École de technologie supérieure
Université du Québec

Université de Technologie de
Compiègne

THÈSE EN COTUTELLE
présentée à l'**École de technologie supérieure**
et à l'**Université de Technologie de Compiègne**
comme exigence partielle
à l'obtention du

Doctorat En Génie
Ph.D.

Grade de Docteur de l'Université
de Technologie de Compiègne

Présentée par **Matthieu BRICOGNE-CUIGNIÈRES**
Thèse dirigée par

Louis RIVEST

Nadège TROUSSIER

Préparée au sein des laboratoires

Laboratoire d'ingénierie des
produits, procédés et systèmes
(LIPPS) et Numerix

Unité de Recherche en
Mécanique - Roberval
UMR 7337
École Doctorale n°71
« Sciences pour l'ingénieur »
Champ disciplinaire
« Mécanique avancée »

MÉTHODE AGILE POUR LA CONCEPTION COLLABORATIVE MULTIDISCIPLINAIRE DE SYSTÈMES INTÉGRÉS : APPLICATION À LA MÉCATRONIQUE

Thèse soutenue publiquement le 13 février 2015, devant le jury composé de :

M. Lionel ROUCOULES
Professeur des Universités, Arts et Métiers ParisTech, centre d'Aix en Provence (président du jury)
M. Clément FORTIN
Professeur associé à l'École Polytechnique de Montréal et Senior Advisor to the President au Skolkovo Institute of Science and Technology de Moscou (rapporteur)
M. Frédéric NOËL
Professeur des Universités, Grenoble INP (rapporteur)
M. Benoît EYNARD
Enseignant chercheur, Université de Technologie de Compiègne
M. Mickaël GARDONI
Professeur, École de technologie supérieure de Montréal
M. Grégory HUET
Product Manager, Parametric Technology Corporation
M. Louis RIVEST
Professeur, École de technologie supérieure de Montréal (Co-Directeur de thèse)
Mme Nadège TROUSSIER
Professeur des Universités, Université de Technologie de Troyes (Co-Directeur de thèse)

PRÉSENTATION DU JURY

CETTE THÈSE A ÉTÉ ÉVALUÉE

PAR UN JURY COMPOSÉ DE :

M. Benoît EYNARD, membre du jury
Enseignant chercheur au laboratoire Roberval à l'Université de Technologie de Compiègne

M. Clément FORTIN, rapporteur
Professeur associé à l'École Polytechnique de Montréal et Senior Advisor to the President au Skolkovo Institute of Science and Technology de Moscou

M. Mickaël GARDONI, membre du jury
Professeur au département de génie de la production automatisée à l'École de technologie supérieure de Montréal

M. Grégory HUET, membre du jury
Product Manager chez Parametric Technology Corporation

M. Frédéric NOËL, rapporteur
Professeur des Universités au laboratoire Sciences pour la Conception, l'Optimisation et la Production de Grenoble (G-Scop) à Grenoble INP

M. Louis RIVEST, codirecteur de thèse
Professeur au département de génie de la production automatisée à l'École de technologie supérieure de Montréal

M. Lionel ROUCOULES, président du jury
Professeur des Universités au laboratoire des Sciences de l'Information et des Systèmes (LSIS) aux Arts et Métiers ParisTech, centre d'Aix en Provence

Mme Nadège TROUSSIER, codirecteur de thèse
Professeur des Universités au Centre de Recherches et d'Études Interdisciplinaires sur le Développement Durable (CREIDD) à l'Université de Technologie de Troyes

IL A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC

LE 13 FÉVRIER 2015

À L'UNIVERSITÉ DE TECHNOLOGIE DE COMPIÈGNE

REMERCIEMENTS

Les remerciements sont une rubrique que je parcours presque systématiquement lorsque je découvre des travaux de thèse car je trouve que les quelques phrases qu'ils contiennent en disent long sur leur auteur et sur le contexte dans lequel la thèse a été effectuée. Je vais à mon tour rendre hommage aux différentes personnes qui m'ont inspiré, poussé ou soutenu tout au long de ces… disons cinq années durant lesquelles j'ai pu murir ces travaux.

En tout premier lieu, je tiens à te remercier Céline pour avoir su me pousser, me motiver à enfin achever cette thèse. Car tu le répètes à qui veut l'entendre, seule une échéance ferme a pour effet de me contraindre à avancer à un rythme soutenu... et pour ce projet, tu t'es auto-attribuée le rôle de gardienne du temps ! Un grand merci également pour m'avoir permis de réaliser les séjours dans cette formidable Belle Province qu'est le Québec, alors même que nos filles demandaient encore tant d'attention. Mais c'est promis, je n'attaquerai plus jamais de front deux ans de travaux quelques mois après l'arrivée de notre seconde fille en parallèle d'une thèse…

Viennent ensuite bien entendu le binôme de choc constitué de Nadège et de Louis. Merci bien entendu à toi Nadège pour le temps consacré et tes précieux conseils, mais surtout pour m'avoir donné le goût et l'envie d'embrasser une carrière d'enseignant chercheur. Car c'est à toi que je dois cette envie de réaliser une thèse (dix ans déjà…), mais aussi de sauter le pas en quittant l'industrie pour revenir de l'autre côté des amphithéâtres. Je tiens également à te remercier très chaleureusement Louis pour m'avoir montré que l'on pouvait faire de la recherche « autrement ». Prendre le temps, aller au fond des choses, c'est la manière dont je me représentais la recherche, mais ce n'est pas toujours la façon dont on peut la pratiquer, en France tout du moins. Merci également pour ta gentillesse, la façon dont ta famille et toi m'avez accueilli et fait découvrir Montréal, sa région et son mode de vie.

Avec Benoît, tout est possible… et c'est en effet toi qui a rendu tout ceci réalisable. Pour avoir recruté un enseignant chercheur sans doctorat et être ainsi sorti des sentiers battus, pour avoir énormément favorisé la logistique franco-canadienne, pour m'avoir mis en relation avec de nombreuses personnes de grande valeur des deux côtés de l'Atlantique et pour m'avoir prodigué de précieux conseils, je tiens à vivement te remercier.

Je tiens ensuite à remercier l'ensemble de mes amis et proches pour m'avoir appuyé lorsque j'ai décidé de changer d'orientation professionnelle, pour s'être régulièrement intéressés à l'avancement de mes travaux, même si la thématique n'est pas des plus évidentes à appréhender…

Je désire également saluer nos camarades de la société DeltaCAD, notamment Harvey Rowson, pour les discussions passionnantes que nous avons eues au sujet de ces travaux et pour avoir cru en leur pertinence au point de monter le projet de laboratoire commun DIMEXP.

De façon très générale, je tiens à remercier les différents étudiants UTC qui ont travaillé sous mon encadrement à élaborer un démonstrateur qui leur semblait tant éloigné de leurs enseignements « traditionnels ».

Enfin, je souhaite exprimer mes remerciements aux équipes des cousines que sont l'École de technologie supérieure de Montréal et de l'Université de Technologie de Compiègne, pour m'avoir permis de réaliser cette cotutelle dans de si bonnes conditions.

J'ai également une pensée pour ceux qui n'auront pas eu l'occasion de partager la joie liée à l'aboutissement de ces travaux.

MÉTHODE AGILE POUR LA CONCEPTION COLLABORATIVE MULTIDISCIPLINAIRE DE SYSTÈMES INTÉGRÉS : APPLICATION À LA MÉCATRONIQUE

Matthieu BRICOGNE

RÉSUMÉ

Ces travaux portent sur la conception multidisciplinaire de systèmes intégrés. Ces systèmes sont soumis à un nombre d'exigences toujours croissant, entraînant des besoins en termes d'intégration fonctionnelle et spatiale. Ces différents types d'intégration relative au produit sont également la source d'une complexité organisationnelle, provenant à la fois de la multitude d'acteurs réalisant différentes activités d'ingénierie, mais également de la diversité des domaines impliqués, désignée dans ce manuscrit par « intégration multidisciplinaire ». Pour favoriser cette intégration multidisciplinaire, les phases de « conception préliminaire » et de « conception détaillée » ont été identifiées comme déterminantes, notamment car elles se caractérisent par la collaboration de nombreux experts, manipulant un grand nombre de données techniques de définition.

Les systèmes conçus lors de conceptions multidisciplinaires restent faiblement intégrés. Cela est en partie dû au cloisonnement entre les disciplines et à un mode d'organisation projet basé sur une planification prédominante, caractérisé notamment par une diffusion de l'information principalement descendante (*top-down*). Afin d'assurer une meilleure collaboration entre ces différentes disciplines, de permettre des prises de décision éclairées par des indicateurs opérationnels et de pouvoir analyser et mieux comprendre les phénomènes d'intégration des expertises, l'introduction d'une méthode inspirée des principes fondateurs des méthodes agiles est proposée pour la conception collaborative de systèmes intégrés.

La contribution de ces travaux s'appuie sur trois concepts complémentaires. Le premier, intitulé *Collaborative Actions Framework* correspond à un cadre de collaboration opérationnelle autour d'actions. Un des objectifs de ce *framework* est de faciliter la collaboration des acteurs des projets de conception, quelle que soit leur origine disciplinaire, mais également d'assurer une traçabilité entre les prises de décision et les corrections/modifications apportées sur les données techniques. Cette traçabilité est

rendue possible grâce aux liens existants avec le second concept intitulé *Workspace*. Apportant un nouvel éclairage sur les possibilités offertes par la collaboration autour de ces espaces de collaboration, ce concept offre un certain nombre de possibilités, notamment la mise en commun continue des travaux, l'intégration multidisciplinaire et la validation des modifications. Les échanges de données techniques entre les *workspaces*, ou le travail simultané sur les mêmes données techniques, s'appuient quant à eux sur la possibilité de pouvoir gérer de façon parallèle différentes versions d'une même donnée technique. Ces possibilités sont proposées par le troisième concept, intitulé *branch & merge*, qui permet également à différents acteurs de travailler simultanément sur les mêmes données.

Enfin, ces trois concepts sont ensuite illustrés par l'intermédiaire d'un démonstrateur composé d'un scénario et d'un prototype informatique. Un produit mécatronique, combinaison synergique et systémique de la mécanique, de l'électronique et de l'informatique temps réel, est utilisé afin d'illustrer les possibilités offertes par nos travaux en termes d'intégration multidisciplinaire lors de la conception collaborative

Mots clés : méthode agile ; conception collaborative ; intégration multidisciplinaire ; mécatronique ; systèmes intégrés ; conception intégrée ; *Collaborative Actions Framework* ; *WorkSpaces* ; *Branch & Merge* ; gestion de versions concurrentes ; *Product Lifecycle Management* ; *Application Lifecycle Management* ; *Cyber-Physical Systems* ; Ingénierie Système

AGILE METHOD FOR THE MULTIDISCIPLINARY AND COLLABORATIVE DESIGN OF INTEGRATED SYSTEMS: APPLICATION TO MECHATRONICS

Matthieu BRICOGNE

ABSTRACT

This work focuses on the multidisciplinary and collaborative design of integrated systems. These systems are subject to an ever increasing number of requirements, leading to the need for more comprehensive functional and spatial integration. These different types of product integration are also at the origin of organizational complexity. This complexity arises not only from the great number of actors performing various engineering activities but also from the diversity of disciplines involved (designated in this manuscript as "multidisciplinary integration"). To encourage this multidisciplinary integration, "preliminary design" and "detailed design" have been identified as the most significant steps, especially since they are characterized by the collaboration of multiple experts handling a large number of product definition' technical data.

Systems that have been designed thanks to multidisciplinary approaches are generally poorly integrated. This is partially due to the compartmentalization of disciplines, as well as to the "project-planned" method, where project planning is predominant and information is mainly spread out "top-down". To ensure better cooperation between the various disciplines, to enable decision making based on operational indicators and to analyze and understand the multidisciplinary integration processes, a method inspired by the founding principles of agile methods (the agile manifesto) is proposed for the collaborative design of integrated systems.

This work is based on three complementary concepts. The first is, the *Collaborative Actions Framework,* an operational framework for collaboration around actions. One objective of this framework is to improve the collaboration among designers, whatever their disciplinary origin. It also ensures traceability between decision making and corrections/changes made to technical data. This traceability is made possible by the use of the second concept, called *Workspace*. Even if this term is already well known, we propose a new definition/usage to transform it into collaboration spaces. This concept

offers great possibilities, including the continuous delivering/sharing of experts' contributions, multidisciplinary integration and change validation. The exchange of technical data between workspaces, or simultaneous work on the same data, relies on the ability to manage several parallel versions of the same item into a single data management system. These opportunities are offered by the third concept, called *Branch & Merge*.

Finally, these three concepts are illustrated through a scenario and a computer prototype. A mechatronic product, "the synergistic combination of mechanical and electrical engineering, computer science, and information technology" (Harashima et al., 1996), is used to illustrate the opportunities offered by our work in terms of multidisciplinary integration during collaborative design.

Keywords : agile method; collaborative design; multidisciplinary integration; mechatronics systems; integrated systems; integrated design; Collaborative Actions Framework; WorkSpaces; Branch & Merge; Concurrent versioning; Product Lifecycle Management; Application Lifecycle Management; Cyber-Physical Systems; Systems Engineering

TABLE DES MATIÈRES

Page

LISTE DES TABLEAUX

Page

LISTE DES FIGURES

Page

LISTE DES ABRÉVIATIONS, SIGLES, ACRONYMES ET ORGANISMES

Liste des abréviations

- cf. : Confer (voir)
- e. g. : exempli gratia (par exemple)
- etc. : et cætera
- *i.e.* : id est (c'est-à-dire)
- *vs.* : versus (contre)

Liste des sigles

- ALM : Application Lifecycle Management
- BOM : Bill of Material
- BI : Business Intelligence
- BIM : Building Information Modeling
- BTP : Bâtiment et Travaux Publics
- CI : Configuration Item
- CM : Configuration Management
- CP : Cyber–Physical
- CPS : Cyber–Physical System
- CRM : Customer RelationShip Management
- CVS : Concurrent Versions System
- ECO : Engineering Change Order
- ECM : Engineering Change Management
- ERP : Enterprise Resource Planning
- GUI : Graphical User Interface
- HW : Hardware
- KBE : Knowledge Based Engineering
- KPI : Key Performance Indicator
- IS : Ingénierie Systèmes
- MBSE : Model Based Systems Engineering
- MVC : Model-View-Controller
- OBS : Organisation Breakdown Structure

- OLAP : OnLine Analytic Processing
- PBS : Product Breakdown Structure
- PDM : Product Data Management
- PLM : Product Lifecycle Management
- PSS : Product Service System
- SCM : Software Configuration Management
- SGDT : Systèmes de Gestion de Données Techniques
- SW : Software
- TRL : Technology Readiness Level
- WBS : Work Breakdown Structure
- WS : Workspace
- XML : Extensible Markup Language

Liste des acronymes

- SysML : Systems Modeling Language

Liste des organismes

- AFIS : Association Française d'Ingénierie Système
- AFNOR : Association française de normalisation
- INCOSE : INternational COuncil on Systems Engineering
- NASA : National Aeronautics and Space Administration
- VDI : Verein Deutscher Ingenieure

STRUCTURE DE LA THÈSE

Le travail de thèse présenté dans ce manuscrit s'inscrit dans les domaines de l'ingénierie collaborative et du *Product Lifecycle Management*. Il s'intéresse à la conception de systèmes intégrés, sous l'angle de l'intégration multidisciplinaire. Il vise à proposer un cadre de collaboration agile, *e.g.* non déterminé a priori, supporté par un ensemble de concepts et outils.

Le chapitre 1 est scindé en deux parties. La première vise à décrire le contexte de nos travaux et à préciser certaines définitions. L'intégration relative au produit, qui se décompose en deux types d'intégration nommés intégration spatiale et intégration fonctionnelle, est mise en relation avec les intégrations relatives aux activités d'ingénierie et aux disciplines (appelée intégration multidisciplinaire dans ce manuscrit). Afin d'illustrer ces définitions, la mécatronique est présentée comme un cadre applicatif pertinent. Il permet également de montrer les convergences qui s'opèrent actuellement entre différents cadres applicatifs représentés par différentes industries et communautés scientifiques. La seconde partie présente notre problématique portant sur le cloisonnement des différentes disciplines impliquées lors de la conception de systèmes et son impact sur l'intégration relative au produit. Les trois objectifs de nos travaux sont ensuite précisés avant de décrire une méthodologie originale qui rappelle les origines de ces travaux.

Le chapitre 2 présente ensuite un état de l'art structuré autour d'un tableau à double entrée. Ces deux « entrées » sont basées sur deux facteurs de positionnement. Le premier a pour but de présenter le type de contribution alors que le second s'attache à définir la portée de cette contribution. Ce tableau nous permet ensuite de préciser notre positionnement et l'originalité de nos travaux.

Le chapitre 3 structure notre proposition autour de trois concepts imbriqués telles des poupées russes. Inspiré par les principes fondateurs des méthodes agiles, ces concepts viennent proposer un cadre de collaboration opérationnelle pour faciliter les échanges d'informations entre les acteurs des projets de conception, quelle que soit leur origine

disciplinaire. Il permet également d'assurer la traçabilité entre les prises de décision réalisées et les corrections/modifications apportées sur les données techniques dans le cadre du projet. Enfin, il propose des espaces de collaboration, permettant le partage et l'intégration au plus tôt des données techniques issues des différentes disciplines. Grâce à ce cadre de collaboration, les acteurs opérationnels des différentes disciplines sont en mesure de prendre des initiatives pour résoudre collectivement des problèmes de conception.

Le chapitre 4 propose un démonstrateur validant la pertinence de notre proposition en regard de notre question de recherche. Ce démonstrateur s'appuie sur un scénario illustrant la conception multidisciplinaire d'un produit mécatronique et sur une mise en œuvre de ce scénario dans un prototype informatique.

Enfin, le chapitre 5 conclut ce manuscrit par une synthèse des apports de ces travaux. Les limites y sont également présentées avant de proposer les perspectives identifiées.

CHAPITRE 1

La conception collaborative multidisciplinaire de systèmes intégrés

1.1 Contexte

Ce chapitre décrit le contexte dans lequel les travaux présentés dans ce manuscrit ont été menés. Ces travaux portent sur la conception multidisciplinaire de systèmes intégrés. Cette thématique, très vaste, est précisée grâce à différentes définitions proposées dans ce chapitre. Ainsi, la conception de systèmes intégrés est définie grâce à différents types d'intégration pour le produit avant de caractériser la conception intégrée, les enjeux liés au travail collaboratif et la notion de multidisciplinarité. Enfin, une illustration de ces enjeux sur le cas des produits dits mécatroniques est proposée, basée sur les singularités de ces derniers.

1.1.1 La conception de systèmes intégrés

1.1.1.1 Un nombre d'exigences toujours croissant

Les produits industriels développés de nos jours sont de plus en complexes et ils doivent répondre à un nombre croissant d'exigences. Une exigence est définie par l'Association Française d'Ingénierie Système (AFIS[1]) et reprise par Badreau et Boulanger comme « quelque chose qui prescrit ce qu'un produit doit faire, avec quelles performances et sous quelles conditions, pour atteindre un but donné » (Badreau and Boulanger, 2014). Dans ce manuscrit, nous regroupons les exigences liées à un produit en trois catégories que nous intitulons « exigences intrinsèques », « exigences extrinsèques » et « exigences liées à la chaîne de valeur ». Ces différentes catégories sont proposées afin d'enrichir les définitions proposées par l'AFIS qui ne considèrent généralement que les exigences liées

[1] www.afis.fr

aux fonctions attendues du système, l'environnement extérieur au système étant considéré comme un ensemble de contraintes auquel il est nécessaire de prêter attention.

La première catégorie d'exigences fait référence aux qualités intrinsèques du produit. Les exigences rassemblées dans cette catégorie sont qualifiées d' « exigences fonctionnelles » par l'AFIS. Ainsi, l'ajout de nouvelles fonctionnalités à un produit, la réduction de son encombrement ou de son poids, l'augmentation de sa fiabilité se traduisent par un resserrement des niveaux d'exigences à prendre en considération lors de la conception du produit et des éventuels services associés.

La seconde catégorie d'exigences est liée à l'intégration du produit dans son environnement. En effet, le fait que les produits soient de plus en plus communiquant, qu'ils doivent s'inscrire dans une logique de développement durable, etc. induit également un grand nombre d'exigences à respecter. Une partie des exigences liées à cette catégorie correspond au terme « contrainte d'environnement » proposé par l'AFIS.

Enfin, une dernière catégorie d'exigences est liée à la chaîne de la valeur associée au produit, ou à une famille de produits. En effet, les différentes activités qui participent à la création de la valeur totale doivent être maîtrisées et les exigences ne doivent donc pas uniquement être associées aux phases d'élaboration du produit. Les coûts de maintenance en conditions opérationnelles, d'infrastructure pour assurer les services liés au produit, etc. sont autant d'exemples permettant de mieux comprendre les activités devant être prises en considération. Une partie des exigences liées à cette catégorie correspond au terme « exigence non fonctionnelle » proposé par l'AFIS.

Bien que les travaux décrits dans ce manuscrit se focalisent sur la conception de systèmes, la prise en considération de l'ensemble des phases du cycle de vie du produit est primordiale et l'ensemble des exigences de ces trois catégories peuvent se référer à l'une ou à plusieurs phases du cycle de vie. La Figure 1.1 rappelle les différentes phases du cycle de vie du produit en regroupant ces dernières en trois grandes étapes que sont le « beginning of life », le « middle of life » et le « end of life » (Terzi et al., 2010).

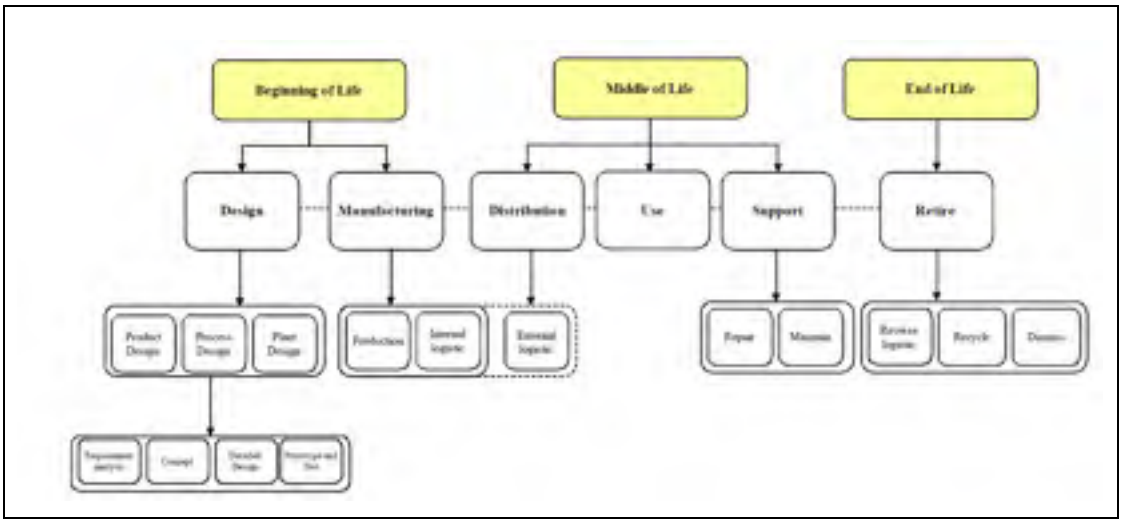


Figure 1.1 Les différentes phases du cycle de vie produit (Terzi et al., 2010)

Dans cette partie, trois catégories d'exigences relatives aux différentes phases du cycle de vie du produit ont été proposées (« exigences intrinsèques », « exigences extrinsèques » et « exigences liées à la chaîne de la valeur »). Dans la partie suivante, nous nous appuierons sur ces catégories afin de reprendre les définitions et typologies relatives à l'intégration en conception de système. Elle se décline en deux parties, l'une relative à l'intégration produit, dont la dimension est principalement technique, et l'autre à la conception intégrée, plus organisationnelle.

1.1.1.2 Les différents types d'intégration dans le cadre de la conception de systèmes

L'intégration relative au produit

Parmi les trois catégories d'exigences présentées ci-avant, les exigences dites intrinsèques ont une importance prédominante sur la conception du produit. Elles ont en effet un impact fort sur les intégrations dites fonctionnelle et spatiale. « Functional integration consists in incorporating as many functions as possible in a single product, either by combining many functions into one component or by dematerialization of certain functions by using information technology » (Warniez et al., 2012).

L'agrégation de fonctions est illustrée par la Figure 1.2. Elle présente l'évolution d'un téléphone portable auquel de nombreuses fonctionnalités ont été ajoutées, telles que la gestion de l'agenda, des contacts et des courriels, la prise de photo, la navigation par GPS, etc.

Figure 1.2 Intégration fonctionnelle : agrégation de fonctions au sein d'un même système[2]

[2] Images recueillies individuellement sur internet

La dématérialisation correspond quant à elle au remplacement de fonctionnalités jusqu'alors assurées par des composants physiques par de nouvelles fonctions logicielles. Nous illustrons ici cette dématérialisation par le choix de différentes solutions pour remplir la fonction de saisie d'adresse sur un GPS automobile (Figure 1.3) : l'interface purement mécanique (a) est remplacée par un clavier tactile (b) puis par une dictée vocale (c).

(a) (b) (c)

Figure 1.3 Intégration fonctionnelle : dématérialisation d'une fonction de saisie d'adresse sur un GPS automobile[3]

Un second type d'intégration relative au produit conçu est l'intégration spatiale. Également appelée intégration physique (Warniez et al., 2012), cette intégration vise généralement à réduire l'encombrement et le poids du système. La Figure 1.4 illustre grâce à un alterno-démarreur les différents niveaux d'intégration possibles, allant jusqu'à la fusion des différents éléments matériels (*hardware*) du système.

[3] Images recueillies individuellement sur internet

Figure 1.4 Les différents niveaux d'intégration pour les produits mécatroniques (Penas et al., 2010)

Ces différents types d'intégration relative au produit sont également la source d'un surcroît de complexité technique par rapport à un produit moins intégré. Celle-ci ne se résume généralement pas à la somme des complexités techniques de chacun des métiers car des phénomènes d'interdépendance des informations apparaissent, augmentant considérablement la complexité de conception. Ces interdépendances, parfois également appelées couplages, existent tant au niveau des données et expertises (interdépendances multidomaines) qu'au niveau des diverses physiques mises en jeu (couplages multiphysiques) (Penas et al., 2010).

La section suivante s'attache à présenter l'intégration des expertises relatives à chacun des domaines impliqués lors de la conception du système. Ces deux types d'intégration (produits et expertises) sont liés par le fait que l'intégration relative au produit tient avant tout à la synergie qui sera mise en œuvre entre les différents experts lors de la conception du produit. Un manque de communication entre les différents experts, notamment issus des différents domaines, devient un obstacle aux intégrations fonctionnelles et spatiales. « Il faut donc une intégration des métiers (et donc des collaborations) (…) pour créer des liens étroits entre les différents domaines » (Penas et al., 2010), et ainsi créer et maîtriser le couplage des informations liées à chacune des expertises tout au long de la conception.

L'intégration relative aux activités d'ingénierie et aux disciplines

Les différents types d'intégration relative au produit présentés dans la section précédente sont également la source d'une complexité organisationnelle. Ils nécessitent un fort niveau de collaboration induisant une complexité organisationnelle provenant à la fois de la multitude d'acteurs, mais également de la diversité des domaines impliqués. Dans ce manuscrit, nous parlons d'intégration multidisciplinaire pour décrire ce type d'intégration.

La notion d'« integrated design », définie comme « the coordinated development effort in timing and substance of the various disciplines and organizational functions that span the life-cycle of new products and services » (Ettlie, 1997), correspond à cette coordination des contributions de chacun des domaines et de chacune des activités d'ingénierie. À l'origine, la notion d'intégration en conception faisait notamment référence à « l'aspect participatif des différents corps de métiers au moment de la conception » et au fait de « prendre en compte au plus tôt des besoins des hommes de métiers qui vont avoir à confectionner le produit, le maintenir ou le détruire » (Tichkiewitch, 1994). Les deux autres notions liées à l'intégration en conception également proposées par Tichkiewitch sont la prise en considération du cycle de vie du produit et des contraintes liées à l'environnement « social » et « physique » du produit.

La conception intégrée a donc, par l'aspect coordination des contributions, de nombreux points communs avec la notion d'ingénierie simultanée, qui est définie comme le « way of work where the various engineering activities in the product and production development process are integrated and performed as much as possible in parallel rather than in sequence» (Sohlenius, 1992).

Dans cette section, deux types d'intégration ont été définis. Le premier type concerne l'intégration relative au produit. Qu'il s'agisse d'intégration spatiale ou d'intégration fonctionnelle, le but est avant tout de répondre aux exigences intrinsèques, tout en prenant en considération les autres types d'exigences décrits précédemment. Le second type d'intégration fait plus écho à la complexité organisationnelle induite par la conception de systèmes intégrés. Il concerne l'intégration des activités d'ingénierie liées aux différentes

phases d'élaboration du produit et l'intégration des expertises en provenance des différents domaines impliqués lors de la conception.

Cette section a notamment permis de mettre en avant la nécessité d'intégration des activités d'ingénierie et des expertises en provenance des différents domaines. Ces intégrations doivent être coordonnées tout au long de l'élaboration du produit. Dans la section suivante, différents modèles de processus de développement produit sont présentés. Ils permettent de définir quelles sont les phases clé permettant de faciliter l'intégration multidisciplinaire.

1.1.1.3 Les processus de développement de produit

Les modèles de processus de développement produit sont en général issus des communautés d'origine mécanique (Hyman, 2003; Pahl et al., 2003; Ullman, 2010). Malgré cette connotation historique, les titres des ouvrages dans lesquels ces processus sont décrits illustrent une volonté de généricité, quel que soit le type de produit développé, ou le type de technologie mis en œuvre: on parle généralement d'« engineering design » et non de « mechanical design ». Dans ce manuscrit, deux modèles de processus de développement produit ont été sélectionnés pour leur notoriété et leur complémentarité, mais un tableau de synthèse plus exhaustif est présenté à la fin de cette section.

Les deux modèles sélectionnés sont présentés sur la Figure 1.5 (Pahl et al., 2003) et sur la Figure 1.6 (Ullman, 2010). Sur ces deux modèles, les phases identifiées comme étant les plus propices à l'intégration multidisciplinaire sont les phases qui suivent le « conceptual design » (Pahl et al., 2003; Ullman, 2010). On retrouve ainsi la phase nommée « embodiment design » (Pahl et al., 2003) et la phase de « detailed design » (Pahl et al., 2003) réunies sous la même appellation « product development » sur la Figure 1.6 (Ullman, 2010). Ces deux phases voient en effet un grand nombre d'acteurs en provenance de différentes disciplines collaborer afin de faciliter les intégrations relatives au produit et tenter de trouver des solutions aux différents problèmes de couplage évoqués précédemment. Comme le montre le tableau de synthèse Tableau 1.1 (Howard et al., 2008), ces termes apparaissent de façon récurrente dans les très nombreux autres

modèles de processus que l'on trouve dans la littérature. Dans ce manuscrit, les termes « conception préliminaire », « conception détaillée » et « développement produit » font donc respectivement référence aux phases d'« embodiment design », de « detailed design » et de « product development ».

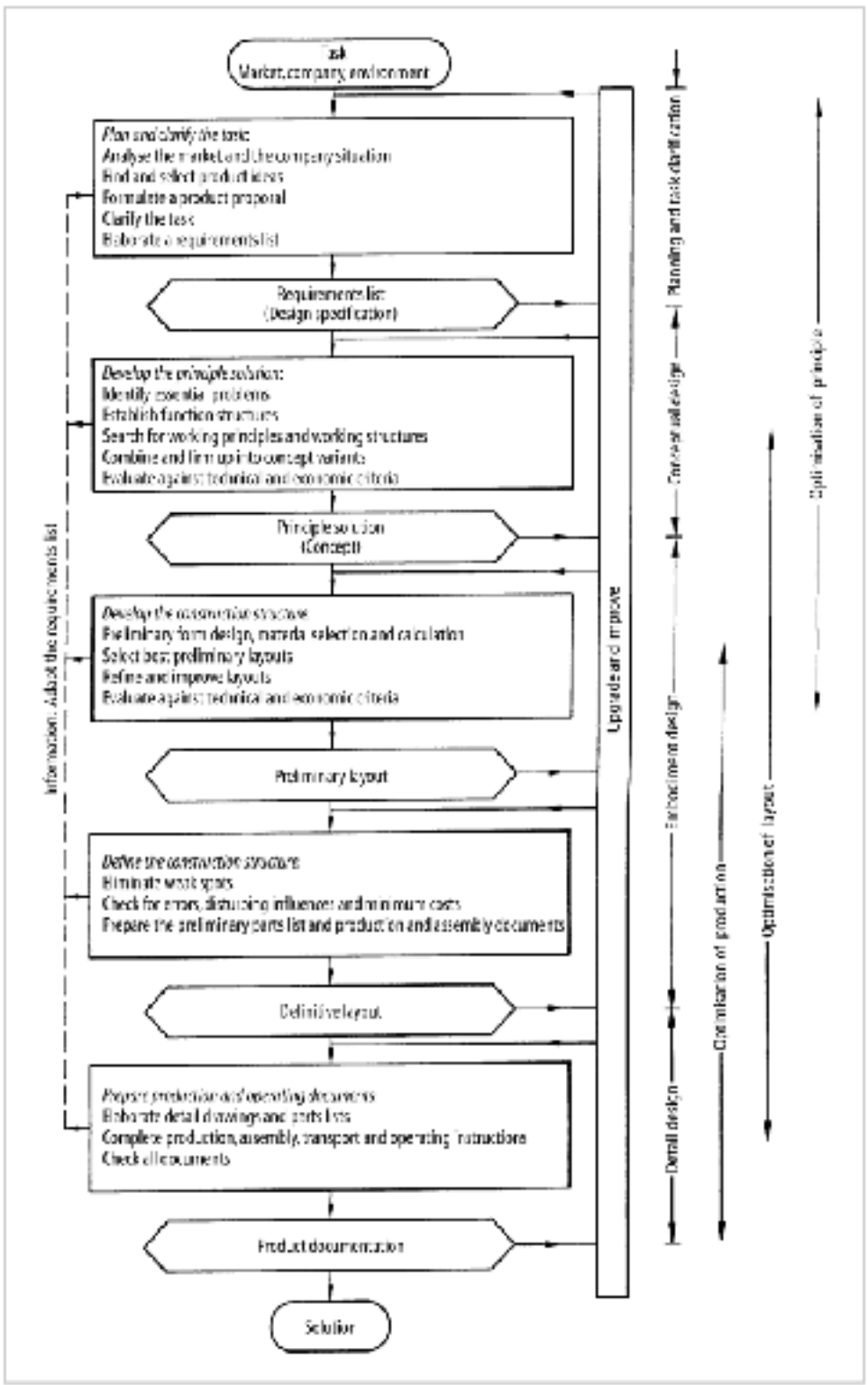


Figure 1.5 Steps in the planning and design process (Pahl et al., 2003)

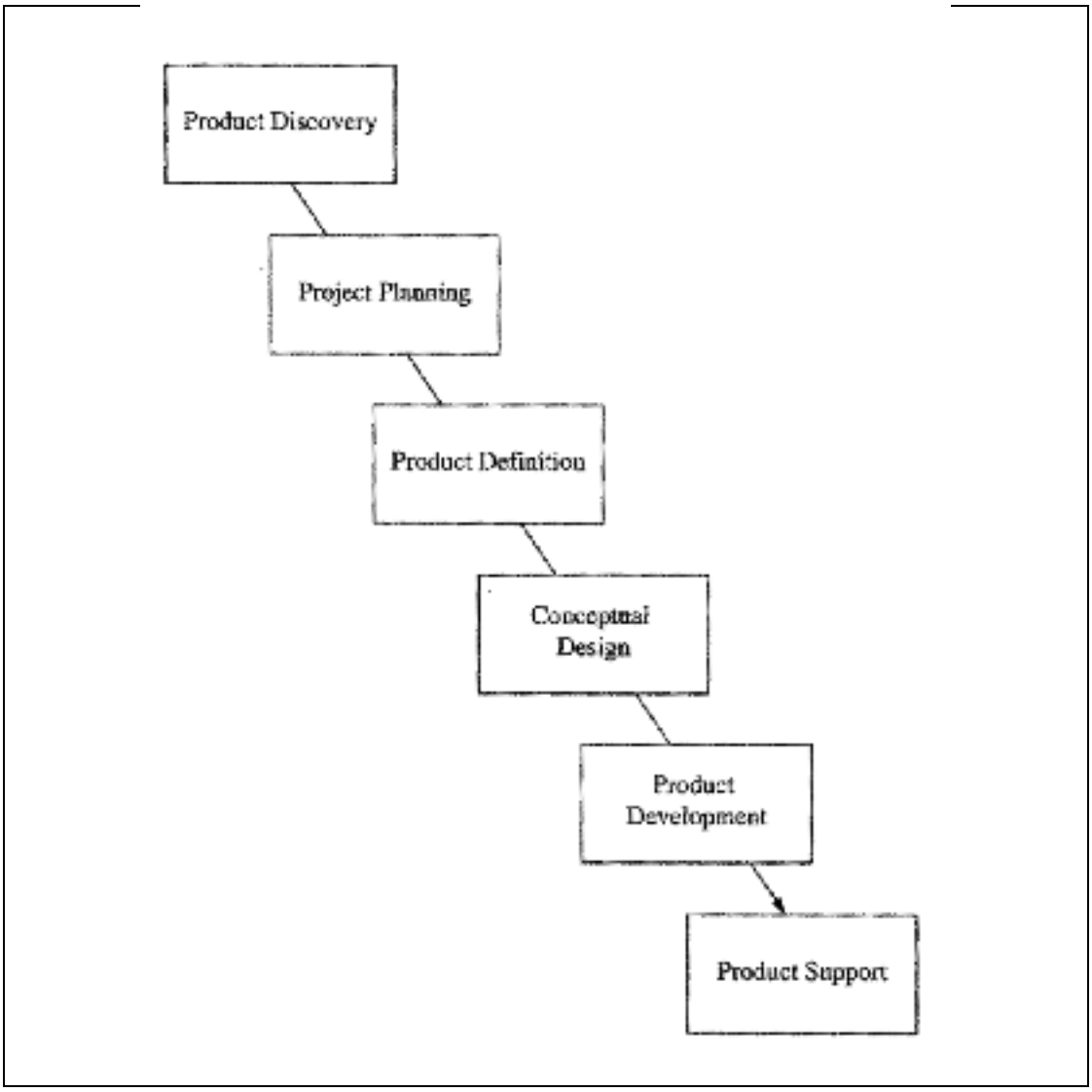


Figure 1.6 The mechanical design process (Ullman, 2010)

Models	Establishing a need phase	Analysis of task phase	Conceptual design phase	Embodiment design phase	Detailed design phase	Implementation phase
Booz et al. (1967)	X	New product strategy development	Idea generation / Screening & evaluation	Business analysis / Development / Testing		Commercialisation
Archer (1968)	X	Programming / data collection	Analysis / Synthesis	Development	Communication	X
Svensson (1974)	Need	X	Concepts / Verification / Decisions		X	Manufacture
Wilson (1980)	Societal need	Recognize & formalize / FR's & constraints	Ideate and create	Analyze and/or test	Product, prototype, process	X
Urban and Hauser (1980)	Opportunity identification	Design		Testing		Introduction (launch) / Life cycle management
VDI-2222 (1982)	X	Planning	Conceptual design	Embodiment design	Detail design	X
Hubka and Eder (1982)	X	X	Conceptual design	Lay-out design	Detail design	X
Crawford (1984)	X	Strategic planning	Concept generation	Pre-technical evaluation	Technical development	Commercialisation
Pahl and Beitz (1984)	Task	Clarification of task	Conceptual design	Embodiment design	Detailed design	X
French (1985)	Need	Analysis of problem	Conceptual design	Embodiment of schemes	Detailing	X
Ray (1985)	Recognise problem	Exploration of problem / Define problem	Search for alternative proposals	Predict outcome / Test for feasible alternatives	Judge feasible alternatives / Specify solution	Implement
Cooper (1986)	Ideation	Preliminary investigation	Detailed investigation	Development / Testing & Validation	X	Full production & market launch
Andreasen and Hein (1987)	Recognition of need	Investigation of need	Product principle	Product design	Production preparation	Execution
Pugh (1991)	Market	Specification	Concept design		Detail design	Manufacture / Sell
Hales (1993)	Idea, need, proposal, brief	Task clarification	Conceptual design	Embodiment design	Detail design	X
Baxter (1995)	Assess innovation opportunity	Possible products	Possible concepts	Possible embodiments	Possible details	New product
Ulrich and Eppinger (1995)	X	Strategic planning	Concept development	System-level design	Detail design	Testing & refinement / Production ramp-up
Ullman (1997)	Identify needs / Plan for the design process	Develop engineering specifications	Develop concept	Develop product		X
BS7000 (1997)	Concept / Feasibility		Implementation (or realisation)			Termination
Black (1999)	Brief/concept	Review of 'state of the art'	Synthesis / Inspiration	Experimentation / Analysis / reflect	Synthesis / Decisions to constraints / Output	X
Cross (2000)	X	Exploration	Generation	Evaluation	Communication	X
Design Council (2006)	Discover	Define	Develop		Deliver	X
Industrial Innovation Process 2006	Mission statement	Market research	Ideas phase	Concept phase	Feasibility Phase	Pre production

Tableau 1.1 Une comparaison des modèles de processus de conception (Howard et al., 2008)

Les données manipulées durant ces deux phases restent néanmoins de natures assez différentes. Durant la phase de conception préliminaire, les principes de fonctionnement, les choix d'architecture, etc. sont déterminés après l'évaluation des différentes solutions techniques au regard de critères techniques, économiques, environnementaux, sociétaux, etc. Des représentants de chaque discipline collaborent alors sous la responsabilité d'architectes système ayant pour but de faire une synthèse, de répartir les fonctionnalités attendues par domaine et d'élaborer la structure produit. Durant la phase de conception détaillée, un nombre plus important d'acteurs travaillent dans chacune des disciplines à l'élaboration de données de définition.

La Figure 1.7 (Raymer, 1999) illustre l'état d'avancement de la définition d'une pièce en fonction des différentes phases du processus de développement. La dernière image de cette figure représente donc un exemple de pièce aéronautique dans la phase de conception détaillée. Durant cette phase, et après avoir sélectionné une variante de concepts et une architecture, les exigences techniques et spécifications sont transformées en définitions de conception finale. Ces définitions se matérialisent sous forme de données techniques qu'il est nécessaire de gérer.

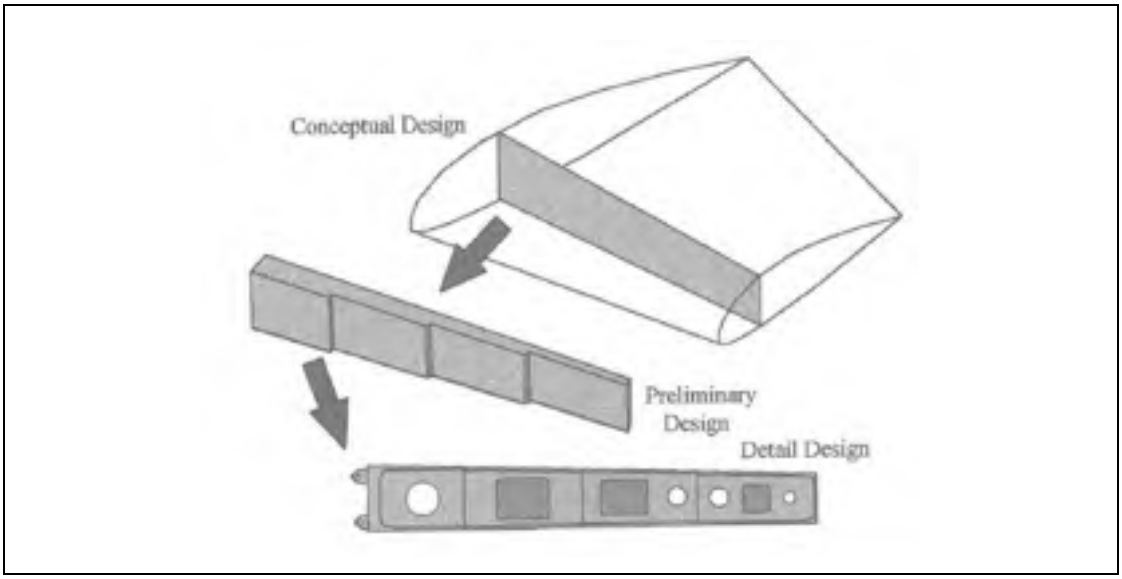


Figure 1.7 Design phases: front wing spar (Raymer, 1999)

Comme présenté ci-avant, nos travaux ciblent tout particulièrement les phases de conception préliminaire et de conception détaillée pour leur impact déterminant sur la finalité recherchée, à savoir l'intégration multidisciplinaire. Au cours de l'avancement de ces différentes phases, il est nécessaire de distinguer les différents états de maturité par lesquels passent les données techniques car ils ont un impact déterminant sur la façon dont ces données doivent être gérées. Dans son ouvrage, Maurino détaille les processus de gestion des changements d'ingénierie et traite de façon distincte les données techniques en cours de conception et les données techniques libérées (« released »). Dans le premier cas, on parle de correction de données et les processus de gestions sont alors très informels alors que dans le second cas, on parle généralement de modification (Maurino, 1994) et les processus sont beaucoup plus structurés (Jarratt et al., 2010). Comme nous le verrons par la suite, bien que les concepts proposés dans le cadre de nos travaux intègrent le formalisme nécessaire au traitement des modifications, leur apport majeur reste surtout l'organisation de la collaboration entre les acteurs des différentes disciplines, et ce même pour la réalisation de corrections.

Dans cette section, une synthèse concernant les différents modèles de processus de conception a été présentée. Sur cette base, deux phases ont été identifiées comme déterminantes pour favoriser l'intégration multidisciplinaire. Elles se caractérisent notamment par la collaboration de nombreux experts, manipulant un grand nombre de données techniques de définition, généralement pas encore libérées.

Afin d'illustrer les enjeux liés à la conception de systèmes intégrés, la section suivante s'intéresse aux systèmes dits mécatroniques, évolution des produits électromécaniques poussée par des exigences toujours plus fortes et toujours plus nombreuses. Ce type de produit a été retenu comme cadre applicatif pour ces travaux car une forte particularité du processus de conception d'un système mécatronique est qu'il nécessite un processus de développement multidisciplinaire et holistique. La seconde particularité de ce cadre applicatif est qu'il permet de s'intéresser aux caractéristiques de l'intégration *hardware-software*, la plupart des travaux traitant de l'intégration multidisciplinaire ne traitant que de l'intégration *hardware*, sans s'intéresser à la partie logicielle des systèmes.

1.1.2 La conception de systèmes intégrés : les systèmes mécatroniques

1.1.2.1 Une évolution des produits électromécaniques vers des systèmes mécatroniques

Les systèmes mécaniques développés depuis les années 80 ont évolué de systèmes électromécaniques présentant des fonctions mécaniques et électriques juxtaposées vers des systèmes intégrant des capteurs, des actionneurs et des systèmes micro-électroniques pilotés par des modules logiciels de plus en plus évolués (Figure 1.8). Ces systèmes intégrés, généralement composés de constituants matériels et logiciels, sont appelés systèmes mécatroniques (Isermann, 2007; Shetty and Kolk, 2010). La Figure 1.9 illustre les différents domaines impliqués lors de la conception de systèmes mécatroniques.

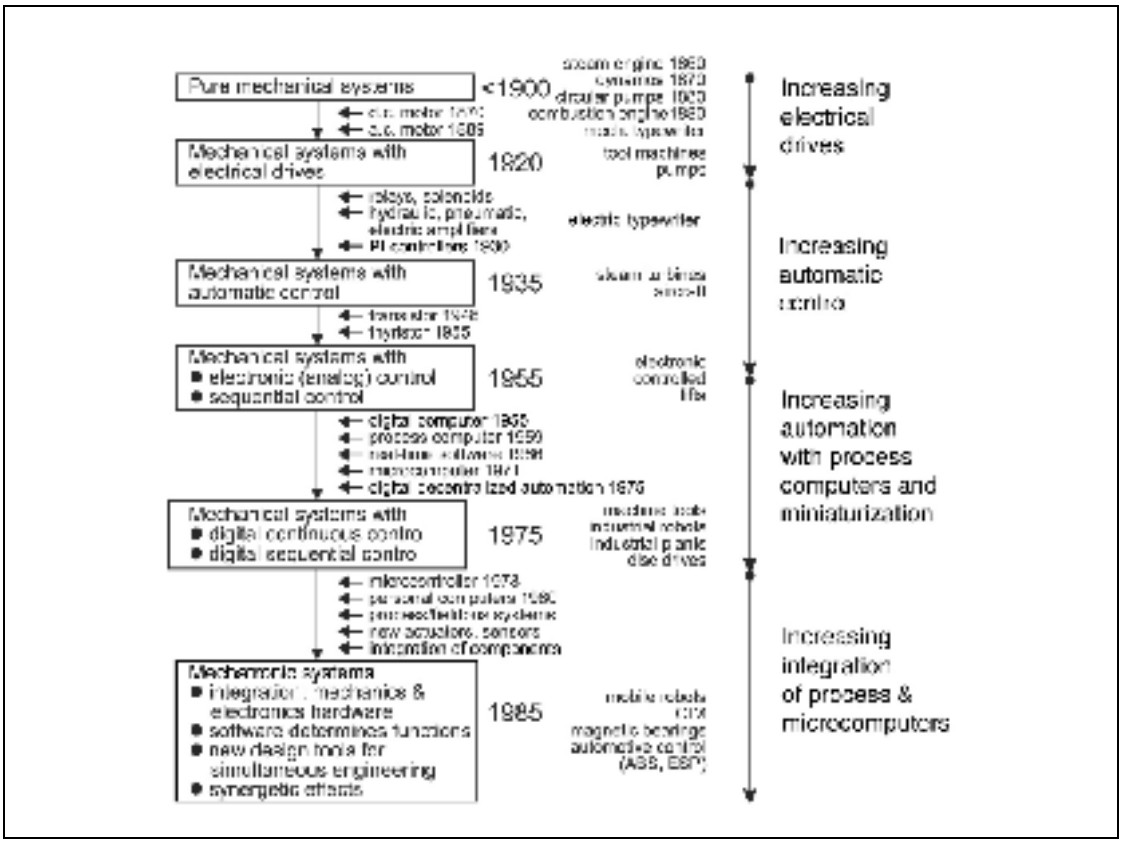


Figure 1.8 Historical Development of Mechanical, Electrical, and Electronic Systems (Isermann, 2007)

Figure 1.9 Interactions entre les différentes disciplines composant la mécatronique (Rensselaer Polytechnic Institute Website (RPI), n.d.)

Lorsqu'un produit traditionnellement électromécanique évolue vers un système mécatronique, différentes étapes sont généralement observées. La Figure 1.10 présente les différentes étapes de cette évolution, impliquant différents domaines et les recoupements qui existent entre ces derniers. Comme l'explique Schöner, un des premiers éléments est généralement l'ajout d'actionneurs (Actuators – A), en bleu. Ils permettent d'augmenter les forces et vitesses d'actionnement. Cette action est menée conjointement par les domaines électronique et mécanique. Afin de fournir la puissance électrique nécessaire à ces actionneurs, une source de puissance externe est généralement fournie par le domaine électrique. La seconde étape correspond couramment à l'introduction de contrôle embarqué (Embedded Control – E), en vert sur la figure. Cette étape permet d'automatiser et de rendre plus systématiques certains fonctionnements et est menée conjointement par les domaines électronique et informatique. La troisième étape correspond à l'introduction de capteurs (Sensors – S). Cette étape, en rouge sur la figure, permet d'obtenir des informations précises sur le fonctionnement du système ou sur son environnement. Cette phase est généralement sous la responsabilité des domaines mécanique et informatique. Enfin, l'intégration de fonctionnalités de communication (Communication – C), représentée en jaune sur la figure, permet l'intégration du sous-système dans un système de plus haut niveau, ou l'intégration du système dans son environnement d'usage.

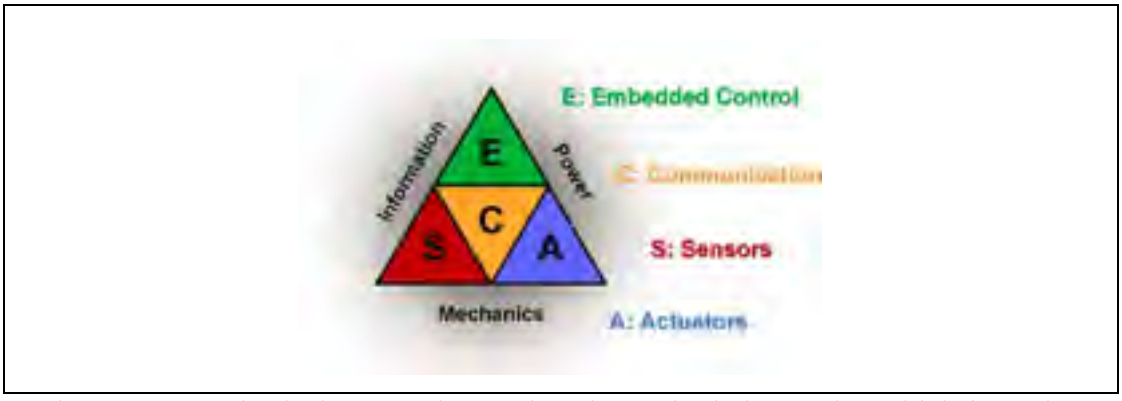


Figure 1.10 Mechanical systems integrating electronics in interaction with information and power (Schöner, 2004)

En synthèse, la mécatronique a été définie dans la norme NF E 01-010 2008 comme « une démarche visant l'intégration en synergie de la mécanique, l'électronique, l'automatique et l'informatique dans la conception et la fabrication d'un produit en vue d'augmenter et/ou d'optimiser sa fonctionnalité. L'objectif de la mécatronique est l'obtention d'une valeur ajoutée supérieure à la simple somme des valeurs ajoutées des fonctions prises séparément » (AFNOR, 2008). Cette définition montre bien tout l'enjeu représenté par ce cadre applicatif. Il ne s'agit en effet pas de créer des systèmes plus ou moins automatisés, mais bien de tirer parti de l'intégration des différents domaines impliqués dans la conception de tels systèmes (Schöner, 2004). Afin de mieux comprendre les enjeux liés à cette intégration en synergie, la section suivante s'attachera à définir les deux catégories de problèmes rencontrés liées à l'intégration multidisciplinaire dans ce cadre applicatif.

1.1.2.2 La conception intégrée appliquée à la mécatronique

Afin de mettre en œuvre une approche basée sur la notion de conception intégrée dans un contexte mécatronique, deux types de problèmes doivent être surmontés (Abramovici and Bellalouna, 2007). Le premier type de problème est relatif aux processus alors que le second a trait aux données techniques de conception.

Les problèmes liés aux processus sont attachés à des problèmes de coordination et de synchronisation des différentes disciplines, notamment des démarches de développement spécifiques, des activités, des tâches et des rendus. Un autre facteur influent tient au fait que la cohérence et les interactions entre les disciplines sont prises en considération trop tard dans le processus de développement (Abramovici and Bellalouna, 2007).

Les problèmes induits par les données techniques de conception proviennent du fait que les outils d'édition et de gestion des données des différentes disciplines sont hétérogènes (Abramovici and Bellalouna, 2007).

Afin de décrire plus précisément les problèmes de coordination et de synchronisation des différentes disciplines, des modèles de processus de développement spécialisés mécatronique sont présentés dans la section suivante.

1.1.2.3 Modèles de processus de développement spécialisés pour la mécatronique

Comme nous l'avons vu, une forte particularité du processus de conception d'un système mécatronique est qu'il nécessite un processus de développement multidisciplinaire et holistique. Malgré ce constat, très peu de modèles de processus de développement spécialisés mécatronique sont disponibles dans la littérature. Depuis plusieurs décennies, différents modèles, comme le modèle en cascade ou Waterfall (Royce, 1970), le modèle en spirale (Boehm, 1988) ou encore le modèle de cycle en V (Forsberg and Mooz, 1992) ont été proposés pour supporter la conception de systèmes. Ce type de modèle peut en effet supporter le processus de conception à un niveau très macroscopique, mais il ne permet pas de supporter la collaboration entre les différents concepteurs provenant des différentes disciplines. Tout particulièrement, l'intégration multidisciplinaire, telle que l'intégration *hardware-software*, n'est pas supportée.

Les quelques modèles proposés ci-après correspondent aux modèles de processus de conception spécialisés pour la mécatronique, qui répondent à des degrés divers à la problématique de l'intégration multidisciplinaire.

Le modèle séquentiel

Les systèmes mécatroniques étant dans une grande majorité des cas une évolution des systèmes électromécaniques, il n'est pas rare de trouver des entreprises qui considèrent que la conception de système mécatronique doit être organisée grâce à une succession séquentielle d'étapes comme l'illustre la Figure 1.11 (Shetty and Kolk, 2010). Mais ce mode d'organisation entraîne généralement de nombreux problèmes d'allocation d'exigences aux différentes disciplines, de communication des évolutions de conception et de validation du modèle numérique du système (Shetty and Kolk, 2010). Les principales conséquences induites par ce mode d'organisation sont donc une mauvaise maîtrise des coûts et des délais de mise sur le marché, et des intégrations relatives au produit limitées.

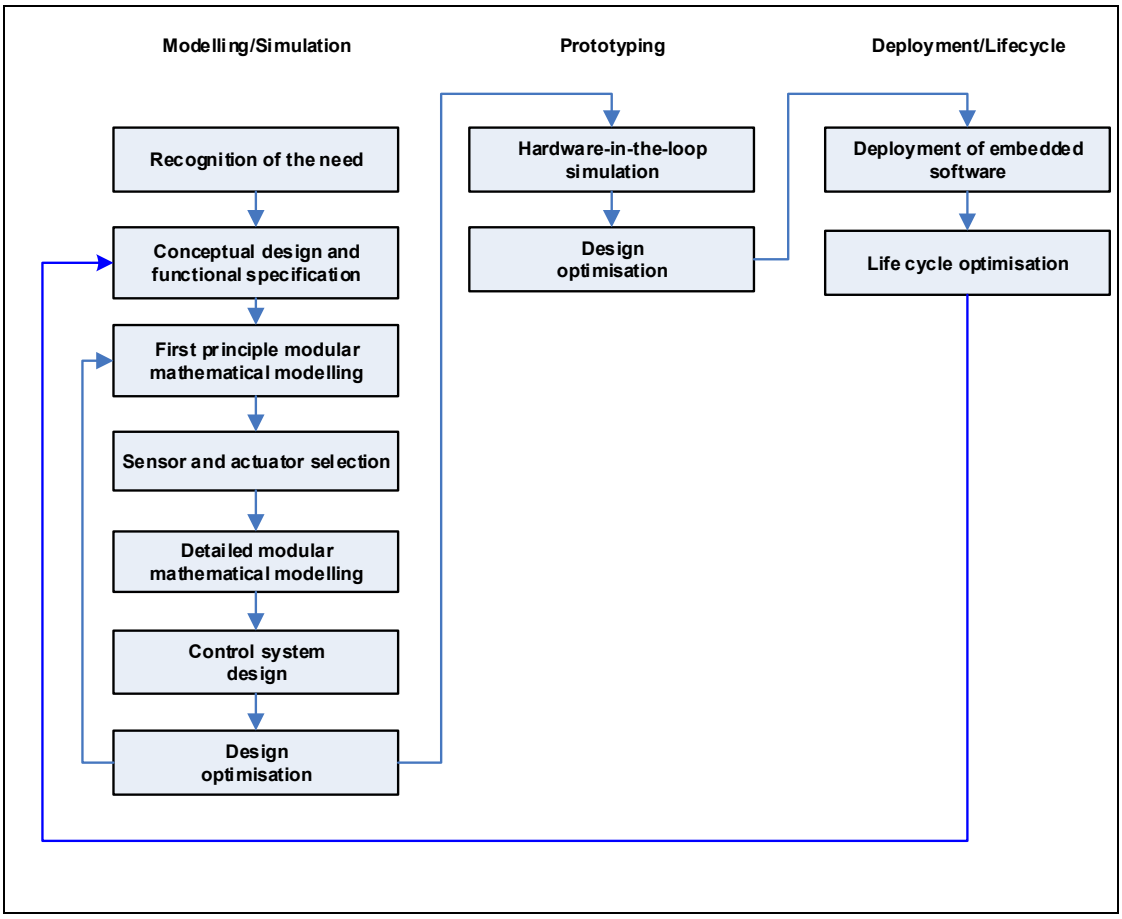


Figure 1.11 Processus de conception mécatronique séquentiel (Shetty and Kolk, 2010)

Le modèle par division disciplinaire

Afin de palier à ces problèmes de mauvaise maîtrise des coûts et des délais, des approches plus rigoureuses basées sur des méthodes de gestion de projet traditionnelles ont été proposées. Aca et al. présentent ainsi un modèle basé sur la répartition des exigences entre les équipes logicielle, électrique/électronique et mécanique. Grâce à cette répartition, les équipes sont en mesure de travailler sur chacun des sous projets et l'intégration multidisciplinaire est traitée comme une étape du processus à part entière, intervenant à l'issue de chacune des conceptions (Aca et al., 2006). La Figure 1.12 illustre cette organisation par division disciplinaire. En raison de cette division, seules quelques personnes impliquées dans le projet ont une vue d'ensemble sur les problèmes de conception rencontrés par les différentes équipes. Les interfaces entre les différents domaines sont définies et ratifiées lors des premières phases. Pour assurer une bonne intégration des différents modules, ces interfaces doivent rester stables. C'est pourquoi, si

un problème de conception se produit dans une discipline spécifique, ce problème est généralement traité directement par l'équipe concernée, même si une solution globale, c'est à dire impliquant plusieurs disciplines, pourrait être plus efficace ou fournir un produit plus intégré. Cette manière d'organiser le processus de conception du système mécatronique est qualifiée de « *project-planned* » (Sommerville, 2010).

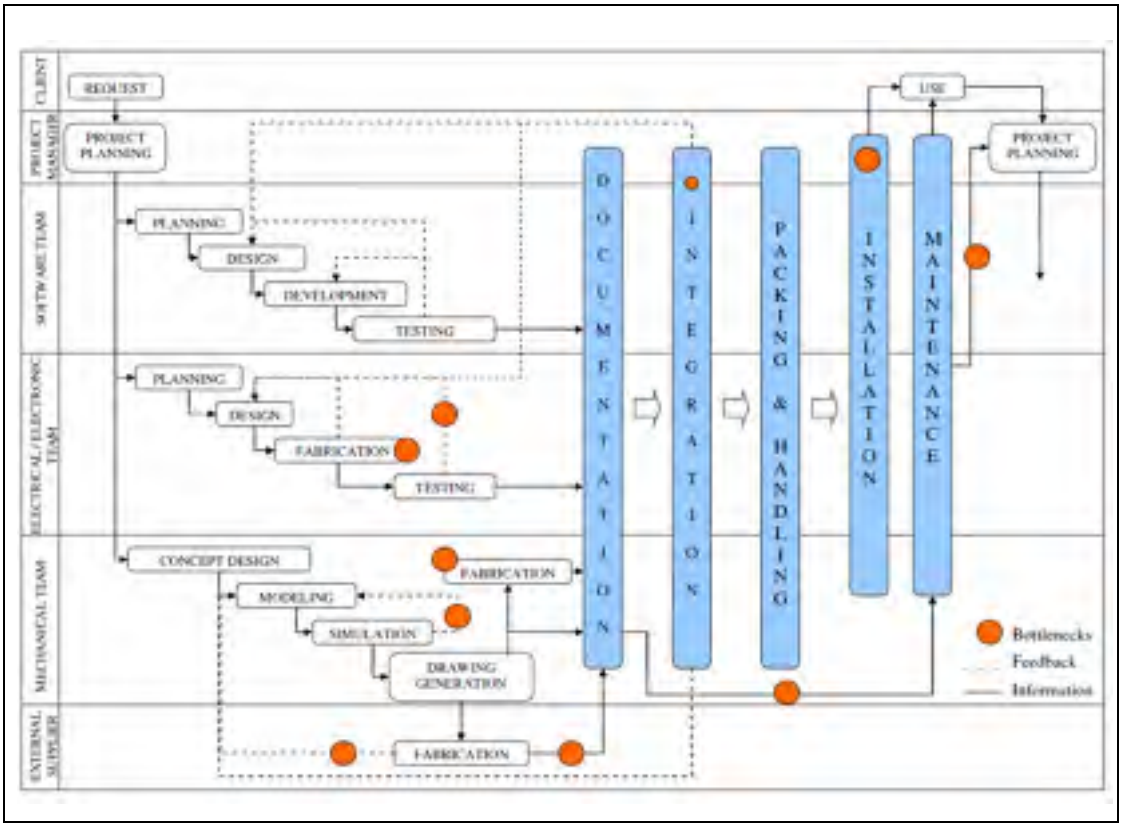


Figure 1.12 Processus de conception par division disciplinaire (Aca et al., 2006)

Le cycle en V et une de ses déclinaisons mécatroniques

Le modèle de cycle en V est un modèle général pour le processus de développement de produits. Il commence par l'identification des besoins des utilisateurs et s'achève par la validation par l'utilisateur. Il se décompose en deux phases principales, la phase dite descendante de décomposition et de définition du produit, et la phase dite ascendante d'intégration et de recomposition, Figure 1.13 (US Department of Transportation, 2007).

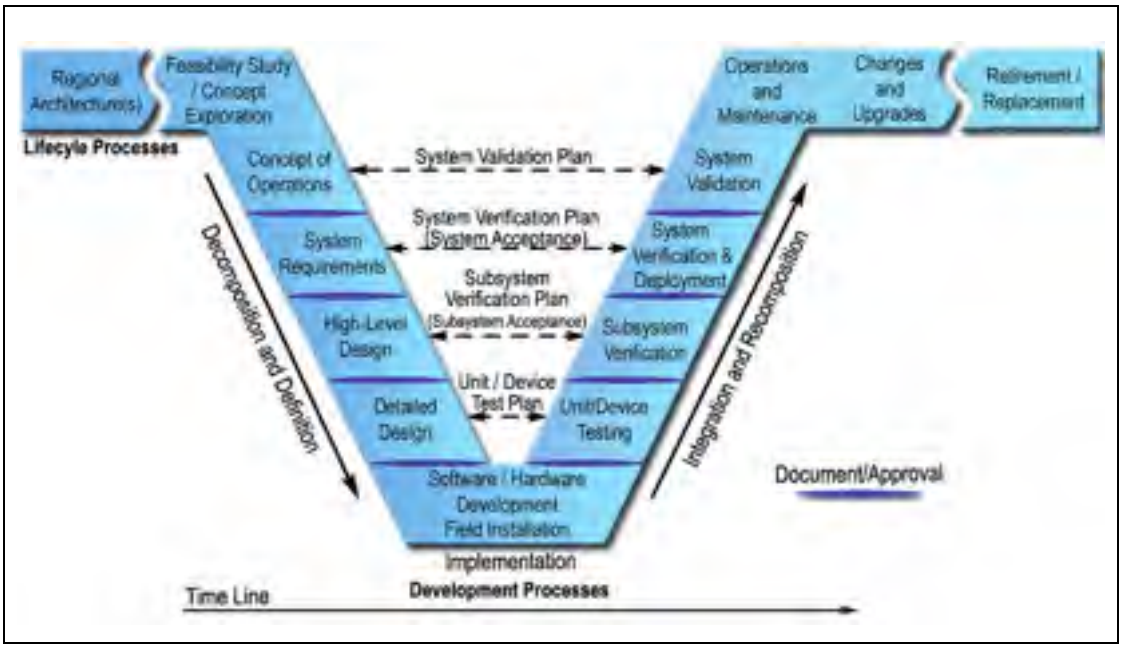


Figure 1.13 Le modèle de cycle en V (US Department of Transportation, 2007)

Une des spécialisations mécatroniques de ce modèle est précisée dans la directive VDI 2206. Cette dernière est élaborée et normalisée par l'association d'ingénieurs allemands « VDI - Verein Deutscher Ingenieure » (Fachbereich Produktentwicklung und Mechatronik, 2004). Il constitue une recommandation axée sur la pratique pour le développement systématique de systèmes mécatroniques.

La directive VDI 2206 fournit un cadre pour la conception de tout type de système mécatronique. Elle divise le processus de conception en quatre phases majeures appelées « system design », « domain-specific design », « system integration » and « assurance of properties », Figure 1.14 (Bathelt et al., 2005).

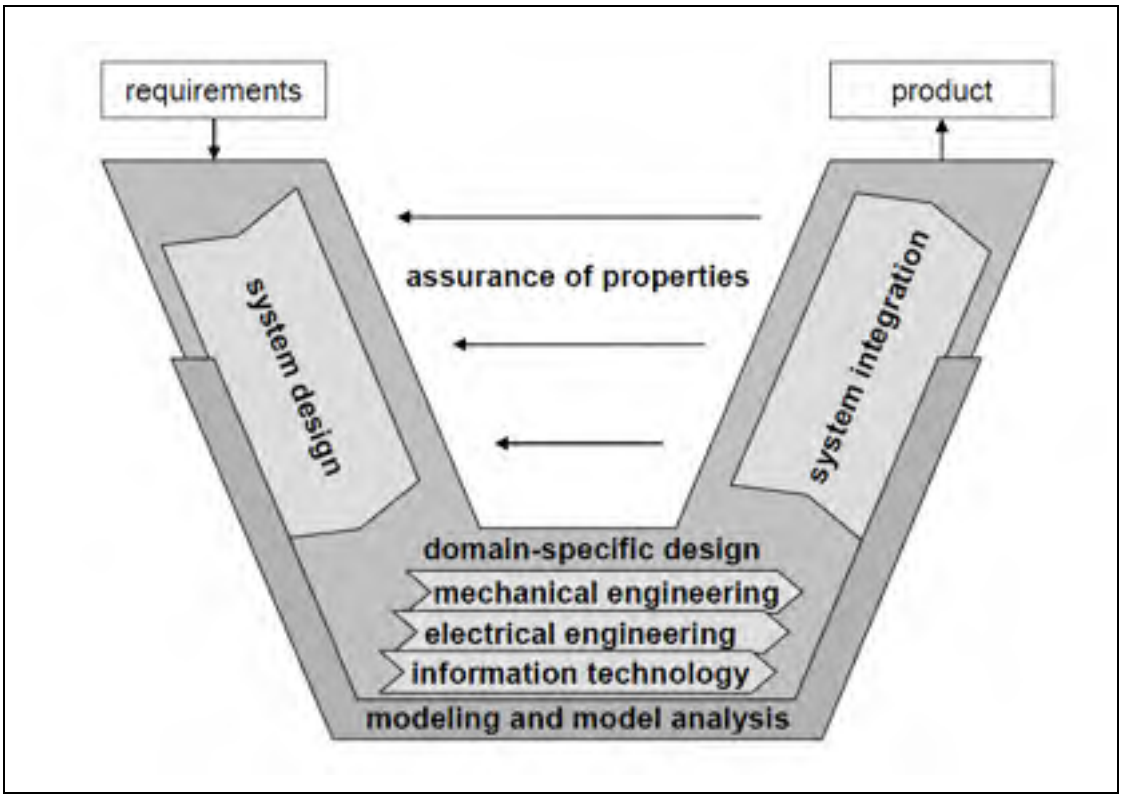


Figure 1.14 Le modèle de cycle en V dans la directive VDI 2206 (Bathelt et al., 2005)

Une fois de plus, l'intégration multidisciplinaire est donc considérée comme une étape intervenant à l'issue des conceptions réalisées dans chacune des disciplines impliquées, ou dans le meilleur des cas lors des différentes mises en commun intermédiaires comme les revues de projet. Cette finalité, même si elle fait partie des considérations omniprésentes pour l'ensemble des acteurs, est donc compliquée par des échanges très peu nombreux et un fonctionnement très cloisonné des disciplines.

Dans cette section, la conception collaborative multidisciplinaire a été illustrée sur un type de système nécessitant un grand nombre d'expertises pour être conçu. Ces expertises sont fortement hétérogènes car elles sont issues de domaines très différents, tels que les domaines logiciel, électronique, électrique et mécanique (intégration *hardware-software* par exemple). Pour faciliter cette intégration multidisciplinaire, deux problèmes ont été identifiés. Ils concernent les processus organisationnels et les données techniques à gérer. Quelques modèles de processus de conception génériques, puis spécialisés mécatronique, ont été présentés, mais aucun ne semble s'intéresser à l'intégration des expertises.

Pour approfondir cette problématique d'intégration des expertises, différents cadres généraux de la conception collaborative multidisciplinaire sont présentés dans la section suivante.

1.1.3 Cadres généraux traitant de la conception collaborative multidisciplinaire de systèmes intégrés

Au sein de cette section, différents cadres généraux traitant de la conception collaborative multidisciplinaire de systèmes intégrés sont présentés. Le premier cadre, intitulé *Product Lifecycle Management* (PLM), a été choisi comme référence pour présenter l'ensemble des autres cadres car les principales thématiques sous-jacentes à ces travaux, comme le travail collaboratif et la conception intégrée, y sont pleinement intégrés. Il servira donc de référence afin de présenter l'Ingénierie Système (IS), cadre général dans lequel s'inscrivent également ces travaux et qui permet de donner un autre angle de vue sur ces derniers. Puis, la cybernétique et les *Cyber-Physical Systems* (CPSs) seront introduits afin de mettre en évidence la nécessité de développer des méthodologies supportant la conception collaborative multidisciplinaire dans ce domaine. Enfin, l'*Application Lifecycle Management* (ALM), considéré comme le pendant du PLM dans le domaine du logiciel, sera également présenté. Cette structuration répond également à une logique de présentation des différents cadres généraux en partant des domaines du *hardware* vers ceux du *software*.

1.1.3.1 Le Product Lifecycle Management

Enjeux et définition de la stratégie PLM

La conception de produits, la fabrication, la réparation ou le recyclage sont des concepts bien connus qui ont évolué à travers l'histoire humaine et qui sont devenus aujourd'hui très complexes. Mais malgré d'énormes progrès dans les domaines des processus et des outils supports à ces activités, l'idée centrale est restée la même : déterminer un ensemble de besoins provenant généralement d'un client et développer un produit ou un système de produit-services (Product Service System - PSS) répondant à ces besoins (Terzi et al., 2010).

L'augmentation des quantités, des variétés et de la complexité des produits a, au cours des années, nécessité la multiplication et la spécialisation des acteurs qui sont maintenant répartis dans de nombreuses structures géographiquement dispersées. Les informations concernant le produit sont, elles aussi, dispersées. Dans cet environnement hétérogène et

distribué, obtenir que les personnes, les informations et les processus soient dynamiquement alignés sur le même référentiel est un défi (Terzi et al., 2010).

Les organisations prennent ainsi conscience qu'il est essentiel de mettre l'accent sur les produits et sur la création d'un référentiel partagé support à ces produits. Au vu de cette préoccupation, le PLM peut être défini comme « a systematic concept for the integrated management of all product- and process-related information through the entire lifecycle, from the initial idea to the end of life » (Sääksvuori and Immonen, 2004). De nombreuses autres définitions du PLM existent. Face à l'hétérogénéité des définitions existantes, (Paviot, 2010) propose la synthèse suivante : « le PLM est une approche intégrée qui utilise les Technologies de l'Information (TI) pour permettre la gestion collaborative des données numériques relatives au produit, au cours de toutes les phases de son cycle de vie ». Ainsi le PLM implique :

- « un point de vue stratégique, selon lequel le produit/service est considéré comme le seul créateur de valeur pour l'entreprise ;
- la mise en œuvre d'une approche collaborative permettant la mise en commun de toutes les compétences de l'entreprise, distribuées parmi différents acteurs ;
- l'adoption d'un grand nombre de solutions informatiques et d'outils. » (Paviot, 2010).

Le terme PLM est également régulièrement employé pour parler de systèmes informatiques support de la stratégie PLM. Dans ces travaux, le terme PLM sera systématiquement différencié du terme Product Data Management (PDM), considéré comme un des éléments les plus importants de l'infrastructure informatique pour la mise en œuvre de cette stratégie, notamment pour les activités de conception.

Mise en œuvre de la stratégie PLM en conception : les outils PDM

Durant le processus de conception, les échanges de données entre acteurs d'un projet s'opèrent quasiment exclusivement de manière numérique. Afin de supporter la stratégie PLM durant ce processus de conception de produit, les outils informatiques peuvent être regroupés en deux familles :

- les outils de création de données ou outils d'édition (outils d'*authoring*) : « ils aident le concepteur à créer des données techniques en fonction des tâches qui lui ont été affectées. Ces données permettent de définir le produit virtuel » (Paviot, 2010) ;
- les Systèmes de Gestion de Données Techniques (SGDT) (Maurino, 1994) également appelés PDM : ils ont pour fonction de gérer les données relatives aux produits tout au long de leur cycle de vie (Eynard et al., 2004). Ils intègrent également sous la forme de modules spécifiques les outils relatifs à l'organisation et au pilotage des activités de conception. Généralement, leur intégration au sein des PDM se décline sous la forme de *workflows* ou de fonctionnalités de gestion de projet.

La Figure 1.15 rappelle les liens qui peuvent exister entre les approches et stratégies PLM et leur mise en œuvre grâce aux différentes ressources et outils comme les PDM (Silventoinen et al., 2011).

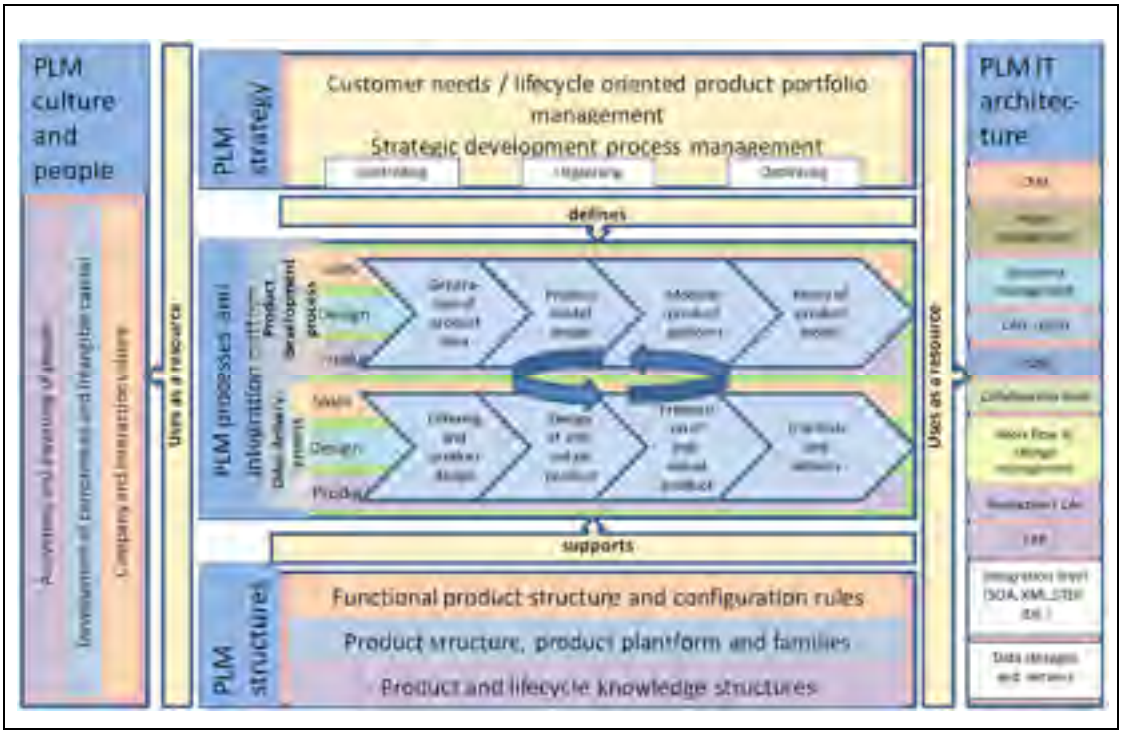


Figure 1.15 Représentation holistique de l'approche PLM (Silventoinen et al., 2011)

Bien que des efforts visent à unifier les pratiques et les outils de modélisation durant les phases amont de la conception de systèmes (Plateaux et al., 2009), une grande hétérogénéité des outils d'édition subsiste (Abramovici and Bellalouna, 2007) et chaque expert utilise un voire plusieurs outils de création de données. En revanche, et bien que les outils de gestions de données soient assez différents selon les domaines impliqués (Abramovici and Bellalouna, 2007), une seule et même solution est généralement privilégiée au sein d'un domaine. Par exemple, dans une stratégie PLM et afin de concevoir un produit, l'infrastructure informatique fournit généralement un certain nombre de services structurés en modules, apportant chacun une liste de fonctionnalités complémentaires. On retrouve ainsi traditionnellement des modules permettant :

- De gérer les personnes, leurs rôles et leurs droits au sein des organisations,
- D'automatiser certains traitements ou certains processus grâce aux *workflows*,
- De gérer la collaboration au sein de l'entreprise, mais aussi avec les clients et fournisseurs (entreprise étendue) tout en respectant la propriété industrielle,
- De gérer les projets : planification, plan de charge, gestion des tâches, …
- De gérer des portefeuilles de produits et l'introduction de nouveaux produits (Portfolio and Program Management),

- De gérer les exigences (Requirements Management),
- De suivre et d'analyser les matériaux présents dans chacun des composants d'un système et de s'assurer qu'un produit est conforme aux différentes normes en vigueur (Compliancy),
- D'optimiser certains processus en fournissant des informations sur les données et les usages (Decision Support – Business Intelligence),
- De gérer les sous-traitants (Sourcing),
- De gérer les demandes et les approbations de changement (Engineering Change Management),
- De gérer la maquette numérique des produits, les différentes nomenclatures associées et les liens avec les différents logiciels d'édition associés (Bill Of Material management),
- De gérer les variantes, la modularité et la personnalisation des produits (Configuration Management),
- De faciliter la capitalisation de connaissances, la classification des modèles et leur réutilisation (Knowledge Capture and Reuse).

Ces modules sont ensuite combinés et adaptés afin de répondre aux besoins de secteurs industriels particuliers. Dans la partie suivante, des précisions sont apportées quant à l'usage d'outils afin de supporter la conception de systèmes mécatroniques.

Les outils supports de la conception de systèmes mécatroniques

Les PDM, traditionnellement issus du domaine mécanique, voient leur usage étendu à la gestion de l'ensemble de la partie *hardware* du produit. La partie *hardware* peut être définie comme l'ensemble des éléments physiques du système, par opposition à la partie logicielle ou *software*. On voit ainsi apparaître les termes Electrical PDM (E-PDM) et de Mechanical PDM (M-PDM) (Abramovici and Bellalouna, 2007). Les E-PDM gèrent les données issues des outils d'édition nommés Electrical/Electronic Engineering Solutions (EES) utilisés par les électriciens/électroniciens, alors que les M-PDM gèrent les données issues des outils d'édition nommés Computer Aided Design (CAD) utilisés par les mécaniciens (Abramovici and Bellalouna, 2007).

Dans le cadre du développement de systèmes mécatroniques, des singularités liées à la nature même du produit ont été observées et/ou rapportées durant des retours d'expérience industriels et lors de collaborations académiques et industrielles avec des électroniciens et mécatroniciens. Ces singularités induisent la nécessité d'adapter les modules traditionnels des PDM et les processus liés à leur usage. Une analyse comparative des solutions PDM proposées par les principaux éditeurs que sont Dassault Systèmes[4], Siemens PLM Software[5] et PTC[6] a permis de déterminer que l'offre proposée par PTC semble se positionner actuellement comme l'une des plus complètes pour gérer les données liées à la conception de systèmes mécatroniques, notamment lors des phases de conception détaillée. En effet, la combinaison des solutions Windchill® et Integrity® permet de prendre en considération à la fois la gestion des données *hardware* et des données *software*.

Le processus décrit dans ce paragraphe est également tiré de journées techniques durant lesquels des industriels[7] ont réalisé des retours d'expériences. Tout d'abord, concernant la gestion des données *hardware* (mécanique, électronique, électrique), l'intégration des données électroniques est généralement réalisée selon le processus suivant. Les fournisseurs de composants électroniques sont référencés dans le module de gestion des sous-traitants et les composants validés et autorisés sont saisis dans le PDM avec leur documentation technique, leurs caractéristiques, leur représentation schématique et leur empreinte 2D à l'échelle. Ces composants sont ensuite accessibles grâce à une référence unique, ce qui permet au logiciel de conception électronique de connaître la liste des composants disponibles. Ces composants sont ensuite utilisés depuis la partie permettant de réaliser les schémas et le routage des différentes pistes de la carte électronique. Le fichier décrivant la carte électronique ainsi créée est ensuite intégré dans le PDM et analysé afin de pouvoir enrichir la nomenclature du système avec la liste des composants

---

[4] http://www.3ds.com
[5] http://www.plm.automation.siemens.com
[6] http://www.ptc.com
[7] Notamment :
- Technofan, filiale du Groupe Safran, gère ses données CAO MCAD et ECAD grâce aux outils PLM PTC Windchill PDMLink et PTC Windchill Supplier Management,
- Valeo travaille depuis plus de 3 ans au développement d'un PLM mécatronique basé sur ENOVIA.

utilisés et les quantités associées. Seuls les soudures et les vernis ne sont pas pris en considération. Ce fichier de CAO électronique est également utilisé afin de générer une géométrie morte permettant de déterminer l'encombrement lié à la carte ainsi créée directement dans la CAO mécanique.

Enfin, les développeurs informatiques utilisent généralement des outils nommés Software Configuration Management (SCM) ou Concurrent Versions System (CVS) pour gérer leur code source (Abramovici and Bellalouna, 2007). Ces outils de gestion de données diffèrent des PDM par plusieurs aspects. Crnkovic et al. ont comparés ces outils dans les années 2000 en listant les fonctionnalités principales des deux systèmes puis en tentant de distinguer les points communs et les différences, comme illustré sur la Figure 1.16 (Crnkovic et al., 2003). Il est cependant nécessaire de noter que même pour des fonctionnalités portant le même nom et répondant au même besoin, des divergences liées aux métiers des usagers demeurent, rendant l'implémentation de ces fonctionnalités assez différentes au final.

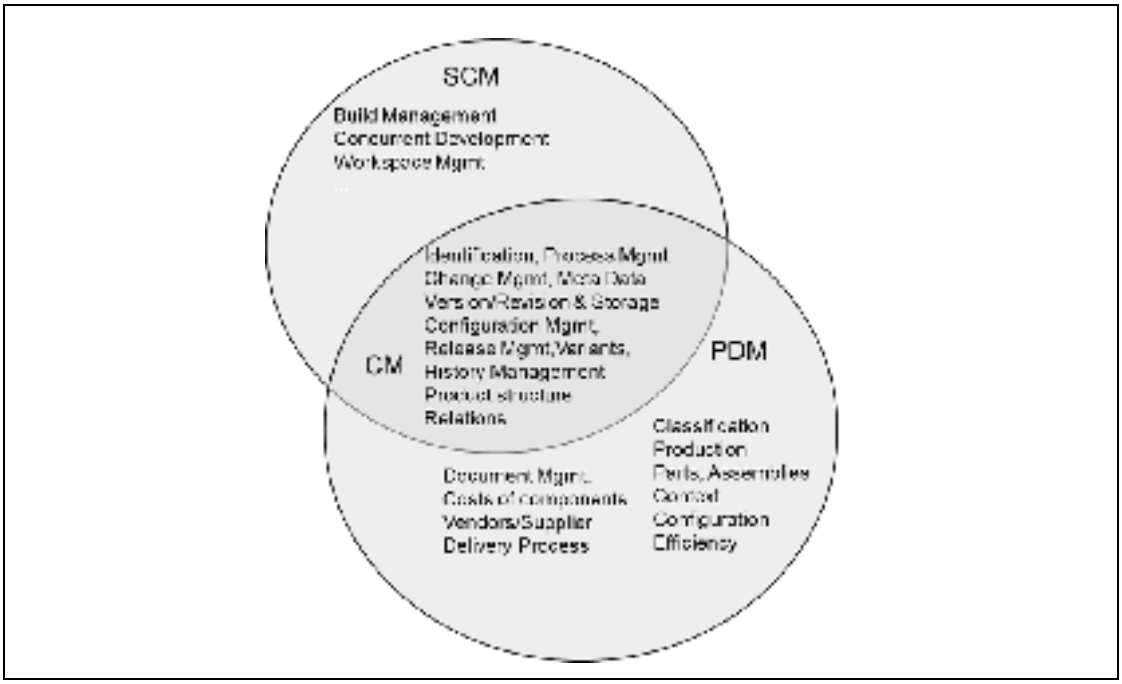


Figure 1.16 Comparaison des fonctionnalités des PDM et des SCM (Crnkovic et al., 2003)

Dans cette section, les enjeux du PLM, dans lequel s'inscrivent ces travaux, ont été présentés avant de décrire les fonctionnalités principales des solutions PDM, considérées comme un des éléments les plus importants dans la mise en œuvre de la stratégie PLM pour supporter la conception de systèmes. Un focus spécifique a été réalisé sur la conception de systèmes mécatroniques en listant les familles d'outils utilisés par les acteurs des différents domaines. Dans la section suivante, l'ingénierie système est présentée comme un autre cadre général dans lequel s'inscrivent ces travaux. Sa description permettra au final de positionner cette dernière par rapport au PLM présenté ci-avant.

1.1.3.2 L'ingénierie système

Historique et définitions

L'ingénierie système est une discipline apparue il y a plus de 40 ans. Pour certains, son avènement date du lancement des principaux programmes spatiaux (Grieves, 2012). Malgré cet historique, Grieves note que l'ancrage de cette discipline dans les milieux universitaires est bien moins important que dans l'industrie (Grieves, 2012). L'explication avancée est que l'ingénierie système, par son caractère pluridisciplinaire, n'est pas bien adaptée à l'organisation très mono disciplinaire et spécialisée du monde universitaire. Des ouvrages de synthèse sont néanmoins apparus (Blanchard, 2008; Kossiakoff et al., 2011) mais les principaux acteurs de ce domaines restent industriels, fédérés par différentes associations comme l'International Council on Systems Engineering (INCOSE) ou l'AFIS, proposant des manuels de référence et des livres blancs. Ces manuels s'accordent à définir l'ingénierie système comme une « methodical, disciplined approach for the design, realization, technical management, operations, and retirement of the system » (NASA, 2007) ou encore comme « une démarche méthodologique générale qui englobe l'ensemble des activités adéquates pour concevoir, faire évoluer et vérifier un système apportant une solution économique et performante aux besoins d'un client tout en satisfaisant l'ensemble des parties prenantes » (AFIS, 2011).

Plus précisément, l'AFIS définit l'ingénierie système comme :

- « un processus coopératif et interdisciplinaire de résolution de problème ;
- s'appuyant sur les connaissances, méthodes et techniques issues de la science et de l'expérience ;
- mis en œuvre pour définir, faire évoluer et vérifier la définition d'un système (ensemble organisé de matériels, logiciels, compétences humaines et processus en interaction) ;
- apportant une solution à un besoin opérationnel identifié conformément à des critères d'efficacité mesurables ;
- qui satisfasse aux attentes et contraintes de l'ensemble de ses parties prenantes et soit acceptable pour l'environnement ;
- en cherchant à équilibrer et optimiser sous tous les aspects l'économie globale de la solution sur l'ensemble du cycle de vie du système » (AFIS, 2011).

Ingénierie Système vs. PLM : différences, complémentarité et tendances futures ?

L'ingénierie système et le PLM s'accordent sur l'idée même qu'un système n'est pas un objet monolithique, mais un ensemble d'éléments interagissant et ayant pour rôle de produire des résultats plus importants que ce qu'ils auraient pu faire individuellement. Mis à part ce point, les approches sont assez différentes, malgré des revendications très similaires.

Une première de ces revendications partagées concerne le fait de s'intéresser à l'ensemble du cycle de vie du système. Dans la réalité, l'ingénierie système se concentre principalement sur la phase de développement du système. L'hypothèse implicite est alors que si les travaux d'ingénierie système sont suffisamment bien réalisés, le système conçu répondra aux exigences attendues. Les autres phases du cycle de vie sont alors de simples exercices d'exécution qui incombent à quelqu'un d'autre (Grieves, 2011).

L'ingénierie système revendique également le fait de favoriser la collaboration. Durant les phases amont, typiquement les phases de « Product Definition » et « Conceptual Design » (Figure 1.6), cette collaboration est effectivement facilitée par l'usage d'un référentiel commun, souvent basé sur un langage de modélisation tel que

SysML (Friendenthal et al., 2008; OMG Group, 2007; Willard, 2007). Le principe sous-jacent à l'ingénierie système, notamment au *Model Based Systems Engineering* (MBSE) (Chhaniyara et al., 2011; Sellgren et al., 2009), est qu'un système peut être décomposé en sous composants et que ces composants doivent interagir de manière à satisfaire les exigences requises. Ainsi, l'ensemble du système est décomposé en unités toujours plus petites jusqu'à arriver à une granularité satisfaisante. Ce processus de raffinement du système est également complété par la création d'un certain nombre de représentations différentes reflétant le comportement du système (Grieves, 2011). Ces différents modèles sont structurés en diagrammes normalisés, permettant de conserver le lien entre chacun d'eux. La collaboration correspond alors à la prise en considération des points de vue des concepteurs issus de différents domaines par les architectes systèmes. À l'issue de ces travaux, le modèle devient alors une référence permettant à chaque domaine de débuter la conception détaillée du système. La collaboration est alors organisée de façon descendante (*top-down*). Le PLM, par opposition, structure l'information et les données autour de nombreux modèles métier en tentant de conserver la cohérence entre ces modèles. L'approche peut alors, sous cet angle, être plutôt considérée comme ascendante (*bottom-up*).

D'autres différences existent entre ces approches, classées par (Grieves, 2012) selon deux critères qui sont le comportement dynamique (statique vs. dynamique) et le caractère spatial de l'information (2D vs. 3D). Mais il semble surtout intéressant de détailler la complémentarité de ces approches. Comme décrit ci-avant, l'ingénierie système apporte des avantages indéniables lors des phases amont de la conception pour définir et structurer les exigences, lancer des premières simulations (généralement 0D ou 1D) (Sinha et al., 2001), définir des architectures système candidates et les évaluer (Graignic et al., 2013), etc. Les exigences peuvent alors servir d'élément de traçabilité tout au long du cycle de conception, et ce grâce aux modules dédiés dans les solutions PDM. L'architecture sélectionnée peut également servir à initier le travail des différentes disciplines.

Afin de combiner ces approches et de mettre leurs forces en synergie, différentes pistes peuvent être suggérées. La première concerne le fait de mieux intégrer l'ingénierie système dans l'approche PLM. L'intégration des bonnes pratiques issues des manuels

développés par les ingénieurs système sous la forme de standards de collaboration ou de patrons (pattern) directement dans les PDM permettrait à l'ingénierie système d'avoir un impact plus important sur le cycle de développement et sur un plus grand nombre de concepteurs. Une seconde piste serait la prise en compte plus systématique de l'ensemble des phases du cycle de vie lors de l'élaboration du modèle issu du travail de consolidation des ingénieurs systèmes. Enfin, une dernière piste serait l'adoption plus systématique des méthodes de vérification issues de l'ingénierie système lors des intégrations successives des conceptions en provenance des différentes disciplines.

Dans cette section, une définition sommaire de l'ingénierie système et un positionnement de cette démarche par rapport à l'approche PLM ont été proposés. Dans la section suivante, la cybernétique et les Cyber–Physical Systems seront également présentés par comparaison à l'approche PLM. Ces positionnements ont pour but de montrer les similitudes entre les finalités recherchées et les complémentarités de ces approches.

1.1.3.3 La cybernétique et les Cyber–Physical Systems

La cybernétique est une science aussi ancienne que vaste. Une multitude de définitions existent, émanant de domaines aussi divers que la sociologie, les sciences cognitives, la robotique, etc. La définition retenue dans ces travaux précise que la cybernétique est une science qui étudie exclusivement les communications et leurs régulations dans les systèmes naturels et artificiels (Wiener, 1965). Il est important de noter que cette science semble à l'origine de nombreux domaines de recherche comme l'intelligence artificielle, l'informatique, l'automatique, etc.

De cette science est née un type de système nommé Cyber–Physical System (CPS). Les CPSs font référence à, selon Rajkumar et al., la prochaine génération de systèmes qui nécessitent une intégration étroite de l'informatique, de la communication et des technologies de contrôle pour atteindre stabilité, performance, fiabilité, robustesse et efficacité dans la gestion des systèmes physiques de nombreux domaines d'application (Rajkumar et al., 2010). Même si le contexte de développement des CPSs fait apparaitre des problématiques et défis spécifiques à ce domaine, une grande majorité des problématiques d'intégration rencontrées sont similaires à celles identifiées lors de la conception de systèmes impliquant des équipes multidisciplinaires comme la mécatronique. De plus, et comme l'illustre la Figure 1.17, la recherche autour des CPSs ne se concentre pas uniquement sur les technologies utiles aux CPSs, mais bien sur les méthodes de conception et d'intégration. Kumar précise d'ailleurs qu'il reste beaucoup de recherche à entreprendre afin de pallier aux problématiques de complexité et de productivité liées à la conception et au développement des CPSs (Kumar, 2012). Il souligne également que des besoins en modélisation des différents aspects d'un système, à différents niveaux d'abstraction et pour divers domaines d'application sont palpables.

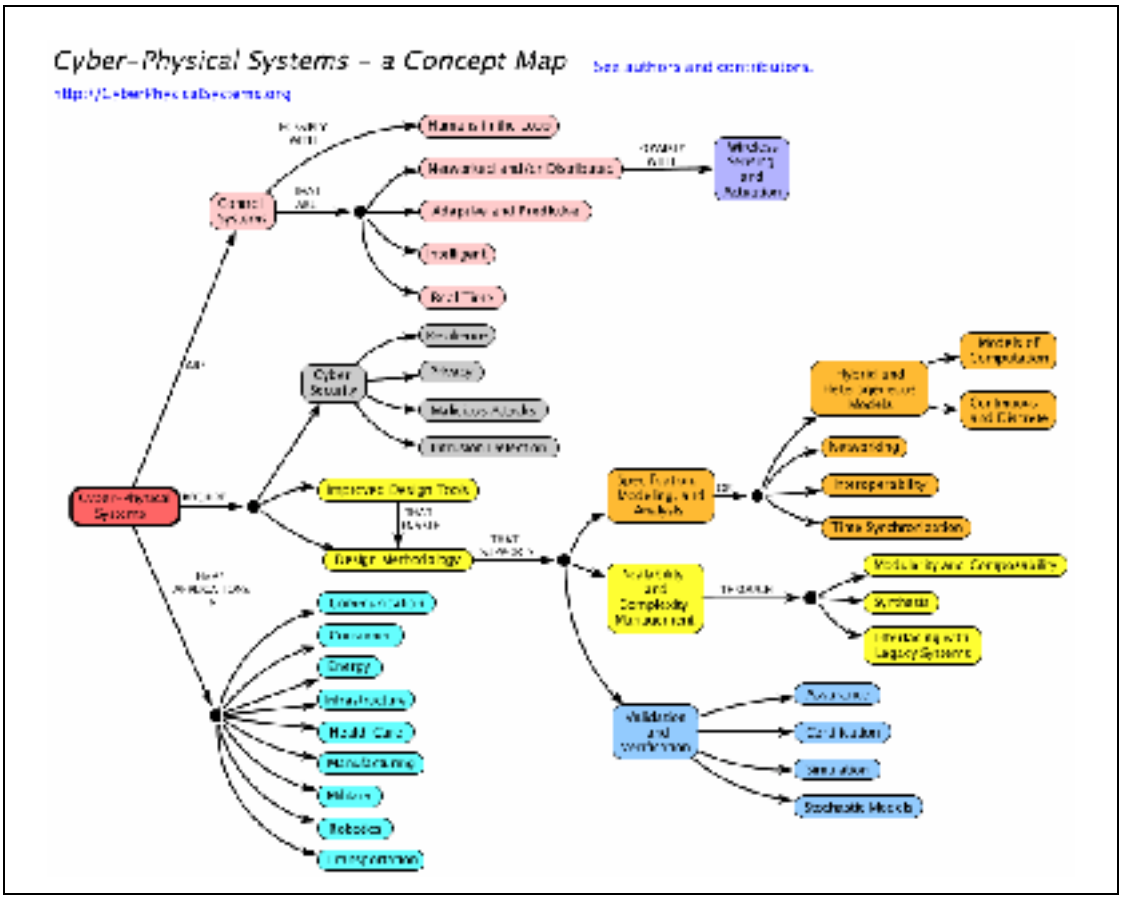


Figure 1.17 Carte mentale définissant les Cyber-Physical Systems (Asare et al., 2012)

Les communautés scientifiques mécatronique et CPS ne semblent que peu interagir et partager leurs expertises. L'explication que nous avançons ici tient au fait que les produits mécatroniques sont historiquement des produits électromécaniques introduisant de plus en plus d'électronique et de logiciel alors que les CPSs sont historiquement des « Cyber-Systems » (Rajkumar, 2012) qui s'intéressent de plus en plus à la réalité physique dans laquelle ils doivent évoluer. La Figure 1.18 illustre la prédominance de l'aspect informatique dans cette vision. Dans cette logique, un parallèle est souvent fait entre la manière dont internet a transformé la façon dont les humains interagissent entre eux, a révolutionné la façon d'accéder à l'information et a même changé la façon dont les produits sont achetés et vendus, et les CPSs qui vont transformer la façon dont les humains interagissent et contrôlent le monde physique qui nous entoure (Rajkumar, 2012).

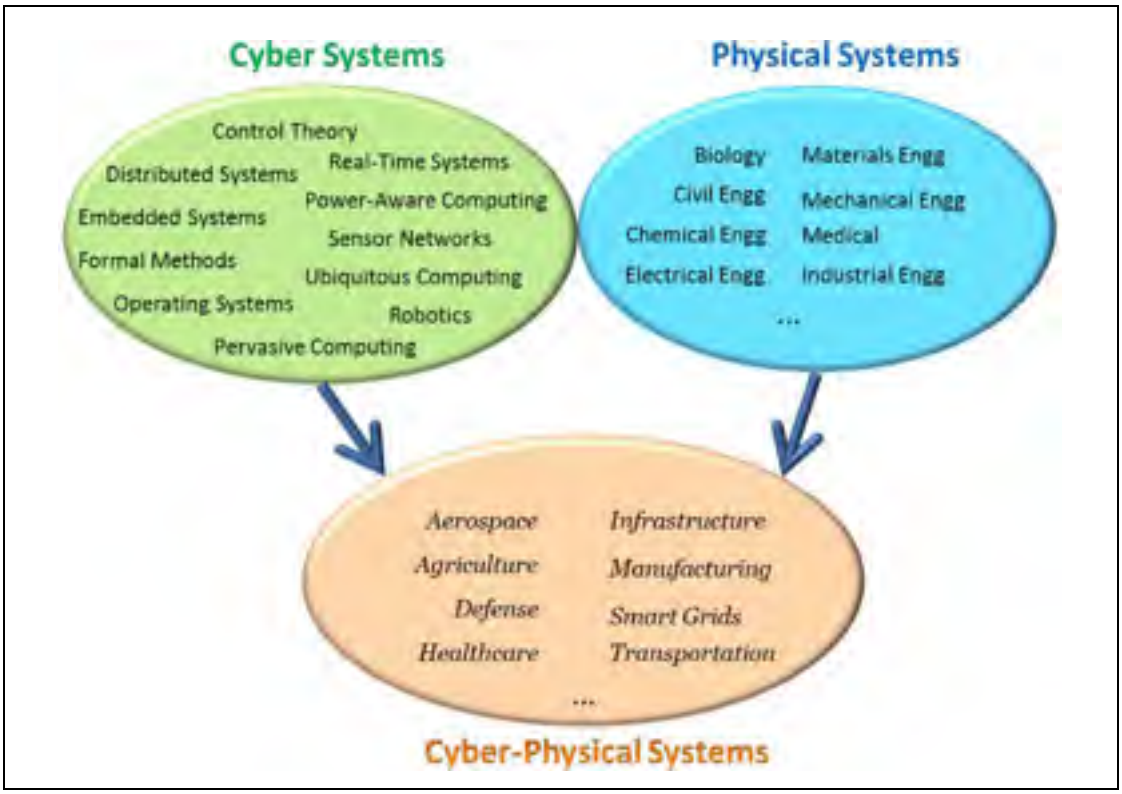


Figure 1.18 La vision des acteurs provenant du domaine des Cyber-Systems sur les CPSs (Rajkumar, 2012)

Comme nous venons de le voir, les spécialités de recherche incluses dans les domaines de la mécatronique et des CPSs sont relativement disjointes mais complémentaires : alors que la mécatronique se focalise tout particulièrement sur les aspects *hardware* d'un système, les CPSs restent centrés sur les aspects *software* de ce dernier. Néanmoins, ces deux domaines partagent un même objectif qui est de concevoir et de produire des systèmes intégrés. Pour ce faire, des méthodologies de conception collaboratives multidisciplinaires, supportées par des outils adéquats, doivent être proposées.

Les deux sections précédentes ont permis de présenter succinctement l'ingénierie système et la cybernétique avec un focus particulier sur les CPSs. Si l'ingénierie système peut être comparée au PLM quant à ses objectifs, les CPSs illustrent quant à eux une certaine évolution du domaine « informatique », couramment appelé « logiciel » dans ce manuscrit, vers la prise en considération des aspects physiques du système et l'intérêt des acteurs de ce domaine pour les méthodologies de conception collaborative multidisciplinaire.

Dans la section suivante, l'*Application Lifecycle Management* (ALM), considéré comme l'équivalent du PLM dans le domaine du logiciel, est présentée. Cette approche montre ainsi l'évolution des systèmes de gestion de code source, Software Configuration Management (SCM) ou Concurrent Versions System (CVS), vers des environnements plus complets prenant en charge de nombreux autres aspects liés au développement logiciel.

1.1.3.4 L'Application Lifecycle Management

Dans les années 2000, de premiers travaux ont été réalisés afin de comparer les fonctionnalités des PDM à la gestion de configuration logicielle (SCM) (Dahlqvist et al., 2001). La Figure 1.16 reprend une partie des conclusions en tentant de distinguer les points communs et différences (Crnkovic et al., 2003). Les SCM y sont considérés comme des environnements de développement logiciel complets, incluant les fonctionnalités de Build Management (gestion de la compilation) et de Release Management (gestion des versions compilées). Toutefois, ses fonctions primaires restent le contrôle des versions (appelé parfois gestion des révisions : version control ou revision control ou source control), le développement concourant, la sélection de la configuration et de la gestion des espaces de travail. En se concentrant sur ces fonctionnalités, les SCM sont très semblables aux PDM, bien que dans le cas des espaces de travail, les fonctionnalités offertes par les PDM restent très différentes et limitées comparées à celles des SCM. Dans le cadre de la mise en œuvre de la stratégie ALM, les SCM restent un des éléments centraux car ils contiennent l'ensemble de la production des concepteurs informatiques.

L'ALM « has emerged to indicate the coordination of activities and the management of artefacts (e.g., requirements, source code, test cases) during the software product's lifecycle » (Kääriäinen and Välimäki, 2009). L'ALM provient du domaine intitulé *Configuration Management* (CM), qui fournit traditionnellement des fonctionnalités de stockage, de gestion des versions et de traçabilité pour les différents items gérés. Comme le montre la Figure 1.19, l'ALM s'appuie sur des fonctionnalités supplémentaires afin de soutenir la communication entre les différents acteurs, la génération de rapports, la

traçabilité quant aux décisions prises et leurs impacts sur les données modifiées, l'intégration d'outils externes, tels que les outils de gestion des exigences et des défauts, le Build Management et le Release Management. Tout comme pour le PLM, cette liste n'est pas exhaustive et l'ALM peut alors être défini comme une approche intégrée qui utilise les technologies de l'information pour permettre la gestion collaborative des données numériques relatives au logiciel, au cours de toutes les phases de son cycle de vie.

Figure 1.19 Éléments principaux constituant l'ALM (Kääriäinen and Välimäki, 2009)

Actuellement, il semble assez difficile de vérifier le niveau de diffusion des concepts liés à l'ALM dans l'industrie logicielle. En effet, cette industrie prend de plus en plus conscience de l'importance des phases d'exploitation et de maintenance avec l'augmentation de la criticité des applications logicielles dans certains systèmes, mais peu d'acteurs maîtrisent véritablement la portée de l'approche ALM (Chappell, 2009). Néanmoins, l'ALM dénote une sorte de convergence entre les méthodes d'organisation et de gestion des données pour le développement de nouveaux produits, qu'ils soient purement *hardware*, purement *software*, ou qu'ils soient hybrides.

1.2 Synthèse de la mise en contexte, problématique et objectifs

1.2.1 Synthèse de la mise en contexte

Dans cette première partie, le contexte lié à la conception collaborative multidisciplinaire de systèmes intégrés a été exposé. Tout d'abord, des précisions ont été apportées quant aux différents types d'intégration, qu'ils concernent le produit, les activités d'ingénierie ou les disciplines. Puis, différents modèles de processus de développement de produit ont été présentés afin de préciser à quelles phases de la conception s'intéressent ces travaux. Les phases de conception préliminaire et de conception détaillée sont ainsi principalement ciblées pour leur impact fort sur la finalité recherchée, à savoir l'intégration multidisciplinaire.

Les systèmes mécatroniques ont été choisis comme cadre applicatif pour la grande diversité des expertises intervenant lors de leur conception. Ils ont donc été utilisés pour illustrer les différents types d'intégration et des modèles de processus de développement spécifiques ont été présentés. Ces systèmes mécatroniques sont également utilisés dans le chapitre 4 de ce manuscrit afin de démontrer la pertinence de notre proposition.

Enfin, afin de préciser dans quels cadres les travaux présentés ici s'inscrivent, le PLM, l'ingénierie système, la cybernétique et l'ALM ont été définis avant de montrer en quoi la finalité recherchée est en lien avec ces domaines. La Figure 1.20 synthétise le cadre général de la conception collaborative multidisciplinaire en fonction des différents domaines impliqués et leur appartenance historique.

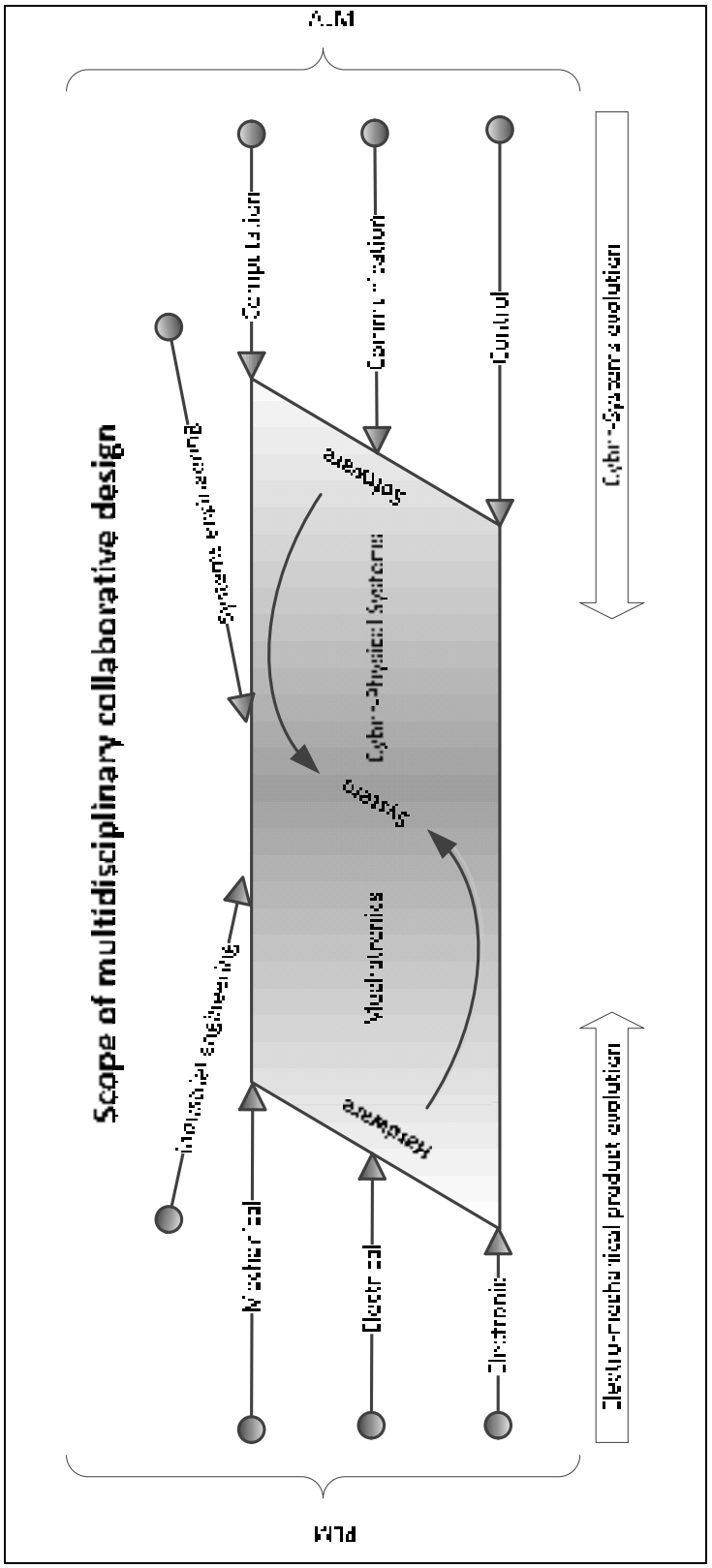


Figure 1.20 Synthèse des différents cadres généraux auxquels contribuent ces travaux

1.2.2 Analyse critique et problématique

La section précédente a mis en exergue les enjeux industriels et scientifiques associés à la conception collaborative multidisciplinaire. Les méthodes visant à augmenter les intégrations relatives au produit, aux activités d'ingénierie et aux disciplines proviennent de multiples communautés scientifiques, sans pour autant apporter de réponse globale et satisfaisante à ces besoins d'intégration.

Ces méthodes, lorsqu'elles sont implémentées sur le terrain industriel, doivent trouver leur place dans un mode d'organisation de la conception « par projet », très largement répandu. Ce mode d'organisation s'appuie sur trois phases (Gidel and Zonghero, 2006). La première, intitulée « formalisation », correspond au passage de l'idée du projet au projet formalisé. La seconde correspond aux « études de définition du projet », qui permettent de « s'organiser pour mener à bien la réalisation du projet ». La troisième, la phase de « réalisation », permet de « maîtriser le projet, contrôler, anticiper, agir » (Gidel and Zonghero, 2006). Différents acteurs de l'entreprise, voire de l'entreprise étendue, s'organisent alors autour d'activités ou d'actions coordonnées afin de répondre à un besoin préétabli tout en respectant la qualité attendue ainsi que le délai et le budget impartis.

Durant la phase de réalisation, la maîtrise du projet est assurée par la mesure des écarts par rapport aux référentiels du projet. Parmi ces référentiels, on note le *Product Breakdown Structure* (PBS), le *Work Breakdown Structure* (WBS) et *l'Organisation Breakdown Structure* (OBS) qui permettent respectivement de définir de façon arborescente le produit du projet sous la forme de livrables, la planification du travail à effectuer sous la forme de tâches et la structuration de l'équipe projet.

Grâce au WBS, ce mode d'organisation par projet permet d'assurer la parallèlisation des tâches et d'éviter la surcharge de certains acteurs. Ces tâches sont ensuite réparties en prenant en considération les liens de dépendance qui peuvent exister entre elles. Si des projets similaires, en termes d'ampleur ou de type de produit conçu, ont déjà été menés, les connaissances acquises lors de la gestion de ces projets sont alors mises à profit afin de définir très en amont les grandes phases du projet ainsi que les jalons et livrables

associés. Il est alors possible de créer des modèles de projet, synthétisant le plan de développement type qui précise pour chaque étape les objectifs et livrables attendus, l'intervention des expertises, les rôles des différents acteurs, les modes de validation, etc.

Lors de la réalisation d'un projet de conception, l'avancement du projet est mesuré grâce au contrôle de la présence et de la qualité des livrables attendus, notamment lors de réunions dites de revue de projet. Ces séances sont essentielles afin d'assurer la cohérence et l'intégration des différentes contributions. Il s'agit alors de « maîtriser le projet ».

Cette organisation « par projet » ou « en mode projet », initiée dans les années 80 (Gidel and Zonghero, 2006), a permis de réduire très largement les temps de développement, améliorant ainsi drastiquement les temps de mise sur le marché. Elle a également donné naissance aux plateaux projet, regroupant des représentants de chacune des activités d'ingénierie impliquées et formant un noyau pluridisciplinaire.

Selon notre analyse et en nous appuyant sur nos expériences et contacts professionnels, cette organisation présente néanmoins quelques limites. La première tient au fait que les plateaux projet ne suffisent pas à pallier au cloisonnement entre les disciplines. En effet, même si des représentants des différentes activités d'ingénierie impliquées sont physiquement présents sur le même plateau, l'ensemble des équipes de développement ne bénéficient pas nécessairement de ce rapprochement et interagissent peu ensemble. Les directives pour ces équipes proviennent alors des décisions prises par les responsables du projet, dans une approche essentiellement *top-down*.

Le terme, *top-down* ou approche descendante, correspond à une logique de prise de décisions réalisées aux niveaux hiérarchiques les plus hauts pour être finalement mises en œuvre au niveau opérationnel (le terme opérationnel sera précisé dans la section relative à la structuration de l'état de l'art). Dans le cadre de l'intégration relative au produit et aux activités d'ingénierie, ceci signifie généralement que l'intégration multidisciplinaire est prévue dès les phases amont, à savoir « formalisation » et « études de définition du projet » (Gidel and Zonghero, 2006). Les tâches sont alors affectées, réalisées et contrôlées. C'est en ce sens que la maîtrise du projet, évoquée précédemment, est basée sur la mesure des écarts par rapport aux référentiels du projet définis a priori, tels que le

WBS, le PBS ou l'OBS. Cette vérification permanente de la concordance entre la réalisation et l'ensemble des référentiels est à l'origine du terme *project-planned* parfois également appelée « méthodes prédictives » ou « plan-driven » (Messager, 2009). Ce terme est défini plus précisément dans la section relative aux méthodes agiles (Sommerville, 2010), auxquelles il s'oppose, dans le chapitre dédié à l'état de l'art. Cette méthode de contrôle s'inscrit dans une logique *top-down*. Par opposition à l'approche *top-down*, le terme *bottom-up*, correspond à l'idée que les contributions des différentes disciplines doivent être intégrées a posteriori de leur réalisation. Une plus grande liberté est laissée aux concepteurs et la remontée de leurs travaux correspond à une sorte de synthèse participative. Ces approches, descendantes ou ascendantes, sont également représentatives de la notion de « portée » des décisions. Cette notion est reprise dans le chapitre dédié à l'état de l'art.

La seconde limite tient au principe même de contrôle de l'avancement du projet. Alors que l'organisation par projet est à l'origine une méthode devant servir à s'organiser pour concevoir et réaliser des produits innovants, la problématique de la maîtrise des risques tend à rationaliser le déroulement des projets. Ainsi, lors de la conception d'un nouveau produit, la structure des projets, la durée de leurs phases principales, les livrables attendus sont standardisés jusqu'à rendre ce mode d'organisation rigide et ne laissant que peu de place aux initiatives et à l'innovation. Tout écart est alors analysé en termes de dysfonctionnement qu'il convient d'éliminer.

Cette organisation basée sur la division et la répartition des tâches entre les acteurs des différentes disciplines (illustrée par la Figure 1.12), est souvent présentée comme une méthode nécessaire pour faire face à la complexité liée aux activités de conception (Aca et al., 2006). Mais en raison de cette division, un nombre restreint de personnes impliquées dans le projet ont une vue d'ensemble sur les problèmes de conception rencontrés par les différentes équipes. Il faut bien comprendre que cette critique des méthodes d'organisation de type *project-planned* ne remet pas en cause toute l'organisation en « mode projet ». Elle vise tout particulièrement la division des tâches et le mode de pilotage du projet basé sur le respect d'un planning, car ils sont fixés a priori, sans tenir compte des aléas du projet.

Dans cette section, les principaux éléments contribuant à la problématique qui est à l'origine de ces travaux viennent d'être présentés. Cette problématique peut être résumée ainsi :

PROBLÉMATIQUE

Les systèmes conçus lors de conceptions multidisciplinaires restent faiblement intégrés car la collaboration entre les disciplines est faible, due à un cloisonnement trop important entre les acteurs de ces disciplines et à un mode d'organisation *project-planned*, caractérisé par une trop grande rigidité et une diffusion de l'information principalement descendante (*top-down*).

La section suivante est consacrée aux objectifs de nos travaux. Dans un premier temps, des objectifs généraux sont présentés. Ils peuvent être appréhendés comme notre vision, notre idéal, concernant la thématique générale de la conception collaborative multidisciplinaire. Puis, ces objectifs sont précisés afin de les rendre compatibles avec la durée d'un doctorat. Ainsi, parmi les différents leviers potentiels permettant de contribuer à résoudre notre problématique, l'aspect organisationnel est sélectionné et les objectifs spécifiques à ces travaux en sont déduits.

1.2.3 Objectifs

En écho à la problématique, l'objectif général des travaux de recherche vise à développer les connaissances nécessaires à la spécification d'une plateforme de collaboration et de gestion de données commune, dynamique et adaptée aux différents acteurs issus des différentes disciplines. Cette plateforme est envisagée alors comme un véritable levier permettant de favoriser les intégrations relatives au produit, aux disciplines et aux activités d'ingénierie. Pour ce faire, les phénomènes d'intégration doivent être mieux compris et des métriques permettant d'instrumenter ces phénomènes doivent être proposés.

La plateforme envisagée doit prendre en considération les spécificités des métiers impliqués, qu'elles soient organisationnelles ou liées aux outils et données manipulés par ces experts métiers. Ces spécificités se déclinent en termes de gestion des données, en termes de méthodes organisationnelles, mais aussi en termes d'adaptation des interfaces utilisateurs à chacune des disciplines impliquées.

Le cas de la mécatronique est considéré pour ces travaux comme le cadre applicatif de référence pour la diversité des métiers impliqués. Mais cet objectif reste trop ambitieux pour être traité dans le cadre d'un doctorat, et trois sous objectifs ont donc été extraits afin de focaliser les travaux synthétisés dans ce manuscrit. Ces trois sous-objectifs ont été choisis car ils visent à proposer une méthode organisationnelle basée sur des échanges d'informations à la fois *top-down* et *bottom-up*.

Le premier de ces sous-objectifs est de :

OBJECTIF 1	Améliorer la collaboration entre les différentes disciplines en réduisant le cloisonnement existant entre les acteurs qui en sont issus. Il répond directement à la problématique qui relève la corrélation qui existe entre le manque de collaboration et le faible niveau d'intégration résultant pour le produit.

Le second sous-objectif est de :

OBJECTIF 2

Rendre la prise de décisions dans le cadre d'un projet de conception plus aisée grâce à la remontée plus fréquente et plus régulière d'informations opérationnelles précises. Ces informations peuvent concerner les tâches en cours, les exigences prises en considération ou encore les difficultés rencontrées par les différentes équipes de conception. Le but recherché est donc d'assurer que l'information se propage également dans le sens ascendant (*bottom-up*).

Enfin, le troisième sous-objectif est de :

OBJECTIF 3

Permettre la traçabilité entre les évolutions apportées aux données de définition du système conçu et les décisions prises tout au long du projet de développement. Cette traçabilité vise à la fois à un meilleur pilotage du projet de conception, tout comme l'objectif numéro deux, mais également à pouvoir tirer des leçons des projets de conception afin de mieux anticiper les projets futurs.

Ces trois sous-objectifs s'attachent donc tout particulièrement à mieux organiser la collaboration multidisciplinaire d'un point de vue organisationnel. Ils ne traitent en revanche pas des spécificités des données manipulées par les acteurs des différentes disciplines, ni même des caractéristiques des vues métiers associées à ces différentes disciplines.

Après avoir décrit la problématique et les objectifs visés, nous décrivons dans la section suivante les origines de nos travaux, qui sont, avec la problématique, à l'origine de notre hypothèse.

1.2.4 Origines des travaux

Afin de mieux appréhender la nature mais également la structure de ces travaux, il est opportun d'en rappeler les origines. La Figure 1.21 rappelle ces origines et l'influence que ces dernières ont eue sur les objectifs et sur la problématique.

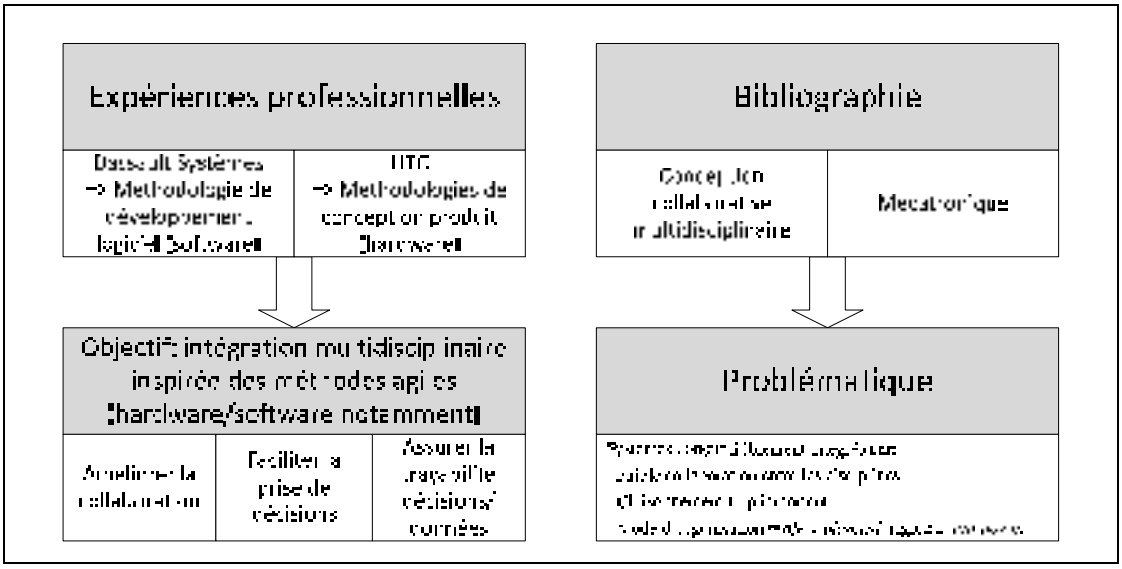


Figure 1.21 Les origines des travaux

Sur la partie gauche de la figure, les expériences professionnelles sont rappelées. La première expérience notable concerne le développement logiciel. Durant cinq années, j'ai en effet eu l'occasion de travailler en tant qu'ingénieur développeur au sein de la société Dassault Systèmes, tout d'abord au service recherche puis sur les solutions *Knowledgeware* (solutions de *Knowledge Based Engineering – KBE*) intégrées à l'ensemble du portefeuille de produits. Durant les trois dernières années, j'ai également été en charge du suivi des indicateurs qualités pour l'ensemble du département développant ces solutions. Ce suivi consiste en la collecte journalière d'informations concernant les interventions considérées comme prioritaires à la fois au sein du système d'information, mais également auprès des développeurs en charge de ces interventions. Ces informations sont ensuite partagées lors d'une courte réunion auprès du responsable de la marque, afin de préparer la réunion exécutive journalière, réunissant notamment l'ensemble des responsables des différentes marques. Lors de cette réunion exécutive, des arbitrages sont effectués et de nouvelles priorités sont définies. Ces dernières sont

renseignées dans le système d'information, afin de permettre la propagation de ces décisions. Ce mode de pilotage basé sur des informations quotidiennes en provenance du terrain et les outils supports à ce pilotage sont en partie à l'origine de l'objectif et des sous objectifs décrits précédemment. Plus globalement, cette expérience professionnelle m'a permis d'acquérir une vision précise quant à une méthodologie de développement logicielle appuyée sur la mise en œuvre de méthodes agiles.

La seconde expérience professionnelle notable, toujours en cours, concerne les méthodologies de conception de produit. Depuis six ans maintenant, je travaille en tant qu'enseignant chercheur au département Génie des Systèmes Mécaniques à l'Université de Technologie de Compiègne. Mes enseignements ont notamment trait aux méthodes de conception collaboratives de systèmes mécaniques. Mes activités de recherche sont menées au sein du laboratoire Roberval, dans le groupe « ingénierie industrielle » qui vise à intégrer les expertises et disciplines impliquées lors de la conception-industrialisation de produit. Ce groupe de recherche travaille étroitement avec le groupe « micro-mécatronique » au sein d'un axe en émergence intitulé « Systèmes Intégrés en Mécanique ». Le travail au sein de ces équipes, ainsi que les différents contacts industriels inhérents à ces activités, ont fortement contribué à obtenir une vision globale sur les méthodologies de conception de produit et sur les outils utilisés durant cette conception.

Ces contacts industriels m'ont notamment permis de découvrir les problématiques induites par la conception de systèmes mécatroniques. Bien plus que l'évolution séquentielle d'un produit mécanique intégrant progressivement des capteurs, des actionneurs, de l'électronique et du logiciel embarqué, la conception de systèmes mécatroniques fait appel à une très grande diversité d'expertises, issues de différentes disciplines. Par exemple, des équipementiers automobiles concevant et fabricant des systèmes comme les systèmes de climatisation, les systèmes d'assistance à la conduite etc. ont ainsi été confrontés à une véritable mutation de leur métier d'origine. En quelques années, ils sont passés de concepteur/fabricant mécanique à intégrateur de solutions basées sur des expertises électriques, électroniques et logicielles. Les outils de travail collaboratif du marché qu'ils achevaient à peine de déployer, ou tout du moins de généraliser et standardiser, se révélaient alors déjà insuffisants pour gérer les échanges entre les différents acteurs et obtenir ainsi une vision globale concernant l'avancement du

projet de conception. Encore maintenant, les initiatives les plus avancées pour ces différents acteurs ont trait à l'intégration des expertises électriques et électroniques au sein des solutions PDM mécaniques. La prise en considération de la partie logicielle du système au sein d'une même plateforme n'est donc encore qu'une lointaine perspective. Certains d'entre eux commencent à échanger autour de l'ALM, mais les solutions mises en place servent à organiser exclusivement le développement de la partie logicielle, et ce, de façon complètement cloisonnée.

Ces contacts industriels m'ont également amené à constater la complexité croissante des organisations et la lourdeur de certains mécanismes mis en œuvre dans les PDM actuels. Pour faire face à la complexité croissante liée aux activités de conception des produits, des processus de plus en plus précis et complexes sont élaborés et formalisés au sein de solutions telles que les PDM. Ces processus permettent en effet de faire collaborer un nombre d'acteurs toujours croissant, mais il nous semble également induire une croissance du temps global consacré aux tâches administratives. De la même manière, les nouvelles fonctionnalités offertes par les logiciels de gestion de données et de collaboration viennent en permanence s'ajouter aux précédentes sous la forme de nouveaux modules, complexifiant l'usage et l'ergonomie de ces solutions. Au final, les utilisateurs ont l'impression de ne plus avoir des outils leur permettant de travailler, mais plutôt de travailler pour les outils; ils n'ont plus l'impression d'être au cœur du processus de conception. Ce ressenti est essentiellement basé sur des échanges réalisés avec des concepteurs et des administrateurs de systèmes de gestion de données et de collaboration issus de différents domaines industriels, notamment du transport. À l'inverse, les processus et les outils utilisés dans le cadre du développement informatique nous apparaissent comme beaucoup plus « simples » et flexibles. Cette simplicité est toute relative car de nombreux mécanismes également complexes sont mis en œuvre, mais ils n'apparaissent pas dans le contexte d'usage. Les interfaces et les fonctions proposées sont épurées. De même, une grande indépendance est laissée aux développeurs qui disposent d'une feuille de route générale et s'organisent ensuite avec une grande liberté pour atteindre les objectifs qui leur sont fixés. Cette forme d'agilité, qui est évoquée tout au long de ce manuscrit, fait également parti des constats ayant contribué à mes travaux.

Sur la base de ces deux expériences professionnelles est né l'objectif de venir proposer une méthodologie de conception de système unifiant les pratiques issues des domaines *hardware* et *software*, qui serait matérialisée par une plateforme de collaboration et de gestion de données commune, dynamique et adaptée aux différents acteurs issus des différentes disciplines. L'idée sous-jacente est ainsi de « prendre le meilleur » de chacune des pratiques et de les adapter pour les rendre compatibles avec les exigences de l'ensemble des disciplines impliquées.

Si les objectifs et sous objectifs ont été principalement déterminés grâce aux constatations industrielles, la problématique a, quant à elle, été formalisée grâce à une étude bibliographique portant sur les méthodologies de conception collaborative visant à faire coopérer des experts métiers issus de différentes disciplines. Cette bibliographie a été complétée par des recherches concernant la mécatronique, considérée dans ces travaux comme un cadre applicatif de prédilection.

La bibliographie menée a été relativement longue et large car elle vise à la fois à connaître les méthodes employées dans le cadre de la conception de produits et de systèmes, les méthodes utiles à la collaboration, les travaux menés visant à l'intégration produit, les contributions liées à l'intégration multidisciplinaire etc. Cette étude s'est dans un premier temps focalisée sur les travaux de la communauté dont je suis issu, à savoir la communauté liée à l'ingénierie collaborative et à la conduite du cycle de vie produit. Mais elle s'est rapidement concentrée sur l'intégration multidisciplinaire, en se basant notamment sur les problématiques induites par la conception de systèmes mécatroniques. Les contributions de type méthodologique pour la conception de ce type de systèmes sont généralement très liées à l'ingénierie système, où j'ai donc dû effectuer quelques recherches. La plupart des travaux développés dans ces diverses communautés se focalisent sur l'intégration via les modèles de données dans les phases amont de la conception. La dynamique des échanges et les aspects organisationnels n'y sont que très peu traités, notamment durant les phases de conception détaillées.

Le fait de ne pas identifier de contribution visant à rendre le processus de conception plus agile et flexible m'a amené à chercher du côté des méthodes dédiées au développement logiciel. L'ALM et les méthodes agiles sont dès lors devenus des sources de connaissances pour proposer une méthode agile pour la conception collaborative multidisciplinaire.

La partie droite de la Figure 1.21 rappelle les deux familles de travaux ayant conduit à l'élaboration de notre problématique portant sur l'intégration multidisciplinaire présentée ci-avant.

En s'appuyant sur ces sous-objectifs et sur la problématique, l'hypothèse a ainsi pu être déterminée. Elle est présentée dans la section suivante.

1.2.5 Hypothèse

Dans les sections précédentes, le manque de collaboration entre les différentes disciplines a été identifié comme un des facteurs majeurs impactant l'intégration finale des systèmes conçus. L'origine de ce manque de collaboration réside dans le cloisonnement des équipes et la gestion de la conception basée sur un mode de collaboration nommé *project-planned*, essentiellement *top-down*. Afin d'améliorer cette collaboration et de prendre des décisions basées sur des informations plus précises en provenance des équipes opérationnelles, l'introduction d'une méthode inspirée des principes fondateurs des méthodes agiles est proposé dans le cadre de ces travaux.

La pertinence des méthodes agiles, originaires du domaine de la fabrication (Matthews et al., 2006) et du développement logiciel (Sommerville, 2010), n'a, à ce jour, pas été évaluée dans le domaine de la conception multidisciplinaire. Matthews et al résument les avantages des méthodes agiles comme « the ability to react rapidly to changes in the environment, whether expected or not » (Matthews et al., 2006). Ils insistent tout particulièrement sur le fait que les méthodes de conception de type *project-planned* n'ont pas la capacité de répondre aux changements tardifs ou imprévus. Dans le cadre de la conception multidisciplinaire, cela se traduit généralement par une prise de risque

minimale et donc à une division et un cloisonnement des travaux des équipes de développement.

Afin de répondre à la problématique proposée ci-avant, les travaux présentés dans ce manuscrit se proposent donc d'étudier l'impact de l'introduction d'une méthode inspirée des principes fondateurs des méthodes agiles sur la conception collaborative multidisciplinaire, et donc, in fine, sur le niveau d'intégration final des systèmes conçus. Cette méthode permet d'assurer la propagation de l'information de façon descendante (*top-down*), mais également ascendante (*bottom-up*), ce qui apparaît comme essentiel pour favoriser le processus d''intégration.

L'hypothèse sur laquelle repose nos travaux peut donc être synthétisée de la manière suivante :

HYPOTHÈSE

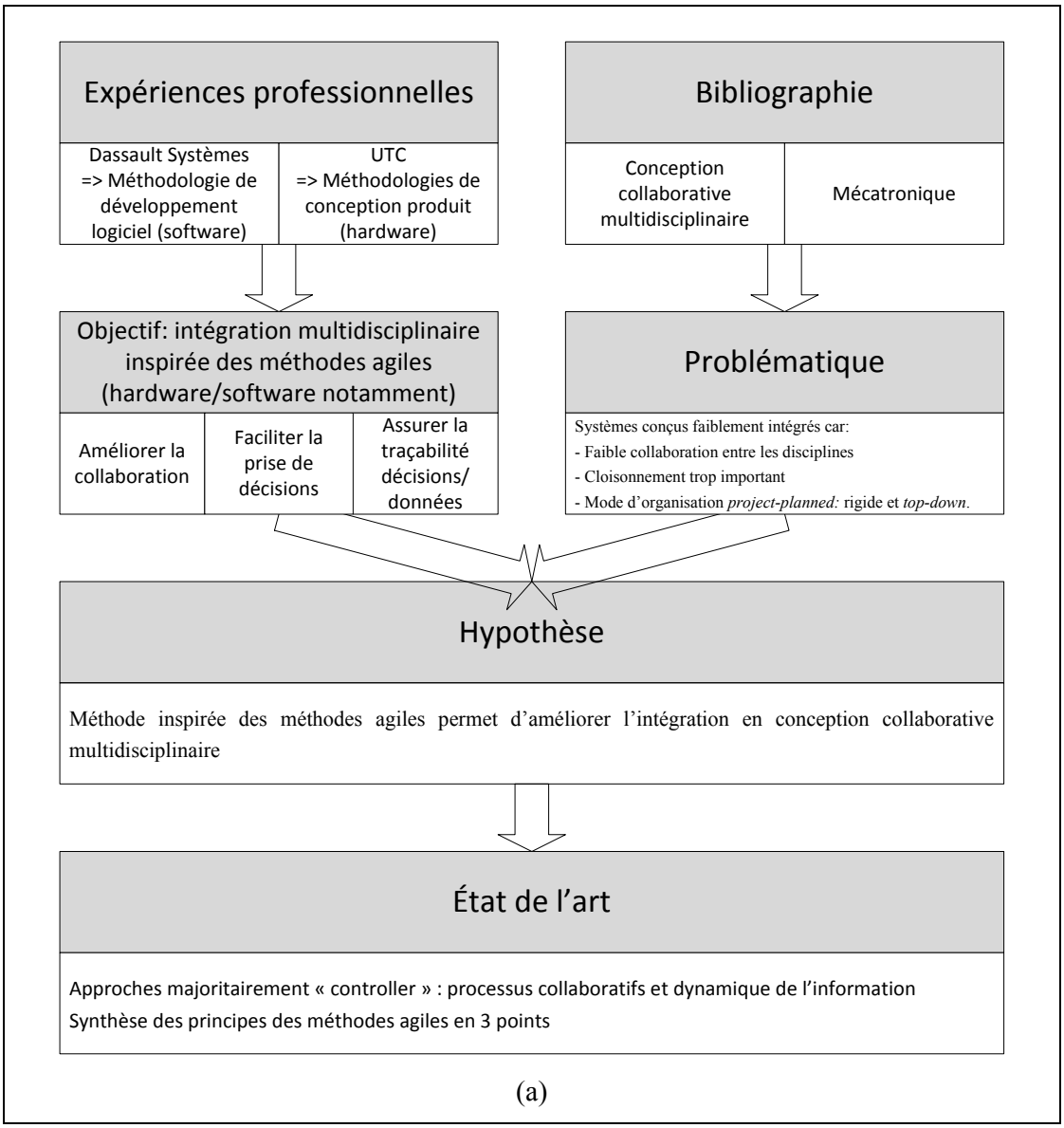
L'introduction d'une méthode inspirée des principes fondateurs des méthodes agiles permet d'améliorer l'intégration en conception collaborative multidisciplinaire car elle permet le décloisonnement des disciplines, elle assure la propagation de l'information de façon descendante (*top-down*) et ascendante (*bottom-up*) et elle fournit des indicateurs permettant d'assurer une traçabilité entre les décisions prises et les données modifiées.

Dans la section suivante, nous présentons la méthodologie générale mise en œuvre dans ces travaux.

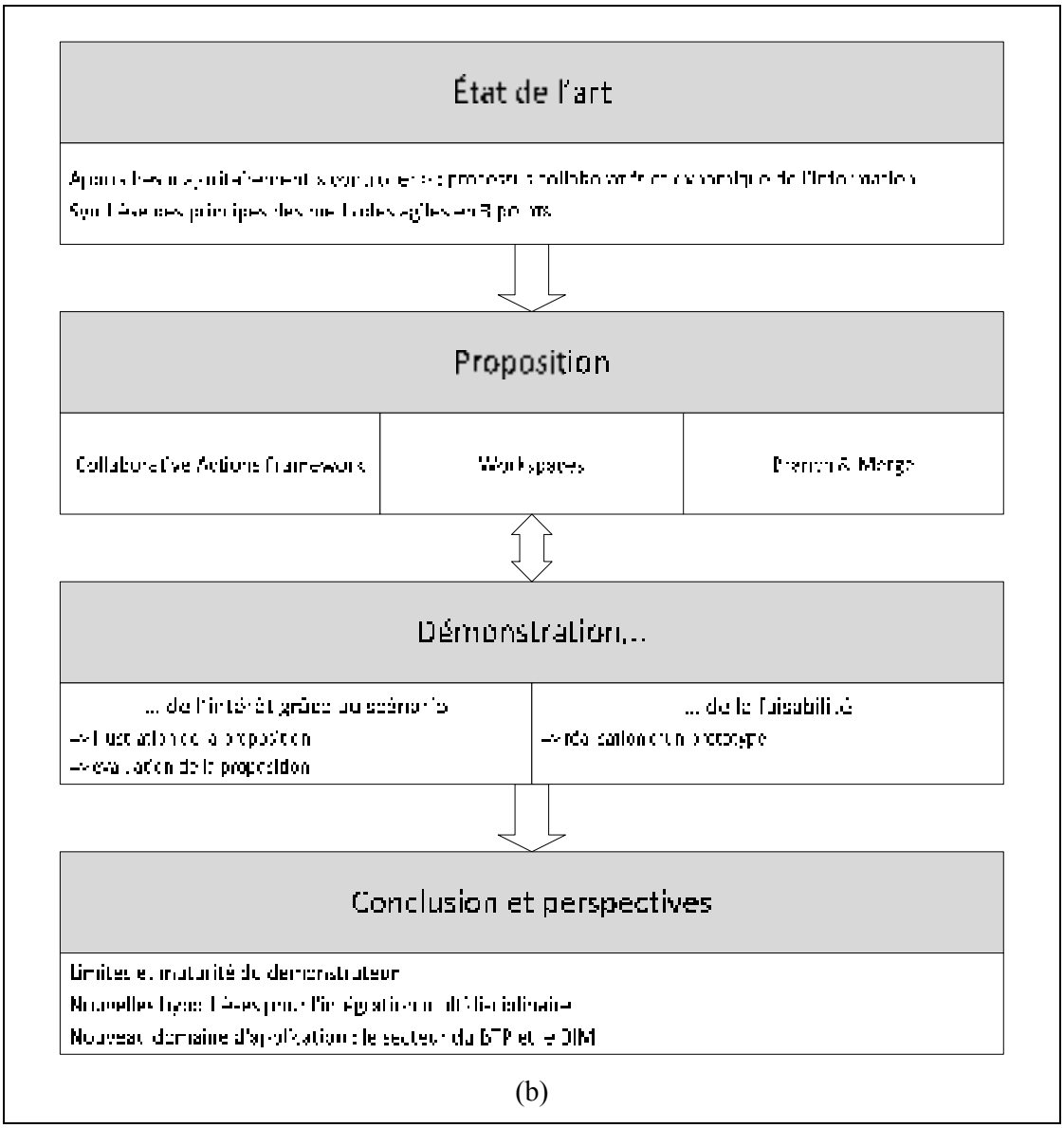
1.2.6 La méthodologie

Sur la base de l'hypothèse présentée dans la section précédente, un état de l'art structuré autour de deux facteurs de positionnement, détaillés dans le deuxième chapitre, a été proposé. Les principes fondateurs des méthodes agiles sont également détaillés dans cet état de l'art et certains de ces principes sont sélectionnés et synthétisés en trois points.

La Figure 1.22 (présentée sur deux pages) précise la méthodologie générale adoptée. On y retrouve les éléments présentés ci-avant (a), la proposition, la démonstration et les conclusions et perspectives (b). Chacun de ces éléments (proposition, démonstration, conclusion et perspectives) correspond respectivement aux chapitres trois, quatre et cinq de ce manuscrit. La proposition est formée de trois concepts complémentaires intitulés *Collaborative Actions Framework*, *Workspaces* et *Branch & Merge*. La démonstration est constituée quant à elle de deux parties. Le scénario mécatronique vise à démontrer l'intérêt de la proposition, alors que le prototype vise à démontrer la faisabilité de l'implémentation de cette dernière. La double flèche entre la partie proposition et la partie démonstration illustre le fait que ces dernières ont été réalisées de manière complémentaires, les démonstrateurs successifs venant enrichir la proposition et vice-versa. Enfin, la conclusion et les perspectives permettent de détailler les limites de l'approche et les travaux futurs qui devraient être menés suite à cette thèse.


Expériences professionnelles
Dassault Systèmes => Méthodologie de développement logiciel (software)
UTC => Méthodologies de conception produit (hardware)
Bibliographie
Conception collaborative multidisciplinaire
Mécatronique
Objectif: intégration multidisciplinaire inspirée des méthodes agiles (hardware/software notamment)
Améliorer la collaboration
Faciliter la prise de décisions
Assurer la traçabilité décisions/ données
Problématique
Systèmes conçus faiblement intégrés car:
- Faible collaboration entre les disciplines
- Cloisonnement trop important
- Mode d'organisation project-planned: rigide et top-down.
Hypothèse
Méthode inspirée des méthodes agiles permet d'améliorer l'intégration en conception collaborative multidisciplinaire
État de l'art
Approches majoritairement « controller » : processus collaboratifs et dynamique de l'information
Synthèse des principes des méthodes agiles en 3 points

(a)



(b)

Figure 1.22 Méthodologie générale

Ces travaux formalisent donc une sorte de convergence entre des intuitions basées sur des expériences professionnelles dans différents secteurs d'activité, sur le recueil informel d'informations du terrain, grâce notamment aux nombreux contacts dans les entreprises où nous avons la chance d'envoyer nos étudiants, aux études bibliographiques menées et aux échanges réalisés avec mes collègues et pairs. Cette convergence s'inscrit donc dans une sorte de recherche-action quelque peu similaire à celle décrite par Nicquevert (Nicquevert, 2012).

Dans ce chapitre, le contexte dans lequel ces travaux s'inscrivent a été présenté. Les différents types d'intégration, les processus de développement produit, sont ainsi autant de clés de lecture nécessaires à la bonne compréhension de notre contribution. La mécatronique, à la fois à l'origine de nos travaux et cadre applicatif choisi pour ces derniers, a ensuite été présentée. Une synthèse des cadres généraux, à savoir le PLM, l'ingénierie système, les CPSs et l'ALM, a également été proposée. C'est dans ce contexte que la problématique liée au cloisonnement des disciplines et au mode d'organisation a été formalisée. Sur cette base, des objectifs ont été précisés. Enfin, la méthodologie qui a été employée pour mener à bien ces travaux a été décrite, en rappelant les origines et les fondements de cette dernière. Dans la section suivante, les travaux contribuant à répondre à cette problématique sont exposés et classés suivant deux facteurs de positionnement, également introduits dans la section suivante.

CHAPITRE 2

État de l'art

Le premier chapitre de ce manuscrit visait à définir le contexte général de la conception collaborative multidisciplinaire et les cadres généraux dans lesquels s'inscrivent ces travaux. Il a également permis de préciser les contributions attendues pour ces travaux de thèse et la méthodologie de recherche déployée. En synthèse, le fait de mieux structurer l'organisation de la collaboration multidisciplinaire a été présenté comme l'objectif applicatif de ces travaux, notamment en s'inspirant de certains concepts issus des méthodes agiles.

Dans ce chapitre, différentes contributions venant contribuer à certains de nos objectifs sont présentées. Afin de pouvoir juger de la pertinence de ces contributions, un tableau de structuration à double entrée est proposé dans la première partie décrivant l'organisation de l'état de l'art. Ces deux « entrées » sont basées sur deux facteurs de positionnement. Après avoir proposé cette structure, les contributions jugées les plus pertinentes sont présentées et positionnées grâce à ces facteurs.

2.1 Organisation de l'état de l'art

La problématique de l'intégration en conception est un problème large et complexe qui a fait l'objet de très nombreux travaux. Ces travaux sont issus de différentes communautés présentées au chapitre précédent (ALM, PLM, Ingénierie Système, cybernétique ou mécatronique). Ils se focalisent sur l'intégration des activités d'ingénierie ou des différentes disciplines, d'un point de vue conceptuel ou appliqué sur des cas concrets comme pour la mécatronique ou la cybernétique. Ces travaux s'intéressent aux méthodes organisationnelles ou à la structuration des données issues de chacune des disciplines. Cette grande diversité nécessite la mise en œuvre d'une grille d'analyse permettant de véritablement juger de la pertinence des contributions relativement aux objectifs de cette thèse. Cette section propose deux facteurs de positionnement qui permettent de juger de cette pertinence. Le premier facteur s'intéresse à la caractérisation du type de contribution (modèle de données, interface utilisateur ou dynamique des échanges) basée sur une analogie avec un patron d'architecture intitulé *Model-View-Controller*. Le second facteur, plus conventionnel, s'intéresse quant à lui à définir la portée de la contribution via le niveau de gestion et de prise de décision (stratégique, tactique, opérationnel). Ces deux facteurs sont décrits dans les sections suivantes puis positionnés par rapport aux objectifs de nos travaux décrits dans le chapitre précédent. Ils sont enfin croisés pour définir de manière précise le positionnement adopté.

2.1.1 1er facteur de positionnement : le type de contribution

Le premier facteur de positionnement permettant de structurer l'état de l'art vise à déterminer le type de contribution. Ce facteur s'inspire, conceptuellement parlant, d'un patron d'architecture informatique (Conte et al., 2001) intitulé *Model-View-Controller* (MVC). Ce dernier est introduit dans son contexte d'origine, le développement informatique, avant d'être décliné dans le domaine cible, la conception multidisciplinaire. Le fonctionnement et l'usage qui est généralement fait de ce type de patron n'est pas primordial pour nos travaux, seul son « esprit » et son facteur structurant sont utilisés ici.

Les patrons informatiques d'architecture, parfois appelés modèles ou patterns d'architecture, sont des solutions génériques à des types de problèmes fréquemment rencontrés (Buschmann

et al., 2007). Ces solutions génériques servent « de source d'inspiration lors de la conception de l'architecture d'un système ou d'un logiciel informatique en sous-éléments plus simples »[8]. Le patron modèle-vue-contrôleur (MVC) est l'un des patron d'architecture les plus connus pour l'élaboration d'interfaces graphiques utilisateur (Bragge, 2013). Il est destiné « à répondre aux besoins des applications interactives en séparant les problématiques liées aux différents composants au sein de leur architecture respective »[9]. Ce patron, comme son nom l'indique, s'appuie sur trois éléments essentiels. Le premier, le modèle, a pour rôle la gestion de l'intégrité des données, sans connaître a priori l'usage qui sera fait de ces données. Fowler précise que « the model is a non-visual object containing all the data and behaviour other than that used for its presentation to the user » (Fowler, 2002). Au contraire, le second élément qu'est la vue, doit exclusivement s'attacher à présenter les informations les plus pertinentes de la façon la plus appropriée à l'utilisateur. La vue doit donc, dans une vision assez simpliste, s'attacher à l'ergonomie de l'application : « Views hand over user actions such as mouse movement, keyboard strokes or touch gestures to the controller » (Fowler, 2002). Entre ces deux éléments, le contrôleur reçoit les notifications de la vue ou du modèle et réagit de façon appropriée en modifiant les données ou en mettant à jour les vues. Il assure donc le côté dynamique de l'application et peut être comparé à un chef d'orchestre, permettant de décorréler les problématiques d'évolution du modèle de données et de l'interface utilisateur. Le pattern MVC a donné naissance à de nombreux autres patterns dérivés de cette logique (Bragge, 2013). Mais étant donné que ces derniers n'ont pour principale vocation que d'apporter des détails quant aux subtilités d'implémentation, il ne semble pas nécessaire de plus détailler ces patterns.

L'introduction de ce pattern a pour but de faire une analogie avec la nature des contributions rencontrées dans le domaine de la conception multidisciplinaire, et même dans un cadre plus large. Le 'M' de *model* renvoie ainsi aux contributions de type « modèles de données ». Elles accordent généralement une large importance à la manière dont les informations liées aux produits conçus doivent être stockées. Ces modèles, traditionnellement présentés sous la forme d'ontologies ou de diagramme de classes UML[10], présentent une vue assez statique de

[8] http://fr.wikipedia.org/wiki/Patron_d'architecture
[9] http://fr.wikipedia.org/wiki/Modèle-Vue-Contrôleur
[10] Unified Modeling Language - http://www.omg.org/

l'information. Dans le cadre de systèmes d'information, le rôle joué par chacun des logiciels mis en œuvre peut être caractérisé grâce à ce type de modèle de données.

Le 'V' de *view* renvoie quant à lui aux interfaces utilisateur, mais également aux dispositifs matériels permettant d'interagir avec un logiciel ou un système d'information. L'ergonomie y joue donc un rôle prédominant et l'interaction avec l'humain est au cœur des préoccupations. La façon de présenter l'information, les moyens d'interaction avec cette information en fonction des profils des utilisateurs et des cas d'usage sont autant de leviers permettant de rendre les systèmes d'information plus pertinents et plus efficaces.

Enfin, le 'C' de *controller* fait référence au caractère dynamique de l'information. Il traite de la façon dont les informations sont échangées entre les acteurs, entre les briques d'un système d'information, etc. Il peut donc décrire, en fonction des situations, quels impacts une action utilisateur va avoir sur la répartition des informations dans le modèle de données, quels échanges doivent avoir lieu entre les différents acteurs, supportés par des outils collaboratifs ou pas, en fonction des étapes du processus de conception, etc. Cet aspect dynamique de l'information est assez difficile à formaliser et à représenter sous la forme de modèles. À un niveau assez détaillé ou « micro », les diagrammes comportementaux UML (diagrammes d'activité ou diagramme des cas d'utilisation) permettent de décrire de façon précise les comportements attendus d'un système dans des cas assez précis. Au contraire, à un niveau beaucoup plus général ou « macro », des modèles de processus sont parfois proposés pour décrire des enchaînements de tâches au sein d'un métier ou entre différentes organisations.

En reprenant les objectifs de nos travaux présentés dans le chapitre précédent, on note que les trois types de contribution *model-view-controller* peuvent contribuer à atteindre le premier des sous objectifs de nos travaux (« Améliorer la collaboration entre les différentes disciplines en réduisant le cloisonnement existant entre les acteurs »). Le décloisonnement des disciplines peut en effet être abordé par le biais des modèles de données, par la dynamique des échanges d'information mais également par la définition d'interfaces ou de dispositifs dédiés. Le second (« Rendre la prise de décisions dans le cadre d'un projet de conception plus aisée grâce à la remontée plus fréquente et plus régulière d'informations opérationnelles précises ») et le troisième (« Permettre la traçabilité entre les évolutions apportées aux données de définition du système conçu et les décisions prises tout au long du projet de développement »)

sous objectifs semblent quant à eux s'inscrire à la frontière des contributions de type *model* et *controller*.

Après avoir présenté le pattern MVC et décrit en quoi ce dernier pouvait être utilisé comme un premier élément de structuration pour les différentes contributions répondant partiellement à la problématique à l'origine de ces travaux, un second facteur de positionnement relatif au niveau de gestion et de prise de décision est présenté dans la section suivante.

2.1.2 2ème facteur de positionnement : le niveau de gestion et de prise de décision

Le deuxième facteur de positionnement utilisé pour classer et différencier les contributions est beaucoup utilisé dans la littérature, bien que rarement décrit formellement (Kéradec, 2012). Il fait référence au niveau de gestion et de prise de décision de l'entreprise. Ce niveau peut être qualifié de stratégique, tactique ou opérationnel. Il se distingue par la typologie des personnes prenant ces décisions et leurs impacts sur l'entreprise. L'impact en lui-même peut être évalué, entre autres, par sa durée, mais aussi par le nombre et le type de personnes impliquées. Ainsi, les décisions stratégiques sont destinées à engager l'entreprise à long terme (plusieurs années) et sont généralement réalisées par les dirigeants et administrateurs de l'entreprise. Ces décisions impactent de près ou de loin l'ensemble des employés, voire l'ensemble de l'écosystème de l'entreprise. Les décisions tactiques engagent quant à elles la société à moyen terme) et sont généralement réalisées par les niveaux hiérarchiques relativement élevés. Elles concernent des secteurs de l'entreprise de sorte que des services complets peuvent avoir à assumer ce type de décision. Enfin, les décisions opérationnelles sont prises, dans un contexte de conception de système, par des designers ou par les membres de l'équipe projet. Elles ne concernent généralement que les membres du projet et ont un impact limité dans le temps.

Ce niveau de gestion peut également se décliner à l'échelle d'un projet de conception. Girard et al. exemplifient ainsi le niveau stratégique comme le niveau auquel les imbrications entre les différents projets de conception sont analysées, le niveau tactique comme le niveau où intervient le découpage d'un gros projet en sous projet avec la répartition des actions principales, et le niveau opérationnel comme le niveau auquel l'ensemble des activités nécessaires pour satisfaire les objectifs des sous projets sont planifiées et réalisées (Girard and Doumeingts, 2004).

Ce deuxième facteur à trois niveaux, stratégique-tactique-opérationnel, a pour principal intérêt de proposer une sorte d'échelle d'impact des décisions prises lors de projet de conception. Mais il nous permet également de venir spécifier les directions dans lesquelles circulent l'information et la portée de celle-ci. En effet, une approche *top-down* ou approche descendante correspond à une logique de prise de décisions aux niveaux stratégique ou tactique mises en œuvre au niveau opérationnel. En ce sens, les méthodes d'organisation de type *project-planned* s'inscrivent donc dans une logique *top-down*. Au contraire, le sens *bottom-up* ou approche ascendante part du terrain opérationnel pour remonter aux niveaux tactique ou stratégique.

En reprenant les trois sous objectifs de nos travaux présentés dans le chapitre précédent, on note que ces derniers visent à faire circuler l'information dans les sens ascendant et descendant. Le sens privilégié pour le moment étant le sens *top-down*, nos travaux se concentreront sur la remontée d'informations opérationnelles vers les niveaux tactique et opérationnel.

Cette première section relative à la structuration de l'état de l'art a permis de présenter deux facteurs de positionnement qui vont permettre de qualifier les différentes contributions identifiées dans la section suivante. Lorsque les approches de type *controller* seront présentées, des précisions supplémentaires seront apportées afin de déterminer si ces dernières sont plutôt dans une logique ascendante ou une logique descendante.

2.2 Positionnement et état de l'art associé

Dans les sections précédentes, deux facteurs de positionnement ont été présentés afin de permettre la discrimination des différentes approches et de déterminer lesquelles contribuent effectivement à l'intégration multidisciplinaire en conception et au décloisonnement des acteurs des différentes disciplines. En croisant ces deux facteurs, le Tableau 2.1 peut être proposé comme support de structuration et de positionnement des contributions[11].

	Model	Controller	View
Strategic			
Tactical			
Operational			

Tableau 2.1 Tableau permettant de synthétiser l'état de l'art

Comme nous le verrons par la suite, la contribution présentée dans ce manuscrit s'inscrit majoritairement à l'intersection de la colonne c*ontroller* et des lignes *tactical* et *operational*. Mais afin de bien comprendre les apports des approches de type *model*, *view* et *controller*, les sections suivantes détaillent quelques contributions pour chacune des colonnes.

2.2.1 Les approches *model*

Dans le cadre général du PLM, de nombreux modèles de données produit ont été proposés afin de devenir, après implémentation, la colonne vertébrale du système d'information PDM. Souvent, les modèles de données produit sont présentés comme une description de l'arborescence structurelle (Bill of Material, BOM) du produit. En réalité, ils ne se réduisent pas à cette description physique (géométrique, structurelle, ...) et ils ont pour fonction de rassembler toutes les informations pertinentes concernant le produit. Ces informations supplémentaires peuvent être :

[11] Pour des raisons de lisibilité et de cohérence avec les figures proposées dans les chapitres suivants, les colonnes sont organisées dans l'ordre *model-controller-view* contrairement au nom du patron d'architecture original.

- la description fonctionnelle ou les fonctions que doit pouvoir remplir le produit durant les différentes phases de son cycle de vie,
- la description comportementale ou comment le produit va fonctionner pour assurer ces différentes fonctions.

Ce triptyque est à la base des approches de modélisation produit *Function Behaviour Structure* (FBS) (Gero and Kannengiesser, 2004) souvent utilisé pour la modélisation et la structuration des données d'un produit mécanique. La Figure 2.1 représente l'espace d'interaction entre ces différents types d'information produit.

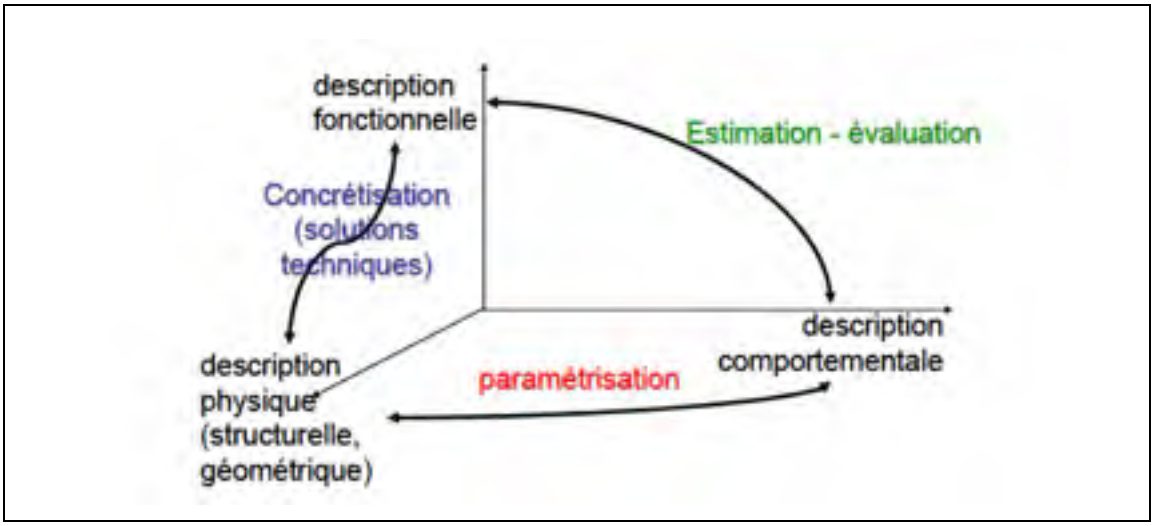


Figure 2.1 Les différentes vues dans un modèle de produit (Belkadi, 2006)

Dans cette logique de représentation et de structuration des informations produit, le *Core Product Model* (CPM) et le modèle Produit, Processus et Organisation (PPO) ont été sélectionnés parmi de nombreux autres modèles de données produit pour leur reconnaissance dans la communauté et leur complémentarité.

2.2.1.1 Core Product Model (CPM)

Le National Institute of Standards and Technology (NIST) propose, par l'intermédiaire du CPM, un support générique, simple et extensible pour représenter les informations liées à la gestion du cycle de vie du produit indépendamment des applications informatiques métier qui vont utiliser ces informations (Fenves et al., 2008; Sudarsan et al., 2005). Basé sur le langage graphique UML, ce modèle représenté sur la Figure 2.2 fournit une représentation multi-vues à travers la notion d'artefact, permettant entre autre une représentation de produits

manufacturiers. Outre les concepts géométriques, cette représentation inclut les notions de forme, de fonction, de comportement, de matériaux, de décompositions physiques et fonctionnelles, de mise en correspondance entre la fonction et la forme, ainsi que plusieurs relations entre ces différents concepts.

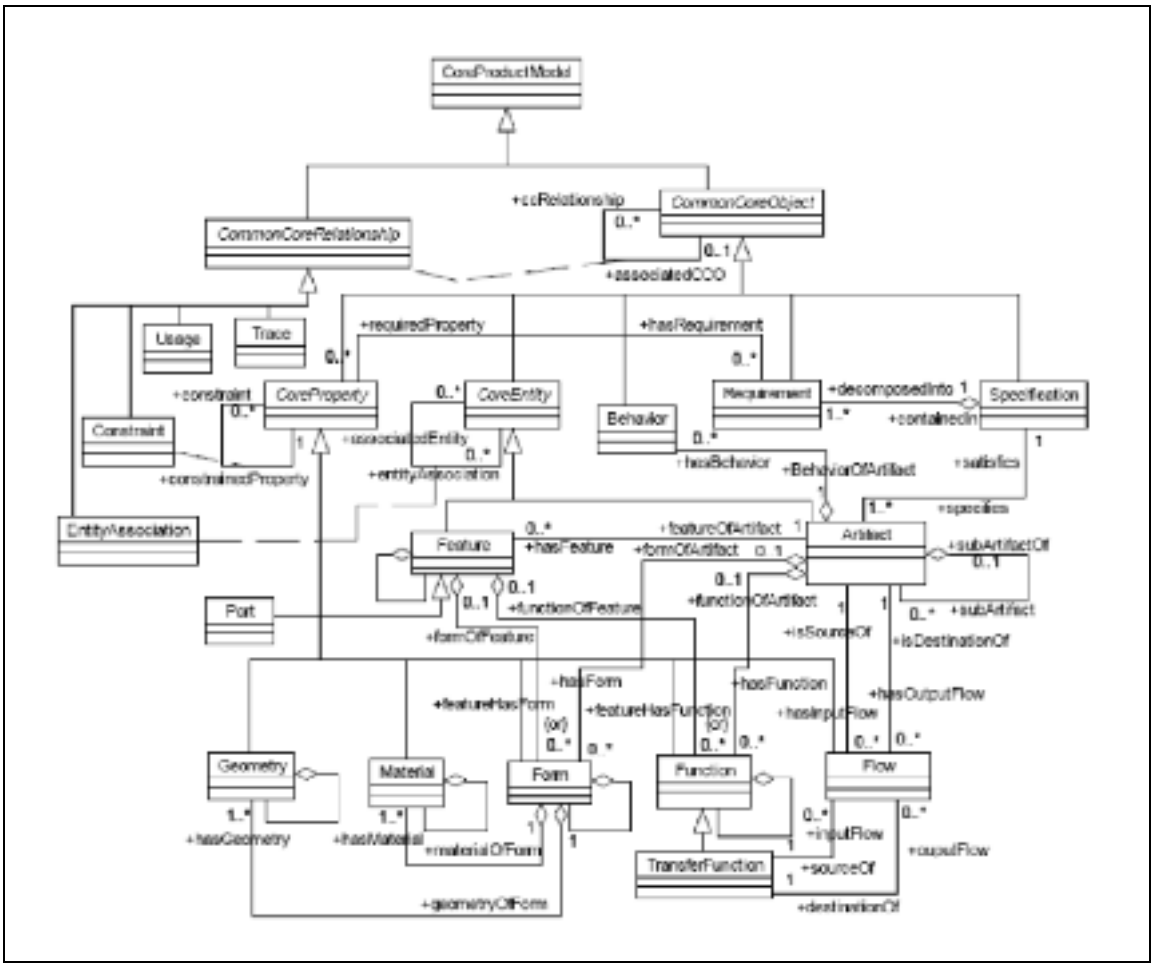


Figure 2.2 Le modèle CPM (*Core Product Model*) (Sudarsan et al., 2005)

Ce modèle s'appuie principalement sur la classe « *Artifact* » qui représente une entité spécifique dans un produit, que ce soit un composant, une partie, un sous-assemblage ou un assemblage. Toutes ces entités peuvent être représentées et interconnectées à travers les liens « sous-artefact » et « sous-artefact-de » modélisés par une relation réflexive sur la classe « *Artifact* » (relation « *subArtifacts* » / « *subArtifactOf* »). Il est donc possible, grâce à la généricité de ce type de modèle, de stocker et de gérer l'ensemble des informations concernant un système, et ce quelle que soit l'origine disciplinaire de ces informations. Mais ce modèle très générique demande à être précisé/spécialisé pour permettre l'usage effectif des

informations stockées. De plus, bien que la collaboration multidisciplinaire gagnerait à être basée sur une plateforme unique ou sur un système d'information fédéré autour d'un modèle de données général comme celui-ci, le verrou identifié dans la section précédente s'attache principalement au mode d'organisation et aux outils supportant ce mode d'organisation, et non à la gestion des données techniques.

Ce modèle de données produit CPM a été choisi comme illustration du type *model* pour différentes raisons. La première est qu'il reste à ce jour un des modèles standards dans la communauté, dans la continuité des approches FBS. De plus, bien que très générique, il s'est progressivement enrichi de diverses extensions permettant de répondre plus spécifiquement à certaines problématiques. On peut ainsi noter les extensions *Open Assembly Model* (OAM) dédiée à l'intégration des différentes informations utilisées dans le processus d'assemblage du produit final (Rachuri et al., 2005), *Reverse Engineering* (RE) dédiée à la gestion des données issues d'opérations de reverse engineering (Durupt et al., 2013), ou encore « Core Product Model extension for environmental evaluation » (CPMe^3) pour capturer et réutiliser les informations de type *Eco-Design* (Kozemjakin da Silva et al., 2013). Mais pour reprendre les formulations de Abramovici et Bellalouna, ce type d'approche ne se focalise que sur les « Data-Based Problems » en laissant de côté les « Process-Based Problems » (Abramovici and Bellalouna, 2007). Dans la littérature, très peu de modèles se sont attachés à relier la dimension modèle de données produit à la dimension organisationnelle et décisionnelle de la conception. C'est en revanche le cas du modèle *product–process–organization* (PPO), présenté dans la section suivante.

2.2.1.2 Produit, Processus et Organisation (PPO)

Le projet IPPOP (pour Intégration Produit, Processus, Organisation pour l'amélioration des Performances en ingénierie) a eu pour objectif le développement d'un système collaboratif pour supporter et partager des informations entre les acteurs tout au long du cycle de vie d'un projet avec une intégration, d'une part, des dimensions Produit, Processus et Organisation, et d'autre part, d'extensions des logiciels de Conception / Fabrication Assistées par Ordinateur (CFAO) et de Gestion de Données Techniques (GDT) existants en prenant en compte les aspects technologiques liés à la conception. La Figure 2.3 synthétise les échanges d'informations entre la dimension Produit et la dimension Organisation via les Processus.

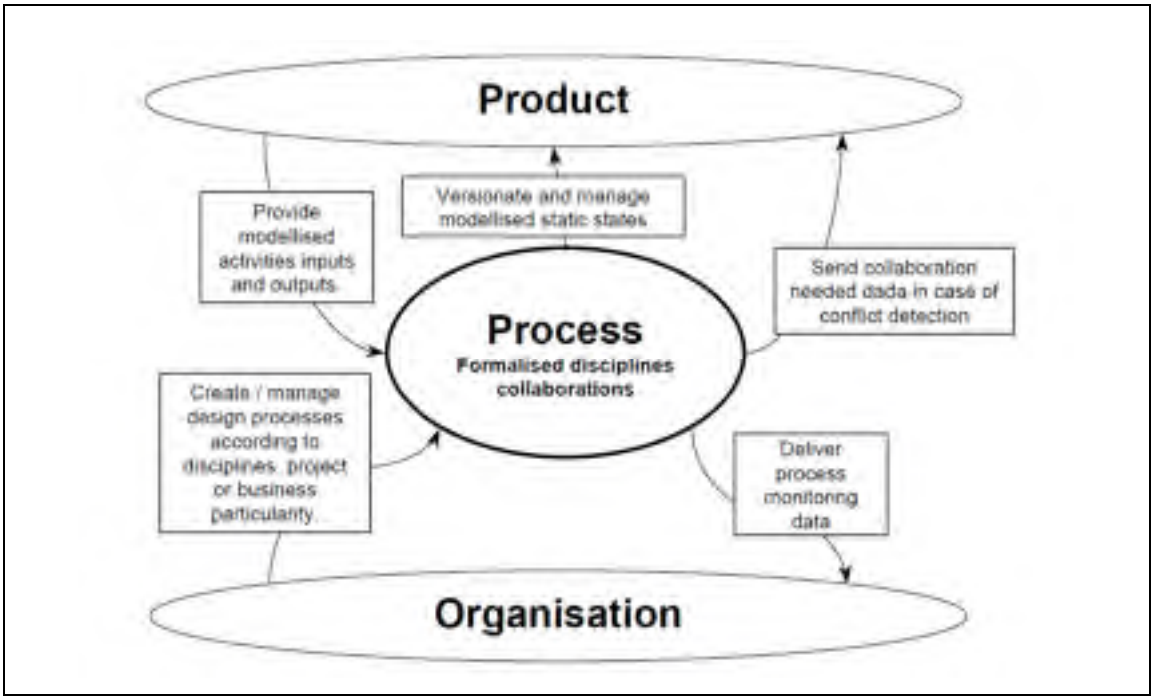


Figure 2.3 Les vues Projet Processus et Organisation du projet IPPOP (Nowak et al., 2004)

La plateforme issue du projet IPPOP repose sur le modèle Produit, Processus et Organisation (PPO) comportant une partie descriptive des informations projet en termes de processus et d'organisation des ressources, ainsi qu'une partie décrivant le modèle produit. La Figure 2.4 illustre que l'organisation et la prise de décision au sein d'un projet de conception s'appuient sur deux *frameworks* nommés « Design_Framework » et « Decision_Framework » (Noël and Roucoules, 2008).

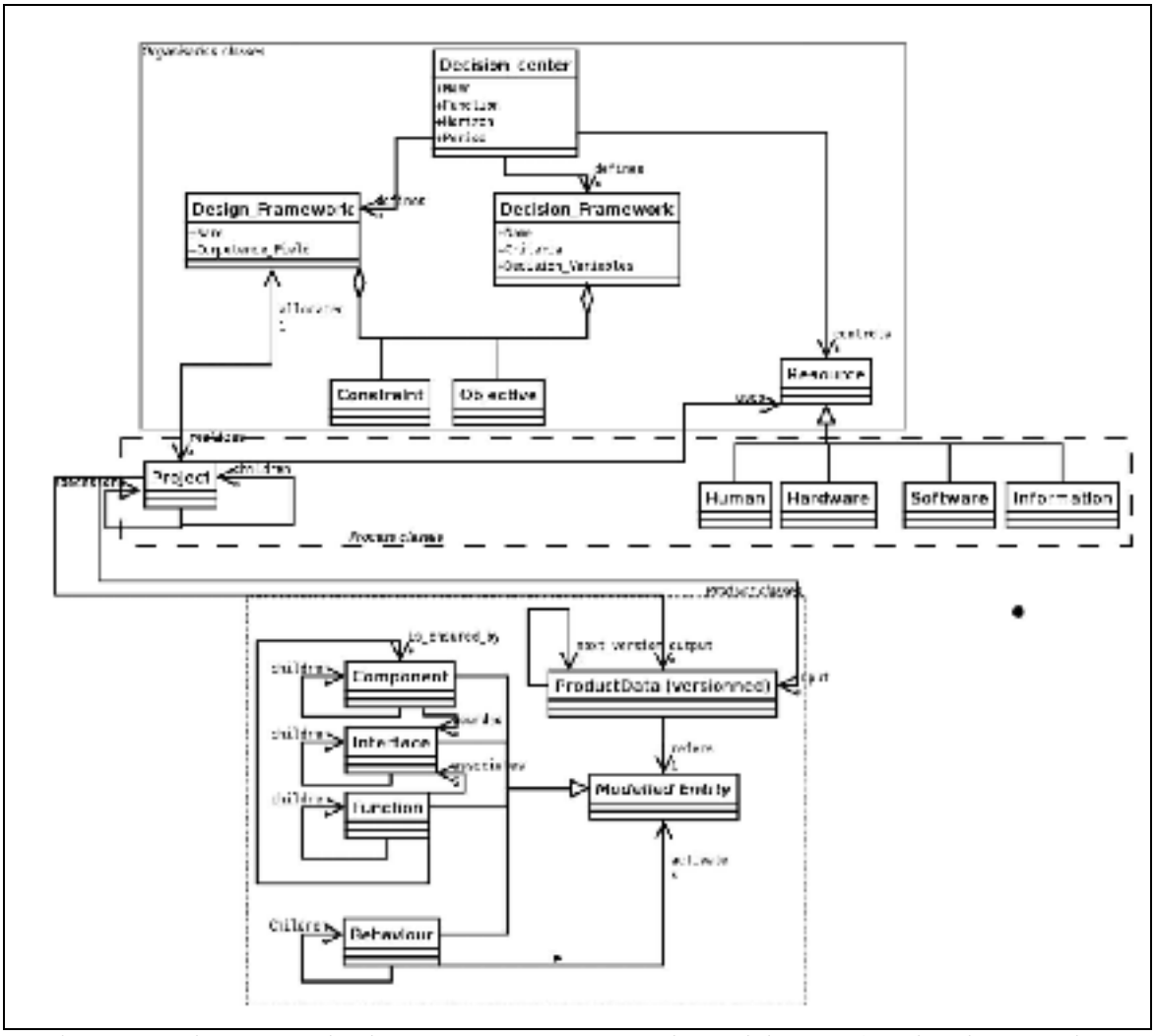


Figure 2.4 Diagramme de classes UML représentant le modèle PPO (Noël and Roucoules, 2008)

Tout comme le CPM, le modèle PPO reste très générique et ne permet pas réellement de montrer le côté dynamique de l'information portée et véhiculée par une plateforme implémentant ce modèle. Il initie la prise en considération de ces informations organisationnelles au même niveau que les informations liées au produit, mais sans apporter le même niveau de précision.

2.2.1.3 Intégration par les outils d'édition grâce aux standards

Avant de synthétiser et de positionner les approches de type *model* présentées dans cette section, il semble nécessaire de présenter un dernier exemple qui permet de préciser une autre facette du positionnement de nos travaux. L'ensemble des contributions présentées jusqu'ici

s'intéresse à l'intégration multidisciplinaire via des solutions de gestion de données techniques. Mais d'autres contributions visant l'intégration des activités d'ingénierie deux à deux méritent d'être mentionnées. Par exemple, dans le cadre de l'intégration des métiers mécanique et électrique/électronique, des travaux ont été menés par Chen et al. (Chen and Schaefer, 2007; Chen et al., 2009). Basée sur l'utilisation des formats standards IGES, STEP (AP203 et AP210) d'une part et Gerber, IDF, Drawing Exchange Format (DXF) d'autre part, ils tentent de définir une méthodologie de collaboration entre les différents acteurs (Figure 2.5) (Chen and Schaefer, 2007). Cette méthodologie est assez proche de celle proposée par Ahn et al. (Ahn et al., 2004) qui propose également d'utiliser des formats neutres pour échanger les contraintes de conception principales d'un métier à un autre. Cette méthodologie s'inscrit donc dans une approche dite fédérative, où plusieurs modèles ou méta-modèles interagissent directement entre eux (Rio, 2012).

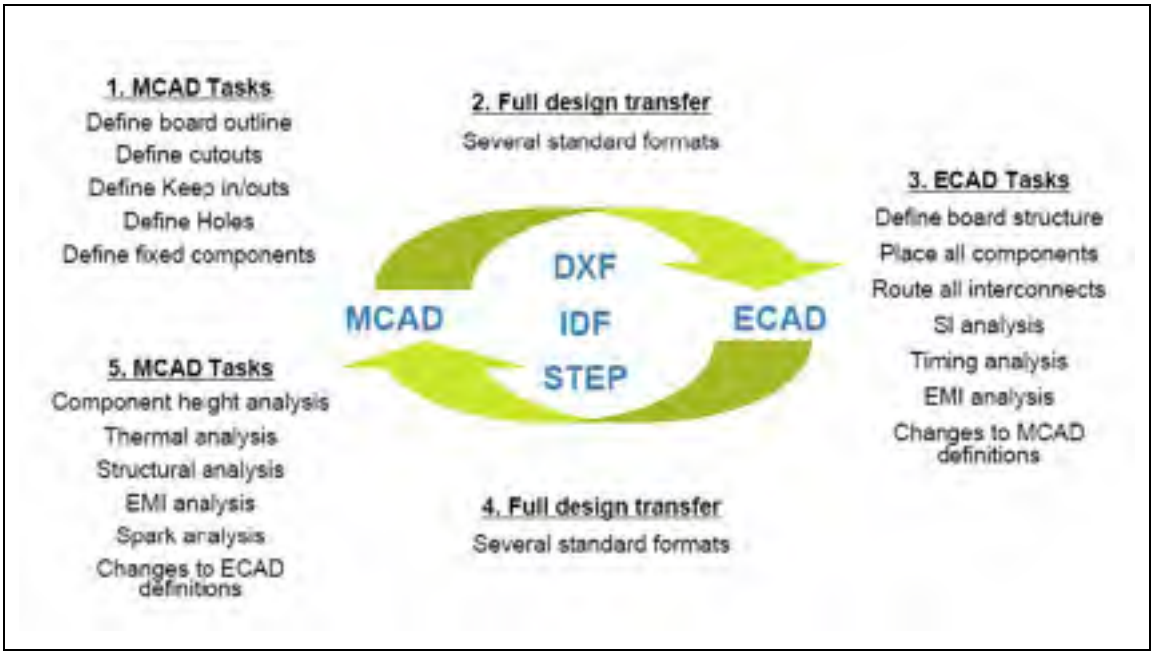


Figure 2.5 Procédure d'échange d'informations entre les métiers mécanique et électrique/électronique (Chen and Schaefer, 2007)

Notre approche se distingue de ce type de contribution par le fait qu'elle n'est pas spécifique à un couple de disciplines. L'intégration par les outils d'édition, bien qu'étant un véritable levier de collaboration et de décloisonnement des métiers, ne répond donc pas de façon générale à la problématique posée.

2.2.1.4 Synthèse des approches *model*

Le Tableau 2.2 positionne ces approches *model* dans le tableau reprenant les facteurs de positionnement *model-view-controller* et *strategic-tactical-operational*. Le modèle CPM, se positionne ainsi exclusivement dans la colonne *model*, gérant principalement des informations produit de type stratégique et nécessitant de nombreuses spécialisations via des extensions pour pouvoir prétendre à un apport de type opérationnel. Le modèle PPO est également principalement dans la colonne *model*, mais il se positionne également « légèrement » à l'intersection de la colonne *controller* et de la ligne *strategic*, pour la prise en considération de certaines informations de type organisationnelles, sans pour autant apporter une solution opérationnelle dans ce domaine. Enfin, les approches visant l'intégration par les standards au niveau des outils d'édition se positionnent très majoritairement à l'intersection de la colonne *model* et de la ligne *operational* car ils traitent avant tout des informations échangées entre les concepteurs.

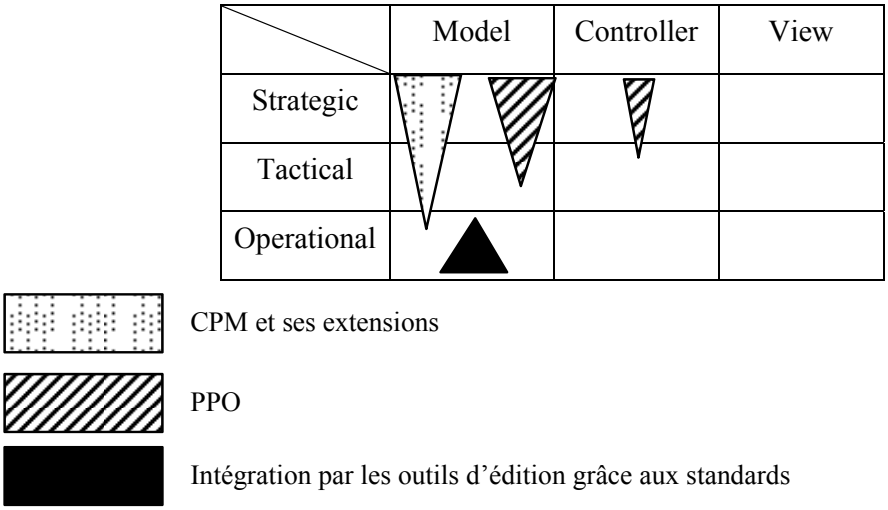


Tableau 2.2 Synthèse des approches *model*

2.2.2 Les approches *view*

Bien que ce type d'approche soit relativement moins présent dans le cadre général du PLM, quelques travaux notables existent quant aux approches de type *view*. Ces derniers correspondent aux initiatives qui permettent d'améliorer la conception collaborative multidisciplinaire via des interfaces ou des systèmes permettant l'interaction entre les concepteurs ou entre concepteurs et systèmes informatiques supportant cette collaboration. Des dispositifs physiques tels que des tables tactiles interactives (Jones et al., 2011), ou encore l'amélioration de l'ergonomie de certaines fonctionnalités proposées par les logiciels PDM (De Pinel et al., 2013) sont donc des exemples de travaux s'inscrivant dans ce type d'approche.

2.2.2.1 Environnements supports à l'ingénierie collaborative synchrone à distance

La distribution géographique des acteurs ainsi que leur multiplicité sont autant de freins à de véritables échanges susceptibles de promouvoir l'intégration des expertises multidisciplinaires. Un environnement de travail efficient pour collaborer devient alors essentiel pour améliorer la compréhension partagée entre les différents acteurs des différentes équipes, partager des informations sur le produit durant les différentes étapes de conception et maintenir la cohérence des travaux durant leurs évolutions, notamment synchrones (Vu Thi, 2012). Vu Thi rappelle que deux catégories de support au travail collaboratif distant existent. La première s'inscrit dans le support au travail collaboratif synchrone et est composée des différents outils tels que la visioconférence, les réunions audio avec partage d'écran, etc. La seconde catégorie traite des activités asynchrones avec notamment l'échange de données structurées provenant d'outils métiers spécifiques. Ses travaux portent donc sur l'amélioration de la « capacité de travail synchrone à distance avec des outils support à la communication technique », de « synchronisation des versions produites par des concepteurs », en identifiant des conflits au plus tôt et d'« assurer la convergence des travaux asynchrones et ainsi supporter la cohérence des données produites tout au long de son développement » (Vu Thi, 2012).

Ces travaux s'inscrivent dans la continuité d'autres travaux s'inscrivant plus spécifiquement dans la colonne *model* et visant à identifier les différences et les potentiels conflits pouvant exister entre des modèles liés (Sadeghi et al., 2009). Le but du système implémenté, nommé

GAM-PPO, est alors de maintenir la cohérence globale en minimisant les incohérences entre les différents modèles révélées lors du processus de mise en commun, dénommé « synchronisation » (Sadeghi et al., 2010).

2.2.2.2 Synthèse des approches *view*

L'exemple présenté ci-avant illustre donc comment des dispositifs s'attachant aux interactions entre les acteurs de la conception collaborative peuvent avoir un impact sur le niveau de collaboration, et comment certaines approches peuvent s'inscrire dans différentes colonnes de notre tableau de synthèse. Ici, les travaux de Vu Thi s'inscrivent donc très majoritairement dans la colonne *view*, car la contribution principale reste dans le domaine des interactions entre les différents concepteurs et les dispositifs innovants pouvant être utilisés lors de la conception. Mais ce genre de contribution peut également être considéré comme interagissant avec les colonnes *model* et *controller*. Pour la colonne *model*, les modifications apportées grâce aux systèmes d'interaction ne doivent pas mettre en péril la cohérence des informations portées par le modèle de données. Pour la colonne *controller*, l'introduction de nouvelles méthodes d'interaction entre les acteurs peut remettre en cause les méthodes de collaboration. Les processus organisationnels peuvent alors s'en trouver modifiés.

Le Tableau 2.3 illustre ce positionnement majoritaire dans la colonne *view*, à un niveau avant tout opérationnel[12].

	Model	View	Controller
Strategic			
Tactical			
Operational			

Env. supports à l'ingénierie collaborative synchrone à distance

Tableau 2.3 Synthèse des approches *view*

[12] Ce tableau est exceptionnellement organisé avec les colonnes *model-view-controller* pour des raisons de lisibilité.

2.2.3 Les approches *controller*

Les approches de type *controller* traitent de la dynamique de l'information, de la façon dont ces informations sont échangées entre les acteurs ou entre les différents éléments du système d'information, à quelle fréquence, etc. Dans le cadre de la conception collaborative, différentes contributions scientifiques et fonctionnalités offertes par les systèmes PDM ainsi que les méthodes dites agiles ont été identifiées comme pertinentes en regard de notre problématique. Les trois sections ci-après visent à présenter ces contributions.

2.2.3.1 Modèles de processus

Les contributions scientifiques dans le domaine des approches de type *controller* se présentent généralement sous la forme de modèles de processus. Ainsi, à une échelle stratégique ou tactique, lorsque l'organisation d'un projet de conception doit être décrite, l'enchaînement des activités et les livrables attendus à l'issue de chaque phase (Pahl et al., 2003) sont les éléments généralement représentés. À une échelle plus opérationnelle, la granularité et l'enchaînement des activités à réaliser pour atteindre un objectif métier (Troussier et al., 2010) ou gérer une activité précise (Jarratt et al., 2010) peuvent être décrits, mais sans pour autant mettre en évidence la complexité des tâches et le nombre d'échanges nécessaires pour assurer la collaboration. C'est là toute l'ambiguïté de ce type d'approche dans une démarche scientifique : la généralisation nécessaire finit par occulter la richesse et par normaliser les échanges entre les différents acteurs.

Afin d'illustrer ces propos, l'exemple de la gestion des modifications d'ingénierie, mis en œuvre lorsqu'une modification technique est nécessaire, est détaillée dans ce paragraphe. Une modification technique est définie par Jarratt et al. comme « an alteration made to parts, drawings or software that have already been released during the product design process. The change can be of any size or type; the change can involve any number of people and take any length of time » (Jarratt et al., 2010). Les auteurs fournissent également un modèle générique (Figure 2.6) pour illustrer les différentes étapes d'un processus de modification d'ingénierie. Tout d'abord, une demande est faite avec la justification de la priorité, le type et les impacts éventuels du changement. La deuxième étape est l'examen des solutions possibles. Les impacts et les risques associés à ces solutions sont évalués dans la troisième étape. La

quatrième étape consiste en l'approbation d'une solution pour atteindre la modification d'ingénierie souhaitée. Habituellement, un comité est chargé de prendre ce type de décision, mais certains raccourcis peuvent être mis en œuvre en fonction du niveau d'urgence. Ce comité doit représenter les différentes parties prenantes, telles que la conception, la fabrication, la commercialisation et autres équipes. La cinquième étape est la mise en œuvre de la modification, qui peut se produire immédiatement ou être introduite progressivement. Enfin, la modification d'ingénierie doit être examinée pour vérifier si les objectifs initiaux ont été satisfaits par la solution.

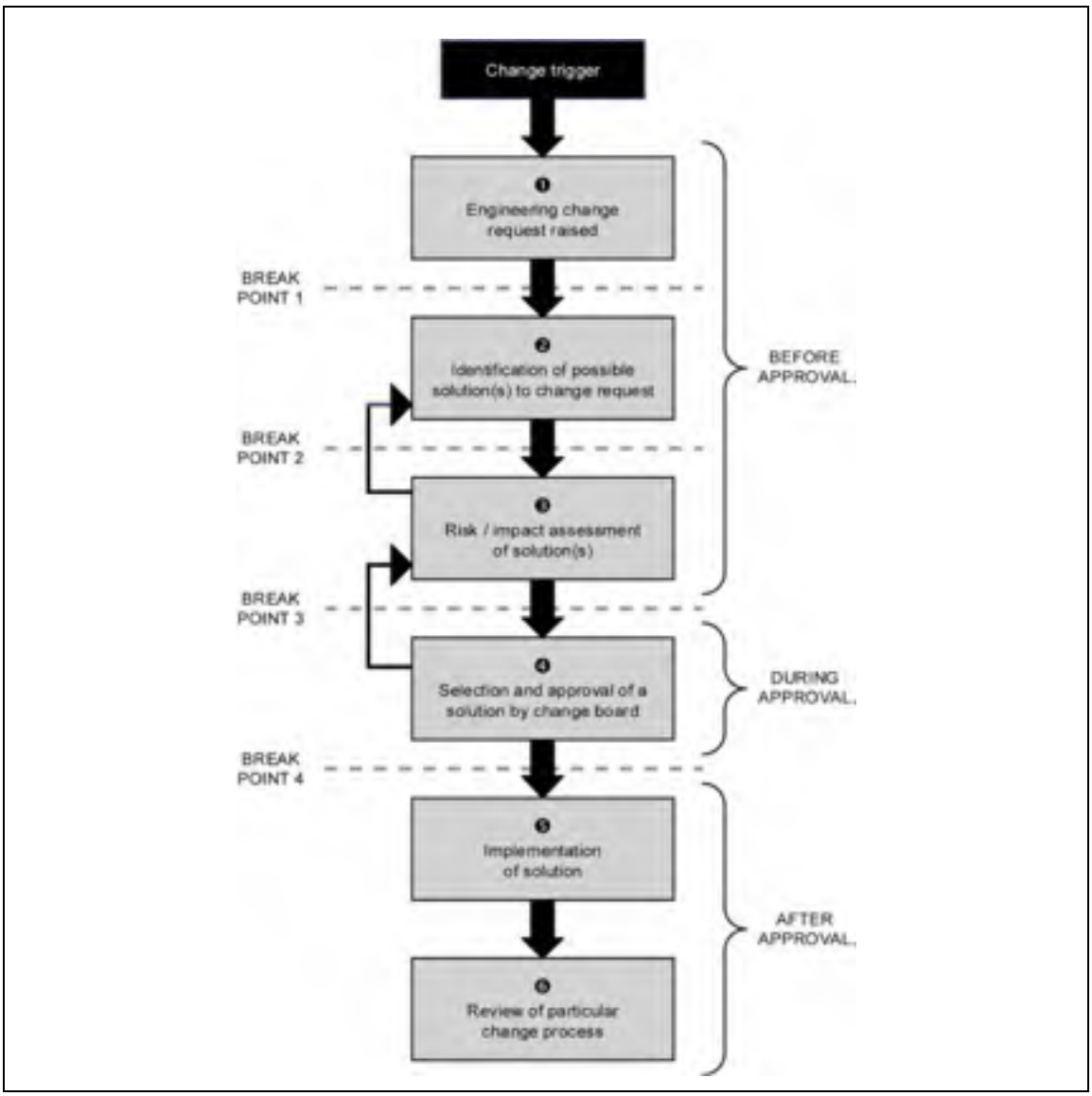


Figure 2.6 Un modèle de processus générique pour la gestion des modifications d'ingénierie (Jarratt et al., 2010)

Ce modèle de processus pour la gestion des modifications d'ingénierie a été choisi pour son positionnement à l'intersection de la colonne *controller* et des lignes tactiques et opérationnelles, mais aussi pour son rôle structurant lors de la collaboration multidisciplinaire. Il apparaît donc comme représentatif du type de contribution se positionnant au cœur de l'état de l'art de nos travaux. Néanmoins, ce processus reste ciblé sur les modifications techniques, n'intervenant qu'à l'issue de la conception.

Les modèles de processus sont des représentations simplifiées et génériques, représentants des processus établis et stables dans le temps. Leur mise en œuvre s'appuie généralement sur la notion de rôle et les transitions entre les différentes étapes peuvent être orchestrées grâce à des *workflows*. Ce type d'implémentation technique est détaillé dans la section suivante, qui a pour vocation de présenter quelques solutions techniques supports de la collaboration.

2.2.3.2 Implémentations techniques des modèles de processus

Comme nous venons de le voir, les modèles, qu'ils soient de données ou de processus, peuvent, pour être rendus opérationnels, être implémentés dans des outils ad-hoc tels que les PDM. Les modèles de données viennent ainsi spécifier les types d'objets gérés et les attributs attendus pour chacun d'entre eux grâce aux mécanismes de personnalisation disponibles dans la plupart des PDM commerciaux. Pour les modèles de processus, deux fonctionnalités ou éléments spécifiques permettent de venir personnaliser le fonctionnement des PDM. Ces deux modules, déjà cités précédemment, sont les *workflows* et les systèmes de gestion de projet.

Les workflows

Les *workflows* sont des éléments essentiels des PDM permettant d'assurer que la juste information est disponible pour les utilisateurs concernés au moment opportun. Ils incluent la définition des étapes de chacun des processus, les règles et activités associées à ces étapes, mais aussi les conditions et règles de validation et enfin les assignations de tâches aux différents utilisateurs impliqués dans le processus. "PDM workflows management provides the mechanisms for modeling and managing defined processes automatically. Data can be submitted to the appropriate workflows for processing. The workflow can transfer the data to nominated users, groups or project roles to carry out a specific business process. Appropriate information is routed automatically. At any location, the data can be assessed as it progresses

through its life cycle. PDM systems record information at each step in a process, and users can review the change history at any time." (Crnkovic et al., 2003).

Les modules de gestion de projet

Les modules de gestion de projet, ou *Program Management* en anglais, incluent des fonctionnalités standards comme la définition de la structure de découpage du projet (ou *Work Breakdown Structure* - WBS), l'allocation de ressources ou encore le suivi de projet. Ainsi, Crnkovic et al. présentent le travail lié à la gestion de projet comme :

- « Planning, or deciding what is to be done;
- Organizing the resources through the activities;
- Directing the activities towards the project goals;
- Controlling the activities concerning the project constraints where the most critical constraints are time, cost, and performance;
- Motivating the project members to accomplish the project objectives. » (Crnkovic et al., 2003).

Ils précisent également qu'en début de projet, ce dernier est souvent divisé en sous projets qui sont eux-mêmes raffinés en tâches plus restreintes et activités. Les tâches sont ensuite structurées dans les différents WBSs qui sont reliés entre eux dans des structures hiérarchiques. Cette structuration permet l'allocation des ressources et le suivi de l'avancement du projet par rapport au planning initialement établi. Ils soulignent enfin que le fait de pouvoir gérer le projet et les données au sein du même système, à savoir le PDM, permet de pouvoir organiser et exploiter la connaissance liée aux processus à mettre en œuvre pour réaliser des tâches d'ingénierie précises générant des données préétablies (Crnkovic et al., 2003).

Si les PDM peuvent être considérés comme des éléments de structuration essentiels de la conception collaborative, ces deux fonctionnalités correspondent à deux éléments essentiels de l'organisation des projets. Ainsi, dans le secteur automobile, un constructeur français vient de redéfinir l'organisation de ses projets de développement véhicule. Un modèle de projet structuré en sept phases a été défini, précisant les livrables attendus à l'issue de chacune de ces phases et les critères chiffrés permettant de définir le passage des différents jalons. Ce modèle sera ainsi instancié pour chaque nouveau projet véhicule et les rôles prédéfinis au

niveau du modèle seront assignés aux différents acteurs du projet. Des *workflows* de validation génériques ont également été définis, précisant notamment les domaines d'ingénierie impliqués lors des différentes approbations. Ce mode d'organisation s'apparente à de la capitalisation de connaissances où le modèle est régulièrement retouché afin de prendre en considération les retours d'expérience des derniers projets menés. Il offre l'avantage de fixer un cadre très structurant pour les nouveaux projets en permettant de contrôler leur avancement.

Mais ce cadre peut également se révéler un frein dans certaines situations. Des formulations d'exigences tardives, du repositionnement face à la concurrence, l'avènement de nouvelles technologies, la signature d'un nouveau partenariat ou encore le rachat d'une société sont autant de cas pouvant nécessiter des réorganisations plus ou moins profondes du projet qui sont perçues comme des déviations par rapport au modèle initial.

La gestion de ce type de projet s'appuie également généralement sur une répartition disciplinaire des tâches comme illustrée par la Figure 1.12. Le système à concevoir est alors divisé en sous-systèmes répartis entre les différentes équipes. Mais cette division implique que seul un nombre restreint de personnes impliquées dans le projet ont une vue d'ensemble sur les problèmes de conception rencontrés par les différentes équipes, résultant dans la recherche de solutions généralement locales.

En conclusion, bien qu'apparaissant comme une méthode nécessaire pour faire face à la complexité liée aux activités de conception (Aca et al., 2006), cette façon d'organiser la collaboration, qualifiée dans le premier chapitre de *project-planned*, révèle certaines lacunes. Nous relevons notamment que ce mode de gestion s'appuie sur des structures organisationnelles stables, sur des rôles préétablis et sur des processus relativement routiniers. Il laisse peu de place aux initiatives et aux innovations, n'incite pas aux échanges réguliers entre les différentes disciplines et est considéré à ce titre comme relativement « rigide » par opposition aux principes des méthodes agiles présentées dans la section suivante.

2.2.3.3 Méthodes agiles

De nombreuses contributions existent concernant les méthodes agiles dans le domaine du développement logiciel (Sommerville, 2010) et du *manufacturing* (Matthews et al., 2006). Cependant, la pertinence de ces méthodes pour les méthodes de conception simultanée commence seulement à être étudiée (Sommer et al., 2014). Matthews et al présentent les avantages des méthodes agiles comme « the ability to react rapidly to changes in the environment, whether expected or not » (Matthews et al., 2006). Ils insistent aussi sur le fait que les méthodes de conception simultanées classiques, appelées ici *project-planned*, ne sont pas prévues pour répondre à des changements imprévus comme une demande client tardive, une défaillance d'un concepteur, ou tout autre impact externe.

L'ensemble des méthodes agiles s'appuient sur les principes fondateurs d'un manifeste nommé « Agile Manifesto »[13]. Il est composé de douze principes énoncés ci-après.

[13] http://agilemanifesto.org/

Principle numbers	Principles
1	Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2	Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3	Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4	Business people and developers must work together daily throughout the project.
5	Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6	The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7	Working software is the primary measure of progress.
8	Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9	Continuous attention to technical excellence and good design enhances agility.
10	Simplicity--the art of maximizing the amount of work not done--is essential.
11	The best architectures, requirements, and designs emerge from self-organizing teams.
12	At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Tableau 2.4 Les 12 principes fondateurs des méthodes agiles

Afin de bien comprendre les tenants et les aboutissants de chacun de ces principes, ils sont repris un à un dans les paragraphes suivants afin d'être précisés et d'évaluer leur pertinence en regard de notre problématique. Ces descriptions synthétisent les idées issues de différents guides de mise en œuvre des méthodes agiles issus d'ouvrages de vulgarisation ou de recherches sur internet. Leur description va donc parfois plus loin que l'intitulé même du principe. Sur cette base, les principes sont classés en trois groupes intitulés « pertinent », « non pertinent » ou « principe général » n'apportant que peu de valeur ajoutée. Enfin, une synthèse de ces différents principes est proposée.

Principe 1 : Livraisons précoces, fréquentes et continues

Si l'objectif de la satisfaction client est déjà bien ancré lors de la conception de n'importe quel système, ce principe défend l'idée que l'utilisateur final doit être impliqué tout au long du cycle de conception, qu'il doit avoir accès aux résultats intermédiaires et que ces résultats intermédiaires doivent être partagés le plus fréquemment possible. Sans aller jusqu'à cette rupture dans les pratiques, l'idée la plus pertinente à retenir de ce principe réside dans le fait que les livraisons précoces, fréquentes et continues sont un véritable atout dans le cadre de la conception collaborative, notamment pour les aspects liés à l'intégration multidisciplinaire. Par le terme livraisons sont désignées les mises en commun des données techniques de définition permettant d'apporter des informations sur le produit. La maquette numérique, pas uniquement 3D, constituée par l'ensemble de ces données techniques, peut alors être considérée comme le résultat de l'ensemble des livraisons déjà réalisées.

Ce principe affiche clairement la nécessité d'avoir une mise en commun des travaux de façon précoce, fréquente et continue et est donc retenu comme « pertinent ».

Principe 2 : Favoriser les modifications d'exigences tardives

Bien que les processus de conception classiques invitent généralement à figer les exigences afin de pouvoir assurer que l'ensemble de ces derniers sont bien satisfaits à l'issue du cycle de développement, les méthodes agiles préconisent de ne jamais réellement figer ces dernières et de laisser l'opportunité au client de pouvoir demander des évolutions, quel que soit le niveau d'avancement du projet. Bien que l'adoption de ce principe puisse avoir des impacts quant à l'organisation, il ne nous semble pas essentiel pour favoriser l'intégration multidisciplinaire.

Ce principe est donc classé dans le groupe « non pertinent » par rapport à notre problématique et notre levier d'action, les méthodes organisationnelles.

Principe 3 : Petites livraisons fréquentes

Le troisième principe vient préciser le premier principe concernant les livraisons. Alors que ce dernier venait notamment avancer que les livraisons devaient être précoces, fréquentes et continues, le troisième principe ajoute que le volume de nouvelles informations ou corrections concernant le produit doit être limité, quitte à augmenter la fréquence des livraisons. Ceci réaffirme le principe de mise en commun au plus tôt, et ce même si le niveau de maturité de l'information n'est pas élevé.

Ce principe réaffirme la nécessité d'avoir des mises en commun des travaux de façon précoces, fréquentes et continues et est donc retenu.

Principe 4 : Collaboration multi-métiers

Le quatrième principe ne fait que réaffirmer un principe bien connu dans le cadre général du PLM. Il prône pour une meilleure intégration des métiers et réaffirme les principes même de la conception intégrée introduite dans le premier chapitre.

Ce principe est donc retenu bien qu'il n'apporte aucun élément d'information supplémentaire.

Principe 5 : Confiance et environnement optimisé

Ce principe prône pour que des conditions optimales soient réunies afin que les différents concepteurs soient les plus efficaces possibles. Il prône également pour que toute latitude leur soit accordée.

Ce principe est donc classé dans le groupe « principe généraux ». Bien que pouvant être intéressant, il s'attache plus à des aspects de pur management, quelque peu éloigné de nos travaux.

Principe 6 : Privilégier la communication directe

Avec ce sixième principe, nous touchons à une des limites des méthodes agiles (Kettunen and Laanti, 2008; Lindvall et al., 2004). Ces dernières ont en effet été imaginées pour de petites équipes et des projets de petite ou moyenne taille. Pour de gros projets, des adaptations sont nécessaires et certains principes ne sont plus applicables au pied de la lettre. La communication face à face, dans le cadre de gros projets collaboratifs, ne semble pas envisageable. En revanche, la transparence quant à l'avancement des différents acteurs, l'échange quant aux difficultés rencontrées et le resserrement de l'équipe projet afin d'éviter au maximum toute lourdeur liée à de la communication, notamment formelle, nous semble intéressant.

Certaines idées sous-jacentes à ce principe, la transparence et la communication « libre », sont donc retenues pour la suite des travaux.

Principe 7 : Redéfinir les indicateurs d'avancement

Dans le cadre d'organisations de type *project-planned*, différents indicateurs sont mis en place afin de mesurer l'avancement du projet tels que le nombre d'heures passées, le nombre de personnes affectées aux différentes tâches, la quantité de documents livrés, etc. En revanche, ce principe défend l'idée que la seule véritable mesure d'avancement est le nombre de fonctionnalités déjà livrées et testables. Bien que, de nouveau, ce principe ne soit pas directement applicable à tous les domaines, il nous semble pertinent de chercher à proposer des indicateurs sur l'avancement du projet qui soient au plus proche des actions à mener et des données de définitions matures effectivement livrées.

La mise en place d'indicateurs opérationnels « au plus proche du terrain » est donc retenue pour la suite des travaux.

Principe 8 : Engagement commun et constant

Le huitième principe défend l'idée que l'ensemble des acteurs doit s'engager de façon commune et constante tout au long du projet. Ce principe général ne nous semble pas apporter de nouvel éclairage ou de nouvelle opportunité quant à notre problématique et il ne sera donc pas retenu.

Principe 9 : Excellence technique

Bien que certainement pertinent, ce principe basé sur la constitution de l'équipe projet et le développement de ses compétences techniques ne nous semble pas un levier sur lequel il est envisageable d'agir en regard de notre problématique. Ce principe n'est donc pas retenu.

Principe 10 : Faire au plus simple

De nouveau, ce principe n'est que très peu en lien avec l'organisation du projet et la dynamique de l'information car il vise avant tout à conserver une certaine simplicité lors du développement du système considéré. Ce principe n'est donc pas retenu.

Principe 11 : Mettre les acteurs au cœur des décisions

Le onzième principe plaide pour une auto-organisation du projet. Même si ce mode d'organisation semble quelque peu utopiste, tout du moins dans le cadre de gros projets de conceptions collaboratives, il repose sur l'idée que les acteurs du projet ne doivent pas être de simples exécutants ou de simples techniciens, mais qu'ils doivent être à l'origine des

propositions. Pour reprendre les définitions proposées plus tôt dans ce manuscrit, ce principe prône donc pour une approche fortement *bottom-up*. Ce principe est donc classé dans le groupe « principe généraux ».

Principe 12 : Constante réorganisation

Par opposition aux organisations de type *project-planned*, ce principe défend l'idée que le projet doit régulièrement se réorganiser, en redéfinissant des objectifs qui prennent en considération les aléas du projet, les éventuelles nouvelles attentes, etc. Le terme « régulièrement » fait référence à la notion de cycles, qui est une notion primordiale pour la mise en œuvre de méthodes agiles. Ce point est détaillé plus loin dans ce manuscrit.

La possibilité de redéfinir les objectifs à court et moyen termes, impactant éventuellement l'organisation même du projet, nous semble donc être une idée potentiellement pertinente dans le cadre de nos travaux.

Synthèse des principales idées retenues des méthodes agiles

Sur la base des explications fournies pour chacun des principes, le Tableau 2.5 reprend les informations du Tableau 2.4 en ajoutant une colonne permettant de spécifier l'appartenance de chacun d'entre eux aux groupes intitulés « Pertinent », « Non pertinent » ou « Principe général » et en ajoutant un code couleur pour en faciliter la lecture :

- Gris foncé pour « Non pertinent »,
- Gris clair pour « Principe général »,
- Blanc pour « Pertinent ».

Principle numbers	Principles	Groups
1	Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.	Relevant
2	Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.	Not relevant
3	Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.	Relevant
4	Business people and developers must work together daily throughout the project.	Relevant
5	Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.	General
6	The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.	Relevant
7	Working software is the primary measure of progress.	Relevant
8	Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.	General
9	Continuous attention to technical excellence and good design enhances agility.	Not relevant
10	Simplicity--the art of maximizing the amount of work not done--is essential.	Not relevant
11	The best architectures, requirements, and designs emerge from self-organizing teams.	General
12	At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.	Relevant

Tableau 2.5 Les 12 principes fondateurs des méthodes agiles classés en trois groupes intitulés « Pertinent », « Non pertinent » ou « Principe général »

Sur la base des principes retenus, des regroupements ont été effectués afin de proposer trois points de synthèse. Ces points, présentés dans le Tableau 2.6, permettent de bien définir le cadre dans lequel s'inscrit notre proposition et de mettre en avant les apports de celle-ci.

Point de synthèse 1 : le partage régulier des données de conception
Le premier point, basé principalement sur les principes 1, 3 et 6, concerne le partage régulier des données de conception. Dans une logique de transparence et afin de rendre le travail de chacun visible, les livraisons (mises en commun des données techniques de définition du produit) doivent être précoces, fréquentes et continues. Cela sous-entend que des données non matures peuvent être partagées et que le système de gestion de données doit être en mesure de porter cette information de maturité. Cela signifie également que des corrections profondes peuvent intervenir sur des données déjà partagées. Enfin, les livraisons peuvent être de faible ampleur, c'est-à-dire contenant peu de nouvelles données.
Point de synthèse 2 : l'initiative aux acteurs opérationnels
Le second point concerne le mode d'échange et de communication entre les acteurs. Il s'appuie sur les principes 6, 11 et 12. Il rappelle l'importance du fait que l'initiative doit pouvoir provenir des concepteurs et qu'ils doivent pouvoir échanger librement autour des problématiques rencontrées. Cette logique cherche donc à promouvoir les approches *bottom-up*, en proposant notamment une redéfinition des objectifs en fonction de l'avancement réel du projet ou éventuellement une remise en cause partielle des exigences établies dans les phases amont du projet.
Point de synthèse 3 : la nécessité de pouvoir bénéficier d'indicateurs opérationnels
Le troisième et dernier point concerne les indicateurs. Il s'appuie sur les principes 7 et 12. La mise en place d'indicateurs opérationnels « au plus proche du terrain » est nécessaire pour permettre une prise de décision éclairée. Ce besoin s'inscrit de nouveau dans une logique *bottom-up*, qui vise à ce que les décisions soient prises sur la base d'informations opérationnelles précises.

Tableau 2.6 Les 3 points de synthèses retenus

Afin de bien comprendre en quoi ces points de synthèse viennent contribuer aux objectifs présentés dans le premier chapitre de ce manuscrit, le Tableau 2.7 montre leurs interactions. Ainsi, le « partage régulier des données » vient contribuer à l'objectif de décloisonnement des disciplines. « l'initiative aux acteurs opérationnels » participe à améliorer la collaboration et peut faciliter certaines prises de décisions en laissant plus de latitude aux acteurs opérationnels. Enfin, « la nécessité de pouvoir bénéficier d'indicateurs opérationnels » facilite bien entendu la prise de décision, mais peut également permettre d'assurer une forme de traçabilité décisions/données.

		Points de synthèse méthodes agiles		
		PdS1 : partage régulier des données de conception	PdS2 : l'initiative aux acteurs opérationnels	PdS3 : la nécessité de pouvoir bénéficier d'indicateurs opérationnels
Objectifs travaux	Obj. 1 : améliorer la collaboration	✔	✔	✘
	Obj. 2 : faciliter la prise de décisions	✘	✔	✔
	Obj. 3 : assurer la traçabilité décisions/données	✘	✘	✔

Pour rappel :

- *Objectif. 1 : Améliorer la collaboration entre les différentes disciplines en réduisant le cloisonnement existant entre les acteurs*
- *Objectif. 2 : Rendre la prise de décisions dans le cadre d'un projet de conception plus aisée grâce à la remontée plus fréquente et plus régulière d'informations opérationnelles précises.*
- *Objectif. 3 : Permettre la traçabilité entre les évolutions apportées aux données de définition du système conçu et les décisions prises tout au long du projet de développement*

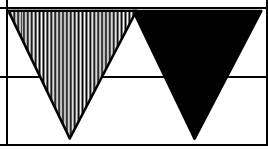
Tableau 2.7 Points de synthèse des méthodes agiles vs. objectifs de nos travaux

De nombreuses méthodes agiles existent, toutes inspirées voire issues de ce manifeste. Sommerville propose une revue de littérature des différentes méthodes disponibles (Sommerville, 2010). Dans le cadre de nos travaux, les méthodes les plus répandues ont été analysées (Scrum, Extreme Programming, Crystal, Adpative Software Development). L'ANNEXE I présente les méthodes agiles les plus couramment employées pour le développement logiciel. Bien que différents éléments supports à ces méthodologies aient inspiré notre proposition, il ne nous semble pas pertinent de décrire chacune de ces méthodes car elles restent fortement caractérisées par leur domaine d'application. Elles ne nous semblent donc pas directement applicables à la conception collaborative multidisciplinaire de systèmes. Seuls les trois points de synthèse présentés ci-avant sont donc retenus pour expliquer la logique dans laquelle s'inscrit notre proposition. Ces trois points permettent en effet d'adresser à eux seuls nos trois sous objectifs (Cf. Tableau 2.7).

2.2.3.4 Synthèse des approches *controller*

Le Tableau 2.8 positionne les approches *controller* par rapport aux facteurs de positionnement *model-view-controller* et *strategic-tactical-operational*. On retrouve ainsi les contributions de type « modèles de processus » qui se positionnent à l'intersection des lignes tactiques et opérationnelles et de la colonne *controller*, dans le sens *top-down*. Les « implémentations techniques » sont également positionnées dans ce tableau. Les *workflows* et les modules de gestion de projet mettent en œuvre au niveau opérationnel des processus établis au niveau tactique. Ils se positionnent donc de façon descendante sur ces deux lignes.

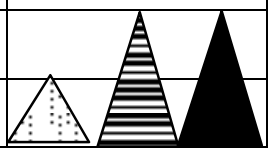
Enfin, les trois points de synthèse concernant les méthodes agiles sont positionnés dans le Tableau 2.9. Le point numéro 1, concernant le partage régulier des données de conception, se positionne au niveau opérationnel, à l'intersection avec la colonne *controller*. Les points numéro 2 et 3, portants sur l'initiative aux acteurs opérationnels et la nécessité de pouvoir bénéficier d'indicateurs opérationnels, se positionnent dans la colonne *controller* et permet de faire remonter des informations opérationnelles vers le niveau tactique voire stratégique.

	Model	Controller	View
Strategic			
Tactical			
Operational			

Modèles de processus

Implémentations techniques des modèles de processus

Tableau 2.8 Synthèse du positionnement des modèles de processus et de leurs implémentations dans l'approche *controller*

	Model	Controller	View
Strategic			
Tactical			
Operational			

PdS1 : Le partage régulier des données de conception

PdS2 : L'initiative aux acteurs opérationnels

PdS3 : La nécessité de pouvoir bénéficier d'indicateurs opérationnels

Tableau 2.9 Synthèse du positionnement des méthodes agiles dans l'approche *controller*

Dans la section suivante, une synthèse concernant l'ensemble des approches pour la conception de systèmes présentées dans ce chapitre est proposée avant de détailler les travaux de Namchul Do qui sont, tout comme nos travaux, une approche hybride *model* / *controller*.

2.2.4 Synthèse des approches *model*, *view* et *controller* pour la conception de systèmes et intérêt d'une approche hybride *model* / *controller*

2.2.4.1 Synthèse des approches *model*, *view* et *controller* pour la conception de systèmes

Dans ce chapitre, quelques approches de type *model*, *view* et *controller* ont été présentées. Cette sélection, loin d'être exhaustive, vise à établir une vision globale sur les travaux contribuant à la conception collaborative multidisciplinaire de systèmes. Le Tableau 2.10 synthétise ainsi le positionnement des principales contributions. Il cherche également à illustrer la prédominance des approches de type *model* et de type descendantes, notamment dans le domaine des approches de type *controller*.

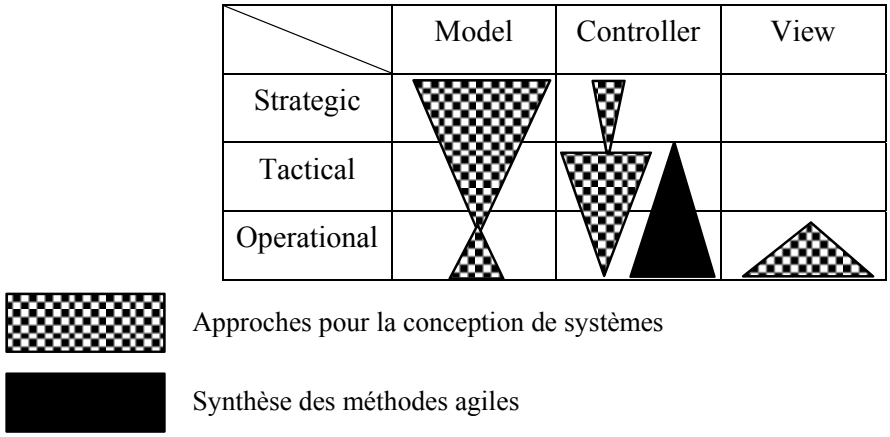


Tableau 2.10 Synthèse des approches pour la conception de systèmes et des méthodes agiles

La dernière section de ce chapitre est consacrée à la description des travaux menés par Namchul Do. Ces travaux ne trouvent pas leur place dans l'état de l'art structuré grâce au MVC car ils sont formés de plusieurs contributions évoluant d'une approche *model* à une approche *controller*. Leur pertinence en regard de nos travaux est liée à la finalité recherchée, l'intégration multidisciplinaire, et par la méthodologie employée, l'évolution des PDM inspirée par des constats réalisés dans le domaine informatique. Bien que les moyens mis en œuvre restent différents et qu'ils ne sont pas présentés sous le volet « agilité », ces travaux restent très complémentaires aux travaux synthétisés dans ce manuscrit.

2.2.4.2 D'une approche *model* vers une approche hybride *model/controller*

Les travaux réalisés par Do et ses coauteurs ont de nombreuses similitudes avec les travaux présentés dans ce manuscrit. Tout comme de nombreux acteurs s'inscrivant dans le cadre général du PLM, ses premiers travaux ont été consacrés à l'élaboration de modèles de données support à la conception collaborative de nouveaux produits (Han and Do, 2005). Puis, ses travaux se sont progressivement orientés vers l'intégration multidisciplinaire avec notamment l'intégration *hardware* (HW) – *software* (SW). Dans les premiers articles traitant de ce sujet, Do and Chae présentent un *framework* permettant de gérer des données SW dépendantes du HW (Do and Chae, 2006). Ils reprennent ainsi quelques fonctionnalités majeures des *Software Configuration Management* (SCM) et cherchent à les intégrer aux PDM afin de pouvoir déterminer certains liens de dépendance entre des données SW et HW et de pouvoir gérer des configurations de produit intégrant ces deux types de données. Puis, en partant du constat qu'un frein majeur à l'adoption des systèmes PDM par les développeurs tient au fait que les modifications du code source logiciel sont réalisées beaucoup plus fréquemment que sur les parties HW, et que le mécanisme de gestion des versions est très différent, ils proposent l'intégration d'un système de versionnement plus complet dans les PDM (Do and Chae, 2008).

Ces propositions sont reprises dans un article plus général (Do and Chae, 2011) où l'on note de nombreux points communs avec nos travaux (Bricogne et al., 2010a, 2010b). Dans cet article, ces mêmes auteurs détaillent les différences qui existent entre le mode de gestion des documents, des structures produit et des configurations produit dans les PDM et dans les SCM (Do and Chae, 2011). La Figure 2.7 détaille les deux modes de gestion des documents. Dans les PDM (a), des articles sont utilisés pour porter différentes versions via leurs métadata et différents documents servent de représentation de ce même article. Lorsqu'une modification est nécessaire, les documents sont alors réservés (*checkout*) pour éviter que deux concepteurs travaillent de façon synchrone sur le même document, ce qui induit une certaine linéarité dans les versions. Au contraire, dans les SCM (b), les documents sont les seules entités manipulées. Les modifications synchrones des mêmes documents sont autorisées, entraînant la création de branches (action nommée *branch*) et pouvant nécessiter la fusion de deux versions concurrentes (action nommée *merge*). Cette opération s'apparente à une réconciliation des deux versions, par intégration des intentions de chacun des développeurs. On parle alors de graphe de gestion de versions, et ce pour un seul et même item. Un item

représente ici, dans le cas du logiciel, un fichier source évoluant au cours du temps, générant différentes versions.

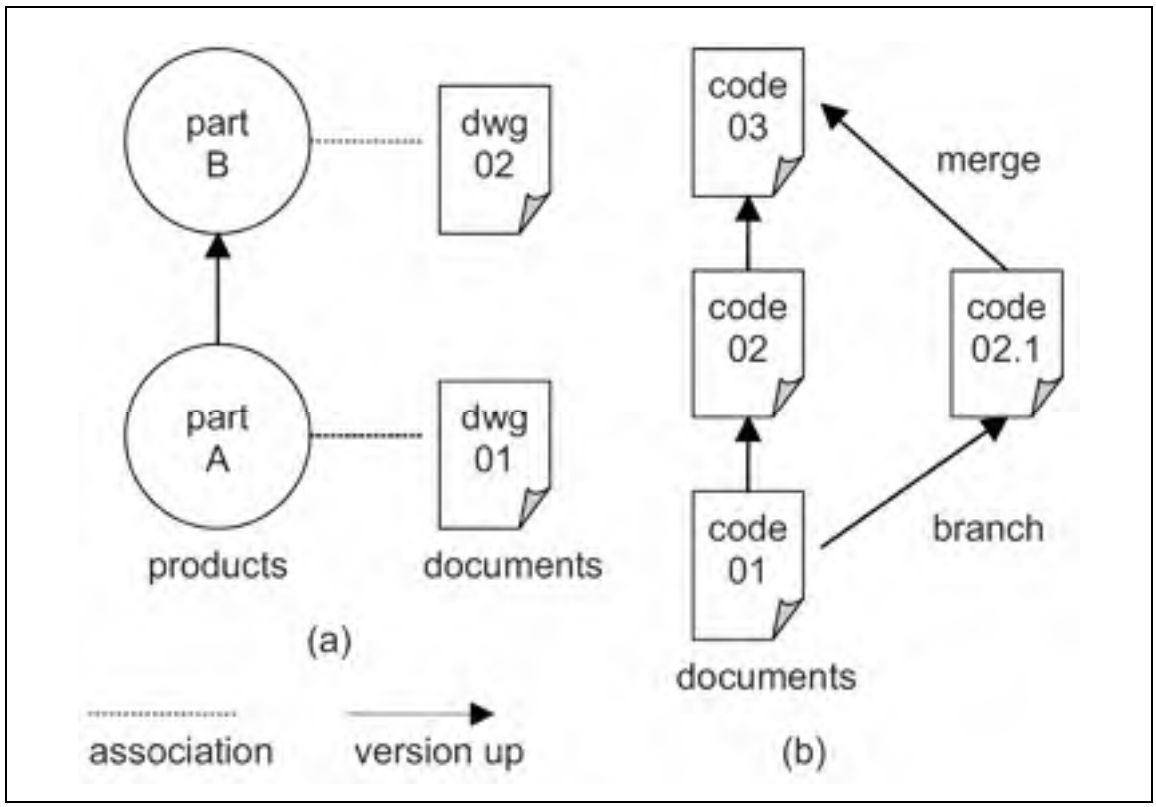


Figure 2.7 La gestion des documents dans les PDM (a) et dans les SCM (b) (Do and Chae, 2011)

La Figure 2.8 traite quant à elle de la gestion des structures et configurations produit. Alors que les PDM (a) s'appuient sur des structures basées sur des modules et des liens d'effectivité pour gérer notamment les différentes variantes du produit, les SCM (b) gèrent les configurations en référençant directement les items (dans notre cas les fichiers code source) dans une version donnée.

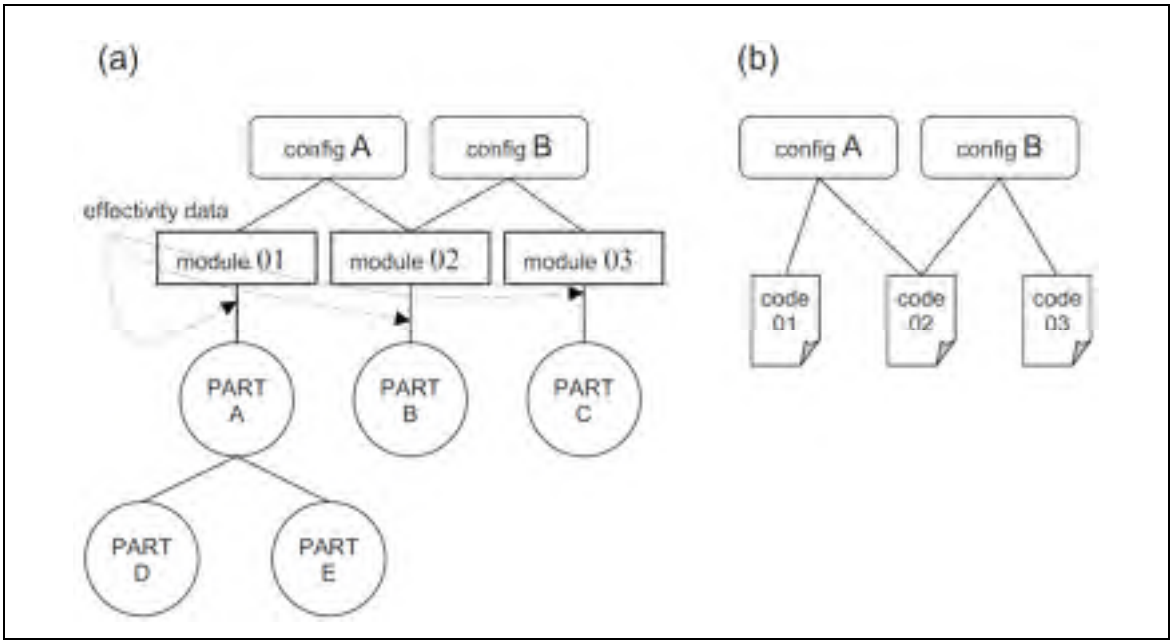


Figure 2.8 La gestion des structures et configurations produit dans les PDM (a) et dans les SCM (b) (Do and Chae, 2011)

Après cet article lié aux mécanismes de gestion des données, Do met en avant dans sa dernière contribution la nécessité de pouvoir bénéficier d'indicateurs permettant d'évaluer l'avancement du développement de produit (Do, 2014). Il définit ainsi un ensemble de Key Performance Indicators (KPIs) issus de données gérées par les PDM en s'appuyant sur une technique d'analyse de bases de données nommée *OnLine Analytic Processing* (OLAP). Il présente cette méthode comme « far more flexible and efficient approach than other result-oriented static evaluation approaches » (Do, 2014). Ces indicateurs opérationnels, relativement proches du terrain, sont donc une prémisse de solution pour répondre au second sous-objectif (pour rappel : rendre la prise de décisions dans le cadre d'un projet de conception plus aisée grâce à la remontée plus fréquente et plus régulière d'informations opérationnelles) présenté dans la section précédente.

Les similitudes de nos travaux avec ceux de Do et de ces coauteurs sont donc diverses. Tout d'abord, l'évolution d'une approche purement *model* vers une approche de plus en plus orientée *controller* est une forte similitude. Bien que la notion d'agilité ne soit pas revendiquée à proprement parler, les objectifs recherchés et les moyens mis en œuvre en restent également assez proches. La proposition, décrite dans le chapitre suivant, permettra de mieux comprendre ces similitudes.

2.3 Synthèse de l'état de l'art

Dans ce chapitre, différentes approches contribuant à notre problématique liée à l'intégration multidisciplinaire ont été présentées. Afin de caractériser au mieux ces approches et leurs apports et de pouvoir les positionner en regard de nos travaux, une structuration basée sur deux facteurs de positionnement a été proposée. Le premier facteur vise à préciser la nature même de la contribution des travaux. Inspiré d'un patron d'architecture intitulé *Model-View-Controller*, il distingue les approches dont la contribution concerne les modèles de données, les approches focalisées sur les interfaces utilisateur ou dispositifs matériels permettant d'interagir avec un logiciel ou un système d'information et les approches centrées sur la dynamique de l'information. Le second facteur s'attache quant à lui à définir la portée de la contribution, grâce à une échelle dont les trois niveaux sont intitulés stratégique, tactique et opérationnel. Ce second facteur permet également de préciser si la contribution favorise la diffusion de l'information dans un sens descendant (*top-down*) ou dans un sens ascendant (*bottom-up*).

Lorsque la problématique de ces travaux a été exposée, le cloisonnement existant entre les différentes disciplines de même que le mode d'organisation, caractérisé par une trop grande rigidité et une diffusion de l'information principalement descendante, ont été mis en avant. L'ensemble des approches présentées dans ce chapitre contribuent à améliorer la collaboration multidisciplinaire. Ainsi, les approches *model* contribuent à proposer un référentiel produit unifié, permettent de faciliter l'échange d'information autour du produit et peuvent, dans certains cas, favoriser l'intégration spatiale du produit. Les approches *View* sont quant à elles très peu nombreuses et s'intéressent principalement à la collaboration distante, sans s'intéresser aux complexités induites par l'aspect multidisciplinaire de la collaboration.

Dans le cas des approches *controller*, si les modèles de processus visent bien à organiser la collaboration, éventuellement même entre différentes disciplines, ils ne s'intéressent en revanche qu'à des situations ou des expertises bien particulières. Ces modèles sont généralement établis sur la base d'observations de terrain qui sont formalisées et décontextualisées afin d'être partagés comme « bonne pratique ». Ils peuvent donc être considérés, dans une certaine mesure, comme une approche *bottom-up*. Mais leur finalité reste néanmoins très orientée vers la standardisation et la normalisation des processus et ils s'inscrivent donc essentiellement dans une approche descendante. Enfin, ces modèles de

processus ne s'attachent que très peu à la conception collaborative multidisciplinaire. La Figure 1.12, illustrant le mode l'organisation *project-planned* basé sur la division et l'affection des tâches aux différentes disciplines, en fournit un exemple.

Les travaux de Do et de ses coauteurs apportent à la fois des contributions de type *model* et de type *controller*. Bien qu'ils ne soient pas présentés sous l'angle de l'intégration multidisciplinaire, nous pensons que certaines connaissances véhiculées par leurs travaux pourraient être mises à contribution afin de répondre aux objectifs de cette thèse. En revanche, Do et ses coauteurs travaillent essentiellement en proposant de nouvelles fonctionnalités pour les PDM et en concluant que ces fonctionnalités permettent aux acteurs des disciplines autres que la mécanique de pouvoir travailler sur cette même plateforme de gestion de données. Cette stratégie, visant à imposer aux autres disciplines l'usage d'un système de gestion de données aux mécanismes différents de ceux proposés traditionnellement, nous semble difficile à mettre en œuvre. En effet, elle risque de faire face à une forte résistance au changement. Dans notre proposition, présentée au chapitre suivant, les systèmes de gestion de données originaux sont conservés alors qu'une méthode s'appuyant sur des outils considérés comme neutre est proposée.

Le manifeste à la base des méthodes agiles, synthétisé en trois points au regard de notre problématique, est quant à lui au cœur et même d'une certaine manière une des motivations de nos travaux. Il apparaît donc pleinement comme une réponse aux besoins d'intégration multidisciplinaire, au décloisonnement des équipes, à la réintroduction d'une certaine flexibilité, et au pilotage basé sur des indicateurs opérationnels. Mais aucune des méthodes agiles actuelles, imaginées pour le développement logiciel, ne semble directement applicable dans le domaine de la conception de systèmes fortement multidisciplinaires. Une nouvelle méthode, s'inscrivant dans la même logique mais prenant en considération les spécificités de la conception de ce type de systèmes, reste donc à conceptualiser, formaliser et implémenter. Une proposition, basée sur trois concepts nommés *Collaborative Actions Framework*, *Workspaces* et *Branch & Merge* est décrite dans le chapitre suivant.

CHAPITRE 3

Proposition

Ce chapitre est constitué de trois parties qui correspondent aux trois concepts complémentaires formant la proposition visant à améliorer l'intégration dans le cadre de la conception multidisciplinaire. Plus précisément, le but de cette proposition est de décloisonner les disciplines en rendant les activités de conception plus agiles, mais également de permettre la propagation de l'information de façon descendante (*top-down*) et ascendante (*bottom-up*). Par sa mise en œuvre, le but est également de pouvoir analyser et ainsi mieux comprendre les phénomènes d'intégration des expertises lors de la conception collaborative de systèmes.

Les trois concepts formant la proposition ne se positionnent pas au même niveau, mais viennent plutôt s'imbriquer tels des poupées russes, comme représenté sur la Figure 3.1. Le concept le plus général est ainsi appelé le *Collaborative Actions Framework*. Il correspond à un cadre de collaboration opérationnelle autour d'actions, majoritairement d'ingénierie dans notre cas. Un des objectifs de ce *framework* est de faciliter la collaboration des acteurs des projets de conception, quelle que soit leur origine disciplinaire, mais également d'assurer une traçabilité entre les prises de décision et les corrections/modifications apportées sur les données techniques. Cette traçabilité est rendue possible grâce aux liens existants avec le second concept intitulé *Workspaces*. Bien que ce nom et concept soient déjà bien connus, ces travaux visent à préciser et étendre les possibles usages des espaces de collaboration. Parmi ces possibilités, un focus tout particulier est mis sur les notions de mise en commun continue des travaux, d'intégration multidisciplinaire et de validation via ces *workspaces*. Les échanges de données techniques entre les *workspaces* et le travail simultané sur les mêmes données techniques s'appuient sur la possibilité de gérer de façon parallèle différentes versions d'une même donnée technique. Ces différentes versions peuvent pour autant être toutes considérées comme pertinentes et on parle alors de branches. L'action de venir réconcilier les versions, ou mutualiser les différentes intentions de conception, est alors appelée

fusion. Ces deux opérations offrent de nombreuses possibilités. Elles sont regroupées sous l'appellation *Branch & Merge* et constituent le troisième et dernier concept de notre proposition.

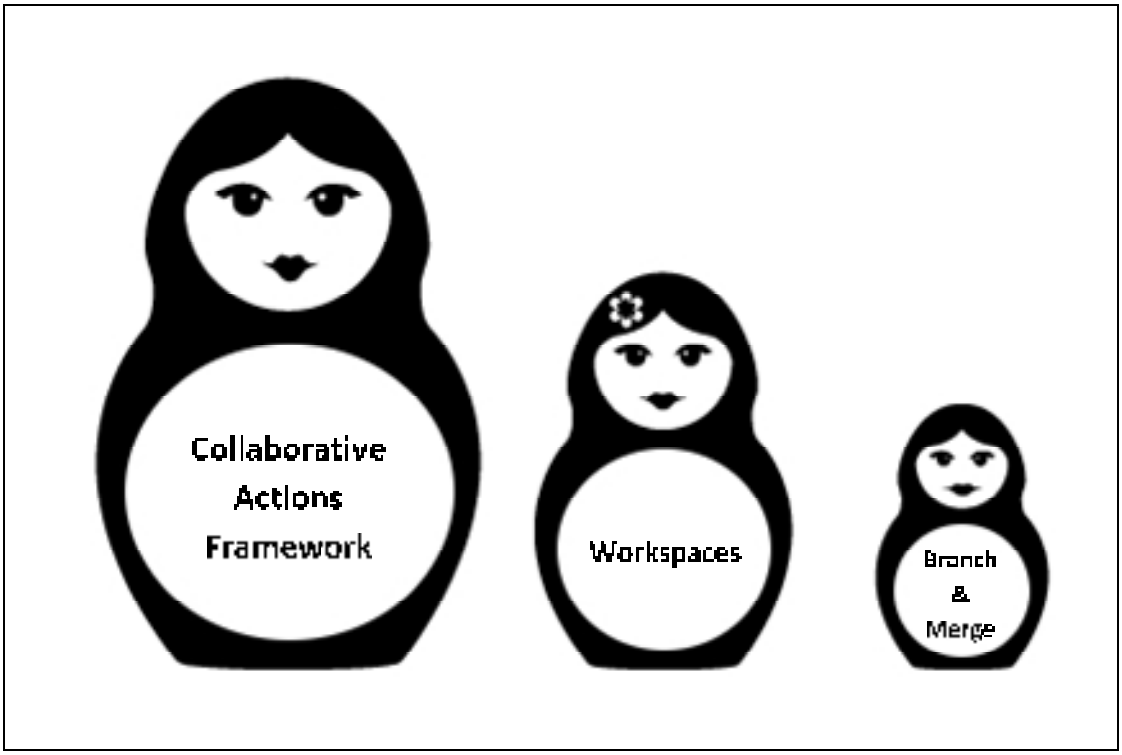


Figure 3.1 Imbrication des trois concepts formants la proposition

Afin de bien comprendre les liens existants entre ces différents concepts et les points de synthèse des méthodes agiles, le Tableau 3.3 montre leurs interactions. Ainsi, comme nous le verrons dans ce chapitre, le concept nommé *Collaborative Actions Framework* contribue au partage régulier des données de conception, permet de laisser l'initiative aux acteurs opérationnels et propose des indicateurs opérationnels pour la prise de décisions. Le concept nommé *Workspaces* permet également le partage régulier des données de conception et de laisser plus de latitude aux acteurs opérationnels. Enfin, le troisième concept intitulé *Branch & Merge* apporte une solution aux corrections/modifications concurrentes sur les données, facilitant le partage régulier des données de conception.

		Proposition : concepts		
		Concept 1 : *Collaborative Actions Framework*	Concept 2 : *Workspaces*	Concept 3 : *Branch & Merge*
Points de synthèse méthodes agiles	PdS1 : partage régulier des données de conception	✔	✔	✔
	PdS2 : l'initiative aux acteurs opérationnels	✔	✔	✘
	PdS3 : la nécessité de pouvoir bénéficier d'indicateurs opérationnels	✔	✘	✘

Tableau 3.1 Concepts formant la proposition vs. points de synthèse des méthodes agiles

La structure de ce chapitre suit l'imbrication des concepts présentée ci-avant, allant du plus général au plus spécifique.

3.1 Collaborative Actions Framework

Le *Collaborative Actions Framework* est un cadre de collaboration opérationnel autour d'actions, majoritairement d'ingénierie dans notre cas. Afin de décrire ses objectifs et son fonctionnement, cette section est structurée en trois parties. Tout d'abord, une description est proposée, avant de décrire sa mise en œuvre et de présenter une synthèse explicitant les principaux intérêts de ce *framework*.

3.1.1 Descriptions

3.1.1.1 Action collaborative

Une action collaborative est un objet permettant de structurer les informations nécessaires à la compréhension de la demande initiale d'action, à la réalisation de cette action et à sa validation. Elle permet de connaître l'historique des échanges entre les différents protagonistes et les données techniques de définitions impactées par les corrections/modifications réalisées dans le cadre de cette action. Enfin, cette action est qualifiée grâce à une échelle de sévérité, grâce à d'éventuelles annotations (également appelées *flags*) et grâce à un statut précisant son état. Les paragraphes suivants proposent de revenir sur chacun de ces attributs afin d'en préciser la signification et l'utilité. Ce paragraphe est structuré dans une logique temporelle en décrivant les phases dites de création, d'aiguillage, de réalisation.

La phase de création de la demande d'action collaborative

Une action collaborative est initiée par un créateur, qui précise l'intitulé et la demande liée à l'action à mener. Cette demande ne précise généralement pas le détail de l'action à mener mais bien le résultat attendu. Lors de cette création, un identifiant unique est attribué à cette action pour assurer un référencement aisé de cette action. Le contexte dans lequel cette demande d'action est formulée est également précisé. Ce contexte peut être le développement d'un nouveau produit, l'industrialisation d'un produit existant, des modifications à apporter à un produit déjà en production, etc. Vient ensuite l'étape de qualification de cette demande d'action. Une priorité est définie pour cette action selon l'échelle choisie par l'entreprise. Des éventuelles annotations, également spécifiques à

chaque entreprise, peuvent également être apposées sur cette demande d'action afin de mieux la qualifier. Par exemple, l'annotation « obligatoire » signifie que le produit/projet concerné ne pourra être considéré comme achevé sans que la demande d'action ne soit honorée. L'annotation « régression » signifie quant à lui que le problème décrit dans la demande d'action est un problème qui n'était pas présent sur une version précédent du produit. Enfin, un statut précisant que la demande d'action vient d'être ouverte est automatiquement positionné sur l'action. Ces attributs permettant de qualifier l'action sont susceptibles de changer tout au long de la vie de cette action collaborative et le statut est gouverné par les différentes étapes du cycle de vie de l'action, également personnalisable selon les modes de fonctionnement de l'entreprise concernée. Ces différentes étapes de description et de qualification correspondent à la phase de création de la demande d'action.

La phase d'aiguillage de l'action collaborative

À l'issue de cette étape de création, l'action est affectée au responsable du produit, sous-produit ou projet concerné qui analyse l'intitulé, le descriptif et le contexte de cette demande d'action et le réaffecte à l'un des membres de son équipe, ou à un autre responsable s'il estime qu'il n'est pas en mesure de répondre à cette demande d'action. Le cas échéant, il peut même renvoyer cette demande d'action à son créateur en précisant qu'il n'est pas concerné par la demande et qu'il ne sait pas à qui réaffecter l'action. Ce processus d'aiguillage est crucial pour maîtriser les délais de réalisation de l'action. Les points importants pour que l'aiguillage soit pertinent sont alors un descriptif précis de la demande d'action, un investissement suffisant de la part des aiguilleurs successifs et une bonne connaissance de la structure organisationnelle de l'entreprise. À chaque réaffectation, un commentaire justifiant cette réaffectation est apposée sur l'action, apportant successivement de nouvelles précisions quant à l'action à mener. Il est important de préciser qu'une demande d'action peut concerner différents concepteurs, issus de disciplines différentes ou responsables de sous-systèmes différents. Des modifications/corrections successives ou parallèles entre ces différents acteurs peuvent alors être nécessaires, et les actions peuvent ainsi être échangées telles des témoins lors d'un relais d'athlétisme. Cette métaphore rappelle également l'idée qu'il n'est pas nécessaire d'avoir le témoin en main pour commencer à courir. En effet, le relayeur commence sa course dès qu'il est en mesure d'appréhender quand et où le témoin lui sera

transmis. De la même manière, les tâches peuvent être parallélisées dès qu'un niveau d'information suffisant est disponible. Cette phase est nommée la phase d'aiguillage ou phase d'échanges.

La phase de réalisation de l'action collaborative

Lorsque la demande d'action collaborative est affectée à un acteur devant réaliser une modification/correction, ce dernier a la possibilité de changer le statut de l'action, en fonction du cycle de vie de l'action déterminé par l'entreprise. Par exemple, les statuts « ouverte - en cours d'analyse », « ouverte - en cours de modification/correction », « ouverte - en cours de test »… permettent de montrer l'avancement lié à cette demande d'action. Grâce au second concept de cette proposition, le concept de *workspaces*, qui est présenté plus loin dans ce manuscrit, le concepteur référence les modifications apportées aux données de définitions qui ont été nécessaires pour répondre, partiellement ou totalement, à la demande d'action. Les références des différentes données modifiées, objets et versions, apparaissent alors sur cette action. Cette phase est nommée la phase de réalisation de l'action.

La phase de clôture de l'action collaborative

Lorsque les différents échanges, ponctués par les éventuelles modifications apportées aux données de définitions, sont achevés et que les éventuels tests sont réalisés, l'action est alors close. De nouvelles informations de clôture sont alors fournies. La première information concerne le type de clôture : si des modifications ont été apportées, le statut de l'action est alors modifié pour « clos - modification/correction ». Si l'action ne peut être menée ou que la fonctionnalité demandée n'a pas à être présente sur le produit, le statut devient alors « clos – limitation du système ». Si une action similaire a déjà été demandée et accomplie, le statut est alors « clos – duplicata » et l'identifiant du duplicata est précisé. Enfin, si le créateur sollicite une action qui n'est pas conforme au comportement attendu du produit, l'action est fermée avec le statut « clos - documentation ». De façon, générale, cette phase est nommée la phase de clôture.

La phase de validation de l'action collaborative

Dès la clôture de l'action, le créateur de cette dernière est notifié afin de vérifier que les actions correctives menées, et/ou les informations apportées lors de la clôture sont acceptables relativement aux attentes initiales. C'est à lui qu'incombe la validation de la clôture. Si la clôture est refusée, l'action est re-transférée à la personne qui a effectué la clôture avec les raisons du refus et le processus décrit ci-avant reprend son cours. Cette étape est nommée la phase de validation.

Synthèse des différentes phases

Ces différentes étapes sont synthétisées sur la Figure 3.2. Cette figure rappelle également les principales tâches réalisées lors de ces étapes.

Les flèches les plus petites illustrent les échanges réalisés par les différents acteurs, alors que la grande flèche illustre qu'un refus de clôture est envisageable lors de sa validation si cette dernière n'est pas acceptable relativement aux attentes initiales de la demande.

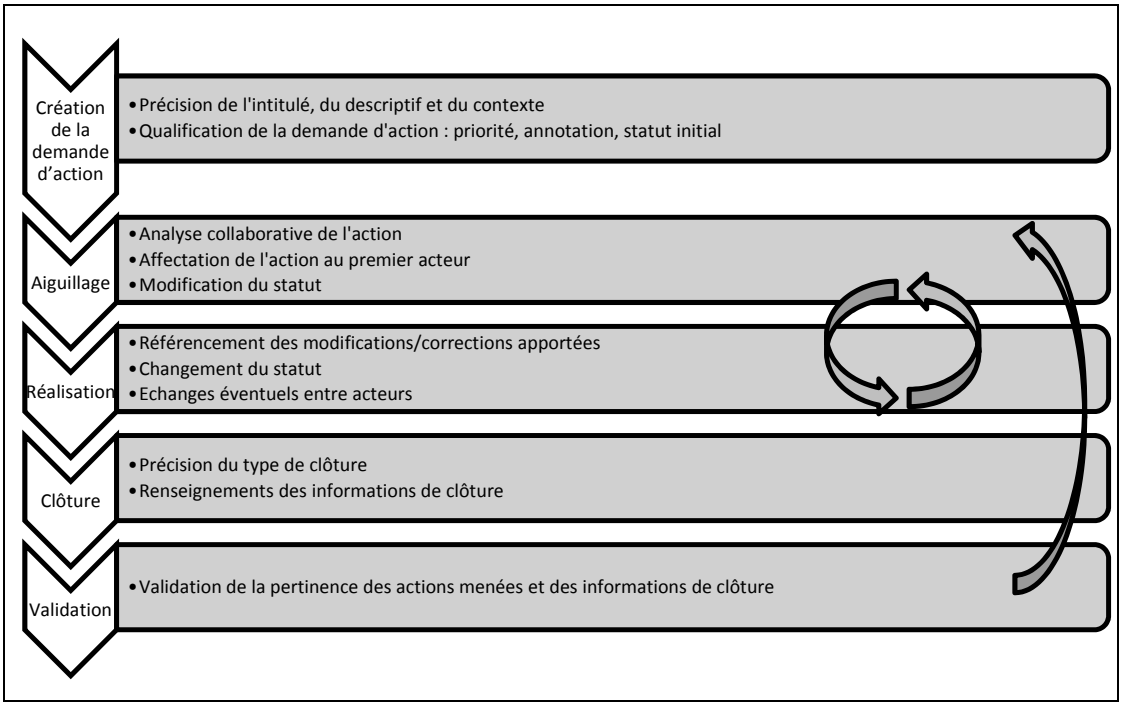


Figure 3.2 Phases successives d'évolution d'une action collaborative

Un diagramme de classes UML précisant une structuration possible des informations portées par une action collaborative est également proposé sur la Figure 3.3. La classe principale nommée « CollaborativeAction » fait ainsi référence à trois énumérations. La première (« CollabActionStatus ») spécifie les valeurs possibles pour le statut de l'action, la seconde (« CollabActionFlag ») montre les différentes annotations (*flags*) que l'action peut porter alors que l'énumération « CollabActionSeverity » précise les trois niveaux de sévérité. Enfin, le type « CollabActionComment », utilisé comme attribut de la classe principale, est spécifié grâce à la seconde classe présente en haut de ce diagramme. La liste de commentaires ainsi gérée permet d'assurer l'historique des échanges réalisés dans le cadre de l'action. Cette figure permet également de mieux appréhender le prototype informatique qui est présenté dans le chapitre 4.

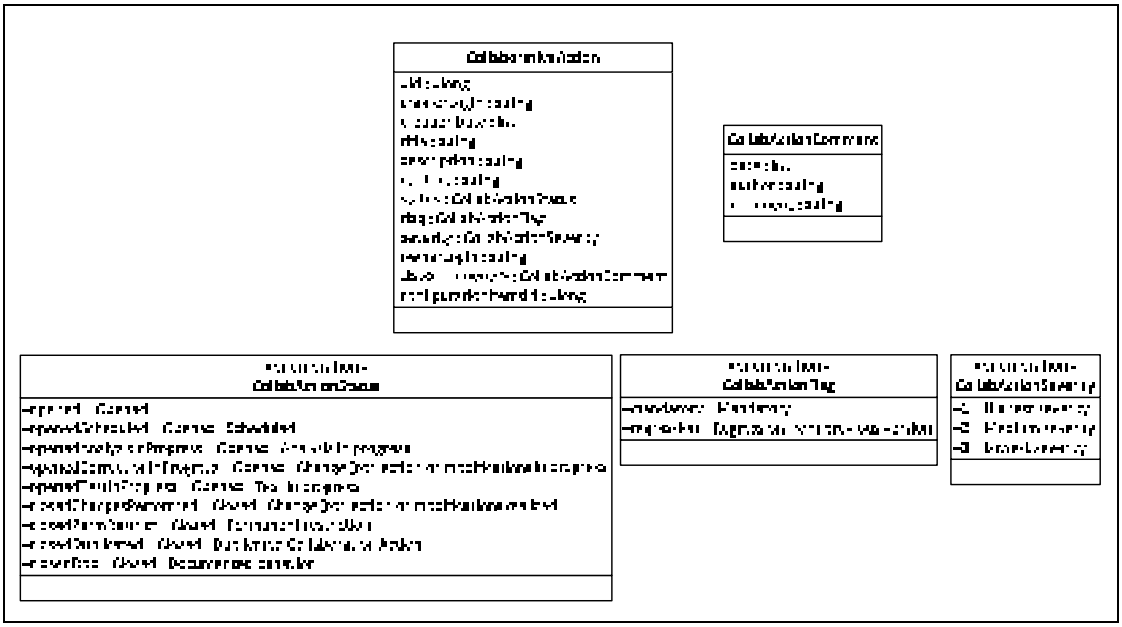


Figure 3.3 Diagramme de classes UML proposant une structuration des informations liées aux actions collaboratives

Les actions collaboratives représentent donc des objets permettant d'échanger des informations structurées autour d'actions opérationnelles précises pouvant impliquer de nombreux acteurs, potentiellement issus de disciplines différentes. Ces objets portent des informations sur l'action à mener, mais ils se révèlent également être une source d'information exploitable pour le pilotage des projets de conception. Le *Collaborative Actions Framework*, défini dans la section suivante, présente ces opportunités de pilotage par les actions.

3.1.1.2 Collaborative Actions Framework

Au-delà de l'intérêt de pouvoir structurer la collaboration autour d'actions nécessitant l'intervention de différents acteurs, les actions collaboratives peuvent être exploitées comme des indicateurs opérationnels permettant d'appréhender le travail quotidien des équipes.

En effet, elles permettent de remonter et de consolider des informations opérationnelles vers les niveaux tactique et éventuellement stratégique, mais également de piloter dynamiquement, sur la base de ces informations, le travail des équipes en modifiant les priorités des actions. Il a donc pour particularité de proposer une solution à la fois *bottom-up* et à la fois *top-down*.

Une solution bottom-up

Comme nous le verrons dans la rubrique précisant les mises en œuvre possibles, une des particularités du *framework* est que tous les acteurs ont la possibilité de venir créer des demandes d'actions collaboratives. Ainsi, des initiatives renseignées et formalisées par les concepteurs issus des différentes disciplines sont portées à l'attention des différents niveaux hiérarchiques tels que les chefs d'équipes, les responsables de départements et les chefs de projet, selon le niveau d'importance de la demande. En couplant ce système à un organigramme hiérarchique, il est possible de consolider ces demandes par personne, par équipe ou par département ou plus généralement à n'importe quel niveau hiérarchique. De la même manière, en couplant le *framework* à un organigramme fonctionnel ou à un organigramme projet, les demandes d'actions peuvent être consolidées par système, sous système ou par discipline. Cette consolidation permet de mieux comprendre la charge de

travail des différentes équipes, les composants pour lesquels les demandes d'actions sont les plus nombreuses etc. De façon plus générale, les actions peuvent alors être considérées comme des indicateurs bruts pouvant être combinés afin d'en déduire les indicateurs nécessaires au pilotage d'un ou de plusieurs projets.

Une solution top-down

La prise en compte de ces indicateurs a pour but une meilleure compréhension des activités opérationnelles afin d'assurer un pilotage plus précis et plus réactif que celui proposé dans le cadre d'une organisation de type *project-planned*. Afin de faire un parallèle avec les méthodes agiles, et tout particulièrement avec la méthodologie Scrum (*Voir ANNEXE I*), il est nécessaire de préciser que la mise en place d'une telle méthodologie s'appuie sur des outils gérant le « *backlog* produit » ou carnet du produit. Un *backlog* est la liste des fonctionnalités attendues d'un produit organisées par priorité. Chaque élément de ce *backlog* représente une fonctionnalité, un besoin, une amélioration ou un correctif. Il possède au minimum une description et une grandeur permettant d'ordonner les efforts entre eux. La méthodologie Scrum s'appuie également sur la notion de « *sprints* » (*Voir ANNEXE I*). Un sprint est une période pouvant durer entre une et quatre semaines pendant lesquelles les fonctionnalités à réaliser sont figées. Cela correspond donc à sélectionner un certain nombre d'éléments cohérents dans le *backlog* produit et de se focaliser sur ces éléments. À l'issue de chaque sprint, cette priorisation évolue. Dans la même logique, des actions collaboratives peuvent être définies comme prioritaires en fonction du contexte du projet. Le fait de mettre l'accent sur une ou plusieurs actions en demandant aux concepteurs de se focaliser sur ces actions permet de mettre en œuvre un pilotage réactif et précis. Les orientations stratégiques et tactiques de l'entreprise peuvent ainsi être mises en œuvre jusqu'à un niveau opérationnel.

La Figure 3.4 présente ce double apport pour permettre un pilotage opérationnel.

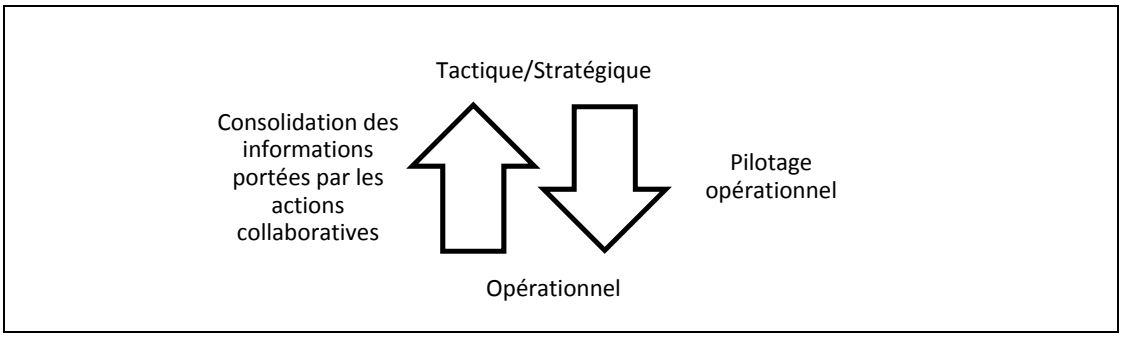


Figure 3.4 *Collaborative Actions Framework* : une solution *bottom-up* et *top-down* pour faciliter le pilotage opérationnel basé sur des informations issues du terrain

Cette section vise à présenter de façon assez générale et à proposer une définition des actions collaboratives et du *Collaborative Actions Framework*. Afin de montrer comment une telle solution peut-être mise en œuvre et les bénéfices qu'elle peut apporter, la section suivante présente les différents usages qui peuvent initier la création d'actions collaboratives et les usages qui peuvent être envisagés en termes de pilotage.

3.1.2 Mise en œuvre & apports

3.1.2.1 Action collaborative

Dans la section précédente, les actions collaboratives ont été présentées de façon assez générale et indépendante des solutions existantes permettant de faciliter la collaboration. Dans cette section, nous nous proposons de comparer ces actions collaboratives aux solutions existantes.

Dans l'état de l'art, nous avons présenté les *workflows* comme les mécanismes permettant la modélisation et la gestion automatique de processus prédéfinis. Mais seuls les processus routiniers peuvent ainsi être généralisés et modélisés. Ils s'appuient sur des rôles standards et prédéfinis. Le cheminement peut être personnalisé en fonction de décisions, d'approbations, etc. mais ces possibilités d'adaptation restent néanmoins assez limitées. En synthèse, les *workflows* sont considérés comme rigides et n'apportant pas une véritable solution permettant de répondre aux besoins de collaboration dans le cadre de la

conception, où les interactions ne sont pas définies a priori. Ce type d'interaction quotidienne/opérationnelle intervient de façon informelle entre les différents protagonistes. Les échanges en face à face, par courriels, par téléphone ou par *tchat* restent alors les usages les plus courants dans un contexte professionnel. Les actions collaboratives sont présentées comme des alternatives aux *workflows* pour les situations dans lesquelles le cheminement de l'information n'est pas prédéfini, permettant de substituer en partie certains échanges informels non capitalisés jusqu'alors.

Différentes situations peuvent être à l'origine de la création d'une action collaborative. Tout d'abord, en début de projet, les exigences formalisées peuvent être retranscrites sous la forme d'actions à mener. Ces actions sont ensuite précisées grâce à la création d'actions filles ayant pour rôle de raffiner les exigences initiales en actions opérationnelles. Cette étape de raffinement permet d'obtenir, après plusieurs étapes si nécessaires, des actions opérationnelles dont l'objet est suffisamment précis pour qu'elles soient considérées comme « unitaires », c'est-à-dire qu'elles puissent être mises en œuvre par un acteur et non par une équipe. Bien entendu, cela ne signifie pas que ces actions ne devront pas être échangées entre différents experts issus de différentes disciplines. Dans le cas où les actions collaboratives sont issues des exigences, elles sont également utilisées dans une logique de traçabilité entre les exigences et les actions menées.

Une seconde situation à l'origine de la création massive d'actions collaboratives est l'affectation de tâches issues de la gestion de projet. Le terme couramment utilisé pour parler de ces informations est le *Work Breakdown Structure* : il représente une décomposition hiérarchique des livrables et y associe les tâches et activités à réaliser pour atteindre les objectifs fixés.

Une troisième situation à l'origine de la création d'une action collaborative peut être la rencontre d'un problème de conception par un des acteurs. Si ce problème de conception peut être résolu localement, à savoir directement par ce dernier ou au sein de l'équipe, l'action collaborative n'a que peu d'intérêt et elle peut au mieux être considérée, si elle est créée, comme un pense-bête. Mais si ce problème concerne plusieurs équipes, l'action porte alors la trace / l'historique de la collaboration, les données impliquées, les décisions prises à son sujet voire, le cas échéant, les sous actions (ou actions filles évoquées

précédemment) lancées pour résoudre le problème initial. Cet apport est d'autant plus important que le nombre d'acteurs, le nombre d'équipes ou le nombre de disciplines impliqués est important.

Enfin, la quatrième situation à l'origine de la création d'une action collaborative est une demande de modification ou *Engineering Change Request* (ECR). Si une demande est formulée, elle est analysée selon un processus bien établi comme décrit sur la Figure 2.6. L'action peut alors être affectée au décideur qui prend la décision de clore l'action sans suite ou définit les opérations à mener. L'action permet alors de recueillir en son sein la plupart des informations liées aux différentes étapes, à savoir la demande de modification, son instruction et son exécution (*Engineering Change Order* - ECO). Les concepteurs peuvent alors réaliser les différentes modifications qui sont alors capitalisées grâce à l'action collaborative. Elle porte ainsi l'ensemble des informations décisionnelles et la liste des références des données impactées.

La section suivante continue à illustrer une mise en œuvre possible du *Collaborative Actions Framework*, dans une logique de pilotage descendante, basée sur la consolidation d'informations opérationnelles.

3.1.2.2 Collaborative Actions Framework

Comme nous l'avons vu dans la section relative aux définitions ci-avant, le *Collaborative Actions Framework* permet, grâce à la consolidation des informations portées par les actions collaboratives, de venir mettre en place un certain nombre d'indicateurs. Ces indicateurs peuvent être proposés aux différents niveaux hiérarchiques, par produit, ou selon tout autre type de regroupement nécessité pour la prise de décisions. Une mise en œuvre envisageable, inspirée par la méthodologie agile Scrum *(Voir ANNEXE I)* est l'organisation de réunions journalières où sont passées en revue les actions en cours afin de prioriser ces dernières. Bien entendu, seules les actions considérées comme prioritaires, après avoir appliqué certains filtres, sont traitées. Lors de cette réunion, des représentants des différentes disciplines, des différentes activités d'ingénierie et des principaux projets sont représentés afin de s'informer mutuellement sur les difficultés

rencontrées et sur les actions collaboratives qui méritent une attention particulière. Elle nécessite donc en amont des réunions d'équipes afin de consolider les informations. Ces actions sont alors classées grâce à la sévérité qui peut être revue et priorisées grâce à l'apposition d'annotations. Ces réunions permettent également la mise en œuvre d'une planification permettant de rendre parfois prioritaires des actions moins sévères mais plus urgentes car nécessaires à la sortie d'un nouveau produit.

Dans cette section, un exemple d'exploitation du *Collaborative Actions Framework* a été proposé afin d'illustrer le pilotage qu'il est possible de réaliser, en se basant sur des informations opérationnelles. Les décisions sont alors prises quotidiennement et le pilotage y est collectif, grâce à la participation des différents spécialistes de l'entreprise; Ce type de réunion permet également d'échanger sur les différents projets en cours et sur les différentes difficultés rencontrées. La section suivante synthétise de façon plus globale les apports de ce concept en s'appuyant notamment sur les trois points de synthèse des méthodes agiles présentés dans l'état de l'art.

3.1.3 Synthèse : le Collaborative Actions Framework support des méthodes agiles

Dans le chapitre dédié à l'état de l'art, différents points synthétisant quelques grandes idées à l'origine des méthodes agiles ont été présentés. Seuls ces points ont été retenus comme cadre pour nos travaux car les méthodes agiles existantes, telles qu'appliquées en génie logiciel, ne nous semblent pas directement applicables. Par exemple, la taille conséquente et l'éloignement géographique important des équipes contribuant à la création de nouveaux produits semble aller à l'encontre des principes mêmes des méthodes agiles. Kettunen et Laanti précisent à ce sujet que les principes de base des méthodes agiles sont « that a small, collocated team working closely together with the business customer can produce high-value, high-quality software efficiently with rapid iterations and frequent feedback. However, in large NPD organizations developing complex products those basic assumptions are not necessarily sufficient » (Kettunen and Laanti, 2008). La localisation, la formation ou encore la taille des équipes sont bien entendu des facteurs pouvant fortement influencer le processus de développement produit et le niveau d'intégration des équipes (McMahon, 2005), mais ces leviers sont considérés

comme hors de propos dans le cadre de nos travaux. L'investissement total du client final dans le processus de développement produit semble également pouvoir modifier les modes d'organisation actuels, mais nous avons pris le parti de proposer des outils et méthodes dans la continuité des pratiques actuelles, et non en rupture avec ces dernières. Le *Collaborative Actions Framework* s'inscrit donc dans cette logique de complémentarité avec les méthodes et outils actuels et vient proposer une mise en œuvre des points de synthèse deux (« initiative aux acteurs opérationnels ») et trois (« nécessité de pouvoir bénéficier d'indicateurs opérationnels ») proposés dans l'état de l'art.

Concernant le point deux, intitulé « initiative aux acteurs opérationnels », le *framework* proposé s'inscrit pleinement dans cette idée en laissant l'opportunité à l'ensemble des acteurs, quelle que soit leur origine disciplinaire, de pouvoir formaliser une demande d'action et la transférer à n'importe quel membre du projet ou de l'entreprise. Ce système d'échange d'information se révèle également neutre en termes de discipline. En effet, il n'est pas inféodé à un système d'information tel qu'un PDM, un SCM ou SGDT électronique, ce qui a parfois un effet négatif sur l'adoption de ce type de système par le plus grand nombre. Les actions demandées par les concepteurs peuvent être des actions multidisciplinaires, favorisant l'intégration des équipes voire des solutions techniques choisies.

Le point trois fait référence à « la nécessité de pouvoir bénéficier d'indicateurs opérationnels ». Comme nous l'avons vu, les informations portées par les actions peuvent être considérées comme des indicateurs bruts pouvant se révéler particulièrement intéressants s'ils sont consolidés et combinés selon les besoins de l'entreprise ou du projet. Les priorités, les annotations, les détenteurs, le cheminement des actions sont autant d'informations qui peuvent permettre d'obtenir des informations sur l'état de maturité d'un produit, sur la charge de travail d'une personne, d'une équipe ou d'un département ou encore la collaboration qui peut exister entre différentes disciplines. Dans le cadre de l'intégration multidisciplinaire, la fréquence des transferts d'actions entre les différentes disciplines ou encore l'analyse du raffinement des exigences entre les différentes disciplines sont autant de mesures in situ particulièrement pertinentes.

En revanche, le *Collaborative Actions Framework* ne contribue que peu au point de synthèse numéro trois qui concerne « le partage régulier des données de conception ». Ce *framework* est en effet avant tout dédié à l'organisation opérationnelle des équipes, et non à la gestion et au partage des données. La Figure 3.5 rappelle d'ailleurs le positionnement de ce concept par rapport aux points de synthèse des méthodes agiles. Il offre néanmoins la possibilité d'assurer une traçabilité entre les prises de décision, représentées par les actions collaboratives, et les corrections/modifications apportées aux données techniques. Ce lien est porté par chacune des actions qui recueillent en permanence et conservent l'historique des créations, corrections et modifications réalisées afin de répondre aux demandes d'action initiales. Ce mécanisme est assuré grâce à la complémentarité qui existe avec le second concept présenté dans la section suivante : les *Workspaces*.

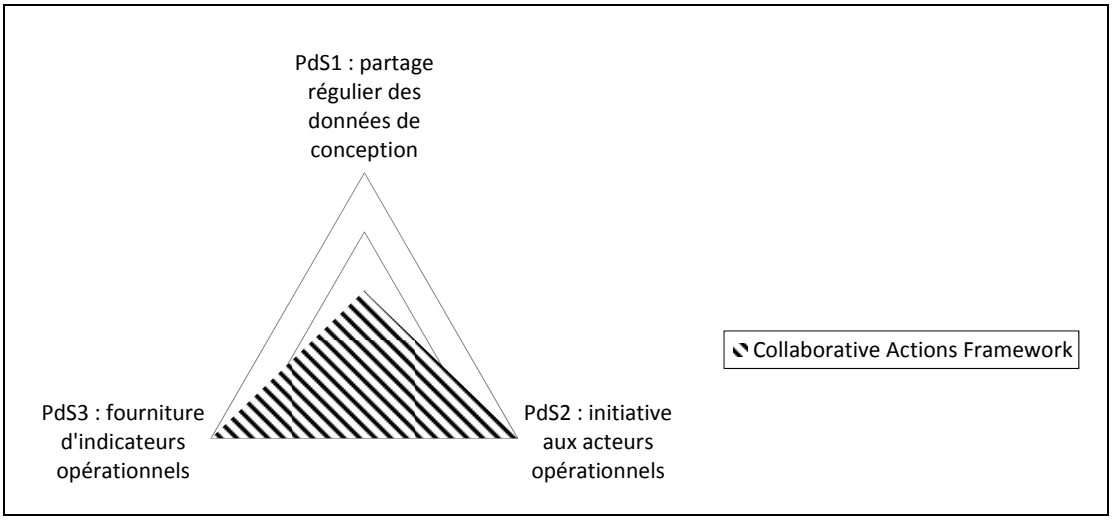


Figure 3.5 Positionnement du concept nommé *Collaborative Actions Framework* par rapport aux points de synthèse des méthodes agiles

3.2 Workspaces

Bien que le nom et concept de *Workspaces* soit déjà bien connu, un nouvel éclairage est donné dans cette section sur les possibilités offertes par ces espaces de collaboration. Parmi ces possibilités, les notions de traçabilité des corrections/modifications, via le lien avec le *Collaborative Actions Framework*, de mise en commun continue des travaux, d'intégration multidisciplinaire et de validation via ces *workspaces* sont détaillées dans cette section. Mais au préalable, une définition et une description des mécanismes d'échanges de données entre *workspaces* sont proposées.

3.2.1 Description

Les *workspaces* (WS) sont classiquement, en génie logiciel mais également dans les PDM, des espaces permettant aux concepteurs de pouvoir sélectionner et isoler certaines données, parmi l'ensemble des données disponibles, le temps nécessaire à la réalisation d'une action au sens large du terme (Estublier, 2001, 2000). Dans le cadre de notre proposition, le concepteur maîtrise la mise à jour des données isolées et choisit à quel moment mettre son travail à disposition de la communauté. Son travail reste privé par défaut, *i.e.* non accessibles aux autres collaborateurs, sauf s'il en décide autrement. L'initiative de création d'un *workspace* est laissée au concepteur, qui choisit également la durée de vie de celui-ci : la création d'un *workspace* peut être décidée afin de répondre à une demande d'action très ponctuelle, ou au contraire, un *workspace* peut servir à la réalisation de nombreuses actions collaboratives. Le contenu d'un *workspace* est en permanence gardé sous le contrôle du système de gestion de versions, seul prérequis au fonctionnement des *workspaces*. Les *workspaces* s'accommodent en effet des méthodes de structuration des données employées.

Sur la base des principes énoncés ci-avant, les paragraphes suivants détaillent les prérequis nécessaires au fonctionnement des *workspaces*, les principes de structuration pour des données multidisciplinaires, l'organisation des *workspaces* entre eux et enfin les mécanismes d'échange de données entre les différents *workspaces*.

3.2.1.1 Prérequis nécessaires au fonctionnement des workspaces

Afin de pouvoir fonctionner, les *workspaces* ont besoin de pouvoir référencer des objets, intitulés dans le cadre de ces travaux, des *Configuration Items* (CI). Ce terme a été repris à la communauté s'intéressant au *Change Management* (CM), avec notamment l'ouvrage de Crnkovic et al. qui définit les *Configuration Item* comme « an element of software or a document placed under version control (…). The most common example of a CI is a source code file, but executables, products, and documents are also CIs. A group of CIs can be defined as a CI (*i.e.*, the group is version controlled itself). » (Crnkovic et al., 2003). Il peut être raisonnablement traduit par « élément versionnable ». Ces CIs peuvent correspondre à toutes sortes de documents, données techniques, modèles, voire features, paramètres etc. pouvant exister sous la forme de fichiers ou d'objets dans une base de données. Suite à leur évolution, les CIs sont versionnés à chaque modification connue du système de gestion de données. Ainsi, quelle que soit la nature de la donnée, son format d'écriture et son système de gestion, seul le fait de pouvoir référencer et accéder à une version précise de cette donnée est nécessaire au bon fonctionnement des *workspaces*. Par la suite, dans ce manuscrit, nous ferons référence à un *Configuration Item* XXX dans la version YYY grâce à la convention suivante : CI_XXX_vYYY ; XXX représentant généralement l'identifiant unique de l'objet et YYY sa version dans un format numérique.

3.2.1.2 Principes de structuration pour des données multidisciplinaires

Les principes de structuration des données sont très différents selon les métiers et les systèmes de gestion de données employés. Après avoir étudié différents logiciels du marché destinés à gérer les données issues des domaines logiciel, mécanique et électronique et avoir étudié certains travaux comme ceux de Crnkovic et al. (Crnkovic et al., 2003) et Do et Chae (Do and Chae, 2011), nous proposons ici ce que nous considérons comme une synthèse de différents mécanismes mis en œuvre dans le cadre de la gestion de données HW et SW.

Tout d'abord, il est nécessaire de préciser que « le grain le plus fin » de la donnée géré par le système n'a pas nécessairement besoin d'être le même selon les disciplines ciblées pour l'intégration recherchée. Ceci signifie dans certains domaines ou selon les besoins,

la donnée la plus fine pourra être un simple fichier, un feature présent dans un modèle CAO, une classe en langage objet informatique, une méthode ou un attribut toujours en langage objet, un composant électronique, etc. Ce grain le plus fin est appelé, conformément au paragraphe précédent, un *Configuration Item* (CI). La Figure 3.6 montre la symbolique choisie pour représenter un CI et illustre le fait que les CIs sont des objets versionnables grâce au Versions System.

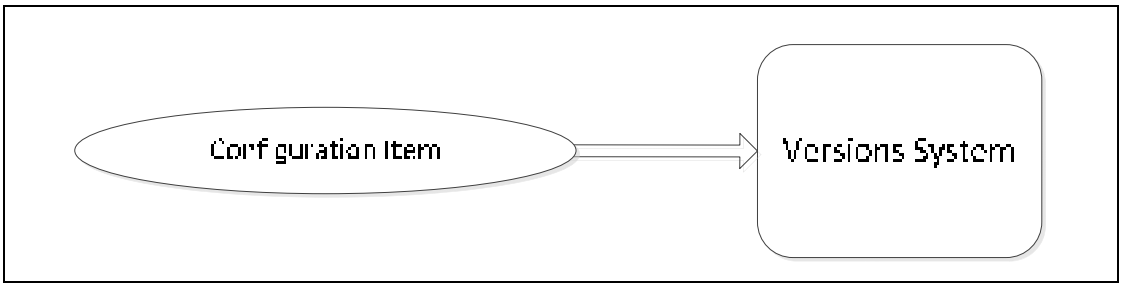


Figure 3.6 Symbolique du *Configuration Item*, « grain le plus fin » versionnable

Lorsqu'un système, sous-système ou pièce physique est conçu, de nombreux modèles relatifs à cette entité physique, dépendant notamment des disciplines d'ingénierie impliquées, sont créés. Ces modèles peuvent être considérés comme des abstractions de la réalité (Gardan, 1991). Une façon de regrouper ces différentes modèles, implémentés sous la forme de représentations, est de faire appel à la notion d'article. Un article est un « bien identifié en tant que tel, constituant un élément de nomenclature ou de catalogue » (Maurino, 1994). C'est, dans nos travaux, également un objet qui vient regrouper les différents couples modèles / représentations qui lui sont relatifs. Ainsi, le maintien de la cohérence entre ces modèles peut être facilité et différentes méta-datas relatives à l'entité physique peuvent être mutualisées. Si chacune des représentations de cet article est un CI (donc versionnable), l'article en lui-même est également versionnable. Si l'une de ces méta-datas est modifiée ou que l'une de ces représentations change de version, l'article doit également monter en version. En effet, un article dans une version donnée ne doit pas porter la moindre ambiguïté quant à sa constitution/composition. Il est à noter que tous les CIs mentionnés dans le paragraphe précédent ne font pas nécessairement référence à un article. Ils sont alors de simple CI que nous considérons comme des « ressources », c'est-à-dire qu'ils ne participent pas directement à la définition d'un produit ou d'un système. La Figure 3.7 montre la

symbolique « de la pâquerette », où le cœur de celle-ci correspond à l'article et chacun des pétales correspond à une représentation. L'ensemble des objets, article et représentations, sont versionnables grâce au Versions System.

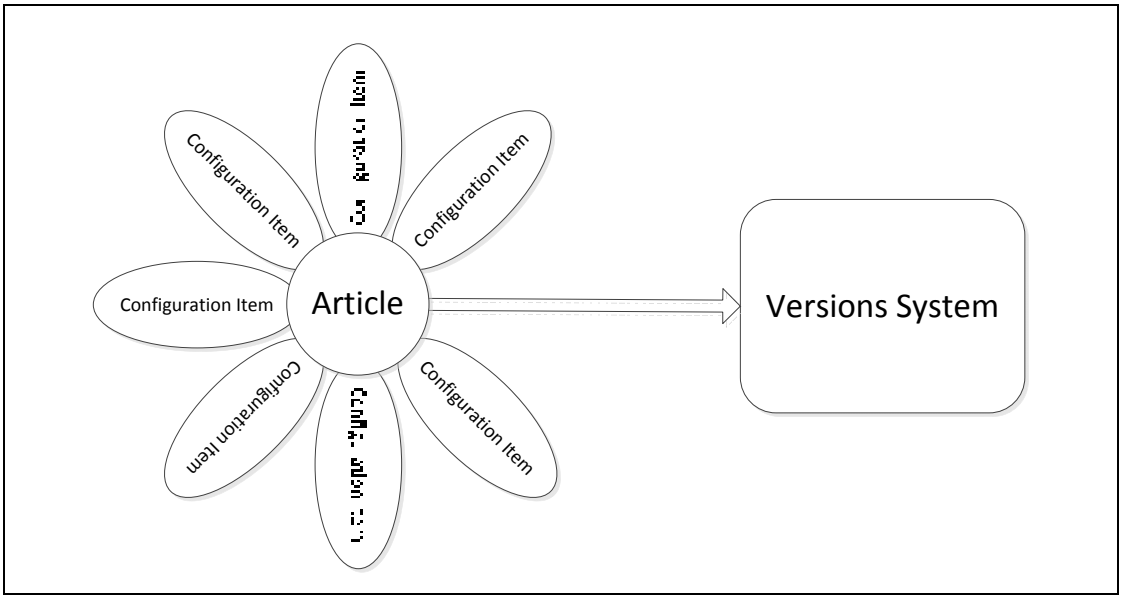


Figure 3.7 Métaphore de la pâquerette pour représenter l'article et ses représentations

Le troisième et dernier niveau de structuration reprend la notion de nomenclature. La nomenclature est « une description des niveaux successifs de décomposition
du produit en objets techniques » (Maurino, 1994). Dans nos travaux, elle est utilisée comme une structure arborescente permettant d'organiser les articles en systèmes et sous-systèmes. Elle est également versionnable : de la même manière que l'article, si l'un des articles composant la nomenclature est modifié, la nomenclature doit subir une montée de version. Cette affirmation peut potentiellement avoir des impacts très importants sur la mise en œuvre d'une telle solution, car la moindre modification d'un CI ou d'une métadonnée induit des montées de versions en chaîne pour l'ensemble des articles et nomenclatures référençant ce CI. Des travaux pour limiter ces impacts de propagation en chaîne ont d'ores et déjà été menés (Şenaltun and Cangelir, 2012), mais ils ne concernent pas directement notre problématique.

La Figure 3.8 résume le rôle des CI, des articles et des nomenclatures pour la structuration des données multidisciplinaires. Elle rappelle également que ces objets sont versionnables grâce à un système de versionnements dénommé Versions System.

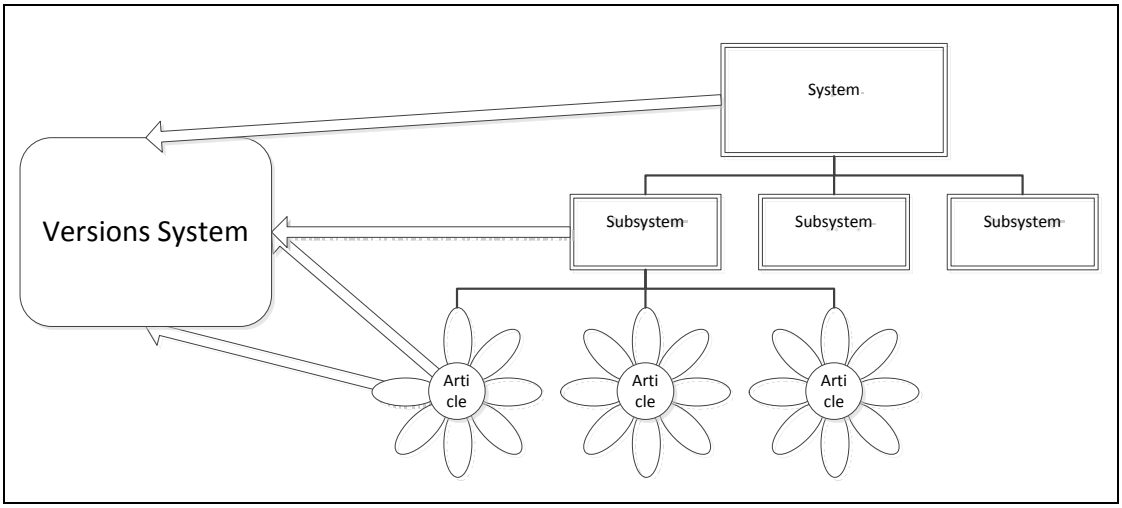


Figure 3.8 3 niveaux de structuration pour les données multidisciplinaire : CI, Article et Nomenclature

Les paragraphes précédents ont permis de détailler la notion de *Configuration Item* et de rappeler quelques concepts, qui, rassemblés, permettent de proposer une structuration simplifiée pour les données multidisciplinaires. Ces deux notions, CI et structure pour les données multidisciplinaires, permettent d'organiser les données HW et SW grâce à un vocabulaire et des concepts unifiés. Elles sont les deux seuls éléments nécessaires à l'explication du mode d'organisation et de fonctionnement des *workspaces* caractérisés dans la section ci-après.

3.2.1.3 Organisation des workspaces et principes d'échanges

Afin de détailler le fonctionnement des *workspaces* et les principes permettant l'échange de données entre ces derniers, les paragraphes suivants détaillent les opérations réalisables de façon chronologique. Ainsi, les étapes de création, d'attachement des données, de création/modification/correction/suppression de données, de synchronisation, de promotion, de collecte, de publication et d'import sont présentées ci-après.

La création de workspace, basée sur une structuration arborescente

Les *workspaces* sont des espaces de collaboration structurés de façon arborescente. La Figure 3.9 illustre cette structure arborescente. Sur cette figure, le WS 1 est ainsi le *workspace* père, qui contient l'ensemble des CI nécessaires à un produit ou à un projet donné. Ces données ne sont pas à proprement parler gérées dans ce WS, elles ne sont que rattachées à celui-ci, la persistance des données étant effectivement assurée par un ou plusieurs systèmes de gestion de données. Chacun des WSs de premier niveau (intitulé WS 1.X) référence tout ou partie des données situées dans le WS père. L'organisation des WSs peut ainsi répondre à des besoins de structuration hiérarchique, disciplinaire ou encore être liée à la division organique du produit à concevoir.

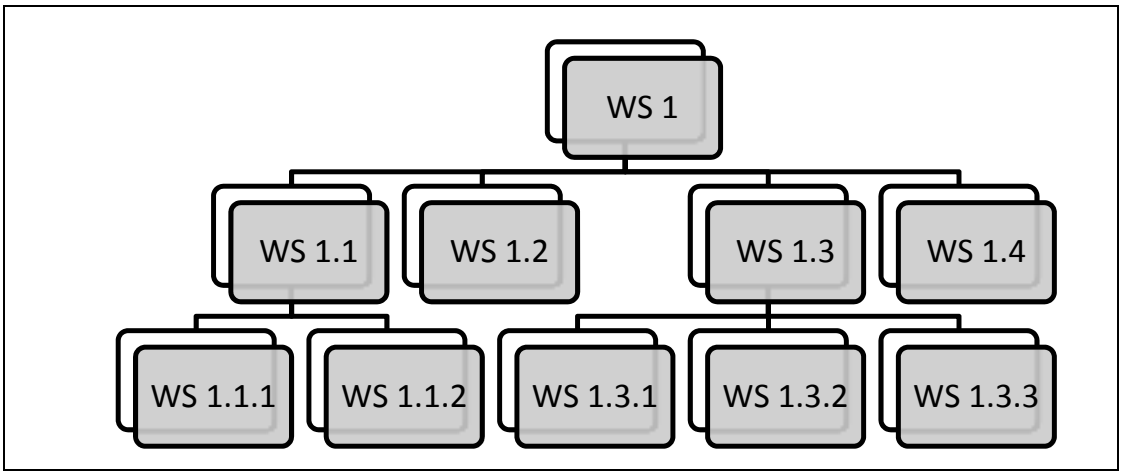


Figure 3.9 La structuration arborescente des espaces de collaboration

Lorsqu'un concepteur souhaite créer un *workspace*, il n'a donc besoin de connaître que le nom ou identifiant du WS auquel il souhaite se rattacher. Ainsi, sur la Figure 3.9, la création du « WS 1.1.2 » ne nécessite que la connaissance du nom du « WS 1.1 ». À sa création, un *workspace* est vide et le concepteur doit donc venir spécifier les données qu'il souhaite venir y rattacher. Le paragraphe suivant explique les principes du rattachement de ces données.

L'attachement des données

Nous avons décrit ci-avant les principes de structuration des données multidisciplinaires envisagés. Dans ce cadre, la nomenclature peut servir de référence afin de demander l'attachement des portions de l'arbre représentant la nomenclature sur lesquelles le concepteur souhaite travailler. Cet attachement correspond à la fois à la copie locale, i. e. sur la machine du concepteur, des données, mais également à la déclaration au système gestionnaire des *workspaces* de la présence de ces données en local qui seront par la suite régulièrement analysées afin de détecter toute modification.

Plus généralement, et quelle que soit la façon de structurer ces données, le système gestionnaire des *workspaces* doit être en mesure de connaître les groupements logiques de données réalisés afin de pouvoir référencer ces grappes de données, facilitant ainsi la sélection des données devant être présentes dans le WS. Mais les liens existants entre les différents CI ne sont pas portés par les WS et seuls le ou les systèmes de gestion de données connaissent ces liens.

Une fois les données présentes en local, le concepteur est en mesure de réaliser des créations de nouvelles données ou des modifications/corrections/suppressions des données présentes. Les WS deviennent alors des espaces de travail où le concepteur attache les données qu'il souhaite manipuler sans interférer avec le travail de ses autres collaborateurs. Il devient un espace d'expérimentation privé jusqu'à ce que le concepteur décide de mettre son travail à disposition de ses collègues.

Les créations, suppressions, modifications et corrections

Comme présenté dans le paragraphe précédent, le processus d'attachement des données copie les données en local. Lorsqu'un concepteur souhaite créer des données, il ne souhaite pas toujours les déclarer immédiatement dans le système. Il préfère généralement les créer en local afin de valider son idée ou concept. Ensuite, afin de déclarer la création de ces données aux niveaux des systèmes de gestion de données et de gestion des *workspaces*, il peut demander la déclaration unitaire de chacune des données qu'il vient de créer, ou demander une comparaison de l'image locale de son *workspace* avec la représentation connue par le système de gestion des *workspaces*. Cette comparaison liste les différences et propose la déclaration des fichiers nouvellement créés.

Le processus pour les fichiers déjà déclarés peut être différent. Les fichiers étant déjà connus dans le système, on voudra informer le système qu'une modification/correction va avoir lieu ou qu'une suppression est demandée. Dans le cas d'une modification/correction, le principe bien connu du check-out semble alors pertinent. Mais alors que le check-out correspond à une réservation dans les PDM actuels, empêchant les autres concepteurs de modifier la même donnée de manière synchrone, il est considéré ici comme un simple avertissement, n'empêchant en rien les autres concepteurs de réaliser cette même opération.

La synchronisation

Durant la réalisation du travail du concepteur, matérialisé par des créations, modifications, corrections ou encore suppressions, l'environnement dans lequel il travaille est susceptible de changer. D'autres modifications peuvent être réalisées par d'autres concepteurs. Un des intérêts du WS est de se prémunir temporairement contre ces modifications en s'isolant, permettant de conserver un cadre stable pour réaliser l'action en cours. Mais avant de promouvoir son travail, le concepteur doit s'assurer que celui-ci est bien compatible avec l'ensemble des modifications réalisées par ailleurs. Cette étape est appelée l'étape de synchronisation. Elle peut être demandée par le concepteur à tout moment et correspond à la diffusion des nouvelles versions ou des CI créés à partir du WS père. La Figure 3.10 illustre la descente des données actualisées du WS 1.1 vers le WS 1.1.1 suite à la demande de synchronisation de celui-ci.

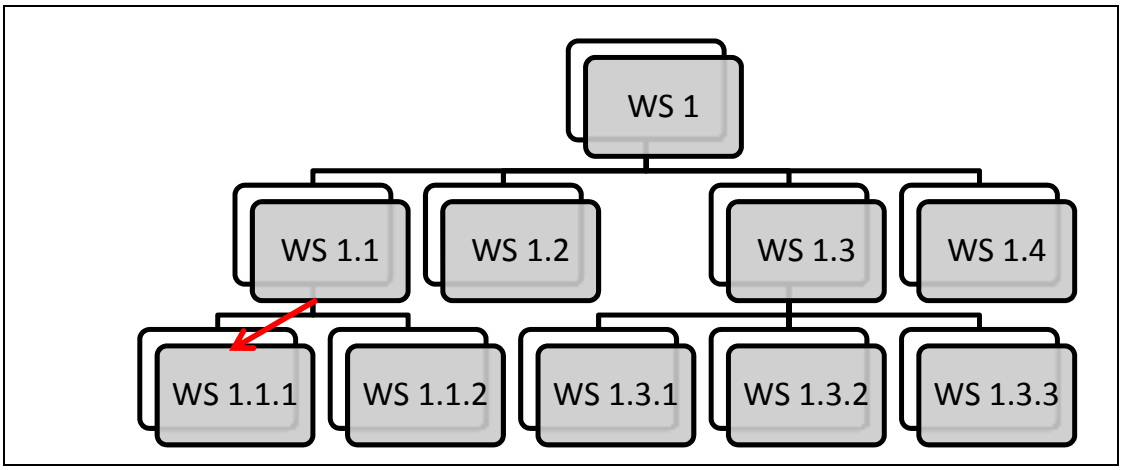


Figure 3.10 Synchronisation d'un *workspace*

La promotion

Lorsque le travail demandé est considéré comme achevé par le concepteur et qu'il a vérifié que celui-ci s'intégrait bien à l'environnement dans lequel il évolue grâce à la synchronisation, il a la possibilité de proposer les créations, suppressions, modifications et corrections réalisé dans son WS au WS père. Le concepteur peut alors choisir de proposer l'ensemble de son travail, ou sélectionner les CI qu'il souhaite proposer. Cette proposition est appelée l'étape de promotion. Elle n'est autorisée que si le WS fils est synchronisé par rapport au WS père et est illustrée sur la Figure 3.11. Il est à noter que, dans certains cas, le WS père peut être fermé et refuser toute demande de promotion. Cela peut s'avérer nécessaire dans certains cas afin de préparer, par exemple, une revue de projet.

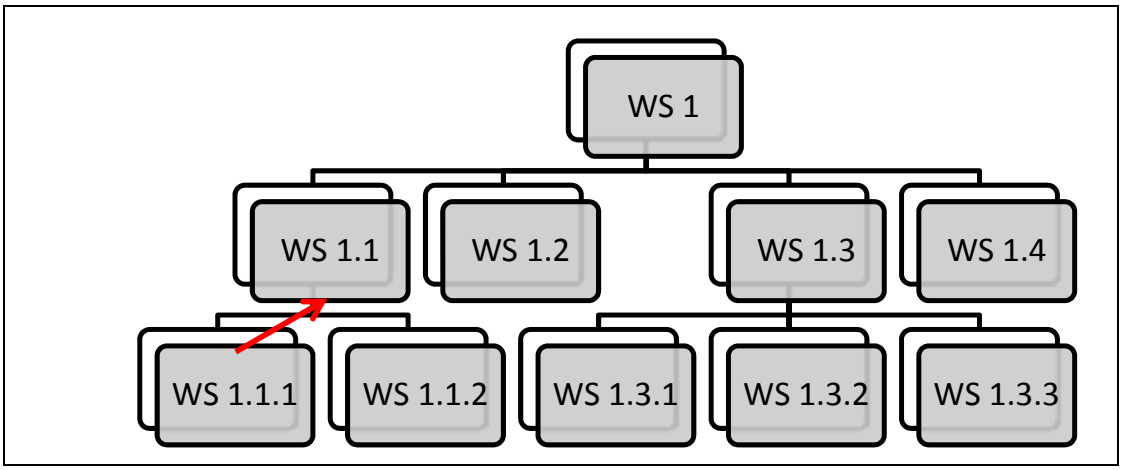


Figure 3.11 Promotion d'un *workspace*

Cette étape de promotion est essentielle dans le cadre de notre proposition car c'est à ce moment que la traçabilité entre les prises de décision, représentées par les actions collaboratives, et les corrections/modifications apportées sur les données techniques est assurée. Lorsque le concepteur promeut son travail, il précise le ou les identifiants des actions concernées. Le fait de renseigner plusieurs actions rend ce lien de traçabilité moins précis, mais il permet parfois de répondre à plusieurs demandes d'action corrélées. Les actions collaboratives recueillent alors l'historique des créations, corrections, modifications et suppressions réalisées afin de répondre aux demandes d'action initiales. Mais la promotion ne doit être considérée que comme une proposition de modification et seule la collecte, présentée dans le paragraphe suivant, par le WS père ne peut être vue comme une prise en considération du travail.

La collecte

À intervalles réguliers prédéfinis, les responsables des WS d'équipe, de département etc. listent les demandes de promotion qui ont été effectuées dans leur WS. Pour chacune des demandes de promotion, les numéros des Actions Collaboratives sont renseignés ce qui lui permet de comprendre l'origine de la demande, sa criticité et son degré d'urgence. Grâce à ces informations, il est en mesure de décider s'il souhaite prendre en compte la demande de promotion et collecter le travail du concepteur. Ainsi, par exemple, lors d'une préparation de revue de projet, seules les demandes d'action portant l'annotation « obligatoire » pourraient être acceptées alors que le reste du temps, toutes les demandes de promotion sont collectées. La Figure 3.12 représente la prise en considération des demandes de promotion via le mécanisme de collecte.

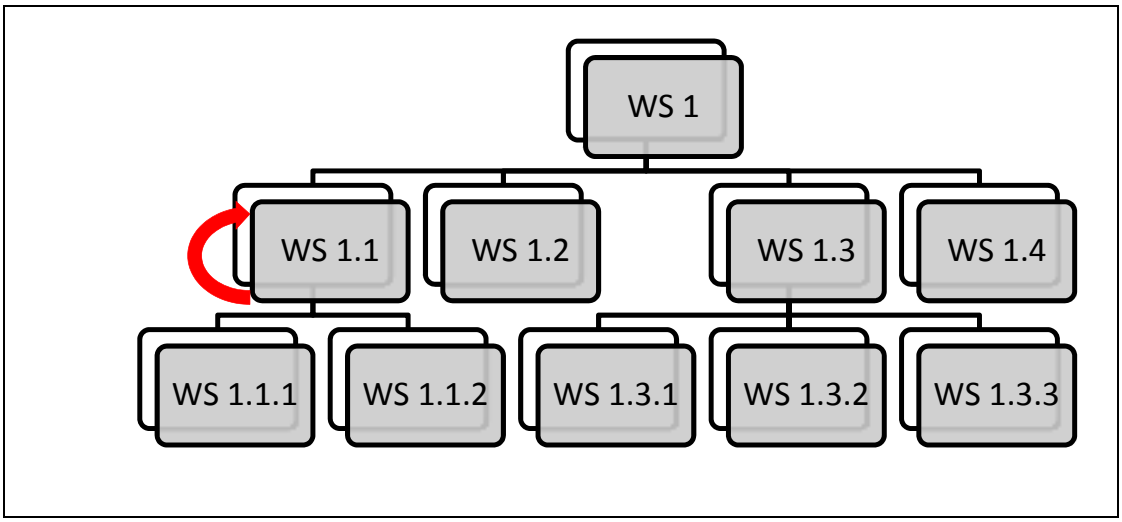


Figure 3.12 Collecte d'un *workspace*

Bien entendu, ces mécanismes de promotion/collecte sont similaires à tous les niveaux de l'arborescence. Ils peuvent être considérés comme des mécanismes de validations successives permettant de contrôler les modifications réalisées. Mais ces mécanismes de filtrage peuvent également avoir un effet pervers qui est la lenteur de la propagation des modifications. Par exemple (Figure 3.13), si une action collaborative nécessite l'intervention successive de différentes équipes travaillant respectivement dans le WS 1.1 et le WS 1.3, les opérations suivantes devront être réalisées :

1) demande de promotion par le concepteur de la première équipe via le WS 1.1.1,
2) collecte réalisée par le WS 1.1,
3) demande de promotion par le responsable de la première équipe via le WS 1.1,
4) collecte réalisée par le WS 1,
5) synchronisation réalisée par le responsable de la seconde équipe via le WS 1.3,
6) synchronisation réalisée par le concepteur de la seconde équipe via le WS 1.3.1.

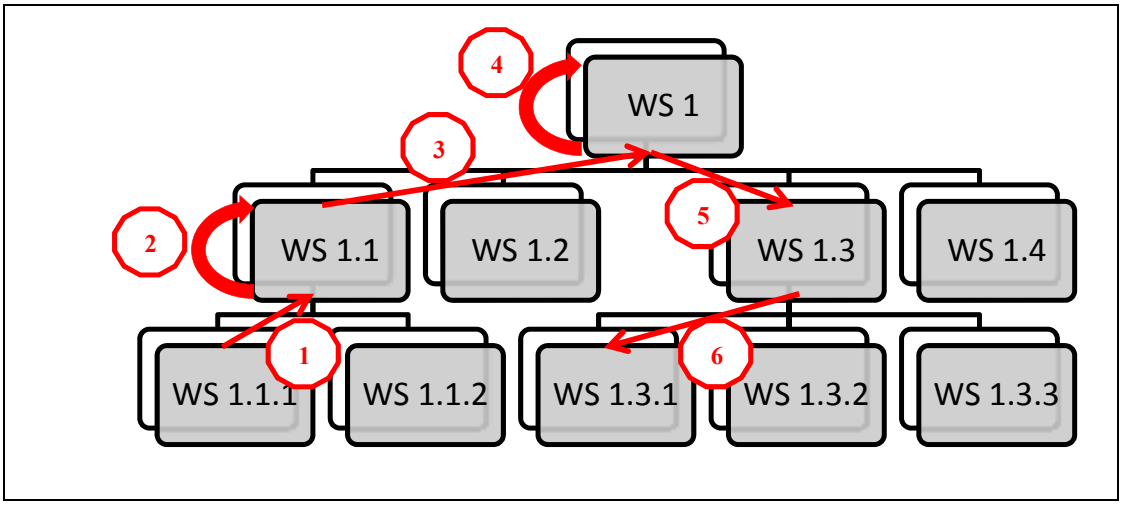


Figure 3.13 Illustration du processus de propagation du changement entre deux équipes

D'autres mécanismes d'échange plus rapides basés sur les publications et les imports sont également possibles. Ils sont présentés dans le paragraphe suivant.

Publication et import

Comme précisé dans la définition des WS proposée ci-avant, le travail d'un concepteur reste privé le temps que ce dernier ne décide pas de le mettre à disposition de ses collaborateurs. Cette mise à disposition est automatique lorsque celui-ci promeut son travail, mais il peut également décider d'effectuer cette opération qui revient à exposer ses modifications sans pour autant considérer que son travail est achevé. Cette exposition est appelée publication et est représentée sur la Figure 3.14. Le concepteur peut alors exposer tout son travail ou choisir les CI qu'il souhaite publier.

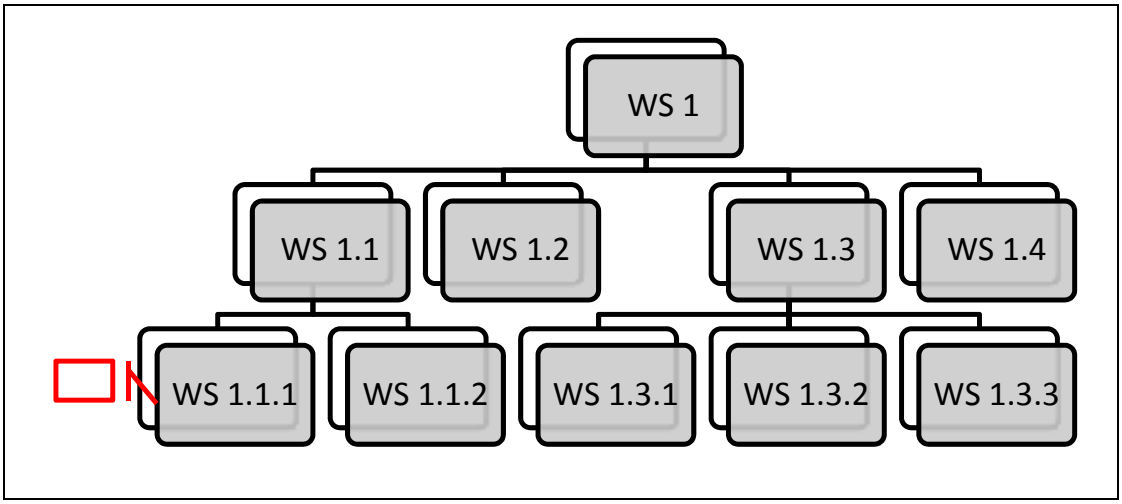


Figure 3.14 Publication réalisée par le *workspace* 1.1.1

Lorsque cette opération de publication est réalisée, le concepteur de la seconde équipe possédant le WS 1.3.1 peut alors demander l'import des modifications, court-circuitant ainsi l'ensemble du processus de diffusion (Figure 3.15). Il injecte ainsi les modifications publiques du WS 1.1.1 de son choix dans son propre WS.

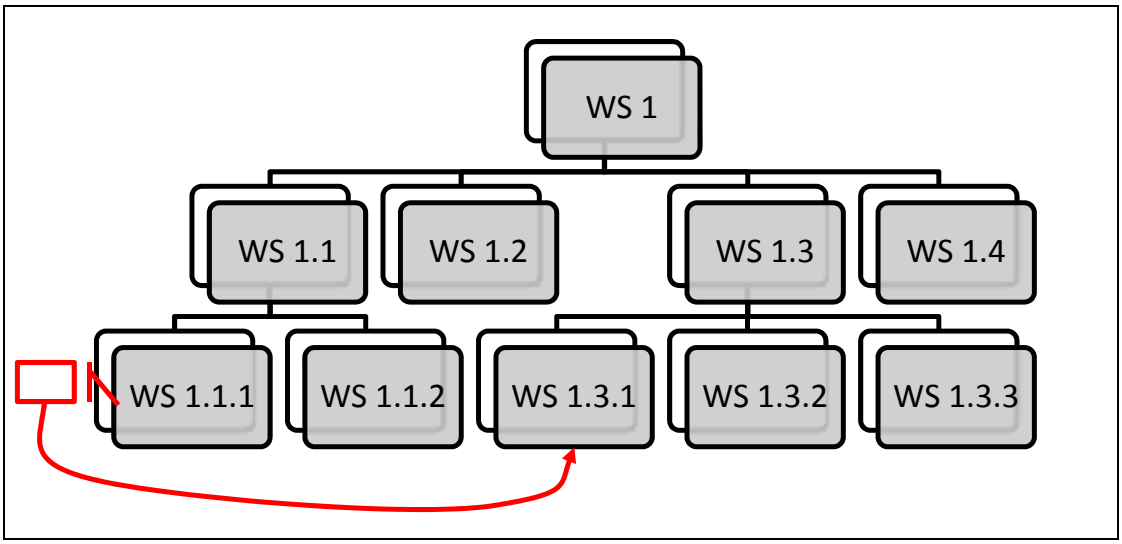


Figure 3.15 Import réalisée par le *workspace* 1.3.1

Cette opération d'import peut être utile lors du transfert d'une action collaborative nécessitant le travail successif de plusieurs équipes. Lors du transfert, le premier concepteur précise alors le *workspace* contenants les CI publiés à importer.

Autre mécanisme offert par les workspaces

Une autre possibilité est offerte par les *workspaces*. Bien qu'elle ne concerne pas la traçabilité produit/décisions ou encore l'intégration multidisciplinaire, elle mérite d'être mentionnée. Cette possibilité correspond au fait d'offrir des vues particulières sur les différents WS. Avec le mécanisme de structuration des données proposé, si une portion de nomenclature est sélectionnée pour être attachée à un *workspace*, l'ensemble des données relatives aux composants présents dans cette portion est censé être rapatrié localement sur la machine du concepteur. La notion de vue propose alors de venir filtrer ces données par activités d'ingénierie et par disciplines en appliquant d'éventuels filtres selon les types de données choisies. On peut ainsi imaginer qu'un responsable industrialisation électronique ne sélectionne en priorité que les fichiers ayant trait au

routage des circuits électroniques et aux références des composants implantés. Un même *workspace* peut ainsi posséder plusieurs vues, signifiant plusieurs projections du WS filtrées selon les choix de l'utilisateur.

Les paragraphes précédents ont permis d'expliciter l'organisation arborescente des *workspaces*, les différents modes d'échange de données entre ces derniers, ainsi que les significations inhérentes à ces échanges. En revenant sur les points de synthèse retenus lors de la présentation des principes sous-jacents aux méthodes agiles, la section suivante détaille les intérêts liés à l'usage des *workspaces* dans un contexte de conception collaborative multidisciplinaire.

3.2.2 Le rôle des workspaces pour la mise en œuvre des principes des méthodes agiles

Dans chacun des paragraphes ci-dessous, le rôle joué par les *workspaces* pour supporter la conception collaborative multidisciplinaire est mis en avant. Parmi les avantages présentés, la traçabilité entre décisions et impact sur les données est détaillée en premier lieu pour exposer le lien avec le *Collaborative Actions Framework*.

3.2.2.1 La traçabilité, ou le lien Produit/Organisation

Comme présenté dans les sections précédentes, la promotion des créations, suppressions, modifications ou corrections par un concepteur dans le WS père est le moment auquel il est possible de préciser le numéro de l'action collaborative pour laquelle ce travail a été réalisé. L'ensemble des CI modifiés sont alors référencés dans l'action collaborative et la clôture de cette dernière peut avoir lieu. La validation de la clôture ne peut intervenir que lorsque les modifications apportées rejoignent le WS de plus haut niveau, qui est généralement celui utilisé pour la réalisation des vérifications nécessaires à la clôture.

Dès lors, une double navigation est possible : à partir de l'action collaborative, il est possible de lister l'ensemble des systèmes, sous-systèmes ou composants impactés. De la même manière, entre deux versions données d'un CI, il est possible de retracer l'ensemble des actions collaboratives ayant contribué à sa modification. Ce lien de traçabilité est donc assez similaire au lien qui existe entre les Engineering Change Order et la liste des documents liés à cet ordre, mais il intervient pour l'ensemble des demandes d'actions collaboratives.

De la même manière que ce besoin d'analyse ponctuel, des indicateurs portants sur les composants modifiés par des concepteurs issus de disciplines différentes, sur le nombre d'imports de données entre équipes de disciplines différentes etc. peuvent être exploités pour mesurer l'évolution du niveau d'intégration entre les équipes. Ces indicateurs contribuent donc également au point de synthèse numéro trois relatif à la nécessité de pouvoir bénéficier d'indicateurs opérationnels.

3.2.2.2 La mise en commun permanente des travaux

La temporalité et la fréquence des échanges entre les concepteurs des différentes disciplines est un élément essentiel de l'intégration multidisciplinaire. En effet, les pratiques semblent, à ce jour, très différentes selon les disciplines. Par exemple, alors que dans le domaine du logiciel les systèmes sont prévus pour prendre en considération jusqu'à plusieurs dizaines de modifications par jour du même fichier, dans des domaines *hardware*, les données sont parfois conservées en local sur la machine du concepteur jusque quelques jours ou heures avant le jalon officiel nécessitant le partage des données. Ce genre de pratiques ne favorise pas l'intégration multidisciplinaire, qui n'intervient ainsi que lors de mises en communs imposées.

Avec l'usage des *workspaces*, le simple fait de travailler dans le contexte du *workspace* permet au système de gestion de données de connaître beaucoup plus régulièrement les modifications en cours de réalisation. Si les actions collaboratives sont suffisamment détaillées, les promotions sont fréquentes et les synchronisations permettent ainsi aux concepteurs de recevoir régulièrement les dernières versions des différents CI modifiés. Les éventuels problèmes d'intégration peuvent ainsi être anticipés. Si une action collaborative nécessite un partage de données, l'unicité du système de référencement qu'est le gestionnaire de *workspaces* permet de faciliter cet échange, notamment grâce aux mécanismes de publication et d'import.

3.2.2.3 L'intégration multidisciplinaire

Si différentes tentatives visant à gérer sur une unique plateforme des données fortement hétérogènes à cause de leurs disciplines d'appartenance respectives ont vu le jour (El-Khoury et al., 2005; Kääriäinen et al., 2000), les spécificités liées aux pratiques et aux outils de chacune des disciplines semblent des obstacles difficiles à franchir. En revanche, le fait de pouvoir accéder aux données issues de ces différentes disciplines via une nomenclature commune et via un système de *workspace* « neutre », *i.e.* non dépendant d'un système de gestion de données issu d'un domaine particulier, semble prometteur. Ce système permet alors de présenter l'ensemble des données de façon unifiée via un unique moyen d'accès.

La Figure 3.16 rappelle une des initiatives de plateforme commune venant fédérer les contributions HW et SW autour d'un méta-modèle commun (El-Khoury, 2006), alors que la Figure 3.17 présente ce à quoi pourrait ressembler une nomenclature unique pour des composants électroniques et logiciels. Ces deux figures illustrent et appuient la nécessaire gestion unifiée des données HW et SW, bien qu'elles s'appuient sur des architectures et des logiques différentes. La Figure 3.16 propose ainsi une base unique alors que la Figure 3.17 fournit une solution où les données restent gérées par leurs systèmes d'origine, mais apparaissent dans une nomenclature unique.

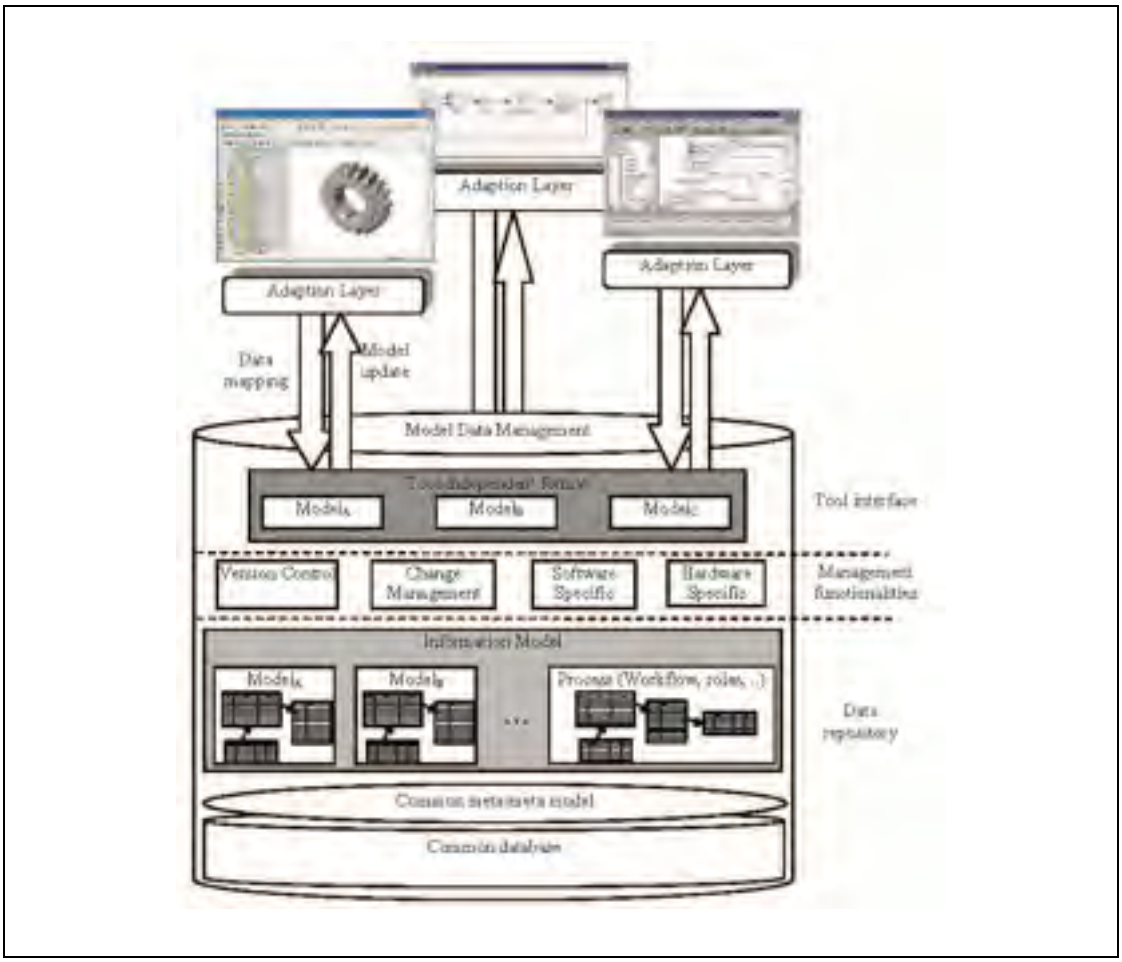


Figure 3.16 Plateforme MDM (*Model Data Management*) pour une gestion unifiée des données HW et SW (El-Khoury, 2006)

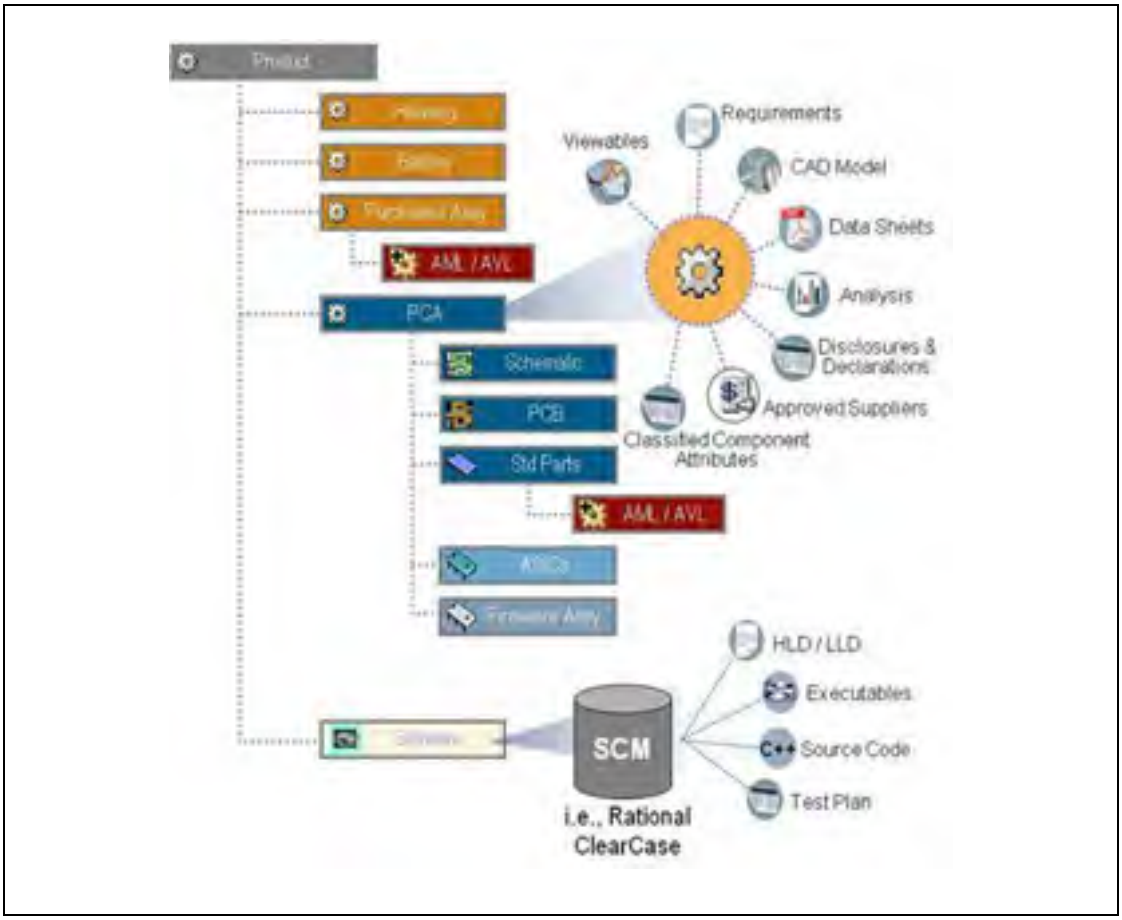


Figure 3.17 Nomenclature unique pour la gestion de données issues de différentes disciplines[14]

Le second aspect permettant d'affirmer que l'intégration multidisciplinaire peut être favorisée par l'usage des *workspaces* tient à la notion même d'intégration au plus tôt. En effet, selon la structuration des *workspaces* choisie, des concepteurs issus de disciplines différentes peuvent promouvoir leur travail dans les mêmes WS d'intégration. Ces WS intermédiaires comme les WS 1.1 et WS 1.3 sur la Figure 3.9 peuvent alors être vus comme des espaces permettant de tester, voire de valider que les contributions issues des différentes disciplines sont bien compatibles et qu'elles atteignent le niveau d'intégration recherché.

[14] Image recueillie sur le site http://fr.ptc.com/

Les WS intermédiaires viennent d'être présentés comme des espaces privilégiés permettant d'améliorer le niveau global d'intégration grâce à un partage multidisciplinaire au plus tôt. Mais ces WS intermédiaires peuvent également contribuer à valider la qualité des contributions par la mise en place de tests, les opérations de promotion et de collecte pouvant alors être considérées d'une certaine manière comme un processus de validation. C'est l'objet du paragraphe suivant.

3.2.2.4 La promotion et la collecte comme processus de validation

Le dernier point d'intérêt relatif aux *workspaces* concerne l'aspect lié à la validation lors des opérations de promotion et de collecte. Pour certaines disciplines, des tests automatiques sont réalisables afin de valider que les modifications/corrections proposées n'introduisent pas de perturbation ou de dysfonctionnement dans le comportement du système. C'est le cas notamment dans le domaine du logiciel avec des tests dits unitaires, de scénario ou de performance. Mais dans d'autres disciplines, la notion même de test automatique ne semble pas envisagée. Pourtant, en mécanique par exemple, pour la préparation de revues de projet, des analyses d'interférence sont par exemple effectuées. Dans certains domaines spécifiques, la compatibilité des matériaux est aussi vérifiée et certains PDM imposent la vérification des modèles CAO par des outils comme Q-Checker[15] afin de vérifier la qualité des modèles. Tous ces tests sont généralement lancés manuellement par le concepteur. Certains de ces tests pourraient être automatisés au niveau des WS intermédiaires afin de valider la qualité des contributions et leur adéquation avec les standards de l'entreprise ; une analyse plus fine des tests réalisables selon les disciplines et les secteurs semblant bien entendu nécessaire. Sur la base des résultats de ces tests, les WS intermédiaires pourraient alors être autorisés à promouvoir dans le WS supérieur. À chaque remontée de WS, des tests de plus en plus globaux pourraient être effectués permettant la validation progressive des données. Le WS de plus haut niveau pourrait alors être considéré comme l'emplacement contenant les données les plus intégrées et validées.

---

[15] http://www.transcat-plm.com/en/software/transcat-software/q-checker.html

3.2.3 Synthèse : les workspaces supports des méthodes agiles

Comme évoqué dès l'introduction de ce chapitre, les trois concepts formant la proposition ne se positionnent pas aux mêmes niveaux. Alors que le *Collaborative Actions Framework* traite de l'organisation de la collaboration et du pilotage agile des activités de conception, les *workspaces* se focalisent plus sur les mécanismes d'échange de données. La section ci-avant nous a permis de découvrir en quoi les *workspaces* contribuent à mettre en œuvre une traçabilité de type données/décisions, comment ils permettent de bénéficier de nouveaux indicateurs bruts permettant d'éclairer les prises de décisions, mais surtout en quoi ils permettent le partage régulier et multidisciplinaire des contributions. La Figure 3.18 résume donc le positionnement relatif aux trois points de synthèse des méthodes agiles. Les *workspaces* se positionnent tout particulièrement comme contributeurs pour le partage régulier des données de conception, mais sont également présents sur les axes liés aux indicateurs opérationnels (pour la traçabilité données/décision) et aux initiatives des acteurs opérationnels (pour les publications/imports).

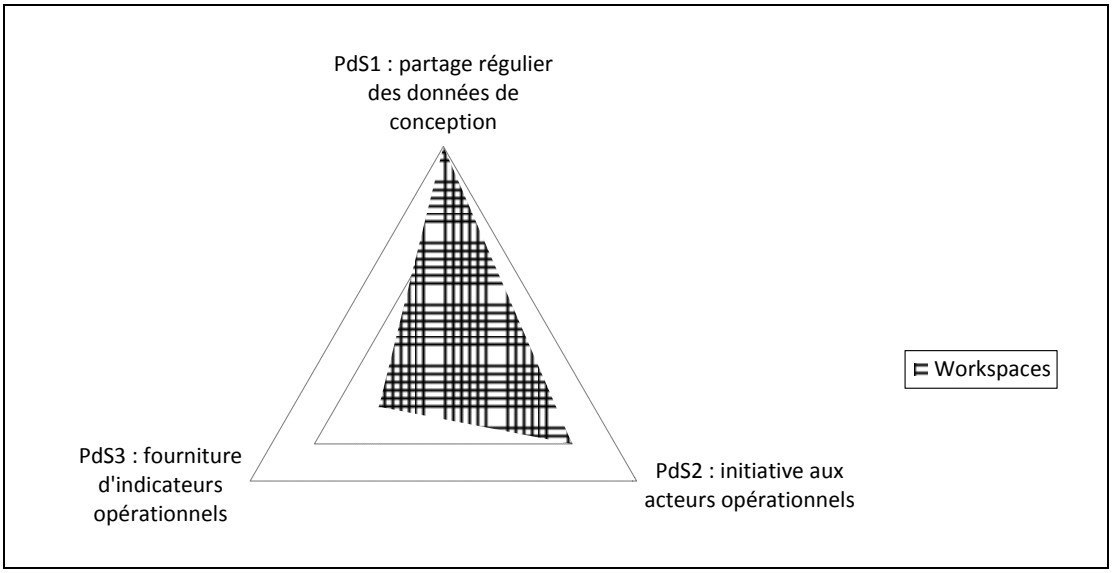


Figure 3.18 Positionnement du concept nommé *Workspace* par rapport aux points de synthèse des méthodes agiles

Lors de ces explications, il a été mentionné que les concepteurs ne sont, grâce à ces mécanismes d'échange, plus obligés de réserver les données sur lesquelles ils travaillent. Le check-out y est considéré comme un simple avertissement, n'empêchant en rien les autres concepteurs de réaliser d'autres opérations sur les mêmes données. Cette liberté conduit à la création de deux versions concurrentes pour un même CI, deux branches dans le graphe de gestion des versions qu'il peut être nécessaire de réconcilier. Cette opération de réconciliation est appelée fusion ou *merge*. La section suivante décrit et illustre les opportunités offertes par ces mécanismes de *branch* & *merge*.

Ce troisième concept formant la proposition peut sembler a priori plus anecdotique quant à la problématique de nos travaux, mais il est à la fois nécessaire pour le fonctionnement envisagé des WS et apporte également une grande souplesse quant aux modifications/corrections.

3.3 Branch & Merge

Cette section reprend les principales idées de l'article intitulé « Concurrent versioning principles for collaboration: towards PLM for hardware and software data management » paru dans le journal « Int. J. Product Lifecycle Management » (Bricogne et al., 2014).

Afin de bien saisir les enjeux liés à ce concept de *branch* & *merge*, il est nécessaire de comprendre les logiques sous-jacentes liées à la gestion du changement dans les disciplines orientées HW et aux mécanismes de contrôle des versions dans les disciplines orientées SW. Les enjeux liés à ces deux activités sont en premier lieu, avant de décrire les outils supports aux concepts de création et de fusion de branches. Ces outils, nommés « *diff* » et « *3-way merge* » sont présentés, définis et illustrés dans la seconde section. Enfin la troisième section vise à illustrer les opportunités offertes par la généralisation de ce concept de *branch* & *merge* à l'ensemble des activités d'ingénierie et des disciplines. Un exemple emprunté à la mécanique, les données de type CAO, y est choisi pour montrer comment les outils de *diff* et de *3-way merge* peuvent être appliqués à des données considérées jusqu'alors comme exclues de ce type d'approche.

3.3.1 Gestion du changement et mécanismes de contrôle des versions : des logiques différentes entre les domaines HW et SW

Comme nous l'avons vu précédemment, les mécanismes mis en œuvre lors de la gestion du changement dans les disciplines HW sont des processus formels et très structurés pour les produits libérés, comme l'illustre la Figure 2.6 (Jarratt et al., 2010). Ils peuvent être pris en charge par les PDM grâce à la fonctionnalité de *workflow*, où les rôles sont attribués aux différentes parties prenantes de chaque projet de conception. Cette fonctionnalité est basée sur des procédures prédéfinies, avec la pré-affectation des rôles, conduisant à des opérations que nous considérons comme lourdes et rigides.

Les approches de gestion du changement et de contrôle de versions dans la conception de logiciels diffèrent de ceux utilisés dans le domaine du matériel. La responsabilité du changement est plus portée par le développeur, qui est considéré comme l'expert. La procédure de validation est quant à elle basée sur un ensemble de tests manuels ou

automatiques. Plus la couverture des tests fonctionnels est élevée, plus la pertinence des tests est importante et donc plus la décision concernant la modification sera sûre. Ainsi, dans le domaine du logiciel, les étapes “Identification of possible solutions”, “Impacts and risks assessment”, “Selection and approval” and “Implementation” sont moins formelles et laissés à la discrétion du développeur, en fonction de la sévérité et de la complexité.

Même si ces deux descriptions ne sont pas exhaustives quant aux méthodes permettant la gestion du changement, elles permettent de mettre en avant les logiques très différentes pouvant exister entre ces deux domaines. Ainsi, la gestion du changement dans le domaine logiciel est basé sur une validation « a posteriori », alors que la gestion du changement dans les domaines HW est basé sur une validation « a priori », ce qui rend le processus beaucoup plus formel et procédural. La seconde constatation porte sur le fait que la gestion du changement dans les domaines HW est basée sur l'organisation, tandis que la gestion du changement en SW est basée sur une plus grande autonomie des individus. Cette logique s’inscrit donc pleinement avec le point de synthèse des méthodes agiles numéro trois, portant sur le fait de laisser plus d’initiatives aux concepteurs.

Afin de permettre à l’ensemble des concepteurs, quelle que soit leur origine disciplinaire, de pouvoir bénéficier de la même souplesse, de la même flexibilité et de pouvoir prendre plus d’initiatives, la généralisation d’un système de contrôle des versions est envisagée. Basée sur le concept de *branch* and *merge*, qui est décrit dans le paragraphe suivant, ce système est ensuite transposé au domaine de la mécanique afin d’illustrer les opportunités liées à son usage dans cette discipline.

3.3.2 Des outils essentiels pour supporter le branch & merge : le diff et le 3-way merge

3.3.2.1 L'origine du branch & merge : des modifications fréquentes pouvant être réalisées par des développeurs différents

En génie logiciel, les données sont constamment conservées sous le contrôle d'un système de contrôle de versions nommé de façon générale Concurrent Versions System (CVS) et ce quelle que soit la phase de conception en cours. Les données sont susceptibles de changer très régulièrement, générant un grand nombre de versions. Cependant, l'ensemble de ces versions sont stockées, grâce notamment à des techniques différentielles utilisées pour stocker uniquement les parties modifiées. Les versions d'un même logiciel sont également très nombreuses, ce qui implique que différentes versions d'un même CI peuvent être impliquées dans différentes versions d'un logiciel. L'effet de cette évolution rapide est que plusieurs versions peuvent être créées ou modifiées en même temps sans pour autant affecter les autres versions. La nécessité de pouvoir gérer plusieurs « branches », contenant les différentes versions concurrentes des données en parallèle, et de pouvoir fusionner différentes versions, est née de ces spécificités.

3.3.2.2 Description des opérations de création de branche et de fusion de branches

Ces deux opérations de création de branche et de fusion sont illustrées sur la Figure 3.19 (Brosch et al., 2012). Cette figure montre comment deux développeurs peuvent travailler sur le même fichier en même temps (t0) et sur la même version (V0). Plus tard (t1), le premier réintègre sa modification (v1). Le second développeur, après avoir terminé sa modification (t2), veut réintégrer sa nouvelle version (v2). La nécessité du *merge* est détectée (éclair) et le processus de fusion (*merge process*) est lancé pour créer une version qui intègre à la fois les modifications des développeurs 1 et 2.

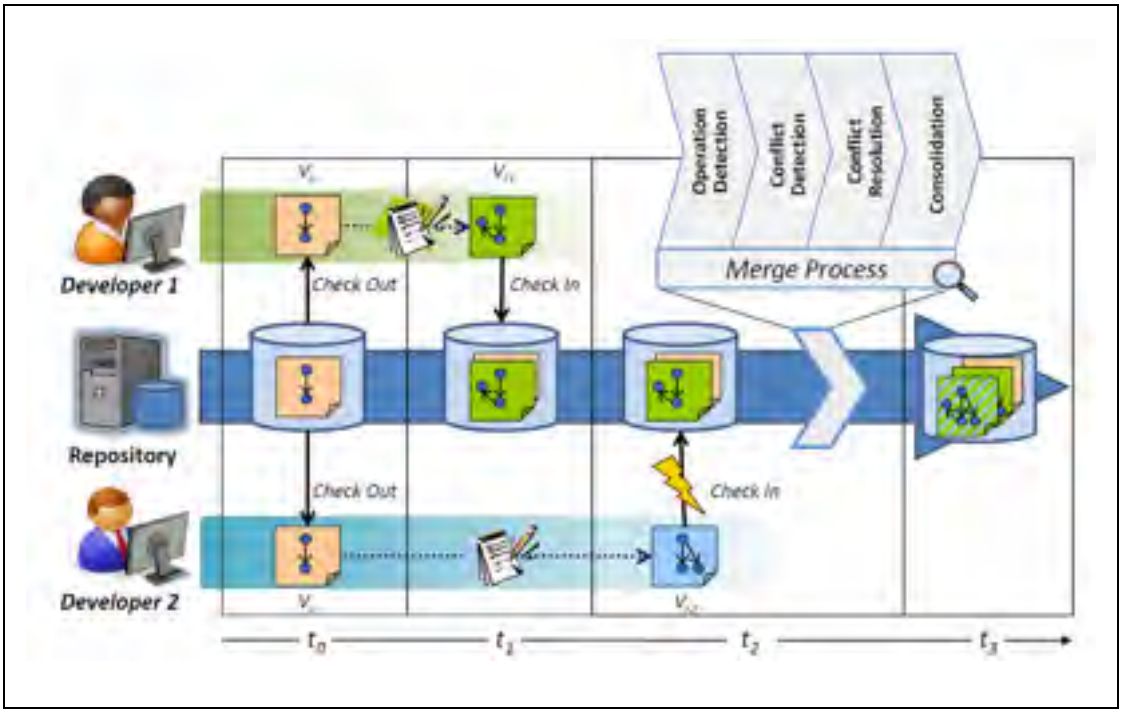


Figure 3.19 Représentation graphique du processus de création de versions concurrentes et de fusion de ces versions (Brosch et al., 2012)

3.3.2.3 Lien entre graphe de versions et intentions de conception

Lorsqu'une branche est créée, elle est généralement liée à une intention de conception spécifique représentée par une action collaborative, qui peut correspondre à la création d'une nouvelle fonctionnalité, la modification d'une fonctionnalité, la correction d'un bug, ou encore une opération de *refactoring* (réingénierie logicielle, consistant à retravailler le code source sans ajouter de fonctionnalités). Au contraire, lorsque deux branches sont fusionnées, la version résultante intègre les intentions de conception provenant de chacune des branches. En conséquence, en fonction de la rigueur avec laquelle le système de contrôle de versions est utilisé, il est possible de suivre les différentes intentions de conception qui ont conduit à la création d'une version spécifique. Cette analyse a déjà été introduite dans la section concernant les WS, mais la description des mécanismes de création et de fusion des branches permet de mieux appréhender le lien entre intentions de conception, représentées par les actions collaboratives, et versions d'un même CI. La Figure 3.20 illustre un graphe de versions pour un même CI, où la version 47 du CI a été créée à partir de la version 43 pour corriger un bug détecté sur la

version 2008 du logiciel. La modification a été reportée dans la 50ème version du CI (a) permettant de corriger le bug dans la version 2010 de ce même logiciel. Cependant, une nouvelle fonctionnalité pour la version 2010 du logiciel qui impacte le CI a été créée, également à partir de la version 43, générant la version 46. Pour éviter une fusion lorsque cette nouvelle fonctionnalité arrivera dans la version 2010 (c), le développeur a décidé de réconcilier les 47ème et 46ème versions à l'aide d'une opération de fusion (b). Cette opération est réalisée par l'intermédiaire d'un outil de fusion appelé « *3-way merge* », présenté ci-après, utilisé avec les caractéristiques suivantes:

- Ancêtre commun : 43ème version
- Première version à intégrer : 46ème version
- Seconde version à intégrer : 47ème version
- Version résultante de l'opération de *merge* : 49ème version

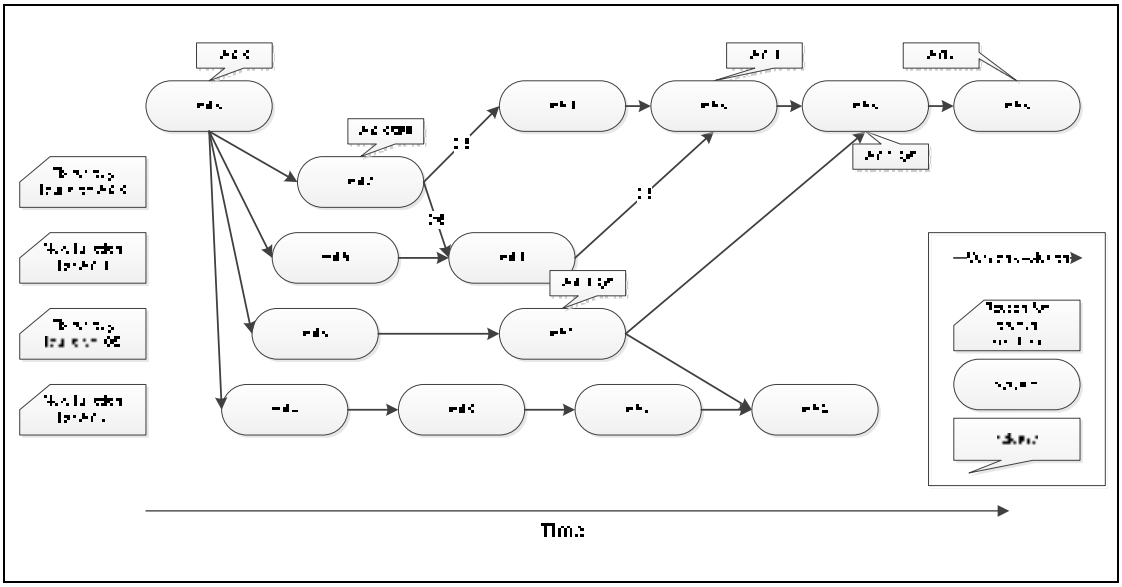


Figure 3.20 Exemple de graphe de versions pour un même CI (Bricogne et al., 2014)

Cet exemple illustre comment les systèmes de contrôle de versions permettent à plusieurs concepteurs de modifier le même fichier de façon synchrone, grâce à la notion de création de branche. Dans cet exemple, les numéros des versions sont donnés de façon séquentielle, mais différentes stratégies peuvent être choisies pour la numérotation, les opérations de fusion, etc.

3.3.2.4 L'apport des outils de « diff » et de « merge »

La capacité à travailler de manière synchrone avec plusieurs versions d'un même CI est principalement fournie par les outils nommés « *diff* » et « *merge* ». Un outil de type « *diff* » calcule les différences entre deux versions, tandis que l'outil de type « *merge* » concilie les différentes versions d'un même CI.

Dans le domaine HW, la plupart des données sont des données binaires, une situation qui a toujours été présentée comme un obstacle à l'adoption de mécanismes de contrôle des versions concurrentes. Cependant, depuis la création de la programmation orientée objet, de l'ingénierie dirigée par les modèles (MDE) (Bézivin, 2005) et d'autres techniques de modélisation, le domaine SW a été confronté aux mêmes problèmes que les domaines HW et les techniques récentes proposées dans le domaine logiciel peuvent être utilisées pour aider la collaboration entre concepteurs issus de disciplines différentes.

L'outil « *diff* » peut être défini comme un système qui permet de comparer deux représentations d'un artefact de même nature et qui présente le résultat de cette comparaison à l'utilisateur d'une manière intelligible. L'outil « *merge* » peut être défini comme un système qui permet de comparer et de réconcilier deux représentations d'un artefact de même nature et qui présente les informations appropriées à l'utilisateur en charge de la fusion. Dans les définitions précédentes, les termes « les représentations de même nature » signifient qu'il n'y a pas besoin d'avoir deux fichiers d'un même type, afin d'effectuer une comparaison ou une fusion. Bien entendu, ceci peut impliquer une complexité supplémentaire, mais elle n'est pas rédhibitoire. Pour effectuer une comparaison / fusion de deux représentations, plusieurs outils de type *diff* / *merge* basés sur des techniques différentes peuvent être fournis. Par exemple, pour la comparaison d'images, une comparaison pixel par pixel peut être réalisée. Mais, si les images comparées sont en fait des plans d'ingénierie ne contenant que des pixels noirs et blancs (ou en niveaux de gris), la comparaison peut être personnalisée afin de reconnaître des entités géométriques pour assister l'utilisateur de l'outil dans sa décision.

Lorsque l'on compare ou fusionne deux versions, la version correspond à l'ancêtre commun (43ème version sur la Figure 3.20) peut être prise en compte. Dans ce cas, le terme de 3-way *diff* / *merge* est utilisé, contrairement à la fusion classique « à 2 voies ». La prise en compte de cette version parente commune apporte beaucoup de fiabilité lors de l'opération de différenciation / fusion. Cet avantage s'explique par le fait que l'outil examine les différences entre les deux versions modifiées, ainsi que les différences entre chaque version et leur parent, afin de détecter des intentions de conception des deux concepteurs qui ont effectué les modifications. Ces modifications peuvent alors être qualifiées de « objet ajouté », « objet supprimé » ou « objet déplacé » et une version résultante peut alors être calculée et proposée à l'utilisateur en charge de la fusion. Ce processus reste donc bien manuel, mais il peut être grandement facilité par ce type d'outil. Bien sûr, une fusion à 3 voies n'est possible que si toutes les versions sont conservées sous le contrôle du gestionnaire de versions, qui conserve en permanence la connaissance de l'ancêtre commun.

Une fusion à 3 voies débute généralement par le choix de la configuration la plus favorable pour effectuer la réconciliation. Le terme configuration fait référence ici à la façon dont les données sont organisées dans l'interface utilisateur graphique (GUI). L'organisation par défaut est généralement d'avoir l'ancêtre du fichier au milieu et les deux versions à réconcilier de part et d'autre. Cette organisation permet l'incorporation des éléments modifiés pertinents, ce qui permet de les inclure dans le fichier de résultat. Cependant, parfois, il peut être intéressant de choisir une organisation différente: si une version introduit des changements majeurs, le fait de choisir cette version comme la version de référence et d'insérer les changements liés à la deuxième version peut se révéler être une bonne option. Le choix de l'organisation correspond en fait à la perception de l'utilisateur qui doit répondre à la question suivante: « est-ce plus facile et/ou plus sûr d'incorporer les modifications apportées par l'autre concepteur dans ma propre version, ou d'intégrer mes propres changements dans sa version? »

Les outils de type *diff* et *merge* sont souvent considérés comme des comparateurs de texte utilisables uniquement par les développeurs de logiciels. Le but de ce paragraphe est de préciser ce que ces outils sont réellement, de montrer leur apport et d'ouvrir les horizons pour la gestion des modifications/corrections techniques. Dans la section suivante, l'utilisation de ces outils dans le domaine mécanique sera envisagée afin de montrer comment leur usage pourrait être généralisé à l’ensemble des disciplines techniques.

3.3.3 Généralisation du branch & merge à l'ensemble des disciplines

3.3.3.1 Illustration du branch & merge appliqués au domaine de la mécanique

Une des possibilités offertes par les systèmes de contrôle des versions est de modifier la même donnée de façon synchrone, grâce à la notion de création de branche. Après que le premier concepteur ait réintégré sa version dans le système de contrôle, le système propose un processus outillé pour assister le second concepteur à réconcilier sa version avec celle du premier concepteur, tout en préservant le travail de ce dernier. Cette deuxième opération, loin d'être anodine, basée sur le concept de fusion, est présentée comme une opportunité pour l'ensemble des disciplines ne bénéficiant pas déjà de ce genre de mécanismes. Par exemple, le fait que deux concepteurs puissent travailler de façon synchrone sur la même pièce CAO nous semble représentatif du type de bénéfice qu'il est possible de tirer de ce type d'opportunité. Ainsi, sur des pièces nécessitant un très long temps de conception/modélisation comme un « wing plank », illustré sur la Figure 3.21, les contributions parallèles de plusieurs concepteurs permettraient de réduire drastiquement le temps de modélisation.

Figure 3.21 Wing Plank : pièce aéronautique de grande taille et présentant nombre de détails (Niu, 1999)

Pour illustrer comment un outil de type « *3-way merge* » peut être utilisé pour un modèle de type CAO, un modèle simple est proposé, composé d'un cube avec un bord arrondi (Figure 3.22) (Bricogne et al., 2014). Ce cube est modifié par deux concepteurs. Le premier crée une première version avec l'ajout d'un bossage (extrusion) sur le côté supérieur. La seconde supprime le bord arrondi et génère ainsi une deuxième version.

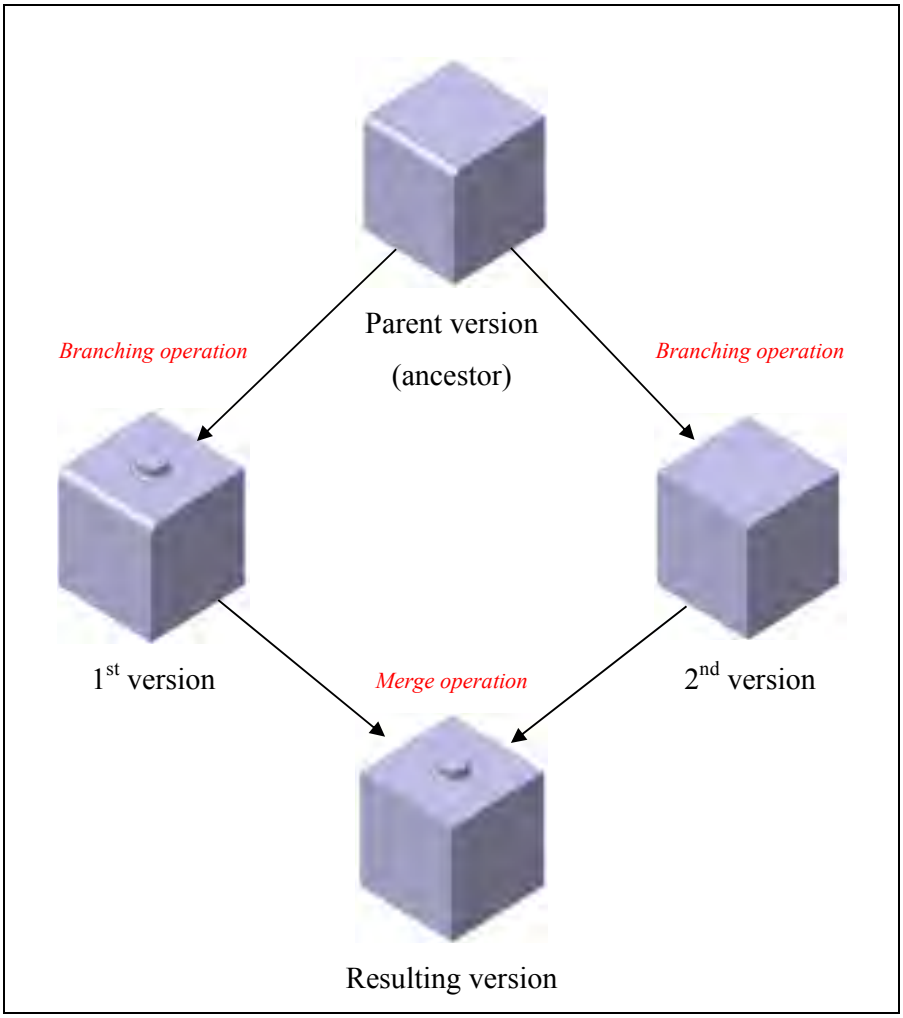


Figure 3.22 Illustration d'une opération de création et de fusion de branches sur une modèle CAO (Bricogne et al., 2014)

L'opération de fusion, résultant en un cube avec le bossage mais sans le bord arrondi, peut être réalisée grâce à une opération de fusion à 3 voies. Les trois configurations GUI décrites dans la section précédente pouvant être utilisées pour résoudre la fusion sont présentées sur la Figure 3.23 (organisation par défaut : initialisation avec la version d'origine), la Figure 3.24 (initialisation avec la 1ère version) et la Figure 3.25 (initialisation avec la 2ème version). Ces figures illustrent l'impact du choix initial sur le travail associé à la fusion. Sur ces figures, les nouvelles faces sont représentées en rouge, les faces modifiées en jaune et les faces inchangées en bleu.

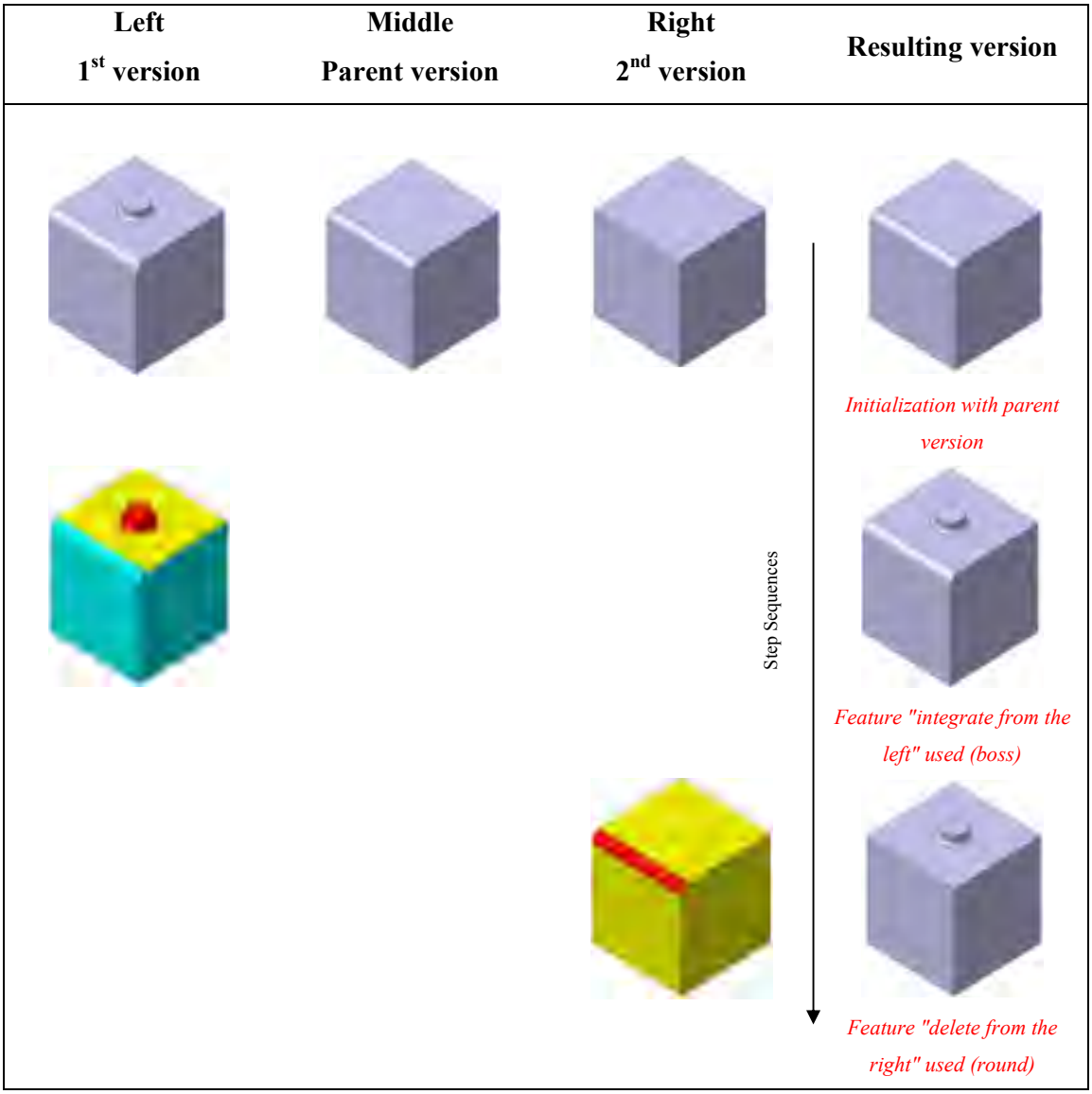


Figure 3.23 *3-way merge*: configuration GUI par défaut (Bricogne et al., 2014)

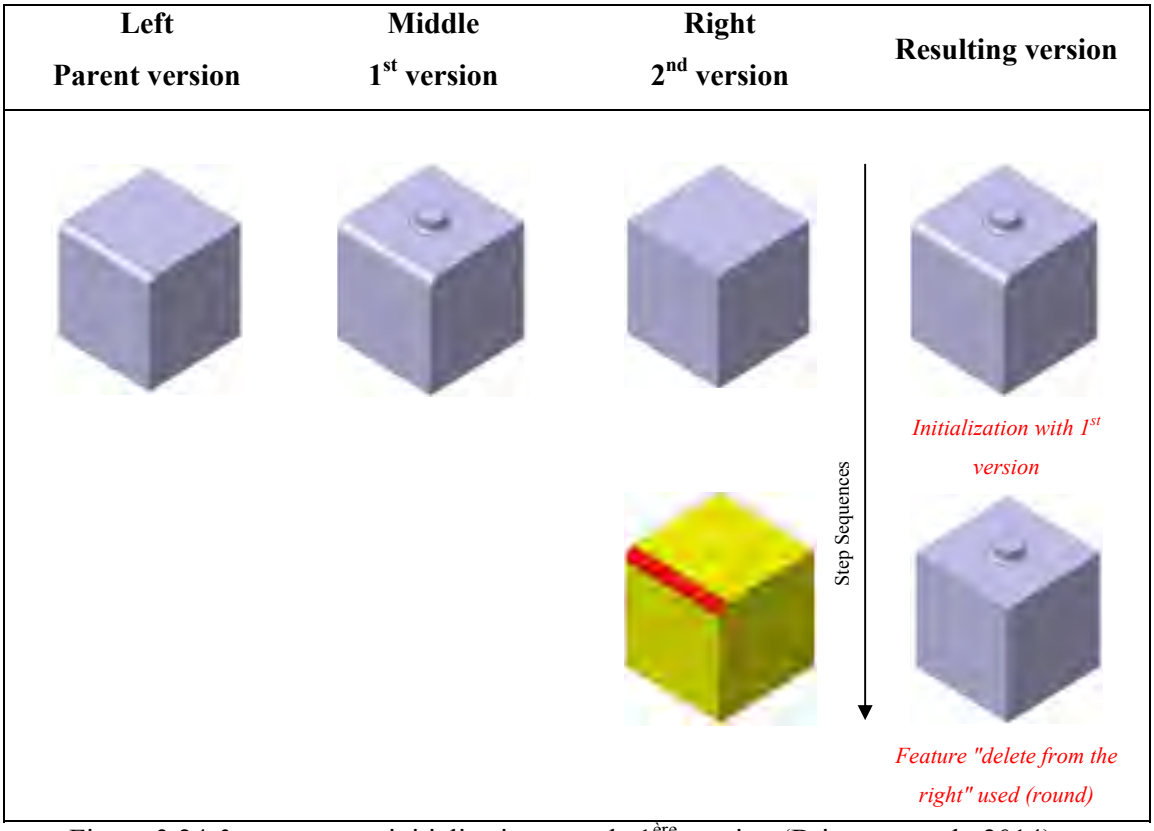


Figure 3.24 *3-way merge*: initialisation avec la 1ère version (Bricogne et al., 2014)

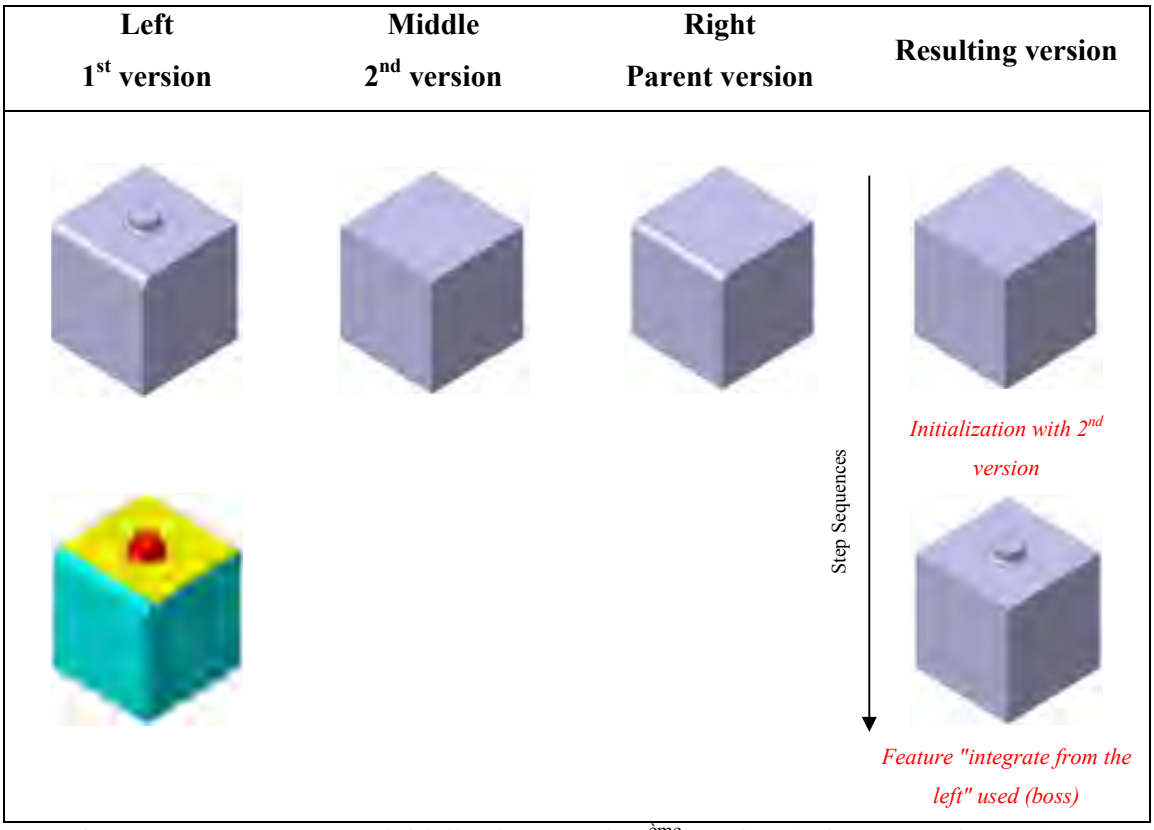


Figure 3.25 *3-way merge*: initialisation avec la 2ème version (Bricogne et al., 2014)

Cet exemple illustre comment ces trois premières configurations peuvent influencer la façon dont les opérations de fusion sont réalisées. Il n'y a aucun intérêt à comparer ces configurations pour déterminer laquelle est la meilleure ou la plus efficace dans ce genre d'exemple. Elles correspondent simplement à des approches différentes. Le concepteur en charge de la fusion peut préférer intégrer ses propres modifications dans la version de son collègue, ou au contraire, préférer intégrer les modifications de son collègue dans sa propre version. Généralement, la configuration par défaut est considérée comme la plus sûre, mais également comme la plus fastidieuse.

Ce type de fusion est une fusion sans conflit, ce qui signifie que la même zone géométrique n'est pas impactée par les deux modifications, permettant ainsi éventuellement une fusion automatique. Lorsque des conflits sont détectés, une intervention humaine est généralement nécessaire pour réaliser l'opération de fusion. Selon la situation, les fusions peuvent nécessiter plusieurs outils spécifiques. Un outil générique pour l'ensemble des données techniques mécanique semble irréaliste, mais des

outils dédiés à des données spécifiques semblent envisageable. L'outil approprié sera alors choisi en fonction du type de données. Même pour fusionner deux versions d'un même CI, plusieurs outils peuvent être utilisés de façon successive ou combinée, par exemple, un *3-way merge* géométrique et un *3-way merge* présentant les caractéristiques des features présents dans l'arbre. Pour illustrer ce genre de solution, la Figure 3.26 présente la fonction de comparaison géométrique proposée par le logiciel de CATIA®[16].

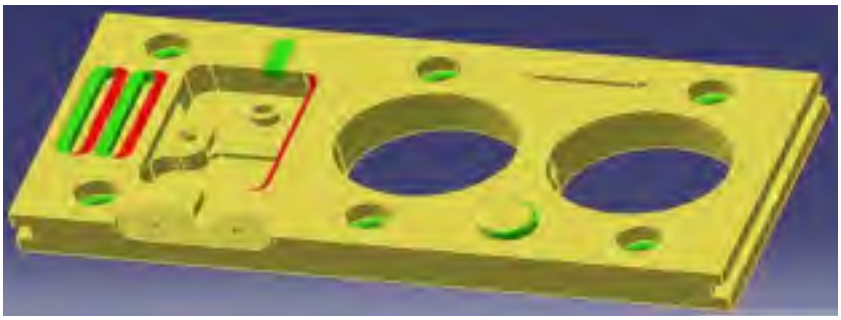


Figure 3.26 Exemple d'outil de comparaison géométrique: la fonctionnalité CATIA® "compare parts" (Brière-Côté et al., 2012)

Plusieurs approches existent pour la comparaison des modèles de CAO (Brière-Côté et al., 2012), mais ces travaux ne rentrent pas dans le cadre des contributions liées à cette thèse.

Sans entrer dans une description détaillée de la façon dont un outil *3-way merge* pour la CAO doit fonctionner, une interface graphique envisageable a été présentée, ainsi que les avantages liés à l'usage de ce type de solution. Dans le paragraphe suivant, les avancées sur les travaux portant sur la fusion de modèles structurés, i. e. portant une sémantique particulière et respectant une structure préétablie, sont présentées comme des solutions pertinentes pour le développement d'outils métiers spécifiques qui permettraient de généraliser les mécanismes de création & fusion de branches à l'ensemble des disciplines pouvant être impliquées dans la conception de système intégrés.

---

[16] CATIA V5, http://www.3ds.com/products/catia/, Dassault Systèmes Inc.

3.3.3.2 3-way merge de documents et modèles structurés: les perspectives pour des outils métier

Comme décrit dans la section précédente, les outils permettant une fusion automatique de versions concurrentes ne semblent pas envisageables à court ou à moyen terme pour des données telles que des données CAO et le but de ces travaux n'est pas de proposer un plan de développement pour atteindre cet objectif. Cependant, ces outils pourraient s'inspirer de techniques récentes permettant la fusion de données structurées. Depuis que le développement de logiciels est de plus en plus basé sur des techniques de modélisation (UML, MDE, ...), les besoins pour modifier de façon concurrente et synchrone ces modèles se fait ressentir. Dès lors, des travaux portant sur la fusion de modèles structurés ont fait leur apparition (Barrett et al., 2008). Cette section présente un bref aperçu de ces techniques, présentées comme une opportunité pour la création d'outils permettant la fusion de modèles métiers et donc la généralisation des mécanismes de *branch* and *merge* à l'ensemble des disciplines.

Une première technique de fusion de modèles est basée sur un environnement de collaboration pour résoudre les conflits. Brosch et al. (2009) considèrent les opérations de fusion manuelle présentent un risque d'introduction d'incohérence dans le modèle, bien qu'ils considèrent pessimiste de "allow only one developer to modify an artifact at the same time" (Brosch et al., 2009). Le problème réside dans le fait que le deuxième développeur est généralement forcé de résoudre le conflit pour que ses modifications soient adoptées. La responsabilité complète de la fusion des deux versions lui incombe alors, sans qu'il soit nécessairement au courant des intentions du premier développeur (Brosch et al., 2009). Afin de partager cette responsabilité, Brosch et al. proposent un solveur de conflits collaboratif qui peut être lancé à partir d'un outil de fusion classique. Basé sur un écran partagé avec un service de tchat, cet outil de fusion nommé « *Model Versioning System Architecture and Collaborative Conflict Resolver* », permet le suivi des conflits et de leur résolution afin de permettre l'apprentissage des décisions des utilisateurs afin d'en déduire des motifs de résolution. Cette solution permet ainsi de ne plus avoir besoin de recourir à la solution basée sur une assignation systématique des actions en fonction des responsabilités des portions de code source.

Cette idée de décloisonnement des responsabilités est reprise dans la synthèse de ce concept.

Une deuxième technique, présentée par Lindholm (2004), est plus axée sur l'usage d'un *3-way merge* pour des documents textuels structurés. Pour Lindholm, un "tree is one of the most frequently used data structures in computer science, and a widely used format for encoding trees has emerged with the advent of the Extensible Markup Language (XML) (W3C, 2008)" (Lindholm, 2004). Ses travaux de recherche, basés sur le XML, peuvent être transposés à tout document qui adopte une structuration arborescente. Il est basé sur deux grands principes. Tout d'abord, tous les nœuds des documents XML à comparer sont identifiés grâce à des étiquettes. Ces nœuds sont comparés deux à deux afin d'en déduire les insertions, les suppressions et les mises à jour, ainsi que les déplacements et les déplacements avec mises à jour. Même si toutes les données métiers ne sont pas écrites en XML, la structure de ces données est très souvent basée sur un arbre ordonné. Par exemple, Wang et al. montrent que la norme STEP pour l'échange de données produit, peut être transcrite directement en XML (Wang et al., 2010). L'utilisation de ce type de technique nous semble donc envisageable pour une grande quantité de données métier.

L'approche décrite dans le paragraphe ci-avant a été classée par Mens comme une « fusion syntaxique » spécifique (Mens, 2002). La fusion syntaxique diffère de fusion textuelle classique par le fait qu'il prend aussi la syntaxe du langage employé en compte. Dans sa revue de littérature sur les techniques de fusion, Mens décrit deux autres types de fusion qui sont la fusion sémantique et la fusion structurelle. La fusion sémantique a surtout la particularité de détecter les dépendances de la modification effectuée. Cela signifie non seulement que l'outil vérifie la syntaxe du document, mais aussi l'impact des modifications effectuées. La fusion structurelle est quant à elle plus dédiée aux opérations de restructuration du code appelées refactoring. Ces opérations « ont la particularité de conserver le comportement du logiciel inchangé, c'est à dire, qu'elles ne changent pas la sémantique du logiciel, bien que la structure de ce dernier, peut, elle, changer de façon significative » (Mens, 2002).

Ces trois paragraphes montrent que de nombreux travaux de recherche continuent dans le domaine des outils de fusion, notamment liés à l'avènement des techniques de

modélisation en génie logiciel. Les différentes techniques, qu'elles soient classiques, *i.e.* sans intelligence particulière, syntaxique, sémantique ou structurelle, peuvent toutes se révéler intéressantes selon le type de donnée, sa structuration ou le type de modification effectuée sur celle-ci. Si un seul et même outil de fusion commun à toutes les données issues des différentes disciplines ne semble pas envisageable, les besoins liés à l'usage de solutions de gestion de données permettant le *branch* & *merge* pourraient pousser certains éditeurs à proposer des solutions de fusion de documents intégrées à leurs propres logiciels. L'illustration réalisée sur des données géométriques, considérées comme particulièrement difficiles à traiter, montre que le développement de ce type d'outils reste envisageable.

3.3.4 Synthèse : le branch and merge support des méthodes agiles

Dans les paragraphes précédents, l'usage possible d'un outil de *diff* et de *merge* sur des données de type CAO a été présenté avant de montrer que les recherches récentes portant sur les techniques de fusion pourraient être utilisées afin de créer des outils de fusion spécifiques aux différentes disciplines. Ces avancées ne peuvent être considérées individuellement comme un support aux méthodes agiles. En revanche, elles rendent possible le déploiement de solutions de gestion de données intégrant des fonctionnalités de *branch* & *merge*.

Outre le fait que l'usage du concept de *branch* & *merge* permet le travail synchrone sur le même CI, il rend surtout possible le « décloisonnement des données ». En effet, les activités de conception, dans de nombreuses disciplines, restent basées sur un découpage et une répartition stricts des tâches, rendant les données créées par un concepteur sa « propriété ». Il devient alors le contact privilégié en cas de modification, ce qui est loin d'être illogique, mais ce qui peut se révéler limitant selon les situations. Dans le cas de composants à l'interface entre disciplines, le fait que la même donnée puisse être éditée par les différentes parties permet à chaque expert de pouvoir réaliser précisément ce dont il a besoin, quitte à ce que le responsable de la donnée vienne affiner son travail ultérieurement. Parfois, le positionnement d'une simple note dans la donnée concernée pourrait améliorer la communication entre les acteurs de la collaboration. Les interactions entre activités d'ingénierie ou entre disciplines pourraient ainsi s'en trouver facilitées.

Le *branch & merge* est un concept rendant possible et donnant un intérêt tout particulier à l'usage des *workspaces*. Leur usage combiné permet de dynamiser les échanges, les rendant plus réguliers et concédant aux acteurs des différentes disciplines l'opportunité de travailler sur leurs propres données, mais également sur les données des concepteurs avec lesquels ils collaborent. Ils permettent de promouvoir la prise d'initiative pour pouvoir, par exemple, trouver des solutions multidisciplinaires aux problèmes de conception.

La Figure 3.27 reprend les trois points de synthèse des méthodes agiles et positionne le concept de *branch & merge* par rapport à ces points. Il apparaît que le *branch & merge* sert surtout comme support au concept de WS.

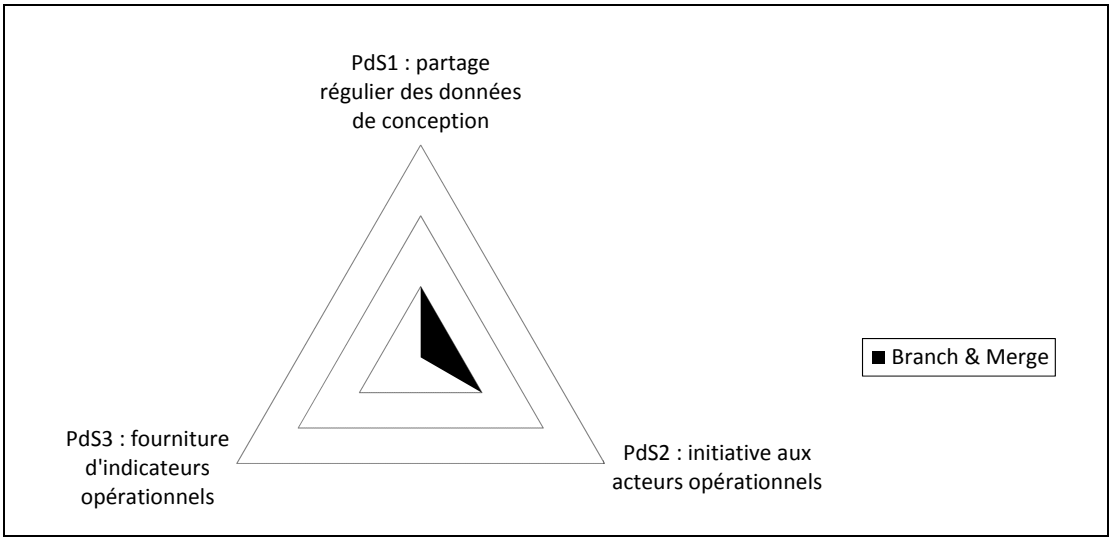


Figure 3.27 Positionnement du concept nommé *branch & merge* par rapport aux points de synthèse des méthodes agiles

3.4 Synthèse de la proposition

Afin d'illustrer la complémentarité et l'imbrication des trois concepts présentés dans ce chapitre, les poupées russes ont été utilisées en introduction (Figure 3.1). C'est sur cette idée d'imbrication que nous souhaitons clore ce chapitre. Pris individuellement, ces concepts sont d'ores et déjà porteurs d'intérêt, mais ils révèlent tout leur potentiel s'ils sont utilisés en synergie. La Figure 3.28 montre la couverture globale des trois concepts par rapport aux points de synthèse des méthodes agiles. Elle fait écho au Tableau 3.1 présenté en début de chapitre. Ainsi, nous pouvons remarquer que l'essentiel de ces trois points sont couverts grâce à ces trois concepts.

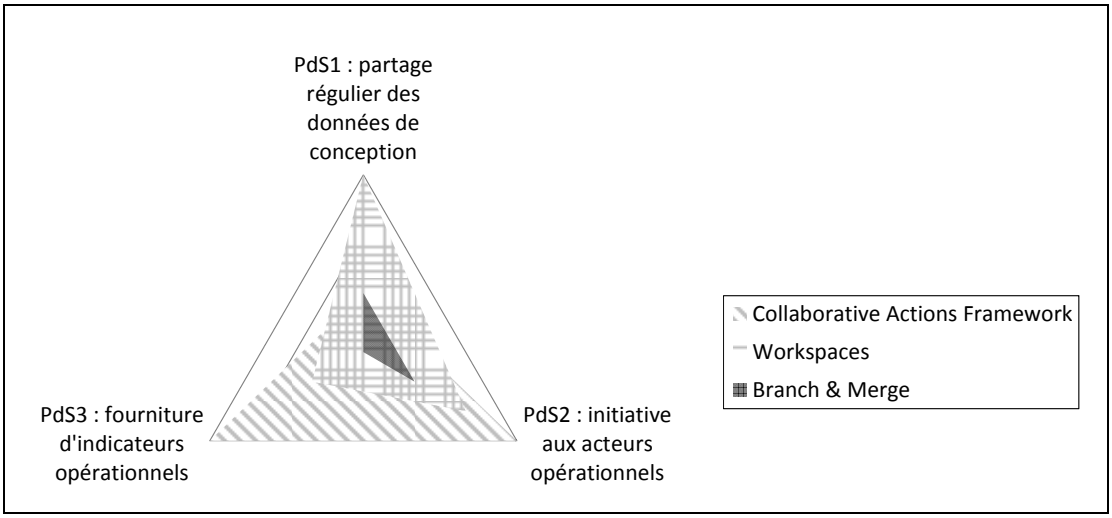


Figure 3.28 Positionnement global des concepts par rapport aux points de synthèse des méthodes agiles

En reprenant les tableaux Tableau 2.7 et Tableau 3.1 montrant 1) les interactions existantes entre les objectifs de la thèse et les points de synthèses des méthodes agiles et 2) les concepts formant la proposition et les points de synthèse des méthodes agiles, nous pouvons proposer un nouveau tableau de synthèse précisant les interactions présentes entre les objectifs de la thèse et les concepts formant la proposition (Tableau 3.2). On y remarque que le *Collaborative Actions Framework*, qui chapeaute l'ensemble des concepts, couvre bien les trois objectifs formulés au premier chapitre.

		Proposition : concepts		
		Concept 1 : *Collaborative Actions Framework*	Concept 2 : *Workspaces*	Concept 3 : *Branch & Merge*
Objectifs travaux	Obj. 1 : améliorer la collaboration	✔	✔	✔
	Obj. 2 : faciliter la prise de décisions	✔	✔	✘
	Obj. 3 : assurer la traçabilité décisions/données	✔	✘	✘

Tableau 3.2 Concepts formant la proposition vs. objectifs de nos travaux

Enfin, le Tableau 3.3 positionne notre proposition au sein de notre tableau de synthèse de l'état de l'art. On retrouve ainsi le *Collaborative Actions Framework* dans la colonne *controller*, partant du niveau opérationnel et s'étirant légèrement jusqu'au niveau stratégique. Les *Workspaces* assurent le lien entre la colonne *model* et la colonne *controller*, le tout au niveau opérationnel. Et enfin, le *branch & merge* se positionne au niveau *model*, à un niveau opérationnel.

	Model	Controller	View
Strategic			
Tactical			
Operational			

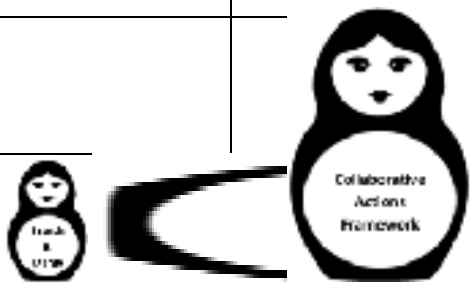


Tableau 3.3 Positionnement de notre proposition dans le tableau de synthèse de l'état de l'art

Alors que le *branch* & *merge* permet de dynamiser les échanges et les modifications, les *workspaces* permettent de mettre dans un même système l'ensemble des données issues des différentes disciplines, mais également de permettre un partage et une intégration multidisciplinaire au plus tôt. Ils rendent possible le lien entre données et décisions, exploitable par la suite de deux manières, à savoir connaître les données impactées par une décision (liste des CI dans une version donnée) et connaître les décisions qui sont intervenues tout au long des modifications apportées à un CI.

Ces décisions sont représentées par des actions collaboratives résultant de la planification et de l'organisation de la conception du système ou d'initiatives provenant des différents acteurs opérationnels. La consolidation des informations portées par les actions collaboratives permet, en les combinant avec d'autres indicateurs comme ceux proposés par l'analyse des échanges intervenant entre WS, de fournir un *framework* de pilotage à base d'informations opérationnelles.

L'ensemble de ces concepts est présenté comme une solution pour la mise en œuvre des points de synthèse issus des méthodes agiles et donc répondant aux objectifs de la thèse (Cf. Tableau 3.1 et Tableau 3.2). Ils permettent de proposer des indicateurs pour un pilotage opérationnel, au jour le jour et prenant en compte les différents aléas liés à la

conception, par opposition aux méthodes de type *project-planned*. Pour résoudre les difficultés rencontrées lors de la conception, une large place est laissée aux initiatives des acteurs opérationnels qui peuvent créer des actions collaboratives et les faire transiter dans l'entreprise en fonction des besoins. Ces différentes initiatives pouvaient, dans certains cas, déjà exister dans les entreprises, mais elles sont, grâce à ce concept, formalisées, mesurées et suivies, remplaçant ainsi des échanges réalisés jusqu'alors par courriel ou téléphone. Pour mener à bien ces actions collaboratives, des itérations sont souvent nécessaires. Ces échanges peuvent s'appuyer sur un nouvel élément du système d'information où l'ensemble des données techniques issues de chacune des disciplines sont partagées plus régulièrement, se retrouvent et s'intègrent.

Le rôle des WS intermédiaires, ou WS d'intégration, est alors primordial pour tester la compatibilité des solutions. Les opérations de promotion et de collecte sont alors des évènements pouvant permettre de formaliser certaines procédures, comme par exemple des procédures de validation.

Afin de démontrer la pertinence de cette proposition en regard de notre question de recherche, le chapitre suivant propose d'appliquer les concepts présentés ci-avant à une discipline très fortement multidisciplinaire, la mécatronique.

CHAPITRE 4

Mise en œuvre d'une méthode agile pour la conception collaborative multidisciplinaire : application à la mécatronique

Dans ce chapitre, la mécatronique a été choisie comme cadre applicatif afin de démontrer la pertinence de notre proposition pour faciliter l'intégration des disciplines, résultant en une meilleure intégration pour le produit. Dans un premier temps, un scénario illustrant l'usage des trois concepts formant la proposition est présenté. Il s'inspire de deux problématiques industrielles réelles bien que les données utilisées soient factices. Dans un second temps, un démonstrateur basé sur deux logiciels du marché nommés Jira et SVN, est exposé. Bien que le concept de WS ne soit pas présent dans ce démonstrateur, le lien données/décision y reste assuré grâce à la présence sur chacune des actions collaboratives de la liste des références des CI modifiés dans leur cadre.

4.1 Un scénario mécatronique permettant d'illustrer les apports de chacun des concepts

Le scénario présenté dans cette section est inspiré de deux problématiques industrielles réelles bien que les données utilisées soient factices. Il a été imaginé par les auteurs et différents groupes d'étudiants se sont attachés à le rendre le plus compréhensible et visuel possible.

4.1.1 Un produit mécatronique : le bras de robot

Beaucoup de produits physiques modernes qui nous entourent peuvent être considérés comme mécatroniques ; ils comportent en effet pour la plupart de l'électronique, du logiciel embarqué et ont *a minima* une enveloppe réalisée sous la responsabilité de concepteurs d'origine mécanique. Mais identifier un produit pour lequel les apports mécaniques, électroniques et logicielles sont équilibrés (ceci n'est pas une finalité mais permet de bien illustrer l'apport des différentes disciplines impliquées) n'est pas aussi aisé qu'il n'y paraît. Pour ce scénario, notre choix s'est porté sur un bras de robot industriel (Figure 4.1) pour sa multidisciplinarité et son intérêt industriel. Sur ce produit, les aspects mécaniques liés au dimensionnement, aux liaisons, à la cinématique et aux matériaux sont aussi importants que les aspects électriques/électroniques pour le contrôle et l'actionnement, tout comme les aspects logiciels avec le pilotage et la programmation.

Figure 4.1 Bras de robot industriel modélisé sur la base de différents catalogues industriels[17]

[17] http://www.kuka-robotics.com et http://www.staubli.fr/

4.1.2 Un scénario regroupant une évolution du produit et une modification

Parce que la conception ne débute que très rarement d'une feuille blanche et qu'elle se base généralement sur des composants existants ou sur une gamme de produits antérieurs, le scénario présenté ici s'appuie sur deux projets distincts qui viennent modifier le bras de robot présenté ci-avant.

Le premier projet, initié par l'équipe s'occupant de la maintenance opérationnelle de ce type de produits, s'attache à solutionner un problème récurrent détecté sur les robots en cours d'utilisation chez certains clients. Il s'agit donc d'une modification. Ce problème concerne la gaine électrique extérieure qui, selon la trajectoire programmée du robot, peut être sur-sollicitée, conduisant à une détérioration de cette dernière. La solution qui sera proposée suite à une simulation vise à ajouter et à déplacer les attaches de ce flexible sur le bras de robot.

Le second projet, porté par le bureau d'études, vise à remplacer le système de freinage présent dans les articulations du robot. Il s'agit donc d'une évolution du produit. Ce système a pour particularité d'être un frein de sécurité à manque de courant : il se doit de stopper les masses en rotation et d'assurer leur maintien en position en cas de panne électrique. Cette fonction était assurée jusqu'alors par des freins de positionnement à aimant permanent, mais pour des raisons de consommation électrique et de surchauffe en phase de fonctionnement, les concepteurs étudient l'opportunité du remplacement de ce système par un frein de sécurité à ressorts présentant des caractéristiques similaires en termes de taille, poids, temps de commutation, etc. Ce remplacement impacte la tension d'alimentation qui passe ainsi de 24V à 12V, mais également la future maintenance de ce système de freinage sur lequel des pièces d'usure seront à remplacer. Cette évolution impactera donc la géométrie du bras qui devra inclure une trappe d'accès. La Figure 4.2 présente le frein de sécurité à ressorts (a) et le frein de sécurité à aimant permanent (b).

(a) (b)

Figure 4.2 Freins de sécurité à ressorts (a) et à aimant permanent (b)[18]

4.1.3 Deux actions collaboratives pour ces deux projets

Lors de l'élaboration de ce scénario, les actions collaboratives n'étaient envisagées que dans le cadre de demandes techniques et elles étaient alors dénommées « actions d'ingénierie » (*Engineering Actions*). L'ensemble des figures et tableaux reprend donc ces termes d'« *Engineering Action* ».

Afin de gérer ces deux projets, deux actions collaboratives sont créées. L'équipe de maintenance, nommée « TMAINTEN » dans ce scenario, réalise une demande d'ouverture d'action collaborative concernant le harnais électrique portant l'identifiant unique « EA131010_1602 ». Le manager du bureau d'études, nommé « MANAGER1 », crée une demande d'action collaborative concernant le système de freinage portant l'identifiant unique « EA131210_0958 ». La Figure 4.3 illustre l'état de cette action à l'issue de sa création : créée le 10/12/2013, cette action concerne la nouvelle gamme de robot nommée « IND_ROBOT_RK_40_3F », et est obligatoire pour la sortie de ce produit (*flag mandatory*). Elle a été incluse au planning de développement et le statut *scheduled* a donc été apposé.

[18] https://www.keb.de/fr/produits/les-produits-embrayages-et-freins.html

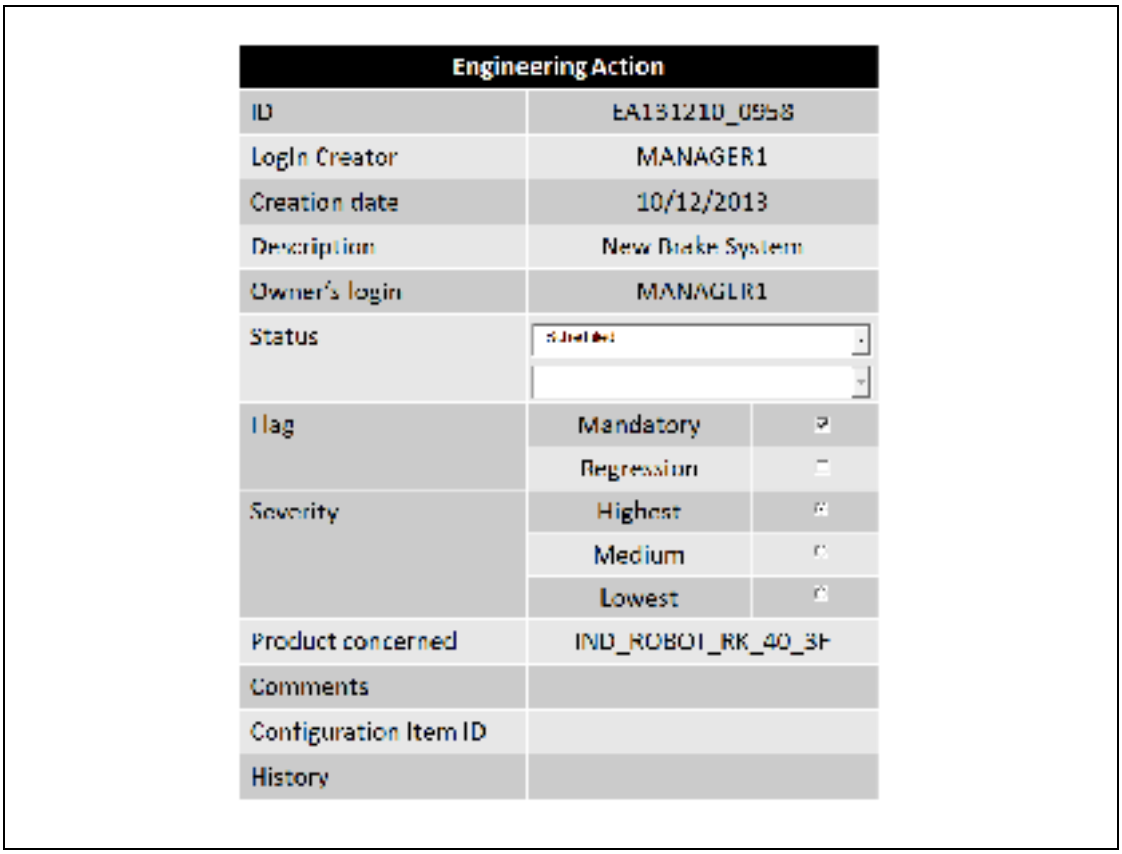


Figure 4.3 Exemple d'action collaborative créée par MANAGER1

4.1.4 Une structure de WS basée sur ces demandes d'actions collaboratives

Afin de répondre à ces demandes d'actions, l'équipe de maintenance TMAINTEN et le manager MANAGER1 ont créé des WS ad hoc correspondants à chacune des actions collaboratives. La Figure 4.4 présente ainsi la structuration des WS mise en œuvre pour répondre aux demandes d'actions collaboratives. Pour la demande d'action liée à la maintenance, le WS EA131010_1602 a été créé et pour la seconde demande liée au système de freinage, le WS EA131210_0958 a été créé. En anticipant un peu sur le déroulement du scénario, nous voyons également que les concepteurs MDESIGN1 et EDESIGN1 seront sollicités pour la première action, alors que les concepteurs MDESIGN2, EDESIGN2 et SDESIGN1 seront sollicités pour la seconde action.[19]

[19] Dans ces dénominations, le 'M' fait référence à la discipline mécanique, le 'E' aux disciplines électrique et électronique et le 'S' à la discipline logicielle.

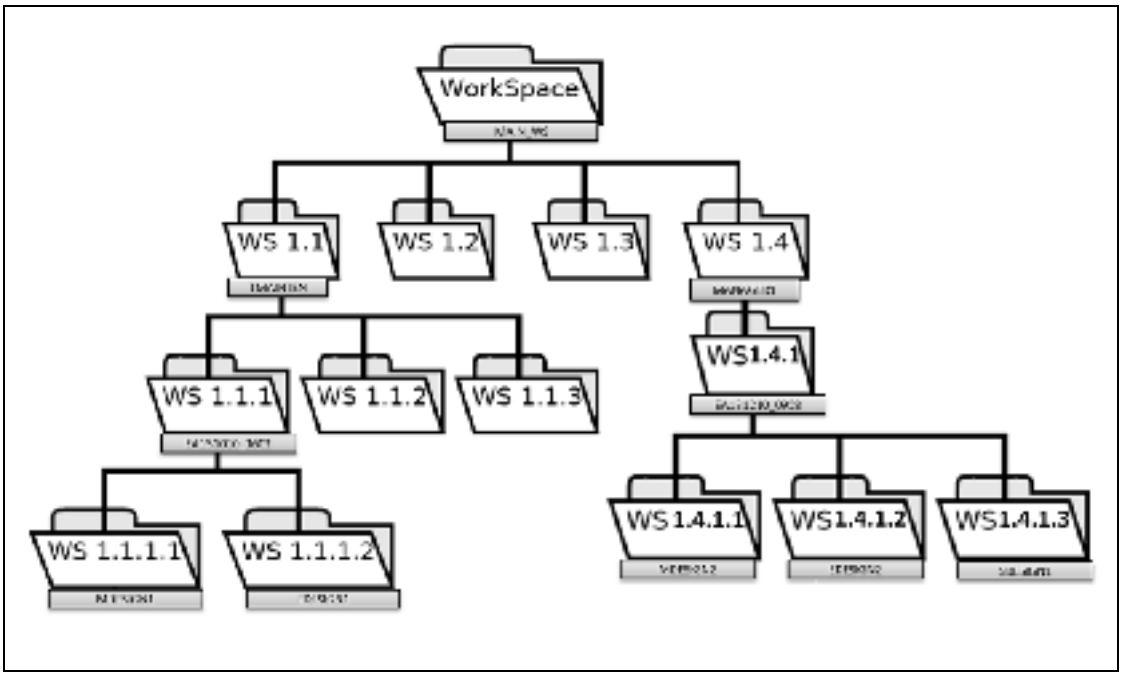


Figure 4.4 Structuration des WS mise en œuvre pour répondre à ces demandes d'actions

À noter qu'en parallèle de cette structuration dédiée à ces actions, une structuration de WS différente existe pour des travaux de conception plus traditionnels. Ainsi, on observe cette structuration sur la Figure 4.5 basée sur une décomposition structurelle du robot en cours de développement.

L'intérêt de pouvoir réaliser ces différentes structurations, relatives aux mêmes données, est de pouvoir bénéficier d'une structure adaptée aux besoins de chacun des projets. Ainsi, un projet nécessitant une forte intégration multidisciplinaire verra les différentes disciplines regroupées dans la même branche de l'arbre, alors qu'un projet plus cloisonné verra les différents acteurs regroupés par disciplines, la structuration de l'arbre reflétant cette organisation fortement disciplinaire.

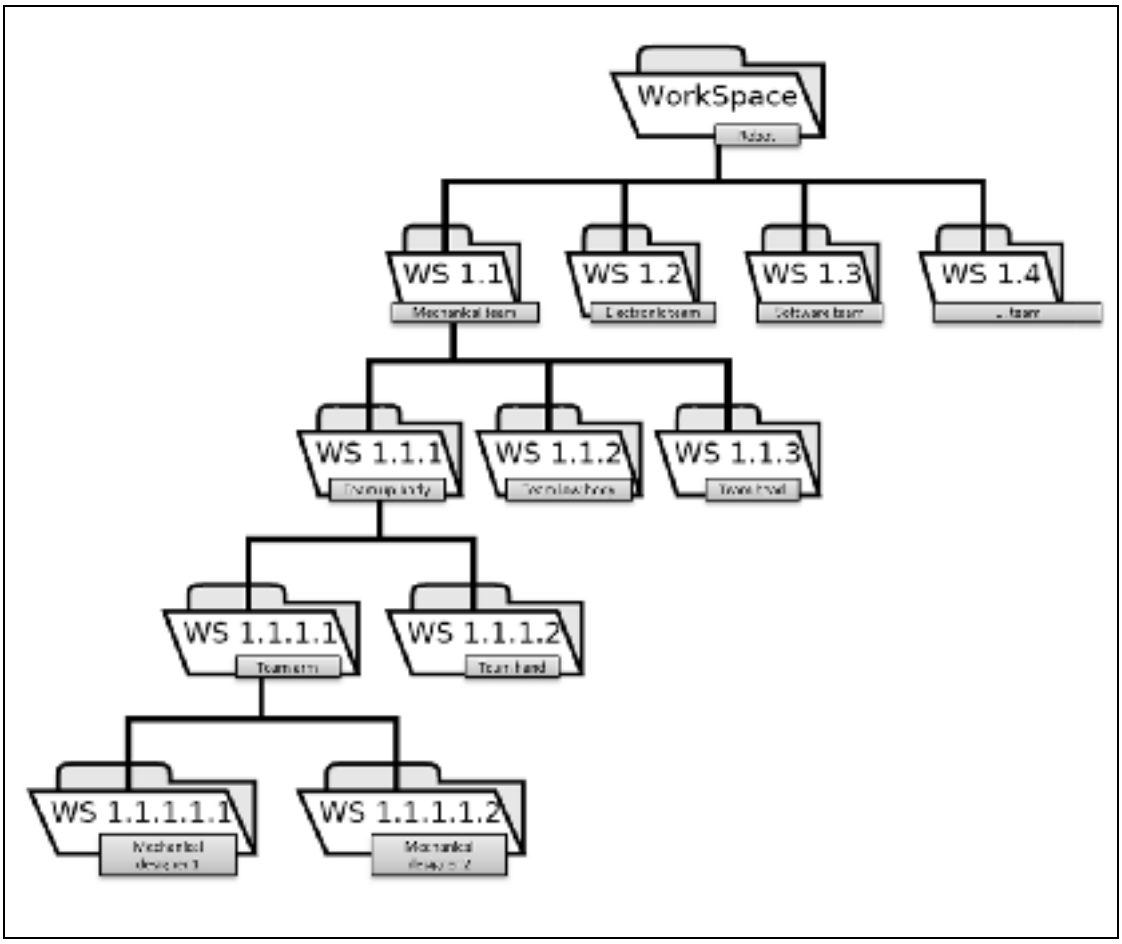


Figure 4.5 Structuration des WS basée sur une décomposition structurelle du robot

4.1.5 Déroulement temporel du scénario

Afin de suivre le déroulement temporel du scénario, deux chronologies sont proposées sur la Figure 4.6. La première chronologie montre qui réalise et à quels moments des opérations concernant le premier projet, structuré autour de l'action collaborative. On voit ainsi que le leader de l'équipe de maintenance analyse et ouvre l'action collaborative (partie gris foncé), demande ensuite à MDESIGN1 de réaliser certaines modifications sur le bras de robot qui vont impacter le harnais électrique. L'EDESIGN1 réalise alors des opérations de modification sur ce harnais électrique. Ces dernières sont intégrées par MDESIGN1 qui clôt alors l'action. Celle-ci est vérifiée par le responsable de la maintenance qui accepte cette clôture.

La seconde chronologie se rapporte de la même manière à la seconde action. La principale différence réside dans le fait que l'ensemble des modifications réalisées dans le cadre de cette action sont vérifiées quotidiennement par MANAGER1. Cette vérification se matérialise par la longue barre gris foncé.

Il semble nécessaire de préciser que ces chronologies ne représentent pas les cheminements de ces actions collaboratives d'acteurs en acteurs, mais bien les implications des différents acteurs au cours du temps. La détention de l'action est matérialisée directement sur l'action collaborative, représentée sur la Figure 4.3 par l'attribut nommé *owner's login*.

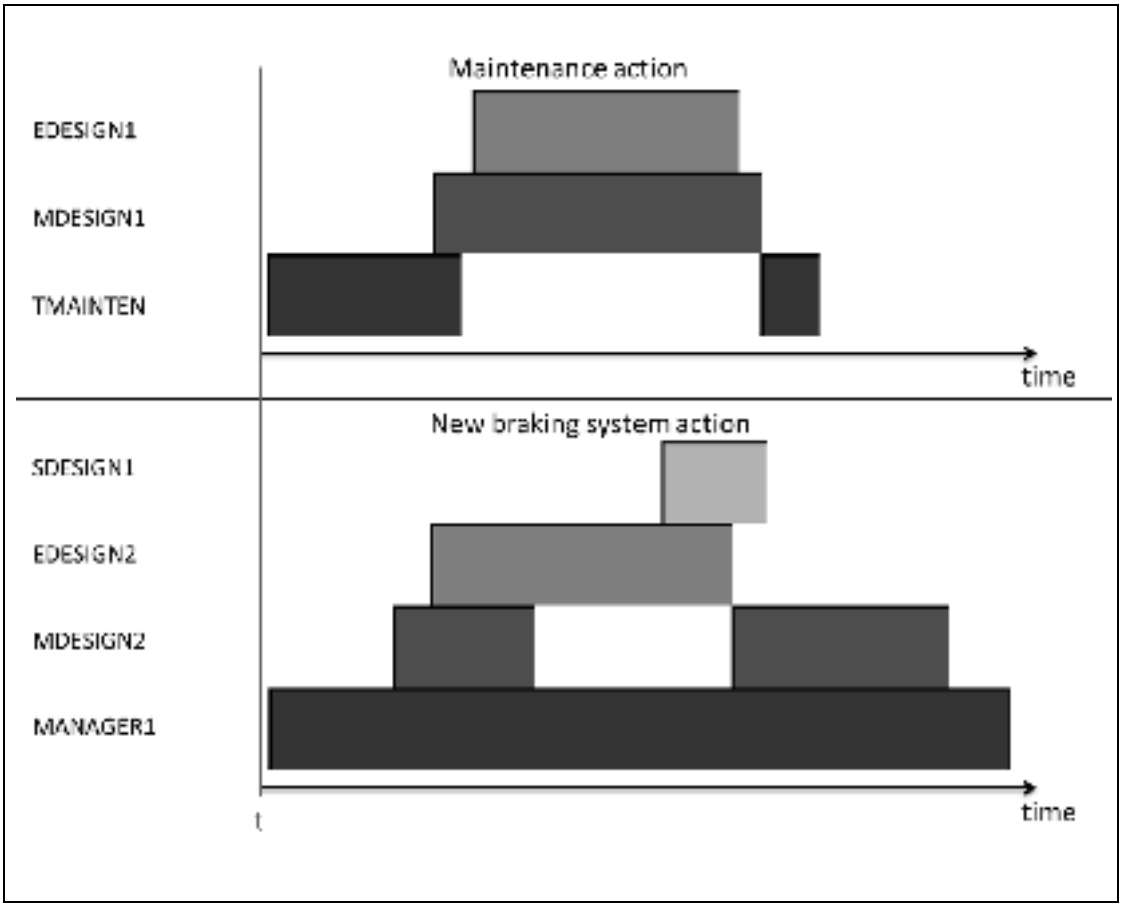
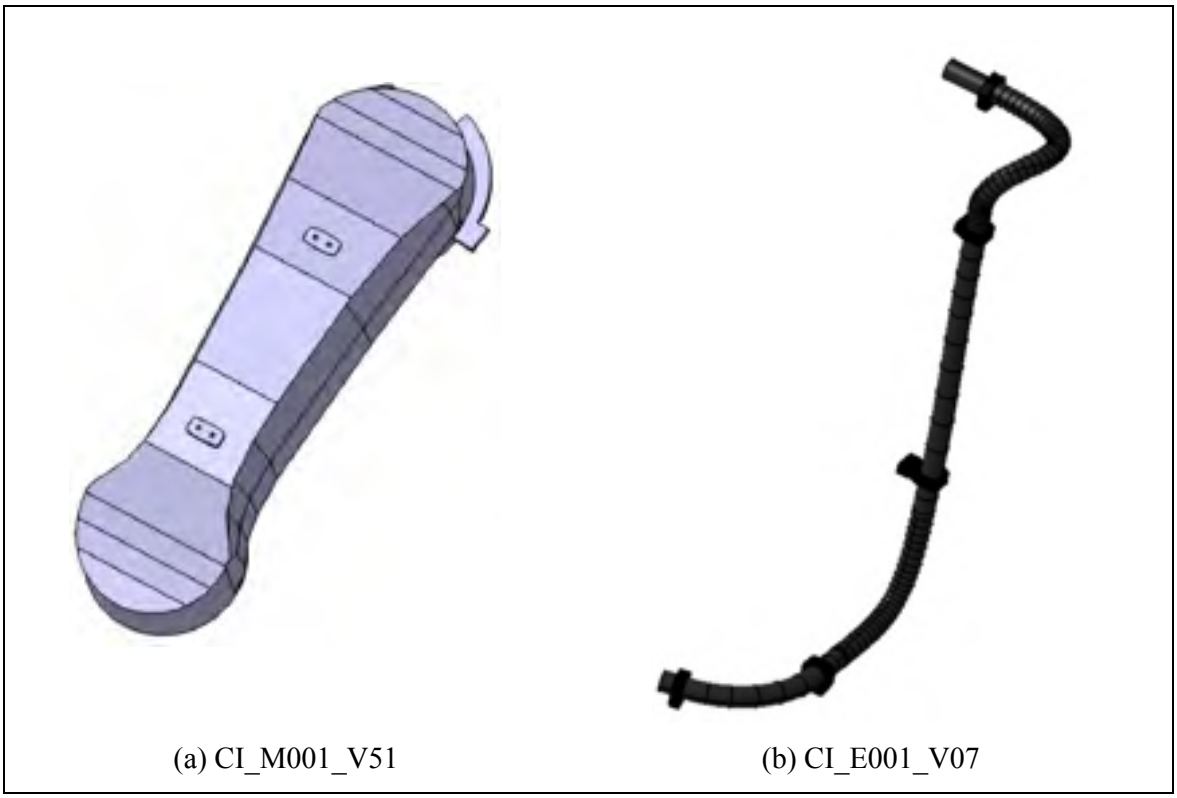


Figure 4.6 Chronologie du déroulement des deux actions

4.1.6 Déroulement de l'action collaborative liée à l'opération de maintenance

Dans le cadre de l'action collaborative EA131010_1602 créée par le leader de l'équipe de maintenance, le concepteur mécanique MDESIGN1 est mandaté via la réception de l'action pour analyser le problème et le résoudre. Il décide alors d'ajouter une troisième fixation pour le harnais électrique et de décaler les deux fixations existantes. La pièce impactée est alors le bras de robot, portant l'identifiant CI_M001. Lors de cette modification, cette pièce passe alors de la version v51 à la version v53. Mais ce déplacement nécessite l'allongement du harnais électrique, portant l'identifiant CI_E001. Lors de cette modification, cette pièce passe alors de la version v07 à la version v08. Les pièces impactées, le bras de robot et le harnais électrique, sont présentées sur la Figure 4.7. L'évolution des versions est matérialisée grâce à la Figure 4.8. L'impact de la modification sur le bras de robot, pièce CI_M001, passant de la version v51 à la version v53, est illustré sur la Figure 4.9.



(a) CI_M001_V51 (b) CI_E001_V07

Figure 4.7 Pièces impactées par les modifications apportées suite à l'action collaborative liée à l'opération de maintenance

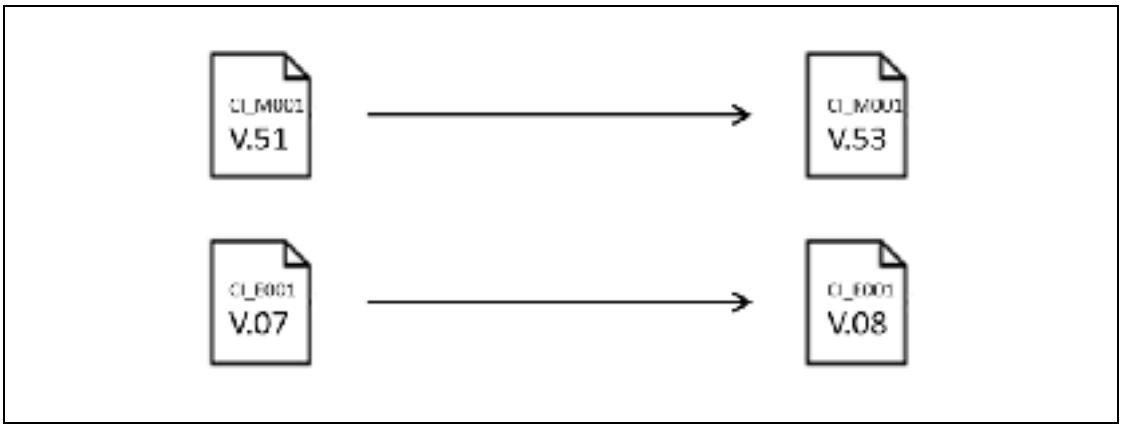
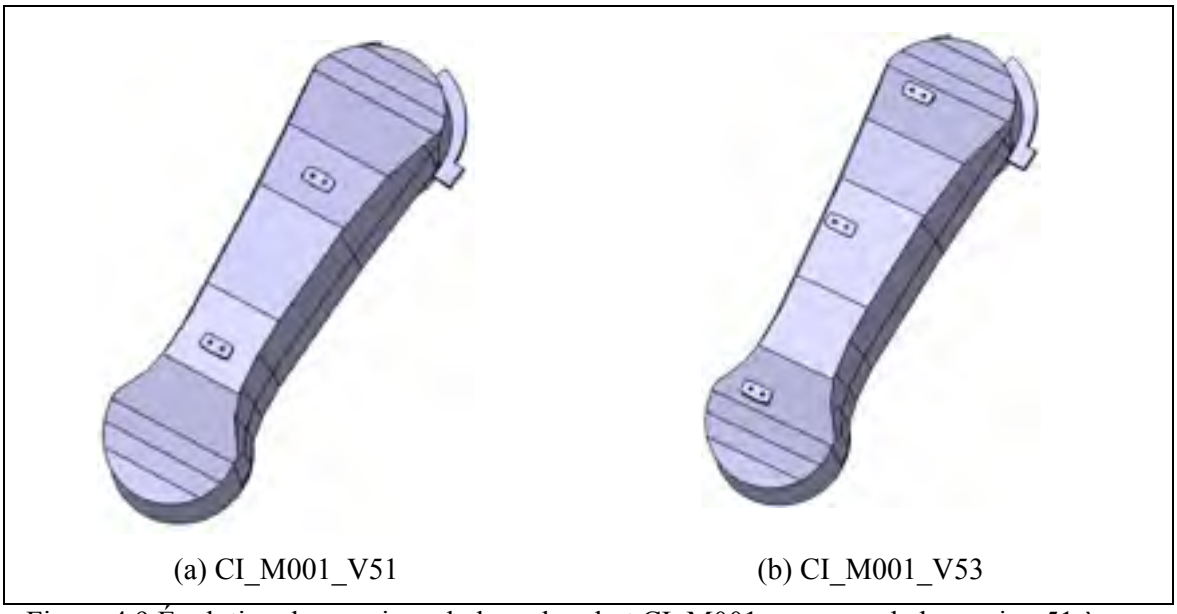


Figure 4.8 Modifications relatives à l'action collaborative liée à l'opération de maintenance : évolution des versions

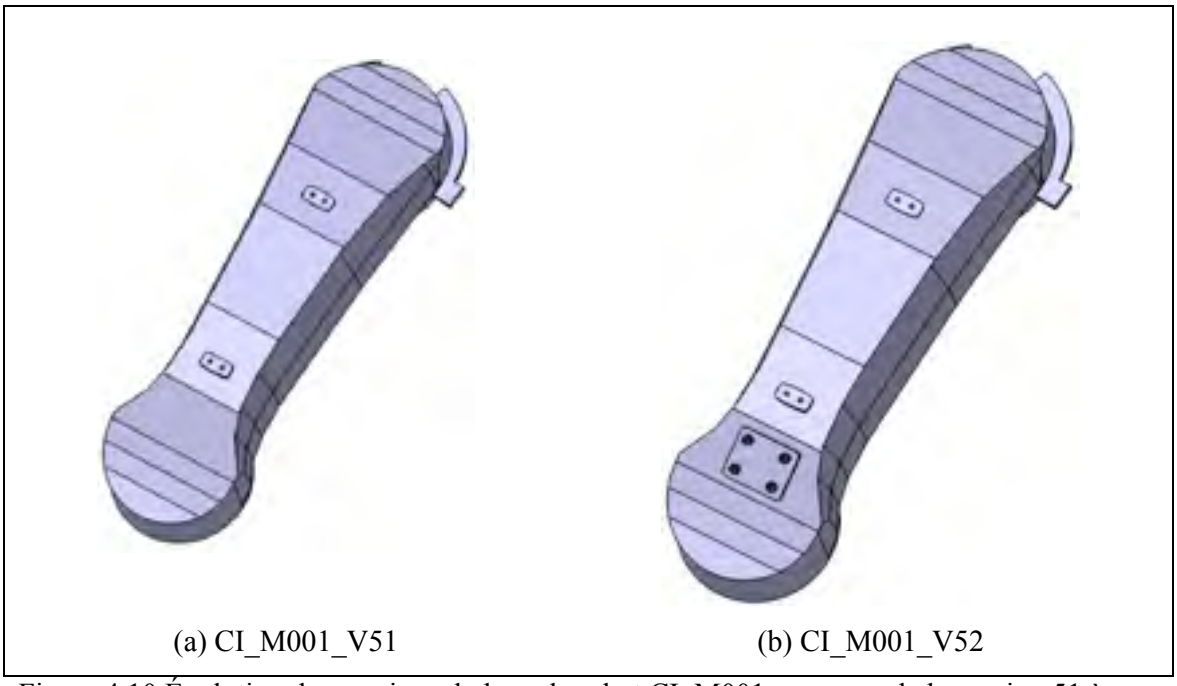


(a) CI_M001_V51 (b) CI_M001_V53

Figure 4.9 Évolution des versions du bras de robot CI_M001 : passage de la version 51 à la version 53

4.1.7 Déroulement de l'action collaborative liée à l'évolution du système de freinage

Dans le cadre de l'action collaborative EA131210_0958 créée par le responsable du bureau d'études MANAGER1, le concepteur mécanique MDESIGN2 se voit affecter cette action afin de réaliser la première opération liée à celle-ci. Afin de rendre le nouveau système de freinage accessible pour pouvoir changer les pièces d'usure, MDESIGN2 réalise une trappe d'accès sur le bras de robot CI_M001 qui passe de la version 51 à la version 52. Cette version sera concurrente de la version 53, créée par MDESIGN1 lors de l'opération liée à la maintenance. Deux branches sont donc apparues sur le graphe des versions du CI_M001 dont un extrait est proposé sur la Figure 4.11. Sur chacune des branches, bien que ces informations ne soient pas encore renseignées dans le système, le lien entre versions et demandes d'action a été spécifié. La chronologie présentée sur la Figure 4.6 montre que le MDESIGN2 a commencé ses modifications du CI_M001 avant le MDESIGN1, ce qui explique qu'il s'est vu attribué la version 52 avant ce dernier.



(a) CI_M001_V51 (b) CI_M001_V52

Figure 4.10 Évolution des versions du bras de robot CI_M001 : passage de la version 51 à la version 52

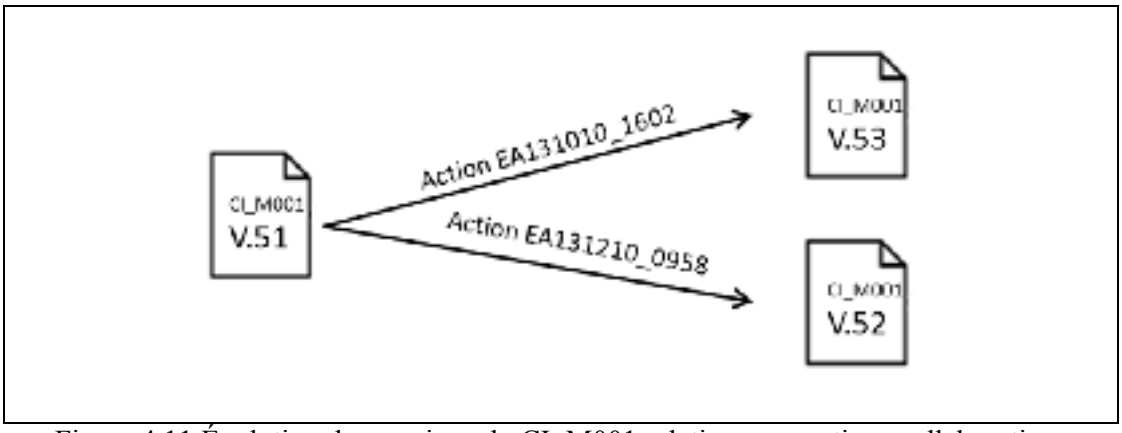


Figure 4.11 Évolution des versions du CI_M001 relatives aux actions collaboratives

Mais le changement de système de freinage implique également une modification électronique (passage de 24V à 12V). Ainsi, le composant CI_E002 est modifié passant de la version 12 à la version 13. De même, les changements liés au système de freinage changent l'inertie du système et, pour pouvoir conserver le même comportement au niveau du bras de robot, des ajustements sont nécessaires au niveau d'un fichier code source logiciel. Ce dernier, le CI_S001 passe de la version 21 à la version 22. La Figure 4.12 résume l'ensemble des modifications nécessaires.

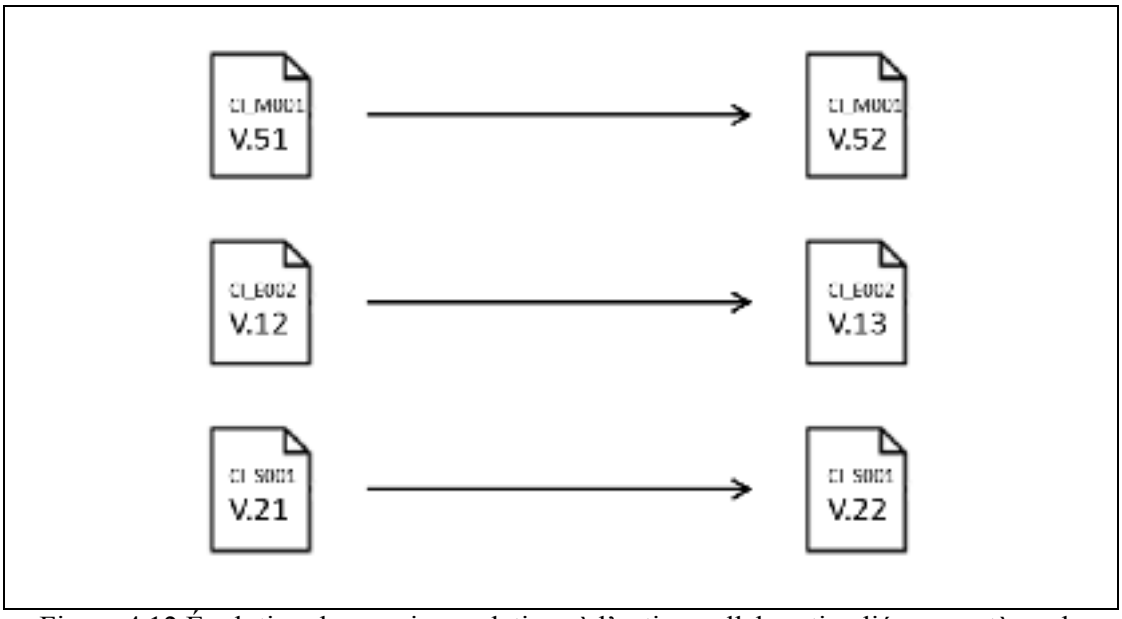
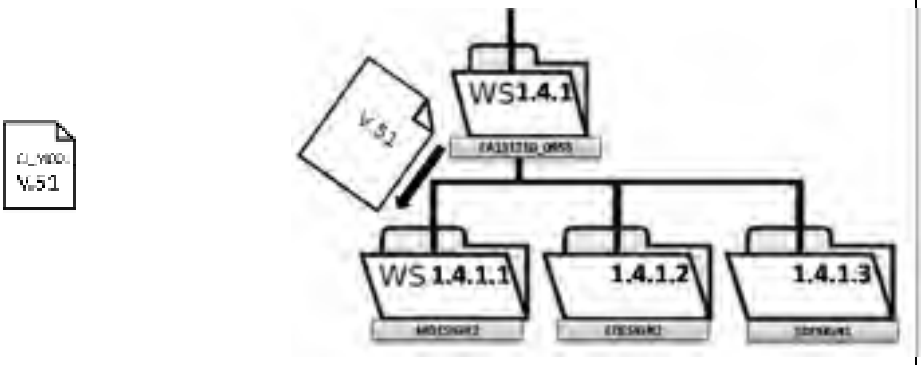


Figure 4.12 Évolution des versions relatives à l'action collaborative liée au système de freinage

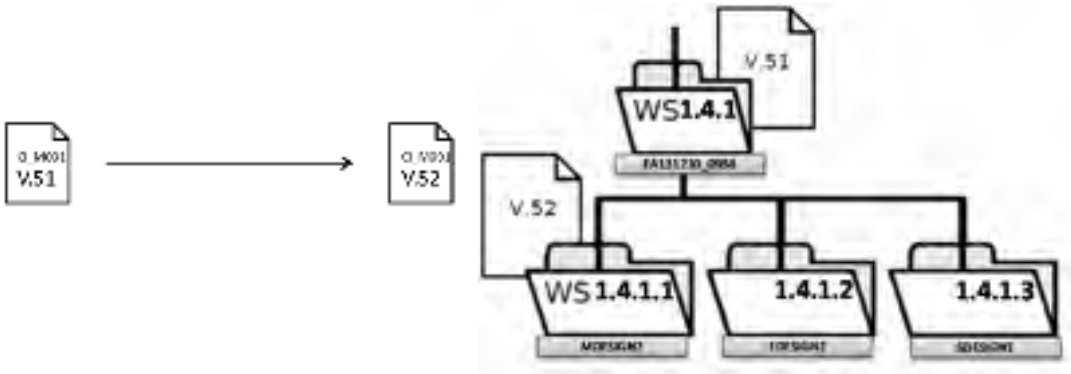
4.1.8 Deux versions du bras de robot concurrentes

La Figure 4.13 décrit les différentes opérations qui sont réalisées, menant à la création de deux versions conflictuelles du bras de robot puis à leur réconciliation. Lorsque MDESIGN1 crée son WS afin de travailler sur l'action collaborative de maintenance (a), il reçoit la version 51 du CI_M001 alors que MDESIGN2 reçoit également la même version 51 dans son WS (b). MDESIGN2 réalise une première modification, il prend alors la version 52 (c). De son côté, MDESIGN1 réalise une seconde modification, il prend alors la version 53 (d). À cette étape, le graphe de versions de la pièce bras de robot montre que deux versions ont été créées (e). MDESIGN1 promeut son travail le premier (f). Le système de gestion a alors l'information selon laquelle l'objet CI_M001_V53 est nécessaire à la résolution de l'action EA131010_1602. Le responsable maintenance accepte alors le travail de MDESIGN1 en collectant son WS (g). Avant de promouvoir à son tour, MDESIGN2 doit synchroniser son WS (h), opération durant laquelle un conflit est détecté et doit être résolu. Le *merge* est alors effectué par MDESIGN2 (i), en collaboration avec MDESIGN1 afin de s'assurer que la fusion respecte bien ses intentions de conception. Enfin, MDESIGN2 promeut son travail à son tour (j). Le système de gestion a alors l'information selon laquelle l'objet CI_M001_V54 est nécessaire à la résolution de l'action EA131210_0958. Le MANAGER1 inspecte le travail en attente de collecte avant d'accepter les modifications proposées (j).

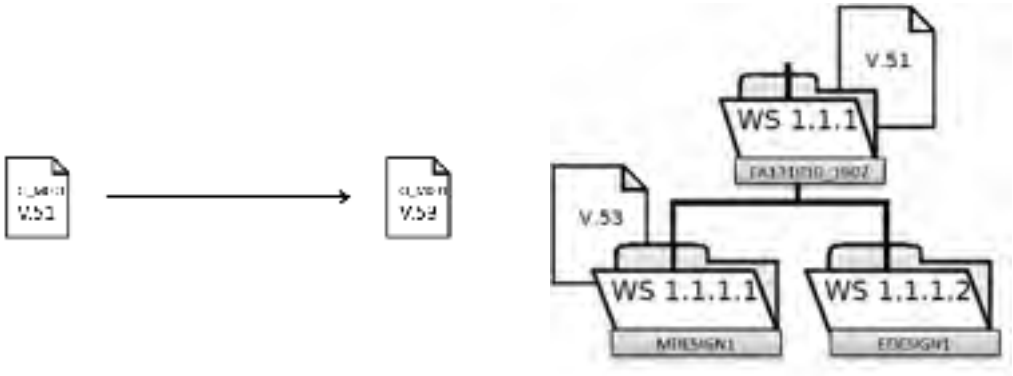
(a) Initialisation du WS du MDESIGN1 avec le CI_M001 version 51



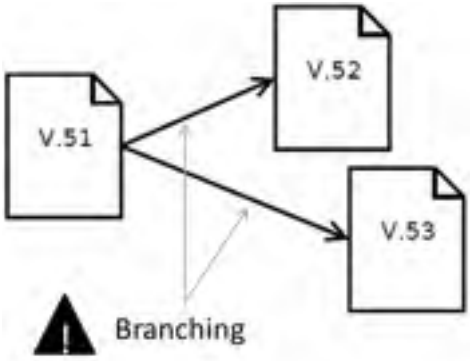
(b) Initialisation du WS du MDESIGN2 avec le CI_M001 version 51



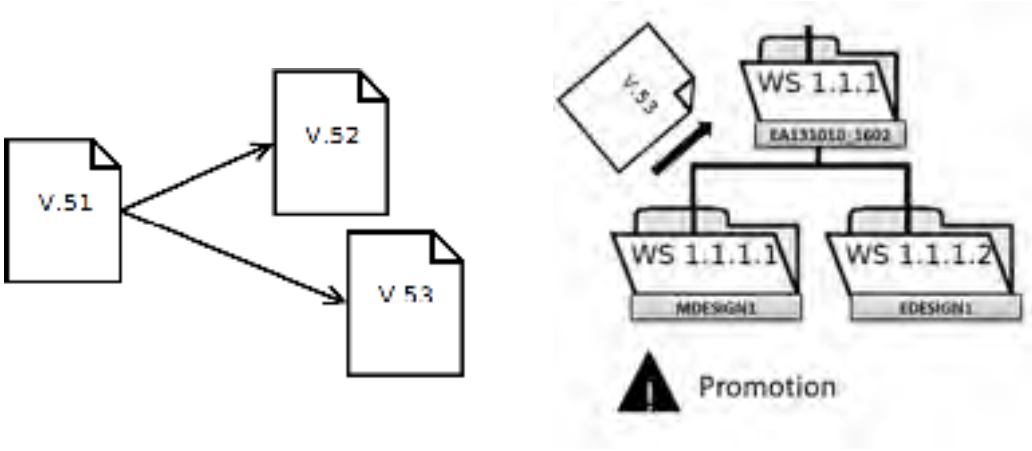
(c) Modification de MDESIGN2 entrainant la montée de version vers la v52



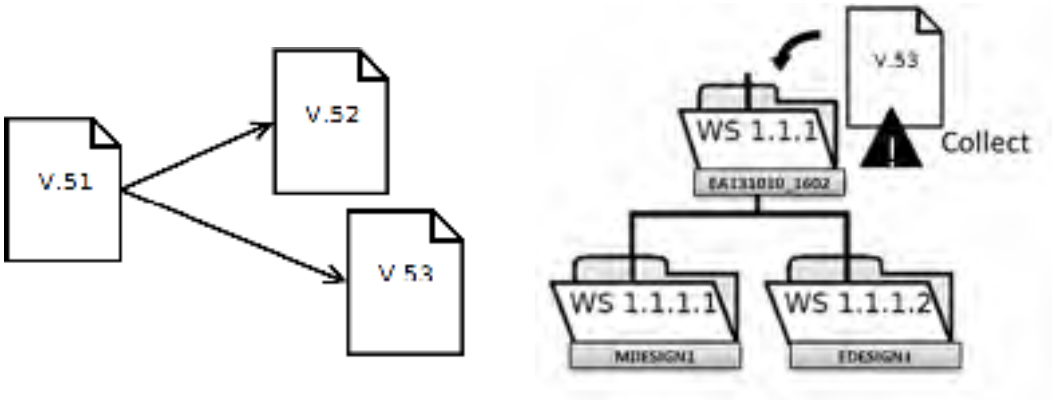
(d) Modification de MDESIGN1 entrainant la montée de version vers la v53



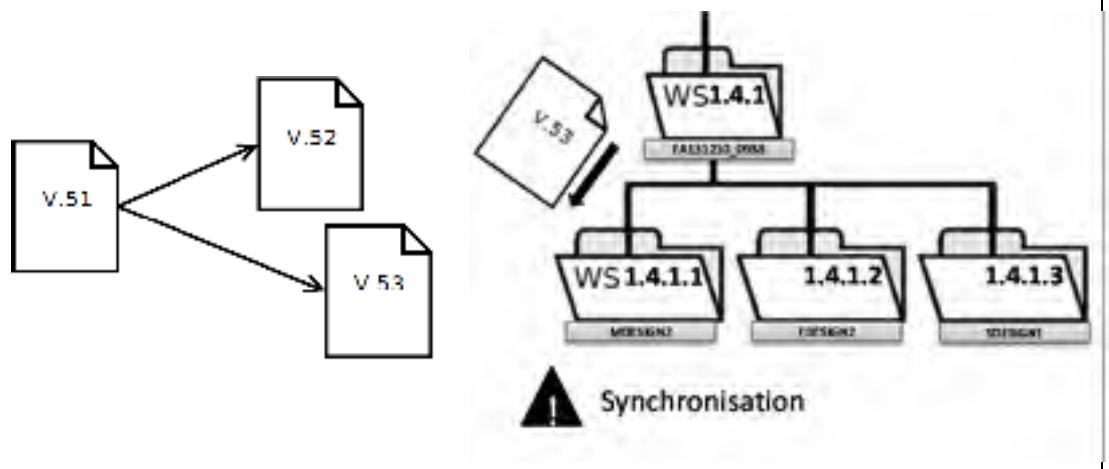
(e) Extrait du graphe de versions du CI_M001 résultant des opérations (c) et (d)



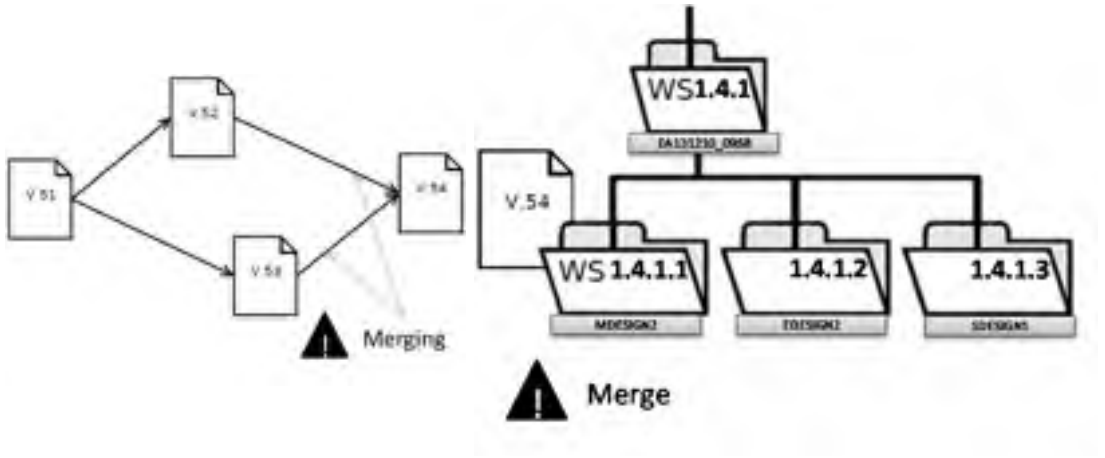
(f) Promotion de la version 53 dans le cadre de l'opération de maintenance



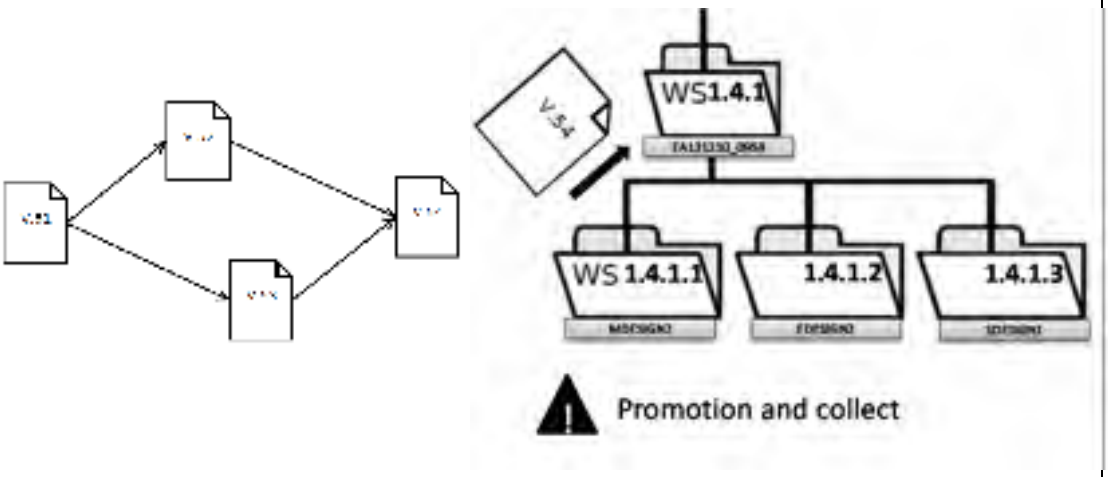
(g) Collecte de la version 53 dans le cadre de l'opération de maintenance



(h) Synchronisation de la version 53 dans le WS de MDESIGN2 dans le cadre du changement du système de freinage



(i) Opération de *merge* nécessitée par l'arrivée de la version conflictuelle dans le WS de MDESIGN2

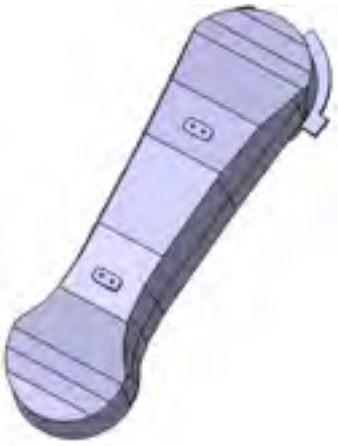


(j) Promotion et collecte de la version 54, réconciliant les versions 52 et 53

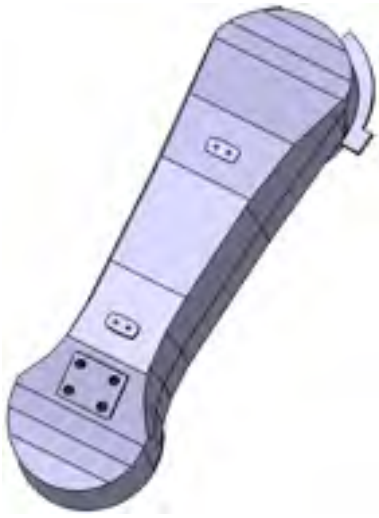
Figure 4.13 Modifications relatives à l'action collaborative liée au système de freinage : évolution des versions

4.1.9 Une fusion des deux versions concurrentes grâce à un outil de fusion adapté

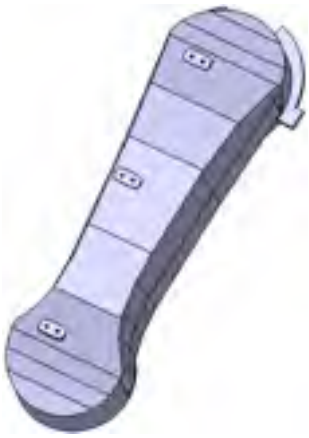
Comme illustré sur la Figure 4.14, les versions 52 (b) et 53 (b') sont en conflit (c) car elles affectent les mêmes zones géométriques. Pour autant, un outil de *merge* s'appuyant sur la version parente (a) n'a pas nécessairement de difficulté à réconcilier les deux versions (d). Dans le cas présent, les intentions de conception sont suffisamment explicites et les changements suffisamment mineurs pour que l'utilisateur réalise la fusion sans difficulté.



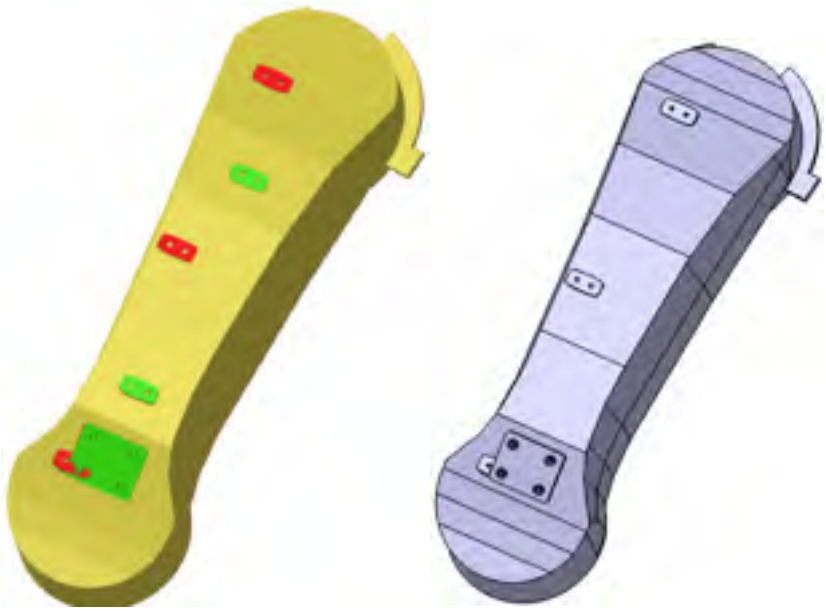
(a) CI_M001_V51



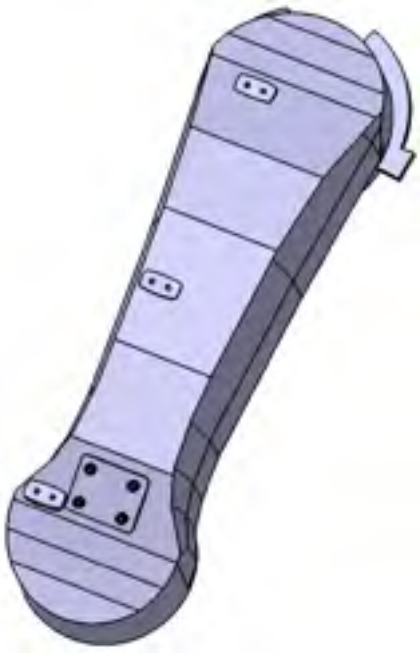
(b) CI_M001_V52



(b') CI_M001_V53



(c) CI_M001 mise en évidence des conflits



(d) CI_M001_V54 (v52 et v53 réconciliées)

Figure 4.14 Évolution des versions du bras de robot CI_M001 : réconciliation menant à la version 54

4.1.10 Des informations ajoutées successivement par les différents acteurs : le cas de l'action de maintenance

La Figure 4.15 illustre l'ajout successif d'informations et le cheminement de l'action collaborative liée à l'opération de maintenance. À l'étape (a), la demande d'action a été créée et qualifiée par TMAINTEN. À l'étape (b), elle vient d'être transmise à MDESIGN1 et le champ commentaire vient donc d'être enrichi par les instructions concernant le travail à effectuer. À l'étape (c), MDESIGN1 demande à EDESIGN1 de bien vouloir importer les modifications qu'il a réalisées sur le bras de robot à partir de son WS pour qu'EDESIGN1 ait les informations nécessaires à la modification du harnais électrique. À l'étape (d), EDESIGN1 vient de finir les modifications et il demande à MDESIGN1 de bien vouloir vérifier que les modifications apportées sont bien conformes à ses attentes. Enfin, l'étape (e) correspond à l'état de clôture avec l'ensemble des échanges et surtout la liste des CI modifiés. Le lien données/décision est ainsi assuré grâce aux livraisons successives ayant été associées à cette action.

Collaborative Action		
ID	EA131010_1602	
Creator's login	TMAINTEN	
Creation date	10/10/2013	
Title	Harness position	
Description	Harness position change needed to resolve the tearing problem	
Owner's login	TMAINTEN	
Status	Opened	
Flag	Mandatory	
	Regression	X
Severity	Highest	
	Medium	X
	Lowest	
Product concerned	IND_ROBOT_RK_40_3F	
Comments		
Configuration Item ID		

(a) État après la création par TMAINTEN

Collaborative Action		
ID	EA131010_1602	
Creator's login	TMAINTEN	
Creation date	10/10/2013	
Title	Harness position	
Description	Harness position change needed to resolve the tearing problem	
Owner's login	MDESIGN1	
Status	Opened	
Flag	Mandatory	
	Regression	X
Severity	Highest	
	Medium	X
	Lowest	
Product concerned	IND_ROBOT_RK_40_3F	
Comments	• TMAINTEN 10/12/2013: check the number and location of the fasteners	
Configuration Item ID		

(b) État après la réception par MDESIGN1

Collaborative Action		
ID	EA131010_1602	
Creator's login	TMAINTEN	
Creation date	10/10/2013	
Title	Harness position	
Description	Harness position change needed to resolve the tearing problem	
Owner's login	EDESIGN1	
Status	Opened – Change in progress	
Flag	Mandatory	
	Regression	X
Severity	Highest	
	Medium	X
	Lowest	
Product concerned	IND_ROBOT_RK_40_3F	
Comments	• TMAINTEN 10/12/2013: check the number and location of the fasteners • MDESIGN1 10/15/2013: due to my modification, the harness is not long enough. Please import CI_M001 in workspace 1.1.1.1 to get the change	
Configuration Item ID	• CI_M001_V53	

(c) État après la réception par EDESIGN1

Collaborative Action		
ID	EA131010_1602	
Creator's login	TMAINTEN	
Creation date	10/10/2013	
Title	Harness position	
Description	Harness position change needed to resolve the tearing problem	
Owner's login	MDESIGN1	
Status	Opened – Change in progress	
Flag	Mandatory	
	Regression	X
Severity	Highest	
	Medium	X
	Lowest	
Product concerned	IND_ROBOT_RK_40_3F	
Comments	• TMAINTEN 10/12/2013: check the number and location of the fasteners • MDESIGN1 10/15/2013: due to my modification, the harness is not long enough. Please import CI_M001 in workspace 1.1.1.1 to get the change • EDESIGN1 10/17/2013 : done	
Configuration Item ID	• CI_M001_V53 • CI_E001_V08	

(d) État après le renvoi d'EDESIGN1

Collaborative Action		
ID	EA131010_1602	
Creator's login	TMAINTEN	
Creation date	10/10/2013	
Title	Harness position	
Description	Harness position change needed to resolve the tearing problem	
Owner's login	TMAINTEN	
Status	Closed – Change realized	
Flag	Mandatory	
	Regression	X
Severity	Highest	
	Medium	X
	Lowest	
Product concerned	IND_ROBOT_RK_40_3F	
Comments	• TMAINTEN 10/12/2013: check the number and location of the fasteners • MDESIGN1 10/15/2013: due to my modification, the harness is not long enough. Please import CI_M001 in workspace 1.1.1.1 to get the change • EDESIGN1 10/17/2013: done • MDESIGN1 10/17/2013: one fastener added and two fasteners moved	
Configuration Item ID	• CI_M001_V53 • CI_E001_V08	

(e) État à l'issue de la clôture

Figure 4.15 Évolution de l'action collaborative EA131010_1602

4.1.11 Synthèse concernant ce scénario

La mise en œuvre des trois concepts issus de la proposition est assez délicate dans le cadre d'un prototype académique. C'est ce que nous verrons dans la prochaine section. Ce scénario a donc pour but l'illustration de l'usage du *Collaborative Actions Framework* mais aussi des échanges et des validations rendus possibles par les WS. Il illustre également un cas dans lequel la même pièce peut être soumise au même moment à deux intentions différentes, générant deux versions conflictuelles. En l'absence d'un véritable outil de fusion, le *merge* de ce modèle a été réalisé de manière manuelle et donc factice, mais la pertinence d'une telle solution a été démontrée.

L'agilité introduite par l'usage de ces différentes solutions provient ici de différents aspects. Tout d'abord, le même CI, ici le bras de robot, a été modifié par deux concepteurs différents pour deux motivations distinctes. Cela montre le décloisonnement entre données et concepteurs évoqué dans le chapitre précédent. Il y a ainsi une plus grande prise en considération des problématiques rencontrées par les collaborateurs, une réduction de la sectorisation, frein à l'intégration pour le produit.

Un second aspect est le partage des données via les WS pour atteindre l'objectif donné par l'action collaborative. Lors des échanges liés à l'opération de maintenance, MDESIGN1 ne demande pas à EDESIGN1 d'augmenter la longueur du harnais de XXmm. Il préfère lui donner accès à toutes les données qui lui sont nécessaires pour réaliser son travail. Ainsi, il n'empiète pas sur son activité, tout en lui facilitant le travail.

Enfin, et c'est certainement un des aspects les plus importants, lorsque TMAINTEN et MANAGER1 créent les actions collaboratives, ils n'ont pas à déterminer à l'avance qui va réaliser le travail, voire même la nature de la solution à mettre en œuvre ou encore des impacts multidisciplinaires liés aux modifications. Ils créent les demandes, les affectent à la personne qu'ils jugent la plus qualifiée pour débuter les opérations liées à l'action, qui à leur tour affectent ces actions aux personnes les plus adéquates, et ainsi de suite. Le cheminement des actions n'est pas prédéfini (il pourrait l'être si nécessaire !) et il s'adapte en fonction des besoins, des charges de travail, des affinités entre concepteurs

etc. La constitution de la mini équipe projet est ainsi directement décidée par les acteurs, ce qui est un des principes fondamentaux des méthodes agiles.

Ce scénario, première pièce du démonstrateur visant à illustrer de la pertinence de notre proposition, a été partiellement intégré à un prototype informatique basé sur deux solutions commerciales nommées JIRA et SVN. C'est ce prototype qui est présenté dans la section suivante.

4.2 Un prototype basé sur JIRA et SVN

Afin de prouver qu'une mise en œuvre informatique de cette proposition est possible, un prototype basé sur deux solutions du commerce, JIRA et SVN, a été réalisé. Les sections suivantes présentent les caractéristiques et apports de ces logiciels et la façon dont ils ont été utilisés.

4.2.1 Présentation des deux solutions commerciales retenues

4.2.1.1 JIRA : le gestionnaire d'incidents

JIRA[20] se présente et était historiquement un système de gestion d'incidents, de suivi de bugs et de gestion de projet développé par la société Atlassian[21]. Créé en 2002, ce logiciel collaboratif accessible depuis un navigateur internet intègre désormais de nombreux modules le rendant plus proche d'une plateforme de collaboration que d'un « simple *bugtracker* ». Parmi ces modules, il est important de noter que certains sont dédiés à la mise en œuvre de la méthodologie de développement agile nommée Scrum. Pourtant, la version installée dans le cadre du prototype est la version la plus basique de JIRA. Dès cette première version, des possibilités de paramétrage très importantes sont disponibles, nous permettant d' enrichir les formulaires de saisie à notre convenance, le but étant de pouvoir retrouver dans ces formulaires l'ensemble des informations portées par les actions collaboratives. D'autres tentatives avec des logiciels similaires, notamment Open Source, ont été lancées, mais il est rapidement apparu que les possibilités de personnalisation de JIRA étaient bien meilleures. La Figure 4.16 présente l'interface principale de JIRA.

[20] https://www.atlassian.com/fr/software/jira
[21] https://www.atlassian.com

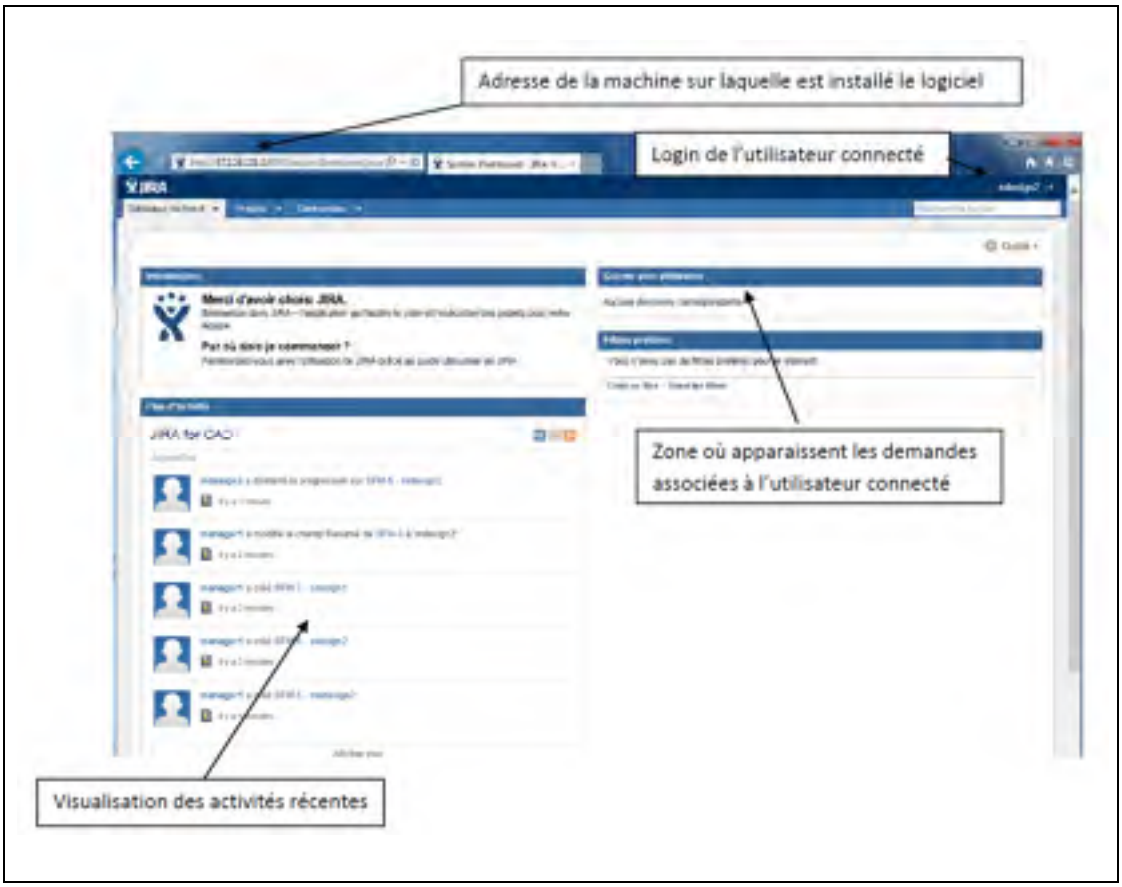


Figure 4.16 Présentation de l'interface principale de JIRA

4.2.1.2 SVN : le gestionnaire de versions

Sur la même machine virtuelle faisant office de serveur, le logiciel Subversion[22] (en abrégé SVN) a également été déployé. Ce logiciel de gestion de versions est, depuis 2010, un projet de la fondation Apache[23] et est distribué sous la licence Open Source Apache. Successeur du système nommé Concurrent Versions System[24] (CVS), il en adopte la plupart des principes de gestion notamment le principe du dépôt centralisé et unique des données (nommé en anglais *repository*). SVN gère ainsi les fichiers et les répertoires, ainsi que l'ensemble des changements qui y sont apportés, même le plus mimine. Cet historique permet ainsi de revenir à d'anciennes versions des données ou d'examiner la

[22] https://subversion.apache.org/
[23] http://www.apache.org/
[24] http://cvs.nongnu.org/

façon dont celles-ci ont évolué. Contrairement à d'autres systèmes concurrents, SVN ne propose pas de gestionnaire de configuration. Il se contente de gérer n'importe quel ensemble de fichiers. C'est ce côté généraliste et modulaire qui rend ce logiciel particulièrement intéressant dans notre cas.

4.2.2 Paramétrage des solutions et déroulement du scénario

L'objet de ce prototype est avant tout d'illustrer, via l'implémentation du scénario ci-avant, la possibilité de mettre en œuvre le lien données/décision. Pour ce faire, le projet « SCM for Mechatronics » a été créé et les utilisateurs du scénario ont été déclarés avant d'être affectés au projet. Des rôles prédéfinis dans JIRA ont également été affectés à ces utilisateurs afin de rendre le comportement du prototype le plus proche possible de notre scénario. Enfin, à l'aide du compte administrateur et des possibilités de personnalisation des formulaires, l'ensemble des champs correspondant aux caractéristiques des actions collaboratives ont été ajoutés aux champs standards de JIRA. JIRA offre la possibilité de créer différentes configurations de demande que l'on enregistre comme « Type de demande ». L'ensemble de ces demandes correspond dans notre cas aux actions collaboratives et les « Engineering Action » sont donc un sous type de ces actions. Dans ce prototype, seul le type « Engineering Action » a été créé. Les intentions de conception, présentées plus tôt dans ce manuscrit, n'ont pas été implémentées dans ce prototype. La Figure 4.17 présente le formulaire de création d'une demande d'action collaborative de type « Engineering Action ».

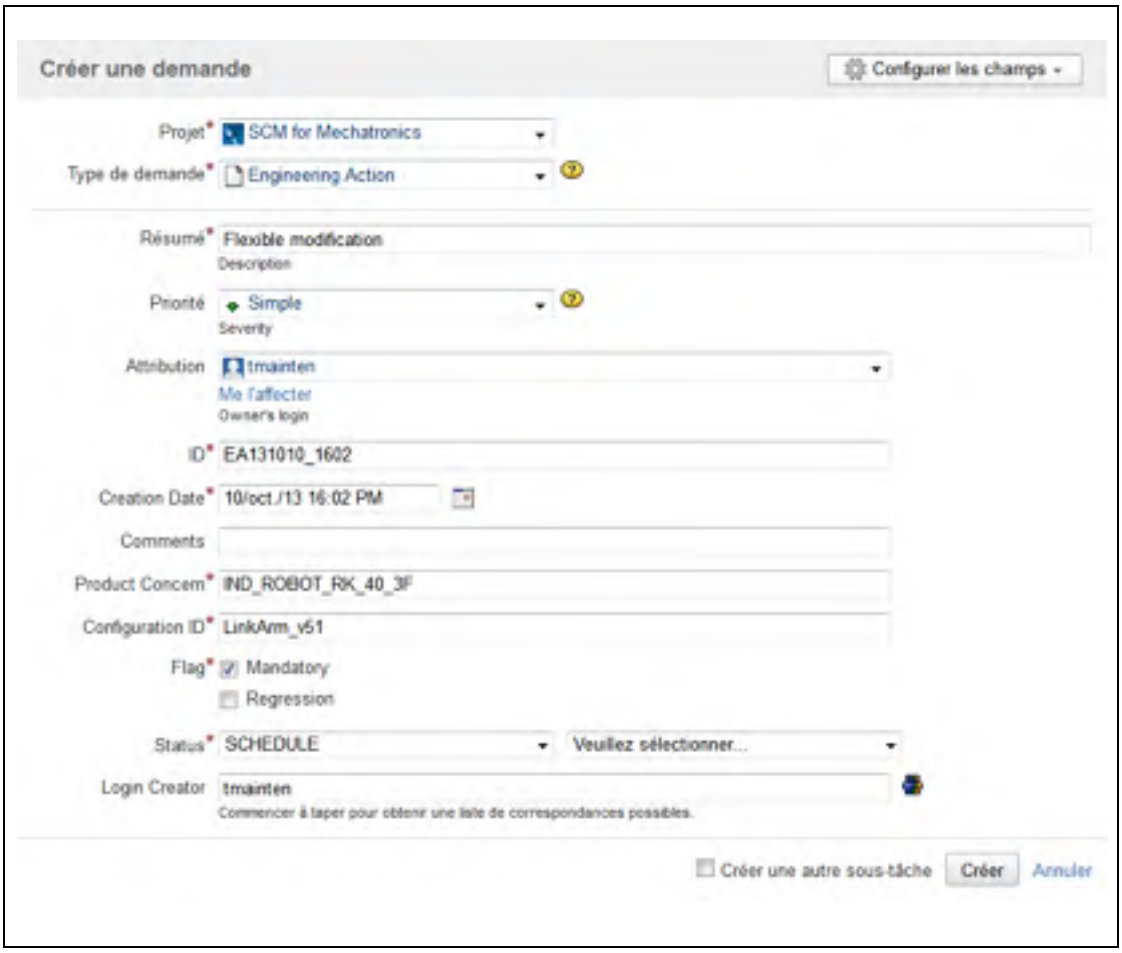


Figure 4.17 Formulaire de création d'une demande d'action

JIRA offre également la possibilité de raffiner des demandes d'actions en sous actions, ou sous tâches également appelées opérations dans ce manuscrit. La Figure 4.18 présente la création de deux de ces sous-tâches, précisant ainsi le travail à réaliser par MDESIGN1 (a) et EDESIGN1 (b) dans le cadre de l'action de maintenance du scénario. Ces deux sous tâches sont rattachées à la demande portant l'identifiant SFM-1 dans JIRA.

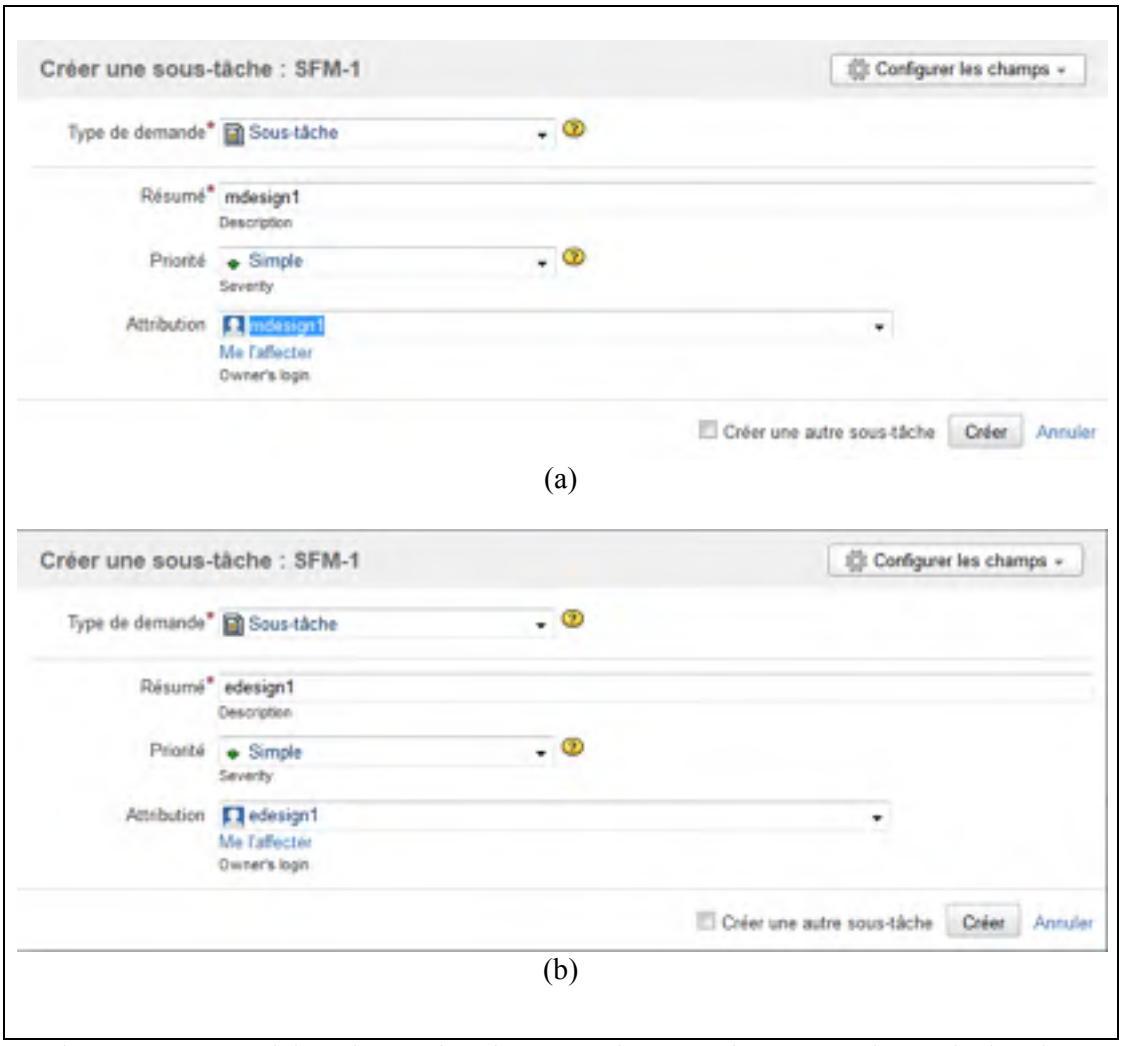


Figure 4.18 Formulaires de création de sous tâches rattachées à une demande d'action

Lorsque ce travail de déclaration est réalisé, différents types de consolidation sont automatiquement réalisés. Les « *backlog*s produit », décrites dans l'ANNEXE I, des différents concepteurs sont mises à jour, ainsi que les tableaux récapitulatifs des demandes affichables à tous les niveaux de la hiérarchie. La Figure 4.19 présente un récapitulatif des demandes en cours dans le cadre de notre projet. La colonne « rapporteur » correspond à la personne qui va suivre l'action et qui devra valider la clôture à l'issue du travail.

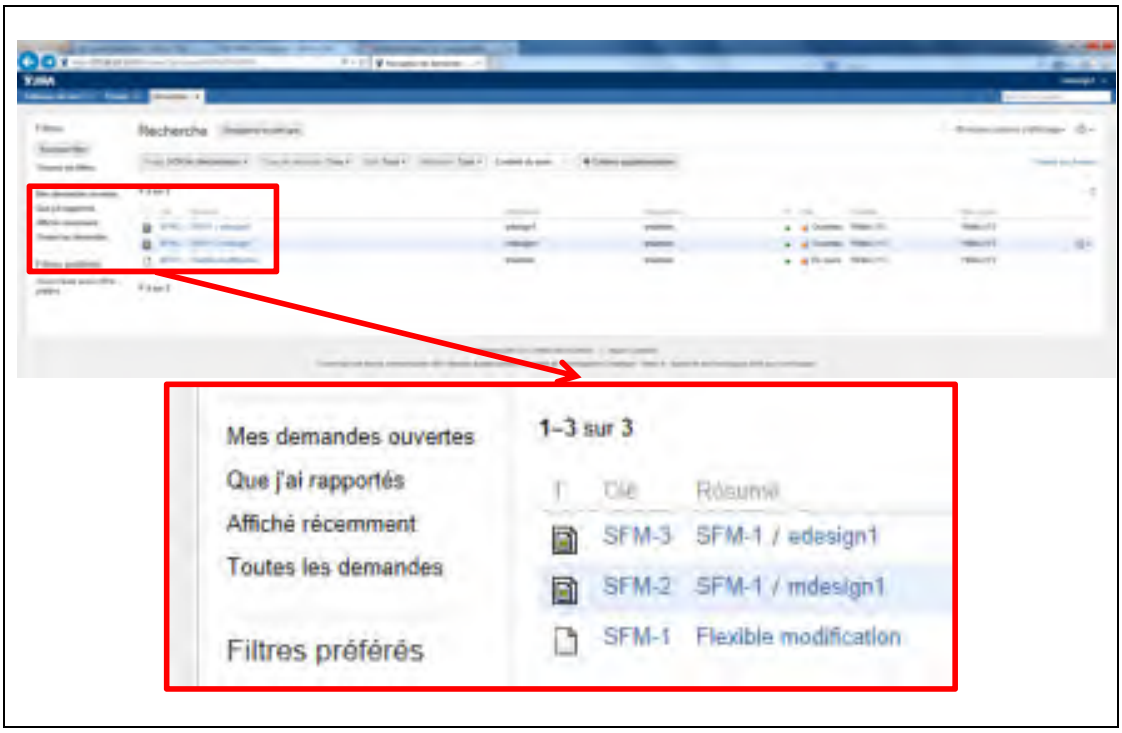


Figure 4.19 Récapitulatif des demandes ouvertes dans le cadre du projet « SCM for Mechatronics »

Une fois les différentes tâches réalisées, TMAINTEN peut clore la demande. Grâce au lien pouvant être réalisé entre JIRA et SVN, il a la possibilité de voir les fichiers qui ont été modifiés pour réaliser cette action. La Figure 4.20 présente la sous tâche de l'action de maintenance nommée « SFM-3 ». On y retrouve dans l'onglet Subversion les références des fichiers modifiés dans le cadre de cette sous tâche. Ces modifications ont été réalisées dans le repository « scm4cad » de SVN. Un fil d'activités y présente également l'enchaînement des opérations et changements de statut réalisés dans le cadre de cette action.

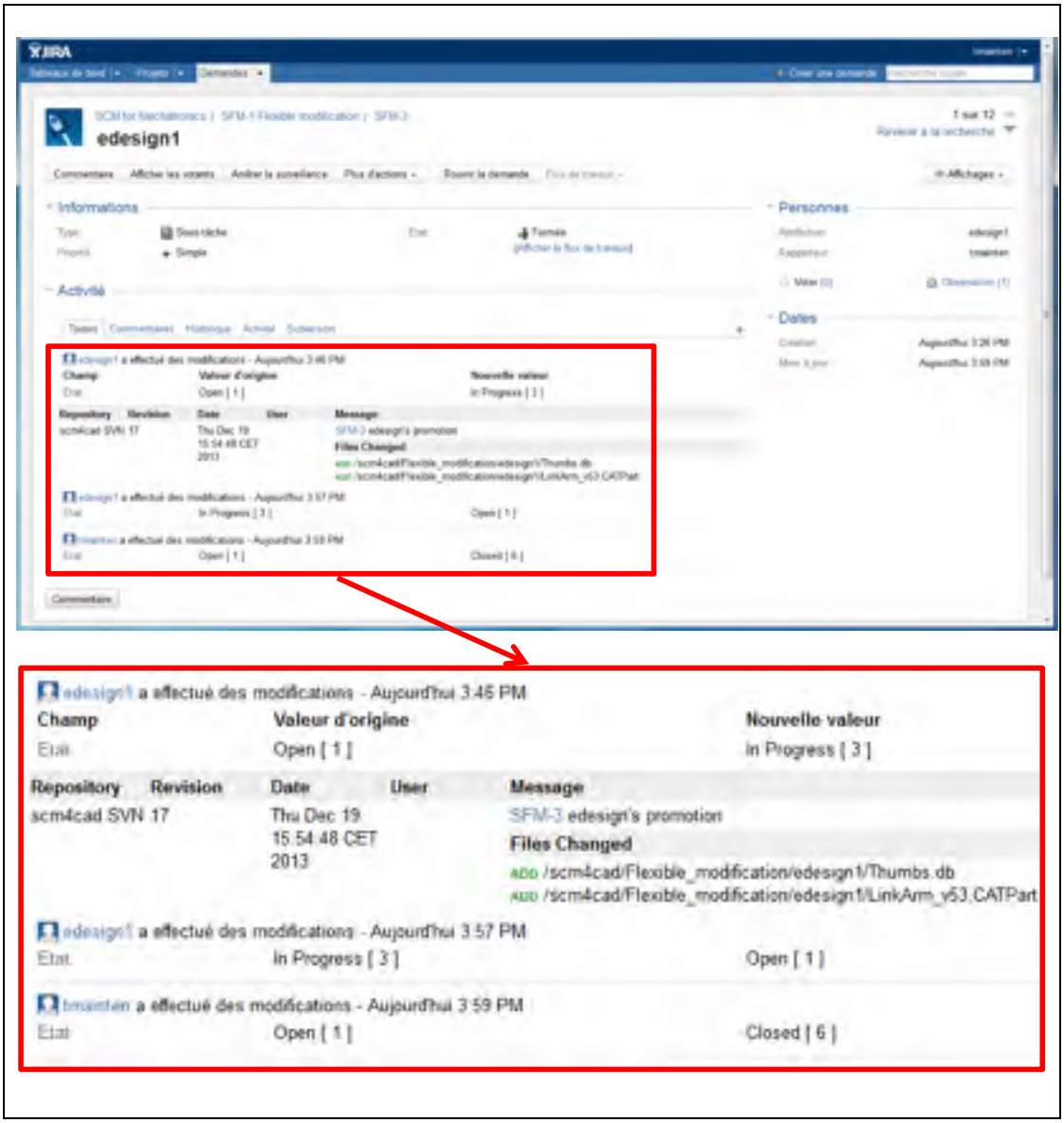
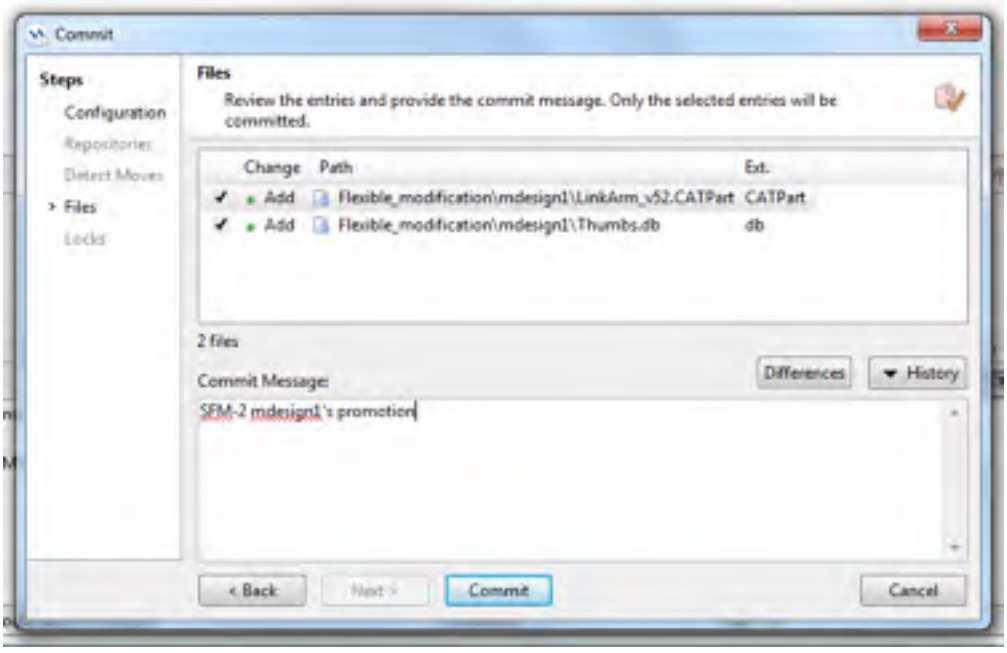


Figure 4.20 Matérialisation du lien données/décision : affichage des fichiers modifiés stockés dans SVN depuis JIRA

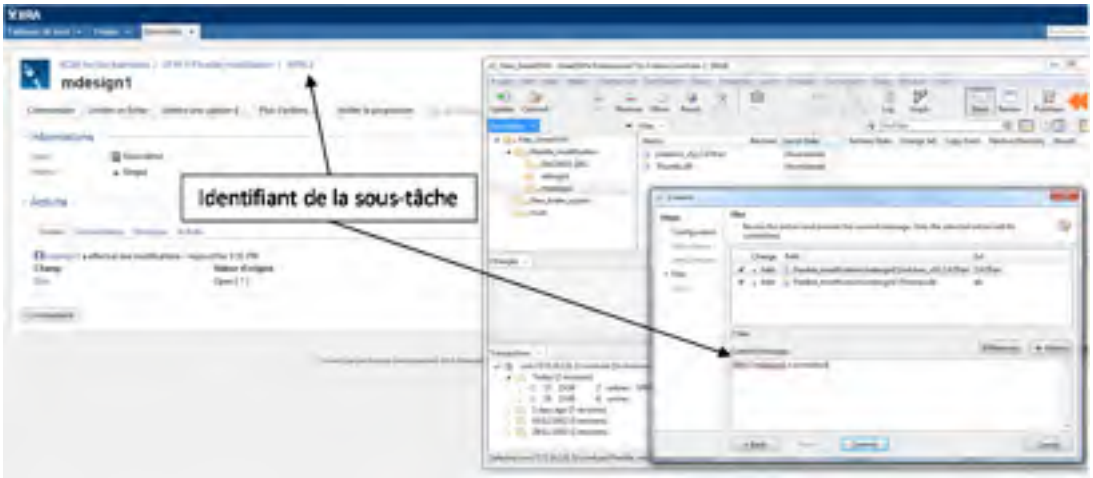
Afin de réaliser ce lien, le concepteur a uniquement dû, lors de l'intégration de ses modifications (nommé *commit* sous SVN), uniquement préciser la référence de la sous tâche. La Figure 4.21 illustre la façon dont ce lien données/décision est créé. Dans la fenêtre de *commit* (a), la référence SFM-2 en provenance de JIRA (b) est précisée.

Pour être tout à fait précis, il est nécessaire de préciser que SVN est un serveur sans aucune interface graphique. Pour manipuler les fichiers et réaliser les opérations SVN plus aisément, nous avons utilisé le logiciel SmartSVN[25] de la société WANDISCO sur les postes clients des concepteurs.

[25] http://www.wandisco.com/smartsvn/home



(a) Zoom sur la fenêtre de *commit* SVN



(b) Création du lien entre le *commit* SVN et la sous tâche

Figure 4.21 Création du lien données/décision : saisie de l'identifiant de la sous tâche lors du commit dans JIRA

À l'issue du travail, TMAINTEN clot la demande après avoir vérifié les modifications réalisées. Un récapitulatif est alors proposé et archivé pour une éventuelle analyse a posteriori. La Figure 4.22 présente ce récapitulatif.

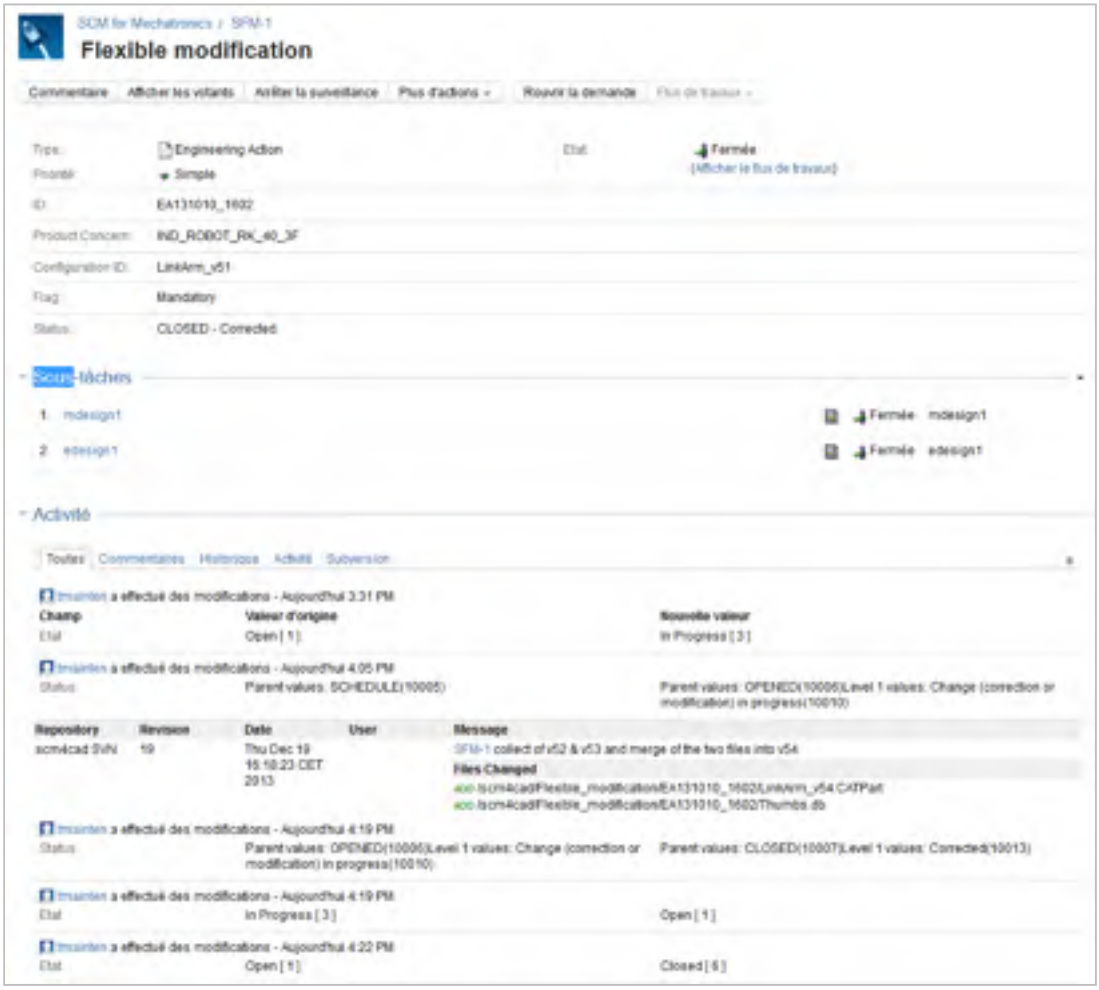


Figure 4.22 Récapitulatif de l'ensemble des informations liées à la demande de maintenance

4.2.3 Synthèse concernant le prototype

Dans cette section, l'implémentation du scénario au sein du prototype basé sur deux logiciels du marché a été illustrée. L'accent a été placé sur la réalisation du lien données/décision qui permet de suivre les modifications apportées sur les données afin de

réaliser une action donnée. Les deux actions détaillées dans le scénario et les sous-tâches associées ont été saisies dans JIRA et la modélisation CAO du bras de robot a été utilisée pour montrer les évolutions de versions dans SVN. Seule la fusion des versions v52 et v53 en v54 a été réalisée manuellement, en l'absence d'outil de *merge* adéquat. Mais si ce type d'outil venait à être développé un jour, il serait tout à fait envisageable de venir l'associer avec un client comme SmartSVN comme l'illustre laFigure 4.23.

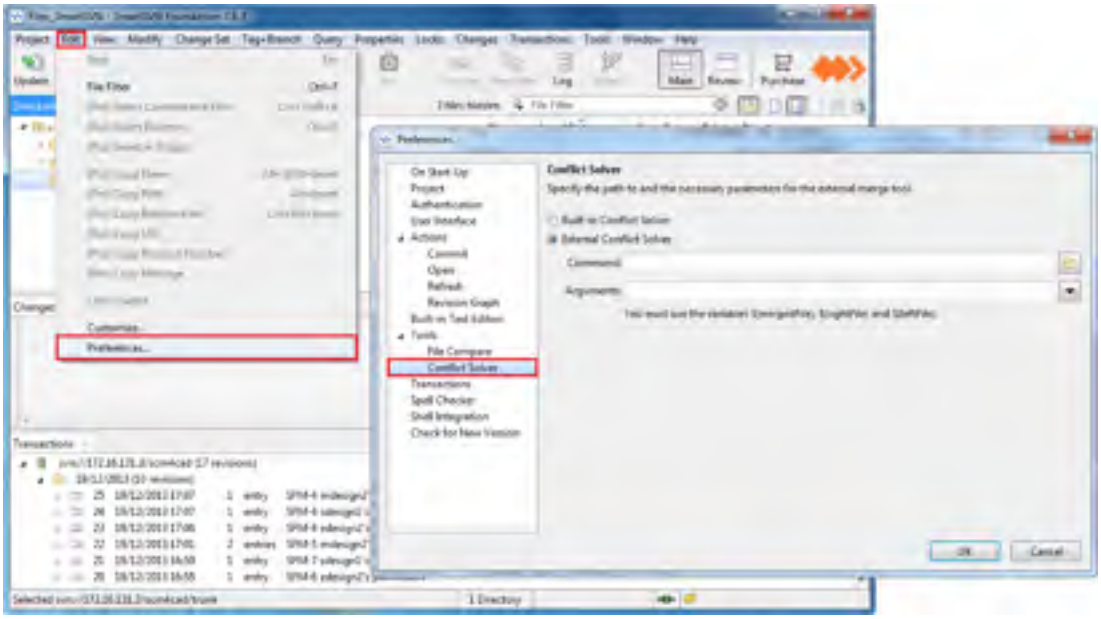


Figure 4.23 Paramétrage de SmartSVN pour pouvoir venir associer un outil de *merge* externe

Ainsi, deux des trois concepts, à savoir le *Collaborative Actions Framework* et le *Branch & Merge* ont effectivement été implémentés dans ce prototype. Le concept de WS n'est donc pas intégré à ce prototype informatique et n'a été illustré que via le scénario décrit dans la section précédente.

Dans la section suivante, nous revenons sur les apports du scénario et du prototype informatique qui permettent de juger de la pertinence de notre proposition en regard de notre question de recherche.

4.3 Synthèse relative à la mise en œuvre d'une méthode agile pour la conception collaborative multidisciplinaire

Le scénario et le prototype informatique présentés dans la section précédente illustrent la mise en œuvre des concepts issus de notre proposition. Afin de juger de la pertinence de ces derniers, nous tentons dans cette section de comparer leur usage aux solutions actuellement proposées, notamment par les PDM.

Les actions collaboratives

Les actions collaboratives ont été utilisées dans ce scénario et dans ce prototype afin de détailler les demandes liées aux deux projets, de les formaliser, d'en assurer le suivi, de tracer le lien avec les modifications apportées aux données techniques, d'assurer la collaboration entre des acteurs issus de disciplines différentes et de conserver un historique de ces échanges. Sans l'usage de ce concept, seuls certains de ces différents points auraient pu être supportés par les fonctionnalités offertes par les différents éléments d'un système d'information classique. Les paragraphes suivants visent à détailler la manière dont ces différents points auraient pu être traités dans le cadre d'un tel système d'information.

Dans le cadre du projet lié à la maintenance opérationnelle, les équipes en charge de ce type d'activité ont généralement à disposition un outil permettant de réaliser le suivi des incidents. Ils permettent d'assurer le suivi des demandes (ou tickets) formulées par les clients, de pouvoir faire évoluer le dossier en ajoutant toutes les informations qui lui sont relatives et de permettre de renseigner le client, quel que soit l'interlocuteur. Il ne s'agit pas de solutions de type gestion de relation client ou *Customer RelationShip Management* – CRM, mais de solutions dédiées à la maintenance. Ce type de système n'est généralement pas transversal à l'entreprise et seule l'équipe en charge de la maintenance assure l'interface entre le client et le reste des équipes. En se basant sur l'usage de ce type de système, la formalisation de la réclamation du client devient le socle des échanges entre les différentes équipes, permettant ainsi de n'avoir qu'un unique référentiel pour le partage d'information.

Une fois les demandes liées aux deux projets formalisées, la plupart du temps dans des documents de type bureautique, une officialisation des projets peut mener à l'intégration des travaux à mener dans le planning du bureau d'études, voire des différentes équipes impliquées. Ces plannings peuvent être gérés sur des tableaux disponibles au sein du bureau d'études, sur des tableurs, sur des logiciels de gestion de projet ou directement dans les modules adéquats dans les PDM. Néanmoins, si ces travaux ne sont pas considérés comme suffisamment importants, cette officialisation n'a pas nécessairement lieu et la communication informelle est alors utilisée pour transmettre les informations. Les courriels et le téléphone servent alors à échanger les informations nécessaires aux modifications. Dans le cas de notre scénario, le projet lié à la maintenance pourrait être traité de façon informelle, i.e. déclaré à son lancement et à sa clôture sans bénéficier d'un suivi régulier. Au contraire, le second projet, lié au système de freinage, est d'une envergure plus importante et bénéficierait certainement d'un suivi plus régulier avec des rapports d'avancement formalisés. Ces explications ont pour but de montrer que selon l'envergure des projets à mener, les processus mis en œuvre ne sont pas uniformisés, et de nombreux échanges interviennent de façon informelle sans aucune traçabilité associée. Le suivi de ces projets d'envergure moins importante (ce qui ne signifie pas que les responsables ne sont pas au courant et qu'ils n'approuvent pas ces projets) peut alors devenir problématique. Il semble également nécessaire de noter que dans le cadre de l'usage du *Collaborative Actions Framework*, le téléphone et les courriels n'ont pas à être bannis. Le courriel semble pouvoir, dans la majorité des cas, être remplacé par le champ commentaire de la *Collaborative Action* pour les échanges qui concernent l'action, alors que les échanges oraux restent, à notre sens, nécessaires.

Concernant le lien entre les décisions et les modifications apportées aux données techniques, aucune solution ne nous semble exister réellement à ce jour pour mettre en œuvre ce lien. Une exception concerne néanmoins les processus de gestion du changement ou Engineering Change Management (ECM) qui formalisent généralement dans les demandes de changement ou *Engineering Change Order* (ECO) les raisons des modifications, les documents techniques à modifier et les modifications à apporter.

En synthèse, le *Collaborative Actions Framework* permet d'assurer la collaboration entre des acteurs issus de disciplines différentes et de conserver un historique de ces échanges.

Sans son usage et comme décrit plus tôt, de nombreux échanges entre les acteurs ont lieu de façon informelle et ces échanges ne sont ni connus du système d'information, ni tracés, ni exploités. Pour réduire le nombre d'échanges informels et promouvoir l'usage du *Collaborative Actions Framework*, il est nécessaire que ce dernier reste très simple d'utilisation et très peu consommateur de temps. Une action collaborative doit être échangée comme le serait un courriel.

Le paragraphe suivant s'attache à comparer les solutions commerciales actuelles au concept de *workspaces* afin de montrer l'apport de ces derniers dans le cadre de la conception multidisciplinaire.

Les workspaces

Les *workspaces* ont été présentés comme des espaces permettant aux concepteurs de sélectionner et isoler certaines données, parmi les données disponibles, le temps nécessaire à la réalisation d'une action. Ces fonctionnalités sont classiquement disponibles dans les PDM commerciaux actuels. En revanche, les mécanismes de structuration des WS entre eux, les mécanismes d'échanges avec les éventuelles opérations de validation successives n'existent pas à notre connaissance dans les PDM actuels, qu'ils soient mécaniques ou électriques/électroniques. Les échanges directs entre WS grâce aux opérations de publication/import permettent également de véhiculer les informations liées aux modifications « au plus tôt ».

Qui plus est, les WS sont ici proposés comme un élément venant en complément des solutions de gestion de données techniques utilisées par les différentes disciplines. Ceci signifie qu'ils sont capables de contenir des données issues de solutions de gestion de données hétérogènes. Les créations de nouvelles données ou modifications/corrections/suppressions réalisées dans le cadre d'une action spécifique peuvent donc être partagées entre les différentes disciplines au plus tôt, et ce même si les données ne sont pas encore considérées comme matures. Les mécanismes d'intégration ne sont plus subis mais peuvent être anticipés.

Le branch & merge

Le dernier concept issu de la proposition et à comparer aux solutions du marché est le *branch* & *merge*. Ce dernier est à l'origine des possibilités de gestion en parallèle de différentes branches, ce qui rend possible la modification synchrone des données. Comme nous l'avons vu dans la proposition, il permet à différents acteurs de réaliser des modifications sur les mêmes données, ce qui a pour effet de décloisonner les données. En effet, les activités de conception restent parfois basées sur un découpage et une répartition stricts des tâches, les données créées par un concepteur devenant alors sa « propriété ». Ce concepteur devient alors la personne à contacter en cas de modification. Dans le cas de composants à l'interface entre disciplines, le fait que la même donnée puisse être éditée par les acteurs des différentes disciplines permet à chaque expert de pouvoir réaliser précisément ce dont il a besoin, quitte à ce que le responsable de la donnée vienne finalement reprendre et mieux intégrer les contributions ultérieurement. Aucun outil commercial connu à ce jour ne permet de venir fusionner des modifications réalisées de façon synchrone à partir de la même donnée technique. Ce type de fonctionnalité est offert par certains logiciels de bureautique comme Adobe Acrobat[26] ou Microsoft Word[27], mais très peu de logiciels d'ingénierie présentent des fonctionnalités comparables.

Ce chapitre a permis d'illustrer la pertinence de notre proposition en regard de notre question de recherche via la description d'un scénario basé sur un domaine fortement multidisciplinaire, la mécatronique. Ce scénario a ensuite été implémenté dans un prototype informatique. Enfin, une comparaison entre les concepts issus de la proposition et les solutions présentes actuellement dans les logiciels commerciaux a été proposée. Dans le chapitre suivant, une conclusion rappelant les objectifs des travaux et les principales contributions est présentée. Elle est suivie d'une section décrivant les différentes perspectives envisagées à court et moyen termes afin de venir enrichir les connaissances issues de ces travaux.

---

[26] http://www.adobe.com/fr/products/acrobatpro.html
[27] http://office.microsoft.com/fr-fr/word/

CHAPITRE 5

Conclusion et perspectives

5.1 Conclusion

Les travaux présentés dans ce manuscrit s'intéressent à la conception multidisciplinaire de systèmes intégrés. Le terme intégration fait ainsi référence à l'intégration produit, qui se décompose en deux types d'intégration, nommés intégration spatiale et intégration fonctionnelle. L'intégration spatiale répond notamment à des problématiques de poids et de compacité, alors que l'intégration fonctionnelle vise l'agrégation de fonctions, voire dans certains cas leur dématérialisation. Cette dernière entraîne un développement de plus en plus important de la partie logicielle intégrée aux systèmes développés. Les produits traditionnellement mécaniques se transforment alors progressivement, par l'ajout d'actionneurs et de multiples capteurs, nécessitant des traitements de données et des apports électriques rendus possibles grâce à l'introduction de modules électroniques et logiciels. Des fonctions de communication peuvent également être ajoutées afin de rendre l'interaction entre les différents sous-systèmes possible. Cette mutation des systèmes est portée par l'avènement de la mécatronique d'une part, et le développement des Cyber-Physical Systems de l'autre (Cf. Figure 1.20). Ces travaux de thèse s'inscrivent pleinement dans cette convergence mécatronique/CPSs, PLM/ALM et Industrial Engineering/Systems Engineering.

Cette intégration produit est rendue possible grâce à l'intégration relative aux activités d'ingénierie et aux disciplines. Ces intégrations sont la source d'une complexité organisationnelle, liée notamment à des phénomènes de couplages ou interdépendances existants au niveau des données techniques, des expertises et des physiques mises en jeu. L'intégration produit rend la collaboration des différents experts nécessaire et une simple division du système associée à une répartition des tâches n'est pas suffisante. Nos travaux visent tout particulièrement à favoriser cette collaboration entre les différentes disciplines, particulièrement difficile à qualifier, à mesurer et donc à organiser.

Peu d'alternatives existent afin de structurer la collaboration entre les différentes équipes pour organiser la conception d'un produit, d'un système ou d'un couple produit/service. La gestion de projet, basée sur un découpage du système à concevoir, sur une segmentation du projet en phases entérinées par des jalons et sur une répartition des tâches en fonction des compétences et des charges de travail des différents acteurs, s'est imposée dans l'industrie comme un véritable standard (Cf. Figure 1.12). Au-delà de ses nombreux avantages, cette organisation présente néanmoins quelques limites. La première tient au fait que l'organisation par projet, voire les plateaux projet, ne suffit pas à pallier au problème de cloisonnement entre les disciplines impliquées. Les directives pour ces équipes proviennent alors des décisions prises par les responsables du projet, dans une approche essentiellement *top-down*. La seconde limite tient au principe même de contrôle de l'avancement du projet. Due à la problématique de la maîtrise des risques, le déroulement des projets est de plus en plus rationnalisé. Ainsi, lors de la conception d'un nouveau produit, la structure des projets, la durée de leurs phases principales et les livrables attendus sont standardisés jusqu'à rendre ce mode d'organisation rigide et ne laissant que peu de place aux initiatives et à l'innovation.

Face à ces constats, ces travaux ont cherché à développer les connaissances nécessaires à la structuration et au pilotage de la collaboration multidisciplinaire. Cet objectif se divise en trois sous objectifs, qui sont : 1) d'améliorer la collaboration entre les différentes disciplines en réduisant le cloisonnement existant entre les acteurs qui en sont issus, 2) de rendre la prise de décisions dans le cadre d'un projet de conception plus aisée grâce à la remontée fréquente et régulière d'informations opérationnelles précises et 3) de permettre la traçabilité entre les évolutions apportées aux données de définition du système conçu et les décisions relatives à la conception de ce système. Ces objectifs s'inscrivent dans le cadre d'une vision idéale où des connaissances relatives à l'organisation, à la structuration des données et à l'adaptation de vues spécifiques à chacun des couples métier/discipline seraient disponibles pour spécifier une plateforme de collaboration et de gestion de données multidisciplinaire et dynamique.

Peu de travaux répondent finalement à ces objectifs d'un point de vue organisationnel. De nombreuses approches traitant de la structuration des données pour favoriser l'intégration des activités d'ingénierie existent. D'autres traitent de l'intégration entre deux disciplines

spécifiques comme la mécanique et l'électronique à un niveau opérationnel. Mais si l'on s'attache à vouloir faciliter l'intégration multidisciplinaire de façon globale lors des phases de conception, en s'appuyant sur des méthodes organisationnelles et sur la dynamique de l'information pouvant exister entre les différentes équipes, la littérature scientifique, tout comme les solutions commerciales disponibles, semblent particulièrement pauvres. Pour arriver à cette conclusion, un tableau à double entrée permettant de structurer l'état de l'art a été proposé dans ces travaux (Cf. Tableau 2.1). Ces deux « entrées » sont basées sur deux facteurs de positionnement. Le premier a pour but de présenter le type de contribution et est inspiré d'un patron d'architecture intitulé *Model-View-Controller*. Le second facteur s'attache quant à lui à définir la portée de la contribution qui peut être stratégique, tactique ou opérationnelle.

Notre approche est basée sur l'idée que des principes sous-jacents à l'ensemble des méthodes agiles, utilisées lors de la conception de logiciels, sont pertinents pour améliorer cette collaboration multidisciplinaire, tout du moins pour les phases de conception préliminaire et détaillée étudiées lors de ces travaux. Ces principes ont donc été scrutés, triés et synthétisés en trois points qui ont orienté et structuré nos travaux (Cf. Tableau 2.6). Le premier préconise le partage régulier des données de conception. Dans une logique de transparence et afin de rendre le travail de chacun visible, les mises en commun de données techniques de définition du produit doivent être précoces, fréquentes et continues. Le second recommande de laisser l'initiative aux acteurs opérationnels. Il rappelle l'importance du fait que l'initiative doit pouvoir provenir des concepteurs et qu'ils doivent pouvoir échanger librement autour des problématiques rencontrées. Le troisième et dernier point concerne la nécessaire mise en œuvre d'indicateurs opérationnels « au plus proche du terrain » pour permettre des prises de décision éclairées.

Sur la base de ces trois points de synthèse, une proposition formée de trois concepts imbriqués telles des poupées russes (Cf. Figure 3.1) a été décrite dans le chapitre 3. Le premier concept, le plus général, est intitulé *Collaborative Actions Framework*. Il correspond à un cadre de collaboration opérationnelle autour d'actions. Un des objectifs de ce *framework* est de faciliter l'échange d'informations et donc la collaboration des acteurs des projets de conception, quelle que soit leur origine disciplinaire. Il permet

également, grâce aux liens existants avec le second concept intitulé *Workspace*, d'assurer une traçabilité entre les prises de décision et les corrections/modifications apportées sur les données techniques. Il fournit également une multitude d'informations concernant les actions à mener, les actions déjà réalisées, les échanges effectués entre les différentes disciplines, la charge de travail des différents acteurs, etc. Ces informations peuvent ensuite être consolidées et utilisées comme de véritables indicateurs opérationnels permettant de prendre des décisions régulières en lien direct avec le terrain.

Les *workspaces*, sorte d'espaces de collaboration, offrent quant à eux différentes possibilités. En plus des capacités traditionnelles de sélection et d'isolement des données, ils permettent de mettre en commun de façon continue les travaux issus des différentes disciplines ou encore d'intégrer et de valider leurs contributions. Les échanges de données techniques entre les *workspaces* ou le travail simultané sur les mêmes données techniques s'appuient sur la possibilité de pouvoir gérer de façon parallèle différentes versions d'une même donnée technique. Ces différentes versions peuvent être organisées, selon leur évolution, en branches. L'action de venir réconcilier les versions, ou mutualiser les différentes intentions de conception, est alors appelée une fusion. Ces deux opérations regroupées sous l'appellation *branch & merge* constituent le troisième et dernier concept de notre proposition. Il permet notamment à différents acteurs de travailler sur les mêmes données, de façon synchrone ou asynchrone, et participe au décloisonnement des disciplines.

Grâce à ces trois concepts imbriqués, les acteurs opérationnels des différentes disciplines sont en mesure de prendre des initiatives pour résoudre collectivement des problèmes de conception qui auraient peut-être été résolus plus difficilement à l'échelle d'une seule et même discipline. Le partage des problématiques rencontrées, les initiatives laissées aux concepteurs sont autant de facteurs pouvant favoriser l'intégration des activités d'ingénierie et des disciplines, conduisant à une meilleure intégration produit. Les actions collaboratives, utilisées comme support pour échanger des informations entre les acteurs, sont également de formidables opportunités pour consolider et exploiter des indicateurs opérationnels, qui font actuellement cruellement défaut. Ces actions collaboratives sont le reflet des décisions de pilotage du projet à l'échelle opérationnelle. Elles peuvent être hiérarchisées, priorisées et permettent ainsi un pilotage fin et réactif aux aléas du projet.

Cette solution permet ainsi de palier aux lacunes mentionnées plus tôt, à savoir le pilotage du projet par le contrôle de l'avancement basé sur des modèles de projets standardisés et rigides. L'introduction de ce mode de pilotage flexible et agile permet également dans une certaine mesure d'introduire une part d'initiative et de créativité supplémentaire dans le processus de conception.

Le décloisonnement des équipes est favorisé par les échanges pouvant intervenir via les actions collaboratives, mais également via la mise en commun précoce, fréquente et continue des données techniques de définition du produit. L'abandon des systèmes de gestion de données techniques spécifiques à chaque discipline ne semble pas envisageable à court ou moyen terme. Mais les WS peuvent se révéler des espaces de collaboration multidisciplinaires neutres, venant « piocher » les données dans chacun des systèmes et proposant des espaces d'intégration et de validation pouvant être organisés selon les besoins. Chacune des opérations d'échange entre les différents WS se révèle alors porteur d'une signification bien particulière, pouvant être exploitée à des fins de test, de validation, d'intégration précoce, etc. Ces échanges multiples s'appuient sur des outils de versionnement autorisant la création de branches, puis la fusion de ces branches, selon les intentions de conception des différents concepteurs. Pour supporter ces activités de création et de fusion de branches, des outils de *merge* spécifiques à chaque type de donnée sont nécessaires. Les récentes techniques concernant les *3-way merge* semblent ouvrir des perspectives quant à la réalisation de ces outils pour tous les types de données utilisés par les concepteurs issus des différentes disciplines.

Les Tableau 3.1 et Tableau 3.2 rappellent le positionnement général de notre proposition par rapport aux points de synthèse issus des méthodes agiles et à nos objectifs de thèse. Ils décrivent la couverture de ces trois concepts par rapport aux objectifs fixés et illustrent que les objectifs sont principalement atteints grâce au concept *Collaborative Actions Framework*. Plus précisément, l'objectif 1 (« Améliorer la collaboration entre les différentes disciplines en réduisant le cloisonnement existant entre les acteurs qui en sont issus ») est atteint grâce au partage d'information uniformisé assuré par le *Collaborative Actions Framework*, grâce à la mise en commun de données multidisciplinaires via les *workspaces*, et grâce au décloisonnement des métiers via le *branch & merge*. L'objectif 2 (« Rendre la prise de décisions dans le cadre d'un projet de conception plus aisée grâce à

la remontée plus fréquente et plus régulière d'informations opérationnelles précises ») est atteint grâce aux indicateurs créés à partir des informations portées par les actions collaboratives et par les espaces d'intégrations que sont les WS. Enfin, l'objectif 3 (« Permettre la traçabilité entre les évolutions apportées aux données de définition du système conçu et les décisions prises tout au long du projet de développement ») est atteint grâce au référencement des données impactées par chaque action collaborative.

Le Tableau 3.3 permet quant à lui de montrer que notre approche est largement positionnée dans une approche *controller*, signifiant que l'on contribue avant tout à la dynamique des échanges. Le positionnement partiel dans la colonne *model* rappelle également le lien entre données et décisions et les possibilités offertes pour le décloisonnement des données issues de chaque discipline.

En synthèse, nos contributions scientifiques principales ont donc trait aux mécanismes d'intégration dans le cadre de la conception de systèmes nécessitant la collaboration de différentes disciplines. Basé sur les principes sous-jacents à l'ensemble des méthodes agiles, nous proposons un cadre de collaboration (méthodologie et outils) facilitant le pilotage de la conception grâce à la prise en compte d'indicateurs opérationnels issus des échanges entre les différents concepteurs et permettant d'assurer la traçabilité entre ce pilotage et les évolutions des données techniques relatives au produit. Dans la section suivante, nous nous attachons à exposer les limites et les perspectives identifiées pour ces travaux de recherche.

5.2 Limites et perspectives

Dans cette section, les limites et perspectives liées à nos travaux sont organisées autour de trois axes. Le premier s'attache à présenter les limites et les perspectives d'amélioration de la maturité du démonstrateur. Le second propose deux nouvelles hypothèses qui pourraient permettre de poursuivre les objectifs de ces travaux. Le troisième et dernier axe présente un nouveau champ d'application envisagé pour nos travaux.

5.2.1 Limites et maturité du démonstrateur

Afin de juger de la faisabilité de notre proposition, les concepts qui en sont issus sont appliqués dans le cadre d'un scénario mécatronique et implémentés dans un prototype informatique basé sur deux solutions commerciales, l'une implémentant principalement le concept d'actions collaboratives et l'autre le concept de *branch & merge*. Les limites principales de ce démonstrateur résident dans sa faible ergonomie liée à l'usage détourné d'une solution logicielle existante qui ne permet pas de réaliser une interface complètement adaptée à nos besoins. De même, le lien créé entre les données techniques modifiées et les actions collaboratives réalisées n'est pas facilement exploitable car aucun outil n'a été développé pour analyser ces dépendances. Un outil de visualisation des versions d'un même CI, sorte de graphe interactif, précisant sur chaque branche la raison de la modification, pourrait ainsi être proposé. Enfin, la dernière limite notable de ce démonstrateur informatique réside dans le fait que le concept de WS n'a pas été implémenté.

Afin de faire évoluer ce démonstrateur, les travaux portant sur le *Collaborative Actions Framework* sont actuellement poursuivis dans le cadre du laboratoire commun DIMEXP entre la société DeltaCAD[28] et l'UTC[29]. Les travaux présentés dans ce manuscrit sont en effet à l'origine d'un des deux axes de recherche de ce laboratoire. Dans ce cadre, la

[28] http://www.deltacad.fr/

[29] Laboratoire commun DIMEXP, dans le cadre de l'appel à projet « LabCom », financé par l'Agence Nationale de la Recherche (Référence ANR-13-LAB1-0006-01).

maturité de ce concept a récemment fait l'objet d'une évaluation grâce à l'échelle *Technology Readiness Level* (TRL), adaptée par la Commission Européenne pour ses appels à projets et rappelée sur la Figure 5.1. La conclusion de cette évaluation a été de positionner le *Collaborative Actions Framework* au niveau 4 de cette échelle. Après avoir démontré l'intérêt du concept (niveau 3), les principales fonctions ont été implémentées sur la base d'une infrastructure *Open Source*, et des interfaces graphiques dédiées sont en cours de développement. Ce partenariat devrait ainsi permettre le passage du premier au second pilier (Cf. Figure 5.1) destiné à valoriser ces travaux sur un terrain industriel.

Figure 5.1 Échelle TRL pour l'évaluation de la maturité d'une technologie (EARTO - Association européenne pour les organisations de recherche et technologie, 2014)

Une seconde limitation du démonstrateur présenté dans le chapitre précédent concerne la réalisation d'un outil de type *3-way merge* pour des données de type CAO. Si la faisabilité technique d'un tel outil a été évoquée dans nos travaux, les développements associés n'ont pas été, à ce jour, très poussés. Pour mener ces développements, deux directions complémentaires pourront être choisies. La première vise à parcourir l'arbre de conception d'une pièce CAO de façon récursive et d'extraire les caractéristiques et valeurs des paramètres de l'ensemble des *features* qui constituent cet arbre. Ces caractéristiques seront ensuite stockées dans un fichier XML. Cette méthode sera appliquée aux deux versions conflictuelles ainsi qu'à l'éventuelle version parente afin d'alimenter un outil de comparaison de fichiers XML. Ce type de comparaison vise à rapidement réconcilier des versions ne présentant que de petites différences, comme par exemple la modification d'un rayon de congé, etc.

Ce type d'outil de *diff / merge* nécessite d'être couplé avec un outil de comparaison géométrique pour des modifications plus profondes, *i.e.* touchant le positionnement des *features*, les références des *features*, l'ajout / la suppression de *features*, etc. Certains

logiciels de CAO intègrent d'ores et déjà des fonctionnalités de *diff* géométrique. Mais elles restent limitées à des comparaisons qui ne prennent pas en considération la version parente et ne permettent en aucune mesure la réconciliation des versions concurrentes.

Bien que n'ayant pas été évalués formellement, nous estimons actuellement le TRL de ces outils au niveau 2 de l'échelle présentée sur la Figure 5.1. En effet, les principes des outils nécessaires à la mise en œuvre du concept *branch & merge* ont été présentés dans le chapitre 3, mais la preuve de concept n'a pas encore été réalisée. Une perspective serait donc de continuer nos efforts concernant ce concept afin de le pousser jusqu'au niveau 4 de l'échelle TRL.

Dans la section suivante, le second axe présente les nouvelles hypothèses qui pourraient permettre de poursuivre les objectifs de ces travaux.

5.2.2 Nouvelles hypothèses pour l'intégration multidisciplinaire

5.2.2.1 Une approche *model* complémentaire

Comme nous nous sommes attachés à le montrer dans le chapitre relatif à l'état de l'art, les différents types d'approches (*model*, *controller* et *view*) sont autant de leviers, souvent complémentaires, permettant de répondre aux problématiques inhérentes à la conception et à l'industrialisation de produits. Dans ces travaux, notre intérêt s'est principalement porté sur la partie organisationnelle et sur le lien existant entre les prises de décisions et les données. Les actions collaboratives y sont présentées comme des « points d'intérêts » permettant de structurer la collaboration. Elles sont des objets venant agréger les informations liées aux raisons de la demande d'action, à la qualification de cette demande, aux échanges ayant eu lieu et aux données impactées.

Dans une logique très proche, mais focalisée cette fois sur les données et non sur l'organisation, une hypothèse concernant l'apport de la définition et du suivi d'interfaces multidisciplinaires tout au long du processus de conception est actuellement en cours d'évaluation (Zheng et al., 2014a, 2014b, 2013). Ces interfaces, définies dès les phases amont, s'appuieront sur l'architecture du système et viendront préciser quelles

informations doivent être échangées entre les composants conçus par les différentes disciplines. Cette première étape s'appuiera sur les différentes techniques proposées par l'ingénierie système. Les interfaces seront ensuite qualifiées par la nature de l'information (géométrie, énergie, contrôle, données) qu'elles auront vocation à véhiculer et par les objets (composants, interfaces, environnement) qu'elles auront à mettre en relation. Ce typage permettra la vérification de l'état de maturité de l'interface et la compatibilité des informations échangées via cette dernière. Ces interfaces seront ensuite précisées et raffinées tout au long du processus de conception. Elles deviendront ainsi de nouveaux « points d'intérêts » à la frontière des différentes disciplines. Formalisées dans le modèle de données, elles feront l'objet de traitements particuliers, tant au niveau de leur gestion informatisée que lors des revues de projet.

La combinaison de la gestion des actions collaboratives et des interfaces a pour objectif une meilleure intégration multidisciplinaire. Elles s'attachent toutes les deux à porter les informations nécessaires à cette intégration autour d'objets manipulables par l'ensemble des concepteurs. Elles permettent la remontée d'informations opérationnelles afin de faciliter les prises de décision contribuant au pilotage général du projet de conception. Mais ces informations ont besoin d'être consolidées afin d'être transformées en indicateurs permettant de mieux appréhender les liens entre les données et les décisions prises dans le cadre du processus d'intégration multidisciplinaire. La thématique de la *Business Intelligence*, permettant la construction et l'exploitation, avant tout graphique, de ces indicateurs, est présentée dans la section suivante comme une nouvelle perspective.

5.2.2.2 Mieux comprendre le processus d'intégration : l'apport de la *Business Intelligence*

Les informations portées par les actions collaboratives et les interfaces décrites dans la section précédente demandent à être consolidées pour être pleinement exploitées. Mais cette consolidation, présentée notamment dans le cadre du *Collaborative Actions Framework*, ne fournit pas nécessairement nativement les informations dont a besoin un décisionnaire pour évaluer le niveau d'intégration du système conçu. Les informations portées par les interfaces multidisciplinaires et les actions collaboratives, couplées aux possibilités de construction et d'analyse d'indicateurs personnalisés proposées par les

solutions de type *Business Intelligence* (BI) semblent être une perspective intéressante pour mieux appréhender le processus d'intégration.

Ces systèmes de type BI sont basés sur un ensemble de modèles, méthodes et outils qui permettent de convertir des données « brutes » en informations pertinentes et utiles pour le contrôle de la performance. Negash présente ainsi les systèmes de type BI comme : « BI systems combine data gathering, data storage, and knowledge management with analytical tools to present complex internal and competitive information to planners and decision makers » (Negash, 2004). Les fonctions les plus couramment disponibles dans les systèmes de type BI sont : « reporting, On Line Analytical Processing (OLAP), analytics, data mining, process mining, complex event processing, business performance management, predictive analytics, and prescriptive analytics » (Bosch-Mauchand et al., 2014). Ainsi, des requêtes complexes et « sur mesure » peuvent être réalisées sur la base de données extraites de l'ensemble des systèmes d'information. Ces requêtes peuvent alors alimenter toutes sortes de formes graphiques avancées organisées sur des pages intitulées tableaux de bord ou *dashboards*. L'aspect temporel est également prédominant pour ces *dashboards* ; l'évolution du résultat des requêtes au cours du temps permet en effet de fournir des vues historiques, actuelles ou prédictives.

De nouveaux indicateurs de mesure de performance (*Key Performance Indicators* – KPI) pourraient alors être imaginés, implémentés et suivis grâce à ce type de système. Le pilotage pourrait ainsi devenir quasi-temps réel (aussi appelé « operational BI » or « just-in-time BI ») et permettrait de réduire le temps de latence entre le moment où les données opérationnelles sont acquises et celui où leur analyse est rendue possible (Chaudhuri et al., 2011).

Les problématiques sous-jacentes à cette thématique portent donc sur : 1) la définition des données brutes qu'il est possible d'obtenir à partir des différents éléments du système d'information utilisés dans le cadre des processus de conception-industrialisation, 2) la déclinaison de l'architecture informatique à mettre en œuvre pour collecter, stocker et exploiter les données et 3) la définition de KPI spécifiques au pilotage des projets ciblés.

Après avoir présenté les perspectives liées à la maturité de notre démonstrateur et les nouvelles hypothèses liées à nos travaux, la section suivante propose comme nouveau domaine d'application le secteur du « Bâtiment et Travaux Publics » (BTP), que le travail collaboratif et la maquette numérique (ou modélisation des données du bâtiment, *Building Information Modeling* – BIM) sont en train de révolutionner.

5.2.3 Nouveau domaine d'application : le secteur du BTP et le BIM

L'implication de différentes disciplines dans un processus de conception et de fabrication / construction n'est pas réservé aux domaines de l'industrie manufacturière. Des rapprochements sont réalisés depuis quelques années entre, par exemple, le domaine du BTP et ces industries manufacturière, notamment au niveau de la collaboration et du partage de données (Bouguessa et al., 2013; Bricogne et al., 2011). Bien que les caractéristiques du produit final, telles que le nombre d'unités produites ou le cycle de vie du produit final, etc. ne soient pas les mêmes, de nombreux points communs existent. Par exemple, la diversité des disciplines impliquées, le nombre grandissant et la spécialisation des acteurs, la nécessaire réduction de temps de mise sur le marché, la maîtrise des coûts de la conception à l'utilisation, la qualité et le respect des normes de plus en plus importantes, etc. sont autant de problématiques communes à ces domaines. Les connaissances acquises depuis plusieurs dizaines d'années sur les processus et les outils supports à la collaboration doivent ainsi pouvoir être transposés aux spécificités du domaine du BTP.

Dans le cadre de nos travaux, les actions collaboratives pourraient être ainsi utilisées pour structurer le partage d'information durant les phases d'appel d'offre / concours, de travaux (suivi et levées de réserves) ou encore d'exploitation du bâtiment. De façon plus détaillée, durant les phases d'appel d'offre ou de concours, les services internes de

commerce, étude de prix, méthodes, achat, structure, fluide, ainsi que les différents bureaux d'études techniques externes (*process*, structure, fluide etc.), et les architectes doivent se coordonner et échanger des informations structurées autour d'un projet de construction, idéalement recueillies au sein d'une maquette numérique de type BIM. Si le concours est remporté, les travaux prennent alors le relai et coordonnent une multitude d'acteurs internes et externes pour mener à bien la phase de réalisation du projet en respectant les contraintes du triptyque qualité / coût / délais. Les notions de traçabilité et de contrôle sont alors déterminantes et des *workflows* de validation pourraient alors être couplés à certaines actions collaboratives pour ajouter un côté systématique à certaines tâches. Une fois le bâtiment livré au client, les phases d'exploitation du bâtiment voient alors une nouvelle typologie d'acteurs intervenir. Les usagers, bien entendu, mais également les exploitants du bâtiment, les prestataires de services (entretien des bâtiments et des espaces verts, gestion des locaux techniques, des systèmes d'incendie, de sécurité, des droits d'accès, des énergies, …), etc. souvent externes (*facility management*[30]) interagissent alors. Afin de coordonner les demandes des usagers et de faire intervenir les prestataires extérieurs au moment prévu, pour une mission précise, avec les informations nécessaires et à jour, etc. les actions collaboratives semblent de nouveau des supports agiles et structurants pour cette forme de collaboration.

[30] http://en.wikipedia.org/wiki/Facility_management

5.2.4 Portée de nos travaux

Pour achever ce manuscrit, il semble nécessaire de noter qu'une idée est omniprésente à l'origine de nos travaux : remettre l'Homme au cœur des décisions. Alors que les systèmes d'information et les procédures sont de plus complexes, spécialisés et formels, il apparaît nécessaire de réintroduire une certaine souplesse, une certaine flexibilité, une certaine agilité dans certains processus d'entreprise tout en conservant le contrôle sur la qualité des produits et services conçus. La conception (surtout si elle est innovante!) est certainement une des activités nécessitant le plus cette flexibilité et cette agilité. C'est en ce sens que les outils proposés dans ces travaux deviennent le support non pas à une méthode, mais à des méthodes de pilotage, que chaque entreprise devra proposer, adapter à son contexte et à ses objectifs. Ces méthodes permettront, par un pilotage agile, une collaboration multidisciplinaire, décloisonnée voire participative et une meilleure circulation de l'information, de manière descendante et ascendante.

SOMMAIRE DES ANNEXES

ANNEXE I

Méthodes agiles

Cette annexe vise à préciser quelques caractéristiques des méthodes agiles considérées comme les plus en lien avec les travaux développés dans ce manuscrit. Elle est inspirée du livre de Sommerville intitulé « Software Engineering » (Sommerville, 2010) et de l'ouvrage de Messager intitulé « Gestion de projet: vers les méthodes agiles » (Messager, 2009).

« Une méthode agile est une approche itérative et incrémentale, qui est menée dans un esprit collaboratif, avec juste ce qu'il faut de formalisme. Elle génère un produit de haute qualité tout en prenant en compte l'évolution des besoins (…) » (Messager, 2009).

L'origine des méthodes agiles

Différents constats sont à l'origine des méthodes agiles. Parmi ces constats, on peut noter le changement rapide d'environnement, comme par exemple l'apparition de nouvelles opportunités de marché, l'émergence de produits/services concurrents ou complémentaires. Le logiciel est particulièrement soumis à ces changements rapides d'environnement, et les entreprises du domaine sont parfois prêtes à faire des concessions sur la qualité ou sur certaines exigences initiales pour permettre d'accélérer le développement. Un autre constat est qu'il est très fréquent que les spécifications du client changent à mesure que le développement progresse. Cela est lié au fait que les clients ne prennent conscience de la nature des fonctions développées que lorsqu'ils sont face au système en cours de réalisation.

Les méthodes agiles apparaissent dans la lignée des développements dits incrémentaux (Mills, 1980). Mais il faut attendre les années 90 pour voir apparaître les premières méthodes agiles, et même les années 2000 pour les méthodes les plus utilisées, à savoir Scrum (Schwaber and Beedle, 2001) et eXtreme Programming (XP) (Beck, 2000). Pour toutes ces méthodes, le logiciel est vu comme une somme d'incréments apportant à chaque fois de nouvelles fonctionnalités. Les phases de spécification, de

conception et d'implémentation sont alors mêlées et l'utilisateur final est activement impliqué pour tester successivement différentes versions afin d'obtenir des feedbacks réguliers.

Les principes des méthodes agiles sont précisés dans le tableau Annexe - Tableau I.1 ci-dessous :

Principle	Description
Customer involvement	Customers should be closely involved throughout the development process. Their role is to provide and to prioritize new system requirements and to evaluate the iterations of the system.
Incremental delivery	The software is developed in increments with the customer specifying the requirements to be included in each increment.
People not process	The skills of the development team should be recognized and exploited. Team members should be left to develop their own ways of working without prescriptive processes.
Embrace change	Expect the system requirements to change and so design the system to accommodate these changes.
Maintain simplicity	Focus on simplicity in both the software being developed and in the development process. Wherever possible, actively work to eliminate complexity from the system.

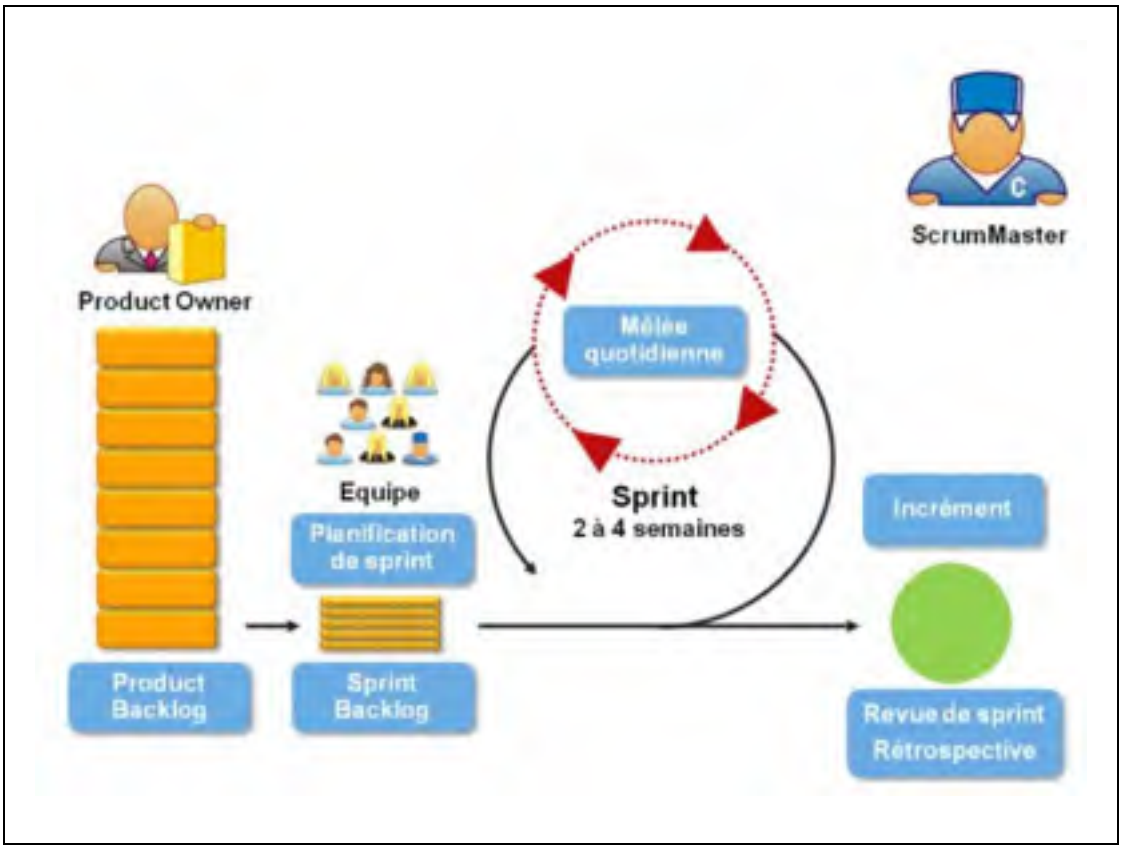
Annexe - Tableau I.1 Principes des méthodes agiles

Scrum

La méthode Scrum fournit un *framework* de management de projet basé sur la notion de *sprints*. Un *sprint* est une période pouvant durer entre une et quatre semaines pendant lesquelles les fonctionnalités à réaliser sont figées. À l'issue de ce *sprint*, une nouvelle *release* est créée et livrée au client pour évaluation des fonctionnalités développées. Cette nouvelle version sert de base pour les retours réalisés par le client, qui viennent alimenter les exigences pour le sprint suivant. Les exigences générales formalisées en début de projet peuvent ainsi être modifiées ou de nouvelles exigences peuvent être exprimées. À l'issue de chaque évaluation, les nouvelles priorités sont fixées et l'organisation des

équipes est modifiée en fonction de ces nouvelles priorités. Lorsque les priorités sont fixées, les équipes se réorganisent en fonction des choix réalisés lors de réunions quotidiennes très courtes appelées *stand up meetings*. Lors de ces réunions, chaque membre décrit son avancement et ses problèmes afin d'assurer une parfaite connaissance de l'état du projet entre tous les membres de l'équipe. Ces membres sont d'ailleurs tous impliqués avec le même niveau de responsabilité. Seul un *Scrum Master*, sorte de facilitateur, organise les réunions quotidiennes, suit le *backlog* produit présenté ci-après, génère les indicateurs adéquat et communique avec le client. Les autres membres de l'équipe n'ont en effet aucun contact avec l'extérieur afin de ne pas subir une éventuelle demande supplémentaire ou modification tardive.

La figure Annexe - Figure I.1 synthétise les principales étapes du processus liée à la méthodologie Scrum.



Annexe - Figure I.1 Processus lié à la méthodologie Scrum[31]

Pour mettre en œuvre ce type de management, différents outils sont généralement nécessaires. Le plus remarquable dans notre cas est le *backlog* produit, qui est la liste des fonctionnalités attendues d'un produit organisées par priorité. Chaque élément de ce *backlog* représente une fonctionnalité, un besoin, une amélioration ou un correctif. Il possède au minimum une description et une grandeur permettant d'ordonner les efforts entre eux. Ce *backlog* produit comporte donc de fortes similitudes avec le *Collaborative Actions Framework*, bien que ces dernières ne portent pas uniquement des informations de type « fonctionnalités à développer », mais également des demandes d'actions entre acteurs issus de différentes disciplines.

[31] Image recueillie sur internet

Extreme programming

Dans le cas de la méthode agile nommée *extreme programming*, les exigences sont formalisées sous la forme de scenarii appelés *user stories*. Ils permettent de définir une liste de tâches à réaliser. Pour chacune de ces tâches, au moins un test unitaire doit être créé et associé à cette action. L'ensemble de ces tests unitaires doit être validé avant la mise à disposition d'une nouvelle version (*release*) au client. D'autres tests basés sur les scénarii sont également créés pour valider l'intégration des fonctions unitaires. Ces *releases* doivent être très courtes afin de montrer l'avancement au client. Cette méthodologie met donc avant tout l'accent sur la qualité du code produit et sur les livraisons régulières.

Toujours dans cette idée de qualité, une autre singularité de cette méthode tient au fait que les développeurs travaillent toujours par paire. Ils partagent alors les informations et les responsabilités pour le développement de fonctionnalités. Des efforts moindres sont apportés quant à la définition a priori de l'architecture du logiciel, mais les efforts de réorganisation a posteriori (*refactoring*) sont encouragés.

Cette méthode s'appuie donc sur les idées de petites *releases* régulières, de tests nombreux et très tôt dans le processus et d'intégration continue. Ce dernier point signifie qu'il n'est pas nécessaire d'atteindre une grande maturité dans le développement d'un module pour le mettre à disposition et qu'il est préférable d'en limiter les fonctions au démarrage, de le mettre à disposition au plus tôt et d'enrichir ces fonctions au fur et à mesure de l'avancement.

ANNEXE II

Support utilisé lors de la soutenance

LISTE DE RÉFÉRENCES BIBLIOGRAPHIQUES

Abramovici, M., Bellalouna, F., 2007. Integration and Complexity Management within the Mechatronics Product Development, in: Advances in Life Cycle Engineering for Sustainable Manufacturing Businesses. pp. 113–118. doi:10.1007/978-1-84628-935-4_20

Aca, J., Ramos, M., Serrano, J.L., Ahuett, H., Molina, A., 2006. Concurrent Engineering of Mechatronic Products in Virtual Enterprises: Selection and Deployment of a PLM System for the Machine Tool Industry, in: Luo, Y. (Ed.), Cooperative Design, Visualization, and Engineering, Lecture Notes in Computer Science. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 318–326. doi:10.1007/11863649

AFIS, 2011. Livre blanc AFIS : vision 2020-2025 de l'ingénierie système.

AFNOR, 2008. NF E 01-010 2008 - Norme "Mécatronique - Vocabulaire" [WWW Document]. URL http://www.boutique.afnor.org/norme/nf-e01-010/mecatronique-vocabulaire/article/793875/fa159146 (accessed 11.26.14).

Ahn, W., Agonafer, D., Novotny, S., 2004. Methodology for an Integrated (Electrical/Mechanical) Design of PWBA. Journal of Electronic Packaging 126 (4), 524. doi:10.1115/1.1827268

Asare, P., Broman, D., Lee, E.A., Torngren, M., Sunder, S.S., 2012. cyberphysicalsystems.org [WWW Document]. URL http://cyberphysicalsystems.org/ (accessed 11.26.14).

Badreau, S., Boulanger, J.-L., 2014. Ingénierie des exigences: Méthodes et bonnes pratiques pour construire et maintenir un référentiel, InfoPro. Dunod.

Barrett, S., Chalin, P., Butler, G., 2008. Model merging falls short of software engineering needs, in: Procedure of the 2nd Workshop on Model-Driven Software Evolution. Athens (Greece).

Bathelt, J., Jonsson, A., Bacs, C., Dierssen, S., Meier, M., 2005. Applying the new VDI Design Guideline 2206 on mechatronic systems controlled by a PLC, in: International Conference on Engineering Design (ICED 05). Melbourne, Australia, pp. 1–14.

Beck, K., 2000. Extreme Programming Explained: Embrace Change. Addison-Wesley Professional, Boston, MA, USA.

Belkadi, F., 2006. Contribution au pilotage des compétences dans les activités de conception : de la modélisation des situations à la caractérisation des compétences. Thèse de doctorat soutenue le 21 novembre 2006, Ecole Doctorale Sciences

Physiques pour l'Ingénieur et Microtechniques, Université de Franche-Comté, France.

Bézivin, J., 2005. On the unification power of models. Software & Systems Modeling 4 (2), 171–188. doi:10.1007/s10270-005-0079-0

Blanchard, B.S., 2008. System Engineering Management, 4th Edition. John Wiley & Sons Inc.

Boehm, B.W., 1988. A spiral model of software development and enhancement. Computer 21 (5), 61–72.

Bosch-Mauchand, M., Bricogne, M., Eynard, B., Gitto, J.-P., 2014. Preliminary Requirements and Architecture Definition for Integration of PLM and Business Intelligence Systems, in: Grabot, B., Vallespir, B., Gomes, S., Bouras, A., Kiritsis, D. (Eds.), Advances in Production Management Systems. Innovative and Knowledge-Based Production Management in a Global-Local World, IFIP Advances in Information and Communication Technology. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 265–272. doi:10.1007/978-3-662-44739-0

Bouguessa, A., Forgues, D., Doré, S., 2013. La complémentarité entre le Building Information Modeling (BIM) et le Product LifeCycle Management (PLM) en passant par le Lean Construction (LC), in: Francis, A., Forgues, D., Attala, M. (Eds.), CSCE 2013 General Conference - 4th Construction Specialty Conference. Montréal, Québec, Canada, pp. 1–10.

Bragge, M., 2013. Model-View-Controller architectural pattern and its evolution in graphical user interface frameworks. Lappeenranta University of Technology.

Bricogne, M., Eynard, B., Troussier, N., Antaluca, E., Ducellier, G., 2011. Building lifecycle management: overview of technology challenges and stakeholders, in: IET International Conference on Smart and Sustainable City (ICSSC 2011). IET, Shanghai, China, pp. 177–181. doi:10.1049/cp.2011.0284

Bricogne, M., Petit, L., Meyine, A.M., Troussier, N., Eynard, B., 2010a. Mechatronics issues for specification of PLM functionalities and features, in: MECATRONICS2010. Yokohama, Japan.

Bricogne, M., Rivest, L., Troussier, N., Eynard, B., 2014. Concurrent versioning principles for collaboration: towards PLM for hardware and software data management. International Journal of Product Lifecycle Management 7 (1), 17–37. doi:10.1504/IJPLM.2014.065457

Bricogne, M., Troussier, N., Rivest, L., Eynard, B., 2010b. PLM perspectives in mechatronic systems design, in: Proceedings of APMS 2010 - International Conference on Advances in Production Management Systems. Cernobbio, Como, Italy, pp. 1–8.

Brière-Côté, A., Rivest, L., Maranzana, R., 2012. Comparing 3D CAD Models: Uses, Methods, Tools and Perspectives. Computer-Aided Design and Applications 9 (6), 771–794. doi:10.3722/cadaps.2012.771-794

Brosch, P., Kappel, G., Langer, P., Seidl, M., Wieland, K., Wimmer, M., 2012. An Introduction to Model Versioning, in: Bernardo, M., Cortellessa, V., Pierantonio, A. (Eds.), Formal Methods for Model-Driven Engineering - 12th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM 2012, Bertinoro, Italy, June 18-23, 2012. Advanced Lectures, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 336–398. doi:10.1007/978-3-642-30982-3_10

Brosch, P., Seidl, M., Wieland, K., Wimmer, M., Langer, P., 2009. We can work it out: Collaborative Conflict Resolution in Model Versioning, in: Wagner, I., Tellioğlu, H., Balka, E., Simone, C., Ciolfi, L. (Eds.), ECSCW 2009. Springer London, pp. 207–214. doi:10.1007/978-1-84882-854-4_12

Buschmann, F., Henney, K., Schimdt, D., 2007. Pattern-oriented Software Architecture: On Patterns and Pattern Language 5.

Chappell, D., 2009. Pro Visual Studio Team System Application Lifecycle Management. Apress, Berkeley, CA. doi:10.1007/978-1-4302-1079-5

Chaudhuri, S., Dayal, U., Narasayya, V., 2011. An overview of business intelligence technology. Communications of the ACM, AIAA Paper 2009-5307 54 (8), 88.

Chen, K., Bankston, J., Panchal, J.H., Schaefer, D., 2009. A Framework for Integrated Design of Mechatronic Systems, in: Wang, L., Nee, A.Y.C. (Eds.), Collaborative Design and Planning for Digital Manufacturing. Springer London, London, UK, pp. 37–70. doi:10.1007/978-1-84882-287-0

Chen, K., Schaefer, D., 2007. MCAD–ECAD Integration: Overview and Future Research Perspectives, in: International Mechanical Engineering Congress and Exposition 2007 (IMECE 2007). Seattle (USA).

Chhaniyara, S., Saaj, C., Maediger, B., Althoff-Kotzias, M., Ahrns, I., 2011. Model Based System Engineering For Space Robotics Systems, in: Proceedings of 11th Symposium on Advanced Space Technologies in Robotics and Automation - ASTRA 2011. Noordwijk, Netherlands.

Conte, A., Fredj, M., Giraudin, J.-P., Rieu, D., 2001. P-Sigma : un formalisme pour une représentation unifiée de patrons, in: INFORSID'01. pp. 67–86.

Crnkovic, I., Asklund, U., Dahlqvist, A.P., 2003. Implementing and integrating product data management and software configuration management. Artech House Publishers, Boston, USA.

Dahlqvist, A.P., Asklund, U., Crnkovic, I., Hedin, A., Larsson, M., Ranby, J., Svensson, D., 2001. Product Data Management and Software Configuration Management - Similarities and Differences. The Association of Swedish Engineering Industries.

De Pinel, P., Maranzana, N., Segonds, F., Leroux, S., Frerebeau, V., 2013. Proposition of Ergonomic Guidelines to Improve Usability of PLM Systems Interfaces, in: Bernard, A., Rivest, L., Dutta, D. (Eds.), Product Lifecycle Management for Society, IFIP Advances in Information and Communication Technology. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 530–539. doi:10.1007/978-3-642-41501-2_53

Do, N., 2014. Application of OLAP to a PDM database for interactive performance evaluation of in-progress product development. Computers in Industry 65 (4), 636–645. doi:10.1016/j.compind.2014.01.014

Do, N., Chae, G., 2006. A Framework for Product Development Process including HW and SW Components, in: Proceedings of World Academy of Science. pp. 130–133.

Do, N., Chae, G., 2008. A Product Data Model for Integrating Product Data Management and Software Configuration Management, in: Bouras, A., Gurumoorthy, B., McMahon, C., Ramani, K. (Eds.), Proceedings of the International Conference on Product Lifecycle Management - PLM'08. Seoul, Korea, pp. 40–48.

Do, N., Chae, G., 2011. A Product Data Management architecture for integrating hardware and software development. Computers in Industry 62 (8-9), 854–863. doi:10.1016/j.compind.2011.09.001

Durupt, A., Remy, S., Bricogne, M., Troussier, N., 2013. PHENIX: product history-based reverse engineering. International Journal of Product Lifecycle Management 6 (3), 270. doi:10.1504/IJPLM.2013.055884

EARTO - Association européenne pour les organisations de recherche et technologie, 2014. The TRL Scale as a Research & Innovation Policy Tool, EARTO Recommendations.

El-Khoury, J., 2006. A model management and integration platform for mechatronics product development. Engineering. KTH - Royal Institute of Technology.

El-Khoury, J., Redell, O., Torngren, M., 2005. A tool integration platform for multi-disciplinary development, in: Software Engineering and Advanced Applications, 2005. 31st EUROMICRO Conference on. IEEE, pp. 442–449.

Estublier, J., 2000. Software configuration management: a roadmap, in: Proceedings of the Conference on The Future of Software Engineering - ICSE '00. ACM Press, New York, New York, USA, pp. 279–289. doi:10.1145/336512.336576

Estublier, J., 2001. Objects Control for Software Configuration Management, in: Dittrich, K.R., Geppert, A., Norrie, M.C. (Eds.), Advanced Information Systems Engineering. Springer Berlin Heidelberg, pp. 359–373. doi:10.1007/3-540-45341-5_24

Ettlie, J., 1997. Integrated design and new product success. Journal of Operations Management 15 (1), 33–55. doi:10.1016/S0272-6963(96)00095-2

Eynard, B., Gallet, T., Nowak, P., Roucoules, L., 2004. UML based specifications of PDM product structure and workflow. Computers in industry 55 (3), 301–316. doi:10.1016/j.compind.2004.08.006

Fachbereich Produktentwicklung und Mechatronik, 2004. VDI 2206 - Design methodology for mechatronic systems.

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 8 (1).

Forsberg, K., Mooz, H., 1992. The relationship of system engineering to the project cycle. Engineering Management Journal 4 (3), 36–43.

Fowler, M., 2002. Patterns of enterprise application architecture. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Friendenthal, S., Steiner, R., Moore, A., 2008. A practical guide to SysML. Elsevier/Morgan Kaufmann, Waltham, USA.

Gardan, Y., 1991. La CFAO, introduction, techniques et mise en oeuvre, 3ème éditi. ed. Hermes, Paris.

Gero, J.S., Kannengiesser, U., 2004. The situated function–behaviour–structure framework. Design Studies 25 (4), 373–391. doi:10.1016/j.destud.2003.10.010

Gidel, T., Zonghero, W., 2006. Management de projet : Tome 1, Introduction et fondamentaux. Hermes Science/Lavoisier, Paris, France.

Girard, P., Doumeingts, G., 2004. GRAI-Engineering: A method to model, design and run engineering design departments. International Journal of Computer Integrated Manufacturing 17 (8), 716–732. doi:10.1080/0951192042000237492

Graignic, P., Vosgien, T., Jankovic, M., Tuloup, V., Berquet, J., Troussier, N., 2013. Complex System Simulation: Proposition of a MBSE Framework for Design-Analysis Integration. Procedia Computer Science 16, 59–68. doi:10.1016/j.procs.2013.01.007

Grieves, M.W., 2011. Virtually Perfect. Space Coast Press LLC, Cocoa Beach, Florida, USA.

Grieves, M.W., 2012. Virtually Indistinguishable: Systems Engineering and PLM, in: Rivest, L., Bouras, A., Louhichi, B. (Eds.), Product Lifecycle Management. Towards Knowledge-Rich Enterprises. Springer Berlin Heidelberg, Montréal, Canada, pp. 226–242. doi:10.1007/978-3-642-35758-9_20

Han, K.H., Do, N., 2005. An object-oriented conceptual model of a collaborative product development management (CPDM) system. The International Journal of Advanced Manufacturing Technology 28 (7-8), 827–838. doi:10.1007/s00170-004-2424-9

Harashima, F., Tomizuka, M., Fukuda, T., 1996. Mechatronics-'what is it, why, and how?' An editorial. IEEE/ASME Transactions on Mechatronics 1 (1), 1–4.

Howard, T.J., Culley, S.J., Dekoninck, E., 2008. Describing the creative design process by the integration of engineering design and cognitive psychology literature. Design Studies 29 (2), 160–180. doi:10.1016/j.destud.2008.01.001

Hyman, B.I., 2003. Fundamentals of Engineering Design, 2nd Edition. Prentice Hall.

Isermann, R., 2007. Mechatronic design approach, in: The Mechatronics Handbook. CRC Press, Boca Raton, USA, pp. 2–3.

Jarratt, T.A.W., Eckert, C.M., Caldwell, N.H.M., Clarkson, P.J., 2010. Engineering change: an overview and perspective on the literature. Research in Engineering Design 22 (2), 103–124. doi:10.1007/s00163-010-0097-y

Jones, A., Kendira, A., Lenne, D., Gidel, T., Moulin, C., 2011. The TATIN-PIC project: A multi-modal collaborative work environment for preliminary design, in: Proceedings of the 2011 15th International Conference on Computer Supported Cooperative Work in Design (CSCWD). IEEE, pp. 154–161. doi:10.1109/CSCWD.2011.5960069

Kääriäinen, J., Savolainen, P., Taramaa, J., Leppälä, K., 2000. Product data management (PDM) Design, exchange and integration viewpoints.

Kääriäinen, J., Välimäki, A., 2009. Applying Application Lifecycle Management for the Development of Complex Systems: Experiences from the Automation Industry, in: O'Connor, R. V., Baddoo, N., Cuadrago Gallego, J., Rejas Muslera, R., Smolander, K., Messnarz, R. (Eds.), Software Process Improvement, Communications in Computer and Information Science. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 149–160. doi:10.1007/978-3-642-04133-4

Kéradec, H., 2012. Epistémologie et didactique de la gestion : Le cas du concept de déecision. Conservatoire national des arts et metiers - CNAM.

Kettunen, P., Laanti, M., 2008. Combining agile software projects and large-scale organizational agility. Software Process: Improvement and Practice 13 (2), 183–193. doi:10.1002/spip.354

Kossiakoff, A., Sweet, W.N., Seymour, S., Biemer, S.M., 2011. Systems Engineering Principles and Practice, 2nd Edition, 2011. John Wiley & Sons, Inc.

Kozemjakin da Silva, M., Guyot, E., Remy, S., Reyes, T., 2013. A Product Model to Capture and Reuse Ecodesign Knowledge, in: Bernard, A., Rivest, L., Dutta, D. (Eds.), Product Lifecycle Management for Society, 10th IFIP WG 5.1 International

Conference, PLM 2013, Nantes, France, July 6-10, 2013, Proceedings, IFIP Advances in Information and Communication Technology. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 220–228. doi:10.1007/978-3-642-41501-2

Kumar, P.R., 2012. Cyber–Physical Systems: A Perspective at the Centennial. Proceedings of the IEEE 100, 1287–1308. doi:10.1109/JPROC.2012.2189792

Lindholm, T., 2004. A three-way merge for XML documents, in: Proceedings of the 2004 ACM Symposium on Document Engineering - DocEng '04. ACM Press, New York, New York, USA, p. 1. doi:10.1145/1030397.1030399

Lindvall, M., Muthig, D., Dagnino, A., Wallin, C., Stupperich, M., Kiefer, D., May, J., Kahkonen, T., 2004. Agile software development in large organizations. Computer 37 (12), 26–34. doi:10.1109/MC.2004.231

Matthews, P., Lomas, C., Armoutis, N.D., Maropoulos, P.G., 2006. Foundations of an agile design methodology. International journal of Agile Manufacturing 9 (1), 29–38.

Maurino, M., 1994. La gestion des données techniques : technologie du concurrent engineering. Masson, Paris (France).

McMahon, P., 2005. Extending agile methods: A distributed project and organizational improvement perspective. CrossTalk 18 (5), 16–19.

Mens, T., 2002. A state-of-the-art survey on software merging. IEEE Transactions on Software Engineering 28 (5), 449–462. doi:10.1109/TSE.2002.1000449

Messager, V., 2009. Gestion de projet: vers les méthodes agiles, 2nd ed, Architecte logiciel. Editions Eyrolles, Paris.

Mills, H.D., 1980. The management of software engineering, Part I: Principles of software engineering. IBM Systems Journal 19 (4), 414–420. doi:10.1147/sj.194.0414

NASA, 2007. NASA Systems Engineering Handbook. Washington, DC, USA.

Negash, S., 2004. Business Intelligence (BI). Communications of the Association for Information Systems 13, 177–195.

Nicquevert, B., 2012. Manager l'interface - Approche par la complexité du processus collaboratif de conception, d'intégration et de réalisation : modèle transactionnel de l'acteur d'interface et dynamique des espaces d'échanges. Thèse de doctorat soutenue le 23 novembre 2012, École Doctorale IMEP-2, Ingénierie Matériaux Mécanique Énergétique Environnement Procédés Production, Université de Grenoble.

Niu, M.C.Y., 1999. Airframe structural design : practical design information and data on aircraft structures, 2nd ed. Conmilit Press Ltd, Hong Kong, China.

Noël, F., Roucoules, L., 2008. The PPO design model with respect to digital enterprise technologies among product life cycle. International Journal of Computer Integrated Manufacturing 21 (2), 139–145. doi:10.1080/09511920701607782

Nowak, P., Rose, B., Saint-Marc, L., Callot, M., Eynard, B., Gzara-Yesilbas, L., Lombard, M., 2004. Towards a design process model enabling the integration of product, process and organization, in: 5th International Conference on Integrated Design and Manufacturing in Mechanical Engineering IDMME'04. Bath, UK.

OMG Group, 2007. OMG Systems Modeling Language (OMG SysML ™), Language.

Pahl, G., Beitz, W., Wallace, K., 2003. Engineering design: a systematic approach. Springer.

Paviot, T., 2010. Méthodologie de résolution des problèmes d'interopérabilité dans le domaine du Product Lifecycle Management. Thèse de doctorat soutenue le 1 juillet 2010, Ecole Doctorale Sciences pour l'Ingénieur, Ecole Centrale de Paris.

Penas, O., Plateaux, R., Choley, J.-Y., Kadima, H., Soriano, T., Combastel, C., Rivière, A., 2010. Conception mécatronique - Vers un processus continu de conception mécatronique intégrée. Techniques de l'Ingénieur.

Plateaux, R., Choley, J.-Y., Penas, O., Riviere, A., 2009. Towards an integrated mechatronic design process, in: 2009 IEEE International Conference on Mechatronics. IEEE, pp. 1–6. doi:10.1109/ICMECH.2009.4957237

Rachuri, S., Baysal, M., Roy, U., Foufou, S., Bock, C., Fenves, S., Subrahmanian, E., Lyons, K., Sriram, R., 2005. Information models for product representation: core and assembly models. International Journal of Product Development 2 (3/2005), 207–235.

Rajkumar, R., 2012. A Cyber–Physical Future. Proceedings of the IEEE 100, 1309–1312. doi:10.1109/JPROC.2012.2189915

Rajkumar, R. (Raj), Lee, I., Sha, L., Stankovic, J., 2010. Cyber-physical systems: the next computing revolution, in: Proceedings of the 47th Design Automation Conference on - DAC '10. ACM Press, New York, New York, USA, p. 731. doi:10.1145/1837274.1837461

Raymer, D.P., 1999. Aircraft design: a conceptual approach, 3rd editio. ed. American Institute of Aeronautics & Astronotics, Inc, Reston, Virginia (Unites States).

Rensselaer Polytechnic Institute Website (RPI), n.d. http://www.rpi.edu/ [WWW Document].

Rio, M., 2012. A l'interface de l'ingénierie et de l'analyse environnementale, fédération pour une éco-conception proactive. Thèse de doctorat soutenue le 7 décembre 2012, Ecole Doctorale Sciences et Technologies, Université de Technologie de Troyes, France.

Royce, W.W., 1970. Managing the development of large software systems, in: Proceedings of IEEE Western Electronic Show and Convention (WesCon). Los Angeles, USA, pp. 1–9.

Sääksvuori, A., Immonen, A., 2004. Product lifecycle management. Springer, New York, USA.

Sadeghi, M., Hadj-Hamou, K., Noel, F., 2010. A collaborative platform architecture for coherence management in multi-view integrated product modelling. International Journal of Computer Integrated Manufacturing 23 (3), 270–282. doi:10.1080/09511920903534321

Sadeghi, M., Noel, F., Hadj-Hamou, K., 2009. Development of control mechanisms to support coherency of product model during cooperative design process. Journal of Intelligent Manufacturing 21 (4), 539–554. doi:10.1007/s10845-009-0237-2

Schöner, H.-P., 2004. Automotive mechatronics. Control Engineering Practice 12 (11), 1343–1351. doi:10.1016/j.conengprac.2003.10.004

Schwaber, K., Beedle, M., 2001. Agile Software Development With Scrum. Prentice-Hall PTR, Upper Saddle River, New Jersey, USA.

Sellgren, U., Törngren, M., Malvius, D., Biehl, M., 2009. PLM for Mechatronics integration, in: Proceedings of the 6th International Product Lifecycle Management Conference (PLM 2009). Bath, England.

Şenaltun, G., Cangelir, C., 2012. Software Management in Product Structure. Product Lifecycle Management. Towards Knowledge-Rich Enterprises, IFIP Advances in Information and Communication Technology 388 (2012), 369–378. doi:10.1007/978-3-642-35758-9

Shetty, D., Kolk, R.A., 2010. Mechatronics System Design, in: Mechatronics System Design: SI. Christopher M. Shortt, pp. 1–40.

Silventoinen, A., Pels, H.J., Kärkkäinen, H., Lampela, H., 2011. Towards future PLM maturity assessment dimensions, in: International Conference on Product Lifecycle Management 2011. Eindhoven.

Sinha, R., Liang, V.-C., Paredis, C.J.J., Khosla, P.K., 2001. Modeling and simulation methods for design of engineering systems. Journal of Computing and Information Science in Engineering 1 (1), 84–91.

Sohlenius, G., 1992. Concurrent Engineering. CIRP Annals - Manufacturing Technology 41 (2), 645–655. doi:10.1016/S0007-8506(07)63251-X

Sommer, A.F., Dukovska-Popovska, I., Steger-Jensen, K., 2014. Agile Product Development Governance – On Governing the Emerging Scrum/Stage-Gate Hybrids, in: Grabot, B., Vallespir, B., Gomes, S., Bouras, A., Kiritsis, D. (Eds.), Advances in Production Management Systems. Innovative and Knowledge-Based

Production Management in a Global-Local World, IFIP Advances in Information and Communication Technology. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 184–191. doi:10.1007/978-3-662-44739-0

Sommerville, I., 2010. Software Engineering. Addison Wesley; 9 edition (March 13, 2010), New York, New York, USA.

Sudarsan, R., Fenves, S.J., Sriram, R.D., Wang, F., 2005. A product information modeling framework for product lifecycle management. Computer-Aided Design 37 (13), 1399–1411. doi:10.1016/j.cad.2005.02.010

Terzi, S., Bouras, A., Dutta, D., Garetti, M., Kiritsis, D., 2010. Product lifecycle management – from its history to its new role. International Journal of Product Lifecycle Management 4 (4), 360. doi:10.1504/IJPLM.2010.036489

Tichkiewitch, S., 1994. De la CFAO à la conception intégrée. Revue internationale de C.F.A.O. et d'infographie 9 (5), 609–621.

Troussier, N., Bricogne, M., Durupt, A., Belkadi, F., Ducellier, G., 2010. A knowledge-based reverse engineering process for CAD models management, in: 8th International Conference on Integrated Design and Manufacturing in Mechanical Engineering, IDMME'10. Bordeaux, France.

Ullman, D.G., 2010. The mechanical design process. McGraw-Hill Higher Education.

US Department of Transportation, 2007. Systems engineering for intelligent transportation systems. Washington, D.C., USA.

Vu Thi, H., 2012. Analyse des environnements supports à l'ingénierie collaborative synchrone à distance : approche ergonomique pour l'amélioration des outils via l'analyse des usages. Thèse de doctorat soutenue le 6 juin2012, École Doctorale IMEP-2, Ingénierie Matériaux Mécanique Énergétique Environnement Procédés Production, Université de Grenoble, France.

W3C, 2008. Extensible Markup Language (XML) 1.0 (Fifth Edition) [WWW Document]. URL http://www.w3.org/TR/REC-xml/

Wang, Q., Ren, Z.-W., Guo, Z.-F., 2010. XML-based data processing in network supported collaborative design. International Journal of Automation and Computing 7 (3), 330–335. doi:10.1007/s11633-010-0511-y

Warniez, A., Penas, O., Soriano, T., 2012. About metrics for integrated mechatronic system design, in: 2012 9th France-Japan & 7th Europe-Asia Congress on Mechatronics (MECATRONICS) / 13th Int'l Workshop on Research and Education in Mechatronics (REM). IEEE, pp. 450–457. doi:10.1109/MECATRONICS.2012.6451047

Wiener, N., 1965. Cybernetics Or Control and Communication in the Animal and the Machine, M.I.T. paperback series. MIT Press.

Willard, B., 2007. UML for systems engineering. Computer Standards & Interfaces 29 (1), 69–81. doi:10.1016/j.csi.2005.12.006

Zheng, C., Bricogne, M., Le Duigou, J., Eynard, B., 2014a. Survey on mechatronic engineering: A focus on design methods and product models. Advanced Engineering Informatics 28 (3), 241–257. doi:10.1016/j.aei.2014.05.003

Zheng, C., Bricogne, M., Le Duigou, J., Eynard, B., 2014b. Mechatronic Design Process: A Survey of Product Data Model, in: Moroni, G., Tolio, T. (Eds.), Procedia 24th CIRP Design Conference. Milano, Italy.

Zheng, C., Le Duigou, J., Bricogne, M., Eynard, B., 2013. Survey of Design Process Models for Mechatronic Systems Engineering, in: 10ème Congrès International de Génie Industriel - CIGI 2013. La Rochelle, France.

BIBLIOGRAPHIE

A – Publications dans des revues internationales avec comité de lecture

Bricogne, M., Rivest, L., Troussier, N., Eynard, B., 2014. Concurrent versioning principles for collaboration: towards PLM for hardware and software data management. International Journal of Product Lifecycle Management 7 (1), 17–37. doi:10.1504/IJPLM.2014.065457

Zheng, C., **Bricogne, M**., Le Duigou, J., Eynard, B., 2014b. Survey on mechatronic engineering: A focus on design methods and product models. Advanced Engineering Informatics 28 (3), 241–257. doi:10.1016/j.aei.2014.05.003

Durupt, A., Remy, S., **Bricogne, M.**, Troussier, N., 2013. PHENIX: product history-based reverse engineering. International Journal of Product Lifecycle Management 6 (3), 270. doi:10.1504/IJPLM.2013.055884

Bosch-Mauchand, M., Belkadi, F., **Bricogne, M**., Eynard, B., 2013. Knowledge-based assessment of manufacturing process performance: integration of product lifecycle management and value-chain simulation approaches. International Journal of Computer Integrated Manufacturing 26 (5), 453–473. doi:10.1080/0951192X.2012.731611

Durupt, A., Remy, S., Ducellier, G., **Bricogne, M**., 2010. KBRE: a proposition of a reverse engineering process by a KBE system. International Journal on Interactive Design and Manufacturing (IJIDeM) 4 (4), 227–237. doi:10.1007/s12008-010-0106-4

B – Chapitres dans des ouvrages de synthèse et revues nationales

Bosch-Mauchand, **M., Bricogne**, M., Eynard, B., Gitto, J.-P., 2014. Preliminary Requirements and Architecture Definition for Integration of PLM and Business Intelligence Systems, in: Grabot, B., Vallespir, B., Gomes, S., Bouras, A., Kiritsis, D. (Eds.), Advances in Production Management Systems. Innovative and Knowledge-Based Production Management in a Global-Local World, IFIP Advances in Information and Communication Technology. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 265–272. doi:10.1007/978-3-662-44739-0

Eynard, B., Le Duigou, J., **Bricogne, M**., 2014. Les fondamentaux - Des SGDT au Product Lifecycle Management : évolution de la place des documents techniques. Documentaliste-Sciences de l'Information 51 (1), 34–37. doi:10.3917/docsi.511.0030

Bricogne, M., Rivest, L., Troussier, N., Eynard, B., 2012. Towards PLM for mechatronics system design using concurrent software versioning principles, in: Rivest, L., Bouras, A., Louhichi, B. (Eds.), Product Lifecycle Management. Towards Knowledge-Rich Enterprises, IFIP Advances in Information and Communication Technology. Springer Berlin Heidelberg, pp. 339–348. doi:10.1007/978-3-642-35758-9_30

Bricogne, M., Belkadi, F., Bosch-Mauchand, M., Eynard, B., 2010a. Knowledge Based Product and Process Engineering Enabling Design and Manufacture Integration, in: Vallespir, B., Alix, T. (Eds.), Advances in Production Management Systems. New Challenges, New Approaches, IFIP Advances in Information and Communication Technology. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 473–480. doi:10.1007/978-3-642-16358-6_59

C – Communications dans des conférences internationales avec actes et comité de lecture

Zheng, C., **Bricogne, M**., Le Duigou, J., Eynard, B., 2014. Mechatronic Design Process: A Survey of Product Data Model, in: Moroni, G., Tolio, T. (Eds.), Procedia 24th CIRP Design Conference. Milano, Italy.

Bricogne, M., Troussier, N., Rivest, L., Eynard, B., 2013. Agile design methods for mechatronics system integration, in: Bernard, A., Rivest, L., Dutta, D. (Eds.), IFIP WG 5.1 International Conference, PLM 2013. Nantes, France, pp. 458–470.

Messaadia, M., Szigeti, M., Bosch-Mauchand, M., **Bricogne, M**., Eynard, B., Majumdar, A., 2013. ICT for design and manufacturing: a strategic vision for technology maturity assessment, in: International Conference on Research into Design – ICoRD'13. Chennai, India, pp. 913–924. doi:http://dx.doi.org/10.1007/978-81-322-1050-4_72

Li, J., Tong, S., **Bricogne, M**., Troussier, N., 2012. Survey on Configuration Management of Modular Mechatronics Products, in: ASME 2012 11th Biennial Conference on Engineering Systems Design and Analysis - Volume 3: Advanced Composite Materials and Processing; Robotics; Information Management and PLM; Design Engineering. ASME, Nantes, France, pp. 535–540. doi:10.1115/ESDA2012-82945

Bricogne, M., Eynard, B., Troussier, N., Antaluca, E., Ducellier, G., 2011. Building lifecycle management: overview of technology challenges and stakeholders, in: IET International Conference on Smart and Sustainable City (ICSSC 2011). IET, Shanghai, China, pp. 177–181. doi:10.1049/cp.2011.0284

Bricogne, M., Petit, L., Meyine, A.M., Troussier, N., Eynard, B., 2010. Mechatronics issues for specification of PLM functionalities and features, in: MECATRONICS2010. Yokohama, Japan.

Bricogne, M., Troussier, N., Rivest, L., Eynard, B., 2010. PLM perspectives in mechatronic systems design, in: Proceedings of APMS 2010 - International Conference on Advances in Production Management Systems. Cernobbio, Como, Italy, pp. 1–8.

Donati, T., **Bricogne, M**., Eynard, B., 2010. PLM platform: integrated support of the enterprise digital chain for Collaborative Product Development, in: The 7th International Conference on Product Lifecycle Management, PLM'10. Bremen, Germany, pp. 236–246.

Troussier, N., **Bricogne, M**., Belkadi, F., Durupt, A., Ducellier, G., 2010. An extension of

the Core Product Model for Reverse Engineering, in: The 7th International Conference on Product Lifecycle Management, PLM'10. Bremen, Germany.

Troussier, N., **Bricogne, M.**, Durupt, A., Belkadi, F., Ducellier, G., 2010. A knowledge-based reverse engineering process for CAD models management, in: 8th International Conference on Integrated Design and Manufacturing in Mechanical Engineering, IDMME'10. Bordeaux, France.

Belkadi, F., Troussier, N., Eynard, B., **Bricogne, M**., 2009. Specification of a collaborative framework for equipment suppliers' integration in product development process, in: 2009 International Conference on Computers & Industrial Engineering. IEEE, Troyes, France, pp. 1347–1352. doi:10.1109/ICCIE.2009.5223952

Durupt, A., **Bricogne, M**., Remy, S., Troussier, N., 2009. Product knowledge based reverse engineering: towards an integrated expert approach, in: International Conference on Designing Pleasurable Products and Interfaces, DPPI09. Compiègne, France.

D – Communications dans des colloques nationaux et groupes de recherche

Zheng, C., Le Duigou, J., **Bricogne, M**., Eynard, B., 2013. Survey of Design Process Models for Mechatronic Systems Engineering, in: 10ème Congrès International de Génie Industriel - CIGI 2013. La Rochelle, France.

Donati, T., **Bricogne, M.**, Eynard, B., 2011. Gestion de la diversité en contexte PLM, in: JD-JN MACS - Journées Doctorales / Journées Nationales MACS. Marseille (France).