieurs configurations.
Enfin, il peut créer une configuration à partir d'un modèle métier, c'est-à-dire parcourir les paramètres du modèles afin d'identifier ceux qui existent dans la base d'ICEs, dès lors la configuration se génère automatiquement.
Pour chaque possibilité de création et particulièrement pour la création depuis un modèle métier, il est impératif de vérifier si les paramètres et les contraintes que l'utilisateur veut instancier existent bien dans la base d'ICEs. Sinon, ils ne pourront pas être utilisés dans la configuration.

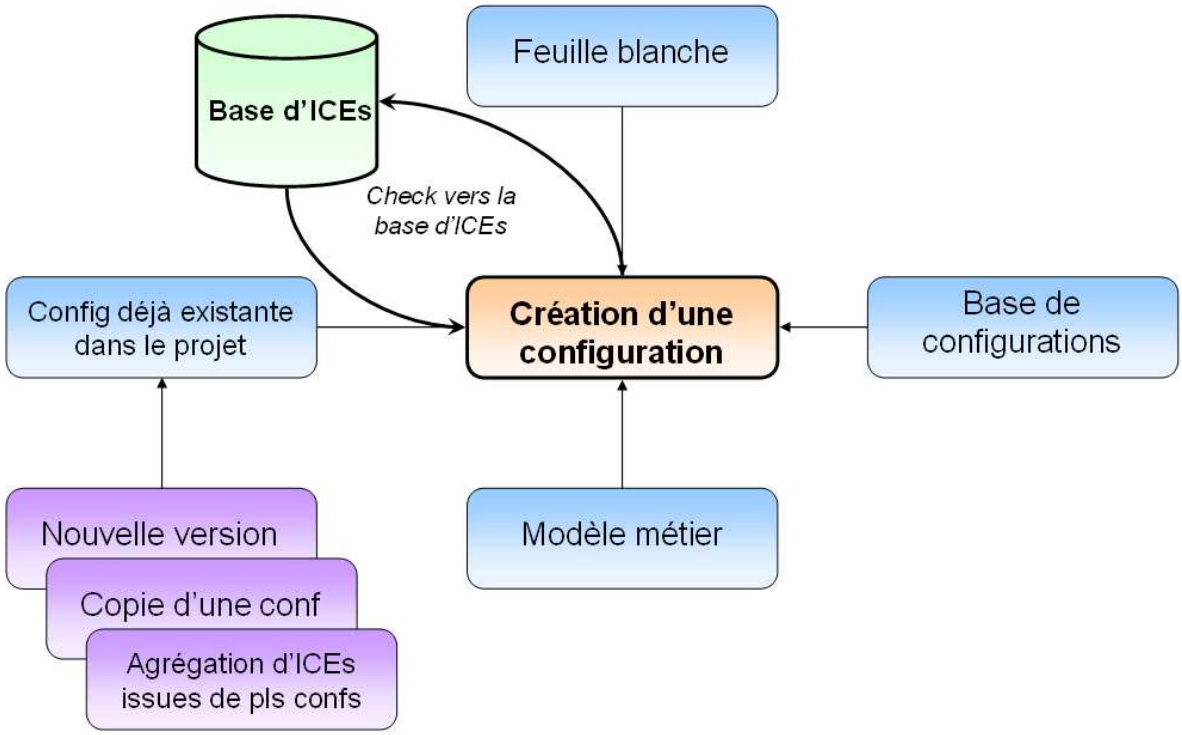


Figure 127 : possibilités de création d'une configuration de connaissance

Annexe 16 : Moteur de CSP et moteurs d'inférences

Dans le cadre de la résolution des conflits, dans et entre les configurations de connaissances, nous utilisons des moteurs permettant de comparer des valeurs de paramètres, vérifier des contraintes mais aussi assurer des mécanismes de propagation, voire d'optimisation. Deux types d'approches sont envisagés : les moteurs d'inférences et les moteurs de CSP (Constraint Satisfaction Problem).

Les moteurs d'inférences :

Un moteur d'inférences est un logiciel faisant partie d'un système expert ; il permet de réaliser des raisonnements logiques et d'en déduire des conclusions afin de résoudre des problèmes spécifiques. Pour ce faire, il utilise une base de connaissances qui se compose d'une base de faits et d'une base de règles :

- La base de faits contient un ensemble d'informations, considérées comme vraies à un instant donné, qui concernent le problème traité. La base de faits se comporte comme la mémoire temporaire de travail du système expert ; elle est modifiée au fur et à mesure de la progression du raisonnement, et éventuellement vidée lorsqu'une consultation se termine.
- La base de règles décrit les relations existant entre les faits et la manière dont le système va raisonner. Une règle est du type : Si P (un fait) ALORS Q (un nouveau fait) où P est nommé la prémisse et Q la conclusion de la règle.

Le processus de résolution de ces moteurs est basé sur une combinaison des mécanismes de Modus Ponens (chaînage avant, méthode de déduction permettant de dégager de nouvelles conclusions en partant des prémisses. A partir de deux formules vraies, en déduire une troisième vraie) et Modus Tollens (Chaînage arrière, par opposition cela permet de remonter aux axiomes à partir des conclusions. De deux formules fausses, nous pouvons en déduire une troisième fausse).

Les moteurs de CSP :

Un problème de satisfaction de contraintes (ou CSP pour Constraints Satisfaction Problem) désigne un type de problème pouvant être décrit à l'aide d'un ensemble de variables définies sur un domaine et d'un ensemble de contraintes qui s'appliquent sur ces variables. De manière formelle, un CSP est un triplet {X, D, C} tel que :

- X = {x1, x2, …, xi} est l'ensemble des variables du problème
- D = {d1, d2, …, dj} est l'ensemble des domaines des variables. D étant la fonction qui associe à chaque variable xi son domaine D(xi) c'est-à-dire l'ensemble des valeurs que peut prendre xi
- C = {c1, c2, ..., ck} est l'ensemble des contraintes ; chaque contrainte ck appartenant à C est une relation entre certaines variables x, restreignant ainsi les valeurs que peuvent prendre simultanément ces variables. Les contraintes peuvent être de type arithmétique (<, =, >, etc.), logique (implication, union, etc.) ou encore sémantique.

Exemple : Soit le CSP(X, D, C) suivant :

- X = {a,b,c,d}
- D(a) = D(b) = D(c) = D(d) = {0,1}
- C = { $a \neq b$, $c \neq d$, $a+c < b$ }

Ce CSP comporte 4 variables a, b, c et d, chacune pouvant prendre 2 valeurs (0 ou 1). Ces variables doivent respecter les contraintes suivantes : a doit être différente de b ; c doit être différente de d et la somme de a et c doit être inférieure à b.

Résoudre un CSP consiste à trouver une instance de chaque variable (i.e. lui affecter une valeur) de telle sorte que toutes les contraintes soient satisfaites. On dit qu'une affectation est consistante si elle ne viole aucune contrainte, et non-consistante si elle viole une ou plusieurs contraintes. Les principaux algorithmes de résolution de CSP sont basés sur des techniques *d'exploration exhaustive* de l'espace

des affectations couplées à des techniques de *propagation de contraintes* qui simplifie la résolution en réduisant l'espace des recherches.

De façon générale, un moteur de CSP, capable de traiter les domaines infinis et continus, est basé sur cinq niveaux d'algorithmes (en fonction de leurs capacités) :

(5) Branch & Bound (Bissection)
(4) Consistance (propagation réduction)
(3) Opérateurs algébriques expo log cos…
(2) Arithmétique de Moore (+ - x /)
(1) Arithmétique flottante (qui s'occupe des arrondis etc.)

Tableau 10 : les cinq niveaux d'un moteur CSP

Les CSP correspondent davantage à notre problématique de cohérence. Si, dans la maquette ADES, c'est PROLOG qui est utilisé, dans la suite des travaux nous envisageons d'implémenter l'outil KADVISER ou ILOG.

- **Gnu Prolog**

Gnu prolog (http://www.gprolog.org/) est un compilateur Prolog libre (sous licence GNU GPL) développé par Daniel Diaz. Il repose sur une machine virtuelle (WAM) et intègre un solveur de contraintes pour la résolution de domaines finis. Gnu-Prolog compile, dans un premier temps, un programme Prolog en un fichier WAM qui est traduit en langage de bas niveau indépendant (mini-assembleur) spécialement conçu pour Gnu Prolog. Il offre l'avantage de produire un exécutable de petite taille qui exécute rapidement un programme Prolog. A noter que la partie Prolog de Gnu-prolog est conforme au standard ISO de Prolog auquel il ajoute un certains nombres d'extensions telles que des variables globales et des sockets. En outre, comme le solveur est ouvert et extensible (contrairement à CHIP par exemple), il est possible à l'utilisateur de définir ses propres contraintes qui viennent s'ajouter au nombreuses contraintes déjà prédéfinies (arithmétiques, boolean, symbolique, etc.). Gnu Prolog existe pour de nombreuses plateformes, telles que linux, mac, solaris ou windows. A noter qu'il existe d'autres implémentations possibles pour Prolog, telles que SWI-Prolog, SICTus-Prolog, Prolog II, Visual-Prolog, Quintus-Prolog,

- **Kadviser**

Kadviser est un progiciel pour le développement d'applications à base de connaissances métiers, créé en 1988 par la société Kadetech. Il est développé aujourd'hui par la société Nimtoth et a été conçu pour tenir compte du caractère évolutif de la connaissance. Il utilise un moteur d'inférences d'ordre 1 à propagation de contraintes, et permet de générer des applications métier d'aide à la conception en fournissant un ensemble des fonctionnalités nécessaires à la visualisation, à l'exploitation, à la maintenance du savoir-faire. En outre, les applications Kadviser disposent de connexions unidirectionnelles ou bidirectionnelles avec des outils de bureautique (Excel par exemple), de bases de données et certains modeleurs CAO, comme CATIA V5 de Dassault. L'architecture souple de Kadviser permet également de l'intégrer facilement sous forme de composant à une architecture logicielle existante, comme dans un navigateur internet, un programme, un PLM (Product Lifecycle Management) ou un ERP (Enterprise Resource Planning)

- **Ilog**

Ilog solver est une librairie commerciale c++, développé par Ilog (IBM) et destiné à la programmation par contrainte. Il supporte les variables entières, flottantes, booléennes et ensemblistes. Les contraintes incluent les opérations d'égalité et d'inégalité (=, ≤, ≥, <, >), les opérations arithmétiques (+, −, ∗, /), les opérations ensemblistes (subset, superset, union, intersection), les opérations logiques (or, and, not, xor), les contraintes cardinality et element et les méta-contraintes (conjonctions et disjonctions de contraintes, ordre sur les contraintes). En outre, les contraintes non linéaires peuvent être résolues à l'aide de méthodes d'approximation par intervalles. Ilog Solver offre une recherche arborescente avec retours-arrières et un algorithme de « branch and bound » couplé avec de nombreuses stratégies de recherche. ILog est une partie de Ilog optimisation, suite dans laquelle il peut coopérer avec CPLEX

un moteur de programmation mathématique. Il a déjà été utilisé pour des applications de grandes tailles dans des secteurs aussi variés que la télécommunication, le transport, la défense et la production industrielle. A titre d'exemple, nous pouvons citer son utilisation dans la gestion des terminaux d'aéroports (British Airways), le commerce électronique (CAMIF), ou encore l'optimisation de découpe de pellicule-photos pour Kodak.

Annexe 17 : Illustration du processus de validation des configurations de connaissances

Point de vue de l'utilisateur

L'utilisateur ne possède des droits que sur le statut de **sa** configuration utilisateur, néanmoins il peut visualiser le statut de la configuration squelette ou le statut des autres configurations utilisateurs (Figure 128).

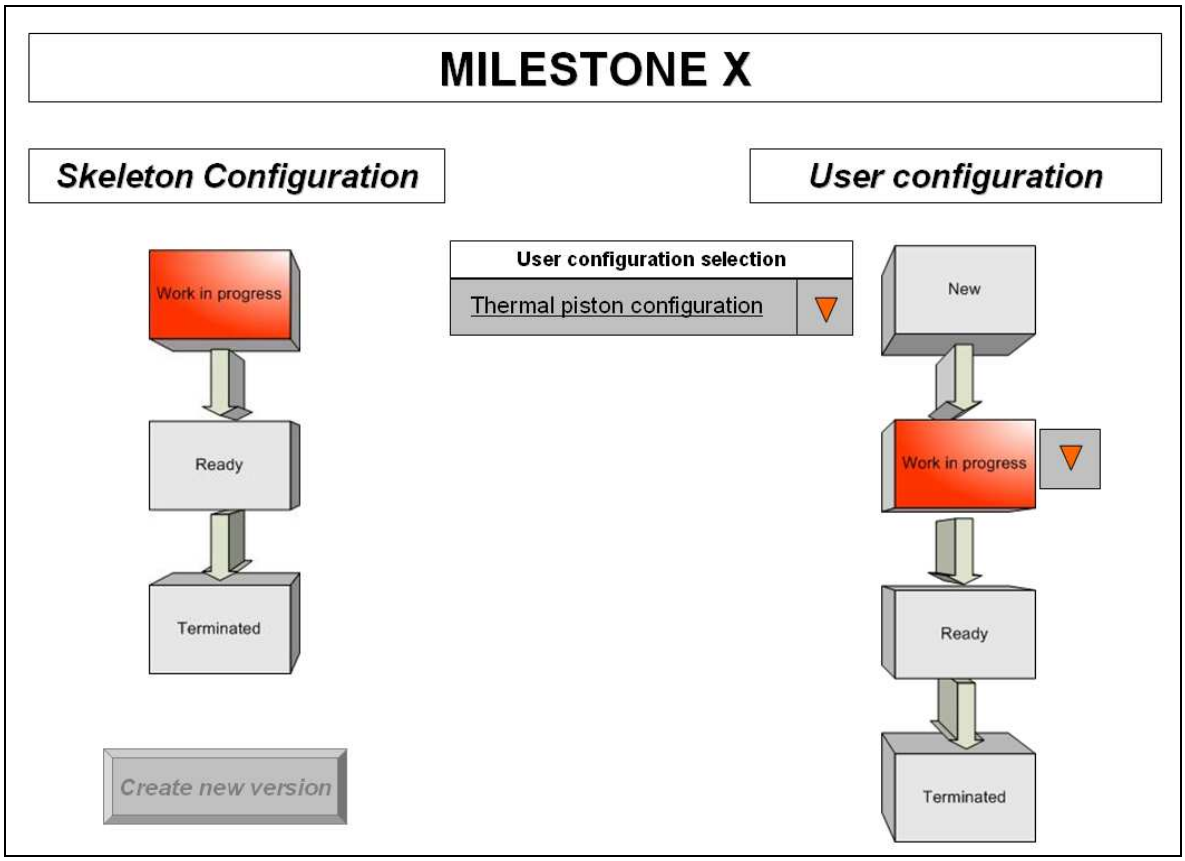


Figure 128 : validation vue par un utilisateur

Quand il crée sa configuration utilisateur, par défaut, elle possède le statut "New". L'utilisateur peut commencer à travailler sur sa configuration, c'est-à-dire la synchroniser avec son modèle métier et créer des versions. Néanmoins, tant que le responsable jalon n'a pas accepté la création de sa configuration (ce qui modifie son statut en "WIP"), il ne peut pas publier de version vers le squelette et modifier le statut de la configuration en "Ready". Le statut "Ready" signifie que cet utilisateur à terminé son travail sur sa configuration et il ne peut plus modifier la version publiée, ni en publier une autre, la configuration est comme figée (mais pas validée). A ce moment, seul le responsable jalon peut faire repasser le statut de la configuration utilisateur en WIP pour que l'utilisateur puisse refaire des modifications.

Point de vue du responsable jalon

Le responsable jalon possède des droits lui permettant de modifier le statut de la configuration squelette mais aussi le statut des configurations utilisateurs (Figure 129).

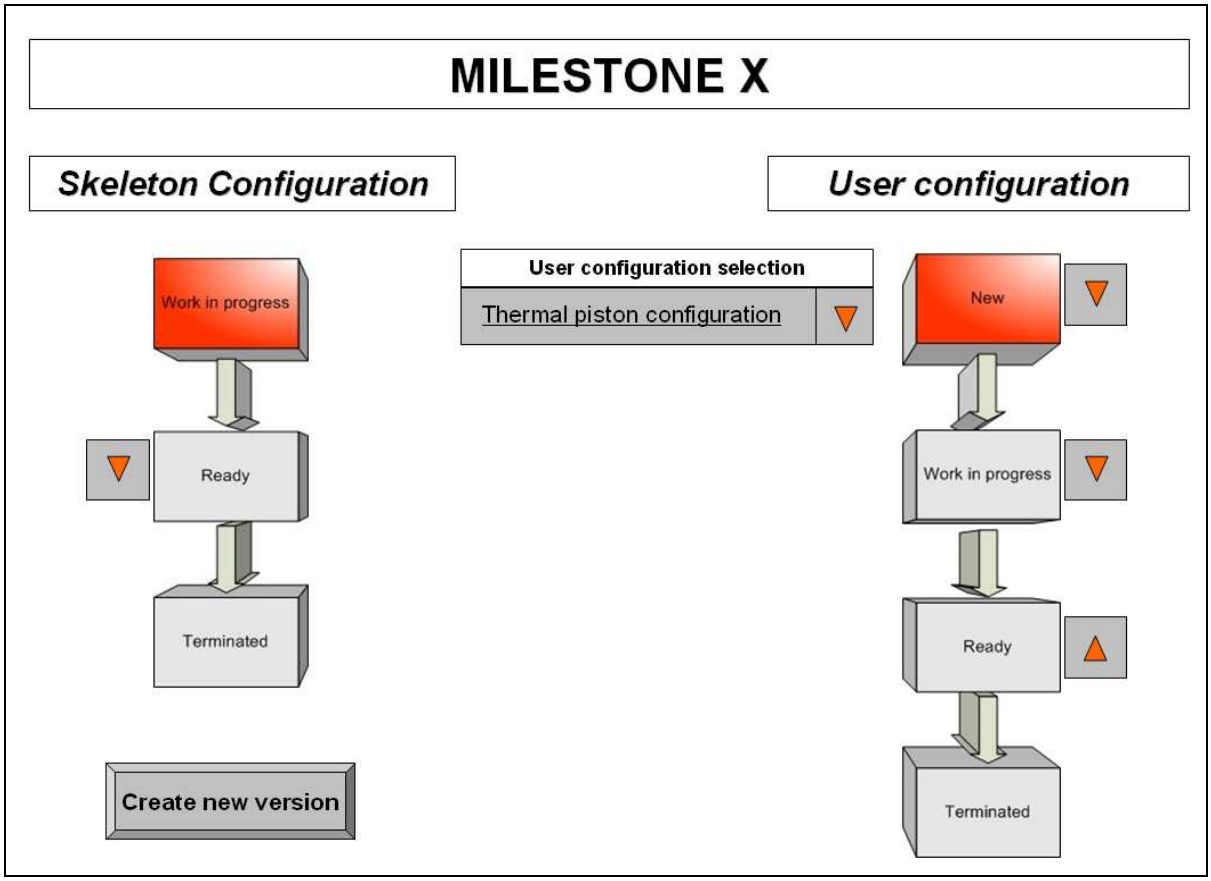


Figure 129 : validation vue par le responsable jalon

A sa création, la configuration squelette est en statut "WIP". Quand une configuration utilisateur est créée, elle devient sélectionnable dans le menu de "sélection des configurations utilisateur" et son statut apparaît. Par défaut, une configuration utilisateur possède le statut New, ce qui signifie qu'elle doit être validée par le responsable jalon.
Le responsable de jalon peut figer une configuration utilisateur officialisée dans la configuration squelette en modifiant le statut de "WIP" à "Ready". Il peut aussi faire le processus inverse pour "rendre disponible à nouveau" la configuration.
Une fois les conflits traités et les statuts de configurations utilisateurs en "Ready", le responsable de jalon est prévenu et il peut valider officiellement l'ensemble des configurations du jalon. A ce moment, il à la possibilité de créer une nouvelle version de la configuration squelette et de placer cette nouvelle version dans un jalon.

Annexe 18 : Package AccessControlDefinition d'après les modèles RBAC

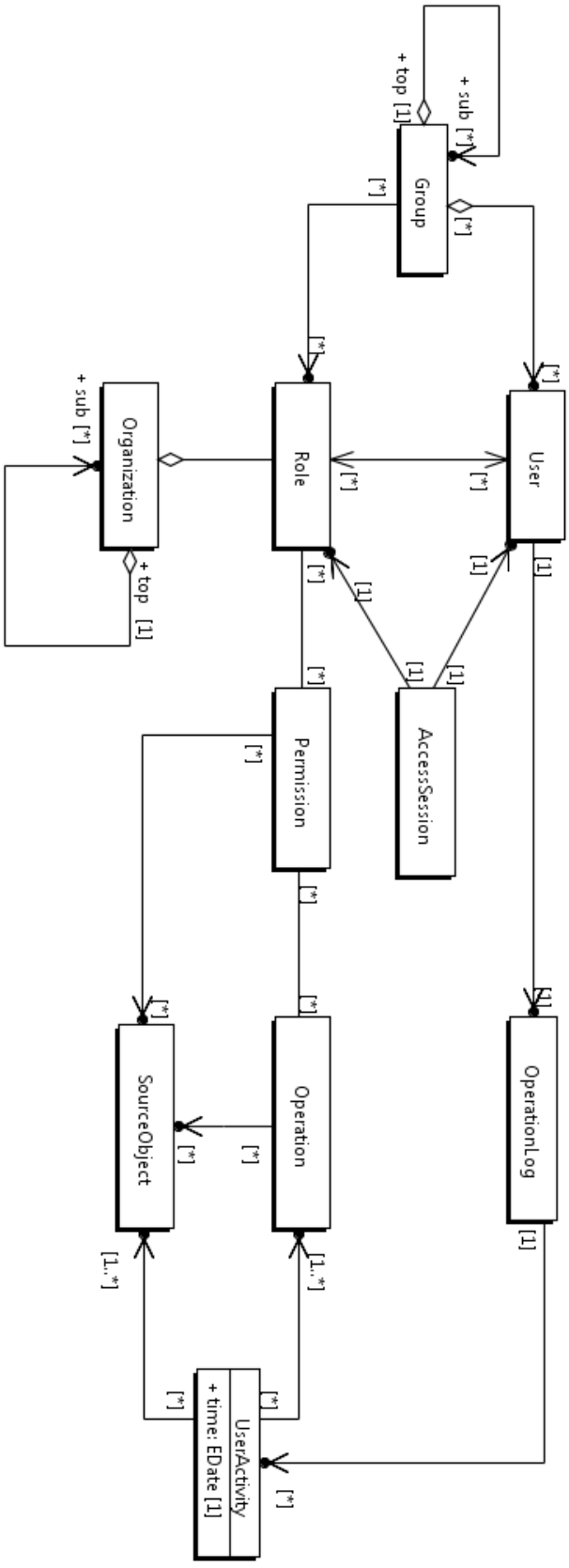


Figure 130 : package AccessControlDefinition

Annexe 19 : Diagramme d’activité Add ICEInstance

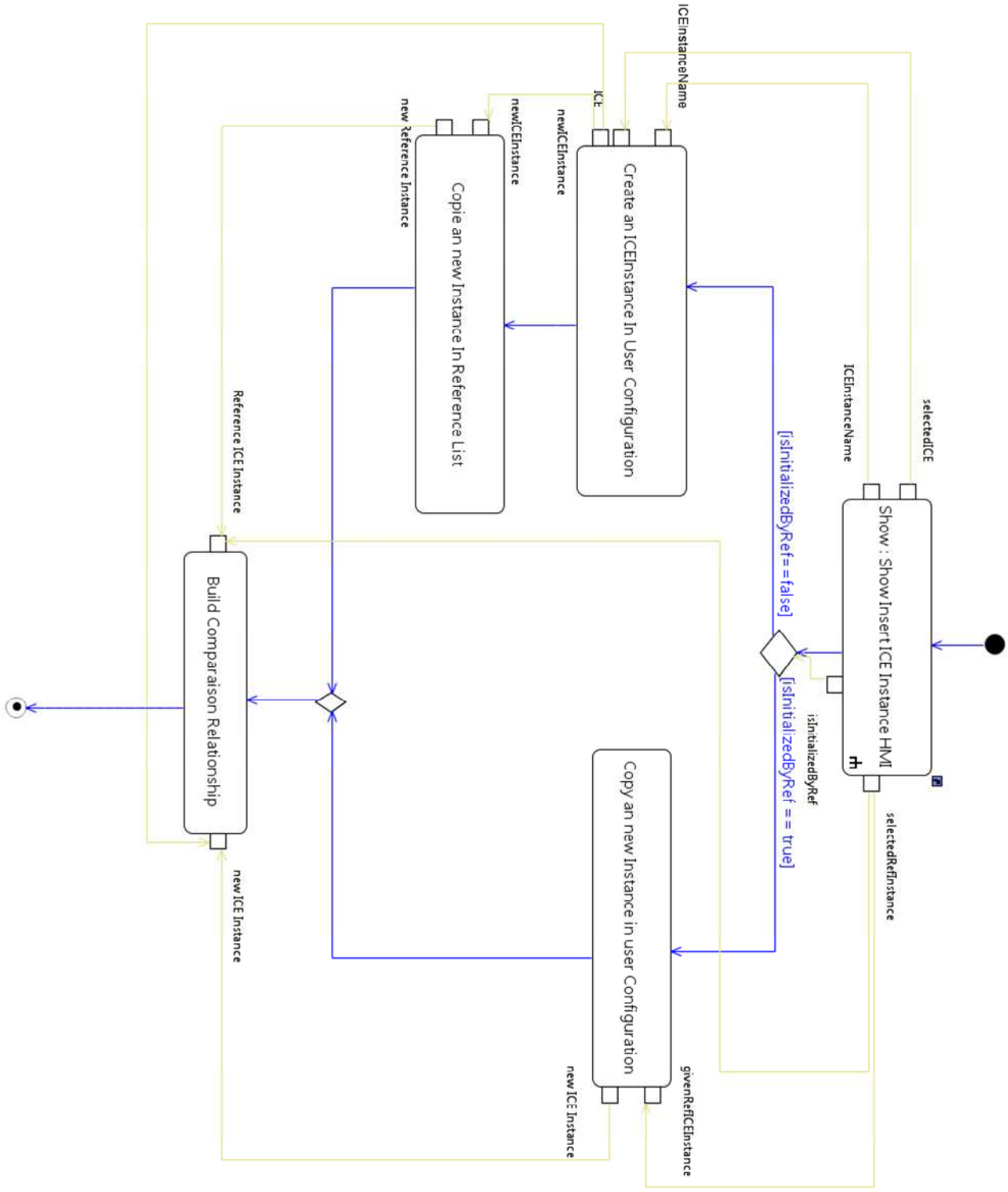


Figure 131 : diagramme Add ICEInstance

Annexe 20 : Diagramme d'activité Insert ICEInstance HMI

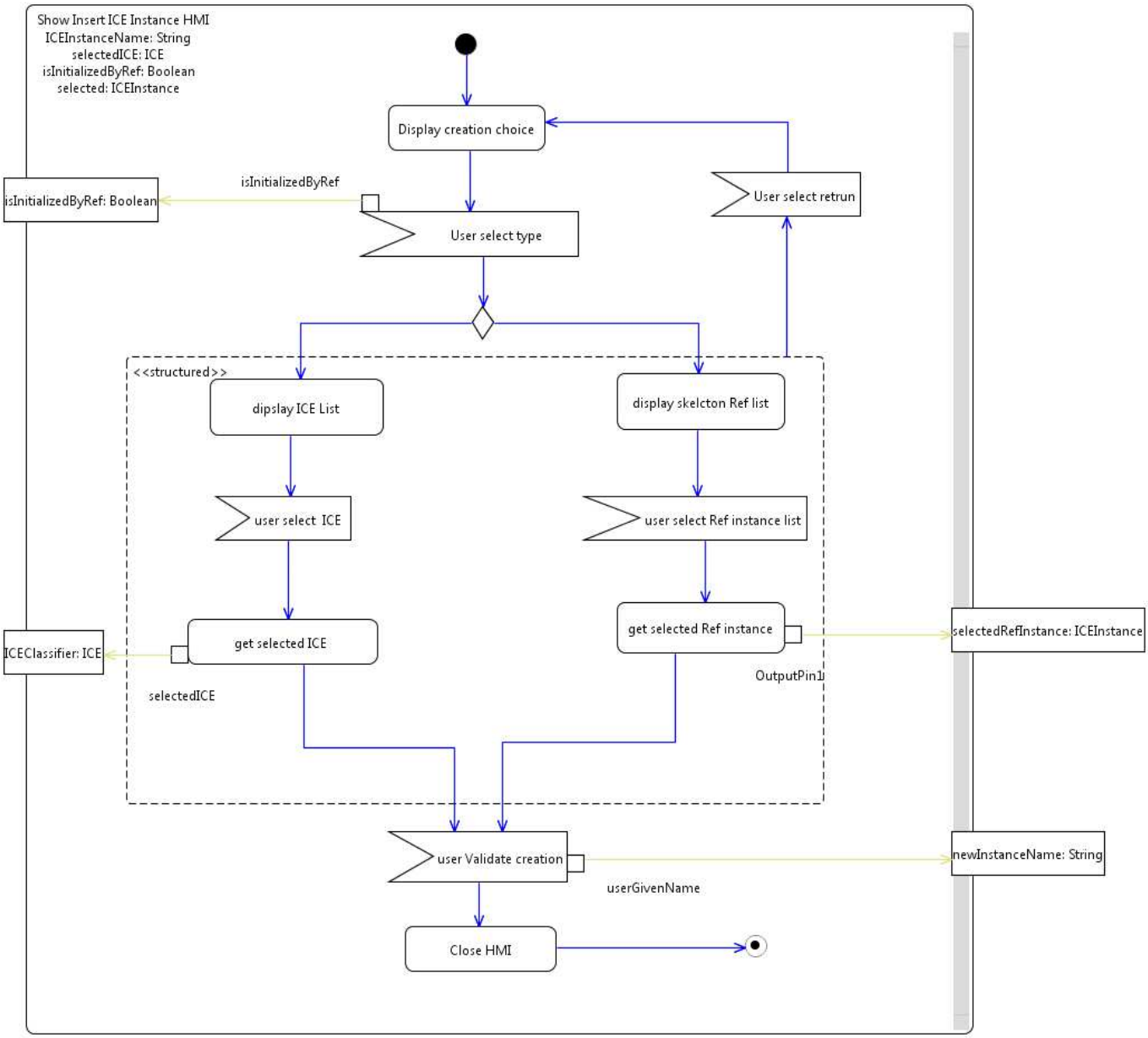


Figure 132 : diagramme Insert ICEInstance HMI

Annexe 21 : Diagramme d'activité SkeletonConfStatusValidation

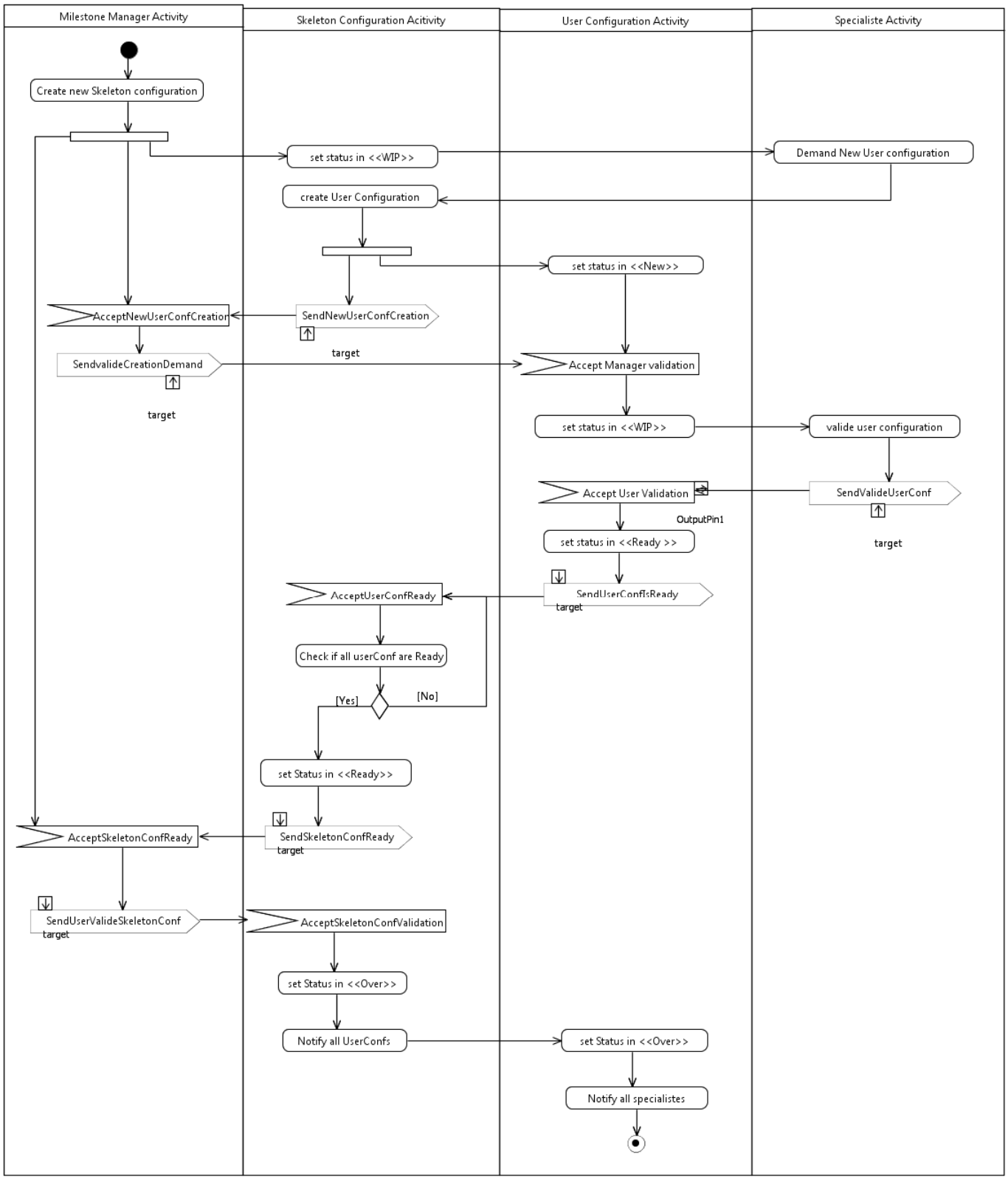


Figure 133 : diagramme SkeletonConfStatusValidation

Annexe 22 : Diagramme d’activité Conflict in SkeletonConfiguration

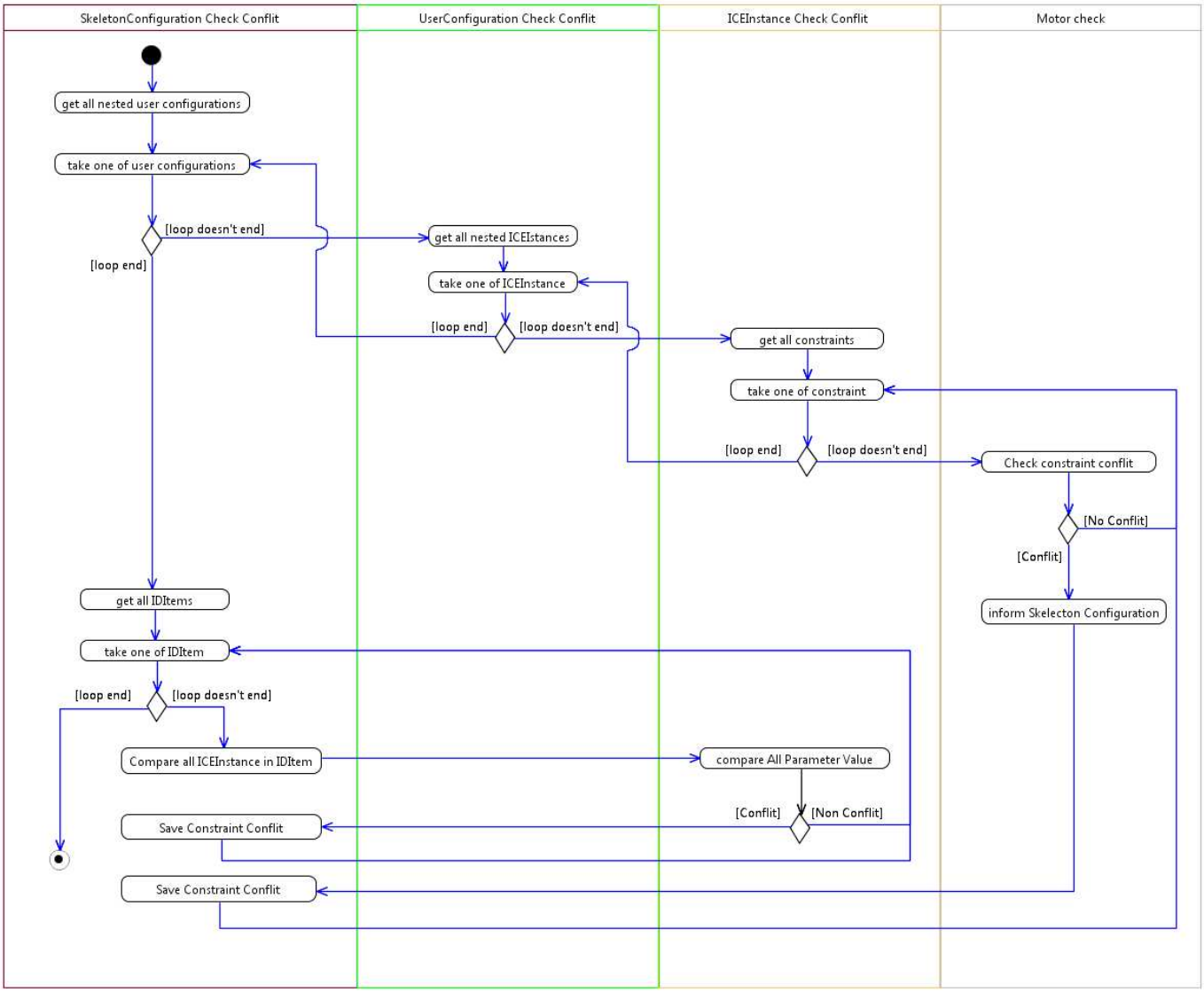


Figure 134 : diagramme Conflict in SkeletonConfiguration

Annexe 23 : Illustration du concept de Knowledge Domain

La Figure 135 illustre un exemple d'utilisation du concept de KnowledgeDomain. Conformément à la partie KCMCore, les ICEs et les ConfigurationTemplate sont stockées dans le KnowledgeDomain, de sorte à être réutilisées dans différents projets appartenant également au KnowledgeDomain.
Dans l'exemple, la configuration Template "CalculPolaireBielle" contient quatre instances d'ICEs issues des ICEs :

- Géométrie Tourillon
- CoursePiston
- ChargementPolaireBielle
- DéplacementMaxBielle

La configuration "CalculEffortMaxBielle" n'est composée que de trois instances d'ICEs.

Chaque ICE, crée dans un projet du KnowledgeDomain, s'ajoute à ce dernier, de sorte à ce que tous les projets qu'il stocke aient accès à l'ensemble des ICEs créées. Le mécanisme est similaire pour la création des ConfigurationTemplate.

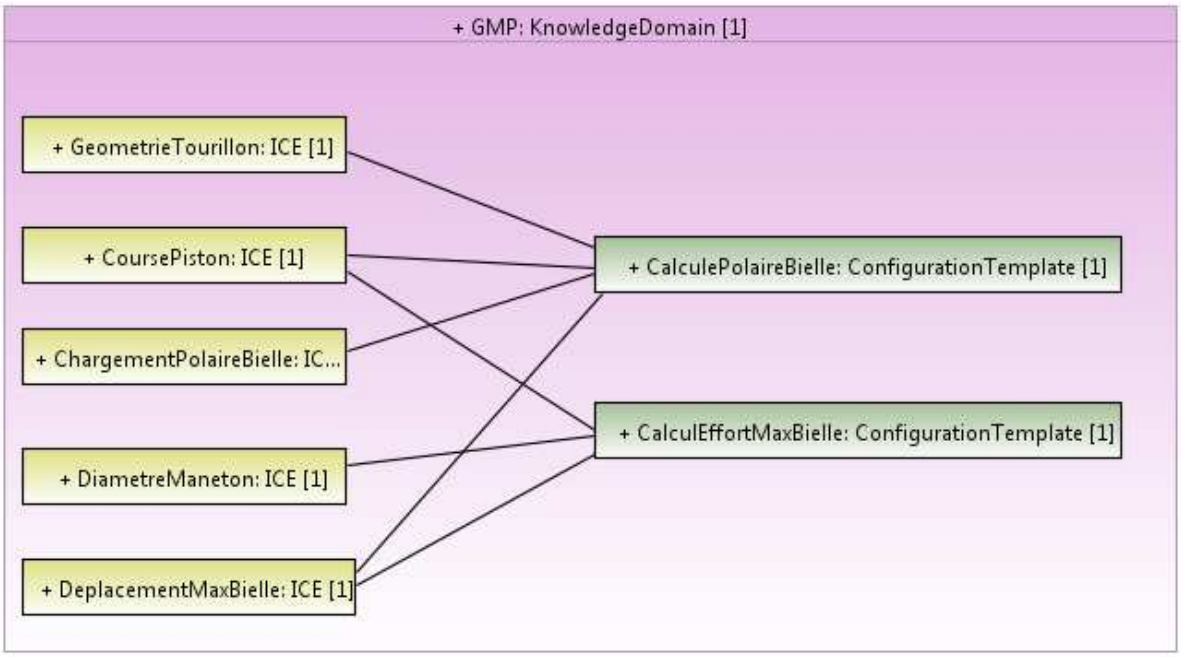


Figure 135 : exemple de KnowledgeDomain

Annexe 24 : Présentation synoptique de l'architecture de TSC

Version "ZEN"

"ZEN" est le nom de code de l'architecture industrielle de TSC (Figure 136).

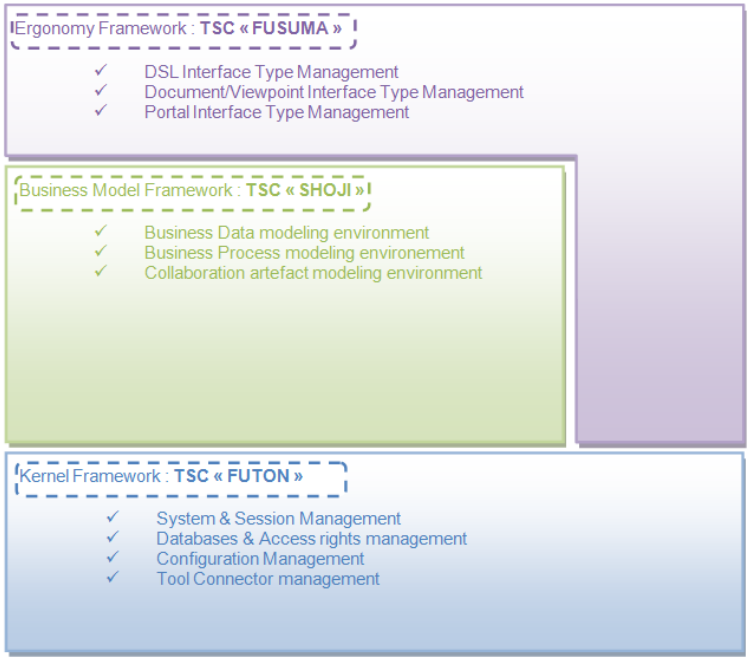


Figure 136: schéma synoptique de la structure de TSC

Le noyau de TSC : "FUTON"[14]

Ce dernier fournit les mécanismes de base de gestion d'un système d'informations de collaboration :

- **Gestion des sessions et des systèmes :** Une session de travail au sein TSC permet de gérer le système d'informations de collaboration pour le développement d'un système. Il sera possible, au sein de TSC, de définir pour un système d'éventuels sous-systèmes dont les systèmes d'informations de collaboration seront manipulés au travers d'autres sessions de travail au sein de la même ou d'autre bases de données de gestion TSC. TSC fournira donc les services permettant de gérer efficacement le triptyque "Session/Système/Base de données hébergeante".

- **Gestion des connecteurs avec les outils et systèmes d'informations externes :** TSC fournira un environnement permettant d'implémenter les services de synchronisations avec des outils/base de données externes. Trois modes de synchronisation seront implémentables :
  - le mode "IMPORT/EXPORT" : le mode le plus simple à implémenter et servant à définir des services d'initialisations de données (utiles surtout dans les contextes de migration de données existantes),
  - le mode "SYNCHRO" : permettant de définir des services de synchronisation entre la (ou les) base(s) de données TSC et un autre (ou plusieurs) outil(s) externe(s). Dans ce mode, des mécanismes de gestion de fusions devront être implémentés si besoin,
  - le mode "TRANSACTION" : permettant de définir des services de synchronisation "temps réel". Dans ce contexte, la modification de données métiers dans le cadre d'une session TSC entraîne simultanément la modification des données les outils externes associées et vice-versa.

[14] Un ***futon*** (布団) est matelas japonais constitué de couches de coton.

- **Gestion de configuration des modèles et des artefacts :** Les bases de données des outils, en interfaces avec une base de données TSC, constituent des modèles à part entière. Une base de données TSC contient un ensemble de modèles "instances" des modèles métiers (méta-modèles) mis en jeu. Elle contient pour chaque système un modèle "instance" d'un système de processus (meta-modèle) mis en œuvre pour le développement de ce système. Une session TSC peut être vue comme l'association de plusieurs modèles :
    - les modèles "instances" des modèles métiers,
    - le modèle "instance" du système de processus techniques,
    - le modèle des artefacts de collaboration (partagé par les modèles "instances" des modèles métiers et le modèle "instance" du système de processus.
    - Tous les modèles au sein de TSC, ainsi que les artefacts, seront gérés en configuration, c'est-à-dire que les relations de cohérences entre eux seront outillées :
    - Gestion de la cohérence des versions de modèles avec les versions de leur meta-modèle,
    - Navigation.

Les environnements de modélisations métiers de TSC : "**SHOJI**"[15]
TSC fournira trois environnements de modélisation différents :

- [PROCESS] : Celui de type "BPMN" permettant de décrire un ensemble de processus techniques,
- [DATA] : Celui de type "ECORE" permettant de décrire un ensemble de modèles métiers,
- [COLLAB_ARTEFACT] : Celui de type "PATTERN" permettant de décrire la structure d'artefacts de collaboration et leurs relations avec les modèles métiers.

Ces trois environnements de modélisations seront interdépendants. En effet, l'environnement de modélisation [PROCESS] utilise des références à des artefacts de collaborations définis au travers de l'environnement de modélisation [COLLAB_ARTEFACT], qui lui-même agrège des données métiers définies avec l'environnement de modélisation [DATA].

Les environnements de modélisations d'IHM de TSC : "**FUSUMA**"[16]
TSC fournira trois environnements distincts de modélisation d'IHM qui permettront d'obtenir des widgets qui seront utilisables directement depuis une session TSC et/ou au travers d'activités de processus techniques. Selon l'environnement de modélisation, on pourra définir des widget de type :

- [Portail] : type de widget idéal pour offrir une visibilité sur le modèle "instance" du système de processus pour le développement d'un système donné, pour accéder aux configurations, pour "exécuter" d'autres widgets TSC ou pour lancer des outils externes dans la bonne configuration de données.
- [Document] : type de widget idéal pour offrir un point de vue documentaire d'un artefact et/ou de données métiers
- [DSL] : Type de widget idéal pour offrir une interface de type DSL sur un ensemble d'artefacts et/ou de données métiers.

[15] un ***shōji*** (障子) est une paroi ou une porte constituée de papier translucide montée sur une trame en bois.
[16] Le ***fusuma*** (襖) est un écran opaque coulissant muni d'une poignée utilisé pour redéfinir l'espace d'une pièce ou servir de porte dans l'habitat traditionnel japonais.

Annexe 25 : Exemple d'IHM ADES dans le cadre du projet ADN

La Figure 137 propose une autre IHM de l'application ADES. Sur la gauche, l'utilisateur dispose des boutons lui permettant d'accéder aux objets majeurs. Au centre, l'écran change en fonction des actions de l'utilisateur. Sur la droite, le cadre rappel à l'utilisateur le projet dans sur lequel il travaille et son rôle dans ce projet.
Sur l'écran de la Figure 137, l'utilisateur est sur la première page faisant suite à sa connexion sur ADES : la page d'accueil appelée tableau de bord. Sur cette page, apparaissent un certain nombre d'informations sur lesquelles il souhaite avoir un suivi (similaire à des widget).

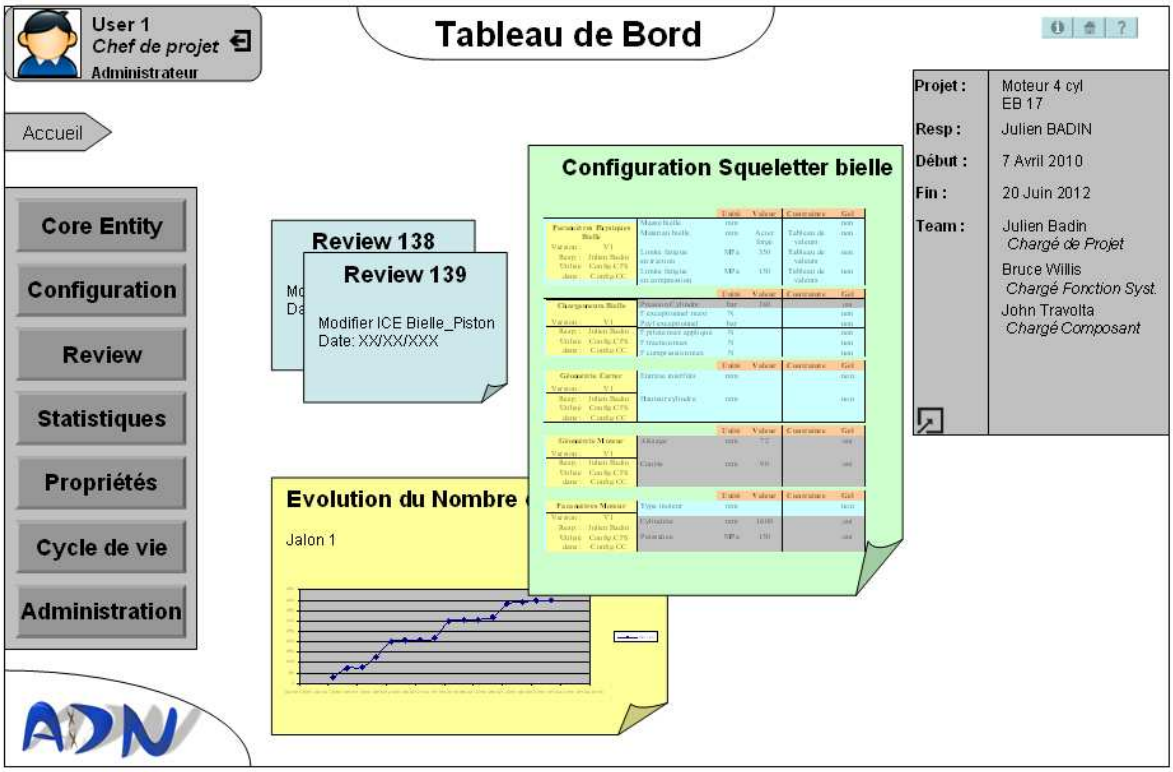
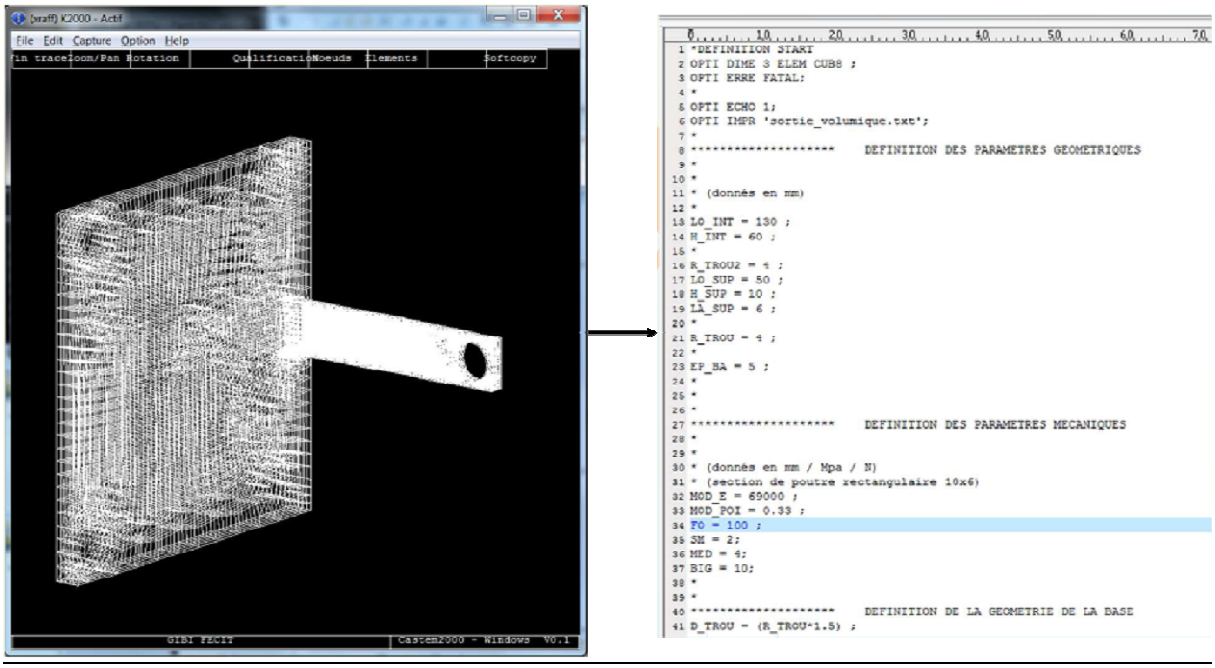


Figure 137 : exemple d'IHM ADES

Annexe 26 : Modèle CASTEM support de fixation




```
1 *DEFINITION START
2 OPTI DIME 3 ELEM CUB8 ;
3 OPTI ERRE FATAL;
4 *
5 OPTI ECHO 1;
6 OPTI IMPR 'sortie_volumique.txt';
7 *
8 ********************    DEFINITION DES PARAMETRES GEOMETRIQUES
9 *
10 *
11 * (données en mm)
12 *
13 LO_INT = 130 ;
14 H_INT = 60 ;
15 *
16 R_TROU2 = 4 ;
17 LO_SUP = 50 ;
18 H_SUP = 10 ;
19 LA_SUP = 6 ;
20 *
21 R_TROU = 4 ;
22 *
23 EP_BA = 5 ;
24 *
25 *
26 *
27 ********************    DEFINITION DES PARAMETRES MECANIQUES
28 *
29 *
30 * (données en mm / Mpa / N)
31 * (section de poutre rectangulaire 10x6)
32 MOD_E = 69000 ;
33 MOD_POI = 0.33 ;
34 F0 = 100 ;
35 SM = 2;
36 MED = 4;
37 BIG = 10;
38 *
39 *
40 ********************    DEFINITION DE LA GEOMETRIE DE LA BASE
41 D_TROU = (R_TROU*1.5) ;
```

Figure 138 : modèle élément finis volumique du support de fixation

Annexe 27 : Description des filières de calculs pour le cas d'étude GMP

- Conversion Pression Couple moteur (simulation système : Figure 139, Figure 140)

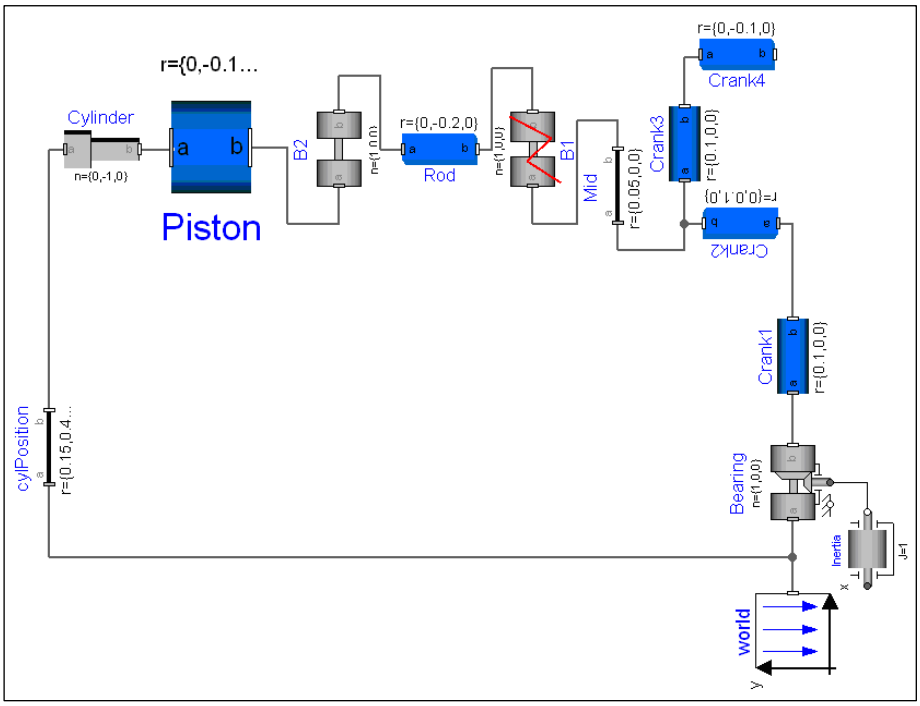


Figure 139 : schéma cinématique de l'attelage mobile

Le modèle permet de simuler rapidement le comportement de l'attelage mobile à partir des paramètres d'entrées du GMP issus du cahier des charges. A partir des paramètres de sortie, tels que vitesse du piston, accélérations induites, pression cylindre en fonction de l'angle du vilebrequin, etc., il est possible de tester et comparer plusieurs architectures.

Ce type de modélisation est utilisé très en amont dans les phases de pré-dimensionnement pour faciliter le choix des principales dimensions envisageables pour le GMP.

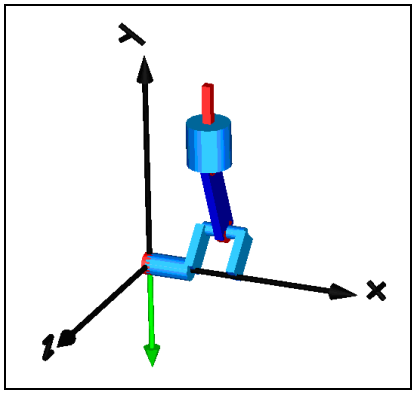


Figure 140 : "représentation 3D" du modèle de simulation système

- Atelier d'architecture GMP

Il s'agit d'un outil basé sur un modèle simplifié d'attelage mobile en 3D avec maillage volumique et simulation par EF (Figure 141).
Cet atelier contient un modèle CAO simplifié et paramétré de base moteur (carter + attelage mobile), maillé automatiquement en éléments volumiques, et 2 cas de calcul contenant un ensemble de blocages et une pression imposée sur le piston.

Dans le cadre du scénario, cet atelier permet le calcul des fréquences libres du système et des efforts maximum sur la tête de bielle en fonction de l'angle du vilebrequin.

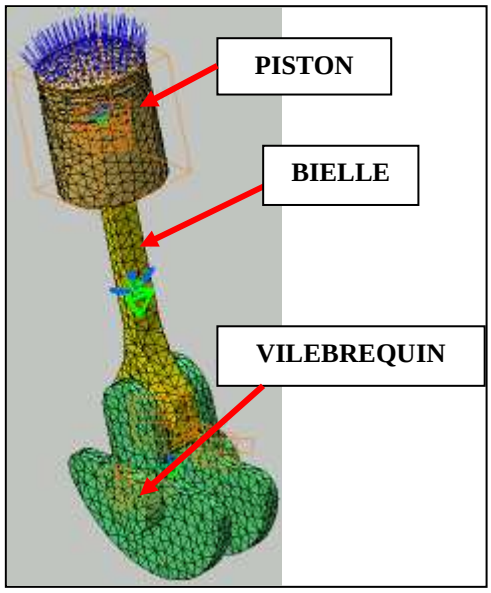


Figure 141 : ensemble Bielle-Piston-Vilebrequin

- Etude tenue de bielle (Figure 142)

Cet atelier permet de simuler le comportement de la bielle sollicitée en traction et en compression, pour analyser la résistance en fatigue et la déformation maximale.

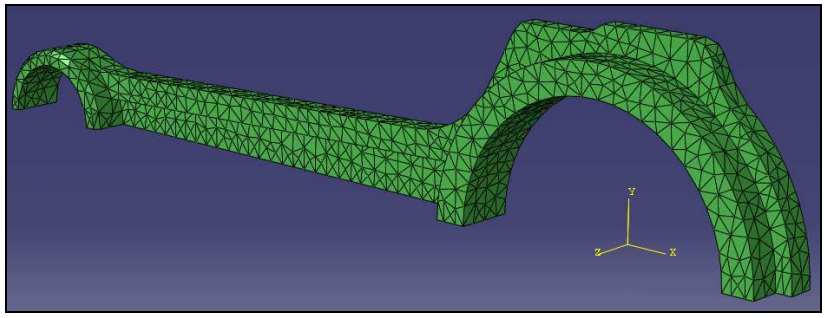


Figure 142 : modélisation EF de la bielle