THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :
Institut National Polytechnique de Toulouse (INP Toulouse)

Discipline ou spécialité :

Systèmes Industriels

Présentée et soutenue par :
Joël ABEILLE

le : mercredi 6 juillet 2011

Titre :

Vers un couplage des processus de conception de systèmes et de planification de projets : formalisation de connaissances méthodologiques et de connaissances métier

---

Ecole doctorale :
Systèmes (EDSYS)

Unité de recherche :
Laboratoire Génie de Production - E.N.I.T. / Centre de Génie Industriel - E.M.A.C.

Directeur(s) de Thèse :
M. Laurent GENESTE - Professeur à l'E.N.I.T. - Directeur de thèse
M. Michel ALDANONDO - Professeur à l'E.M.A.C. - Co-directeur de thèse

Rapporteurs :

M. Jean RENAUD - Professeur à l'I.N.S.A. Strasbourg
M. André THOMAS - Professeur à l'Université Henri Poincaré Nancy 1

Membre(s) du jury :
M. Samir LAMOURI - Professeur à l'E.N.S.A.M ParisTech - Président du jury
Mme Nadège TROUSSIER - Maître de Conférences H.D.R. à U.T.C - Examinateur
Mme Elise VAREILLES - Maître de Conférences à l'E.M.A.C. - Membre
M. Thierry COUDERT - Maître de Conférences à l'E.N.I.T. - Membre

Remerciements

Je tiens tout d'abord à remercier mes encadrants de thèse sans qui ces travaux n'auraient pas vu le jour : Laurent GENESTE, directeur de thèse, Michel ALDANONDO, co-directeur, et Elise VAREILLES et Thierry COUDERT, encadrants de thèse. Merci pour votre disponibilité, vos conseils toujours précieux et votre bonne humeur que j'ai parfois mise à l'épreuve.

Je remercie également Thierry ROUX, gérant de la société Pulsar Innovation, pour m'avoir permis de réaliser cette thèse CIFRE, d'une part, en m'intégrant dans l'entreprise, et d'autre part, en me prodiguant son expérience du milieu industriel.

Je remercie aussi les personnes qui ont acceptées de participer à mon jury de thèse :

- M. Samir LAMOURI, Professeur à l'ENSAM ParisTech, d'avoir accepté de présider ce jury,
- M. Jean RENAUD, Professeur à l'INSA de Strasbourg et M. André THOMAS, professeur à l'ENSTIB d'Epinal, qui m'ont fait l'honneur d'être rapporteurs de ma thèse,
- Mme Nadège TROUSSIER, Maitre de Conférences à l'UT de Compiègne, pour l'intérêt qu'elle a porté à ces travaux.

Je tiens à remercier l'ensemble des laboratoires et des entreprises qui ont pris part au projet ATLAS, pour leur travail, les échanges que nous avons eus et qui portaient souvent une vision nouvelle sur mon travail et pour leur bonne humeur permanente.

Je tiens à remercier toutes les personnes de l'ENIT et de l'EMAC qui, de près ou de loin, techniquement ou moralement, m'ont soutenu et/ou supporté. Merci pour les bons moments passés en votre compagnie.

Enfin, je souhaiterais remercier ma famille pour son amour et son soutien sans faille sans qui je ne serais probablement pas en train d'écrire ces mots aujourd'hui.

Sommaire

Introduction générale

Introduction générale

Dans de nombreux domaines d'activité comme la production manufacturière ou les services, les entreprises font face à des contraintes de plus en plus difficiles à satisfaire. L'une des raisons peut être une évolution systématique et continue de la complexité de leurs produits dans un contexte concurrentiel sévère. Afin de rester compétitives, les entreprises cherchent généralement à maîtriser leurs coûts, la qualité de leurs produits et à garantir des délais fiables et courts à leurs clients. Beaucoup d'entre elles se situent dans des contextes où l'organisation, les acteurs et les ressources matérielles se trouvent distribuées sur de vastes réseaux sur l'ensemble de la planète. La connaissance sur la conception des produits, sur l'organisation, sur la manière d'industrialiser, de produire, de distribuer et de recycler n'est plus centralisée mais distribuée sur des équipes, des sites et des acteurs différents. Dans un tel contexte, la maîtrise des processus de conception de produits ou de systèmes est indispensable afin de garantir au plus tôt : la bonne adéquation entre ce qui est attendu par le client et ce qui lui est délivré, la détection des non faisabilités, la répartition adéquate du travail de conception sur les équipes en évaluant les ressources nécessaires, les coûts engagés et les délais de livraison. Une telle maîtrise passe nécessairement par la mise en œuvre de processus clairs et non-ambigus qui permettent de cadrer l'activité de conception. Afin d'y parvenir, le déroulement des activités de conception doit être intégré dans un projet de conception. Un tel projet doit être défini, planifié et réalisé efficacement. Ainsi, la planification de projet est au centre de ce cadrage.

Classiquement, en gestion de projet, la planification intervient tardivement lorsque l'architecture des artefacts à concevoir est relativement bien connue (voir par exemple la description des processus de planification dans [**Project Management Institute, 2004**] dont la réalisation est basée sur la connaissance de l'architecture WBS – *Working Breakdown Structure*). Une telle démarche séquentielle peut entraîner une détection des problèmes tardive lors de la conception ainsi que des délais importants. Pourtant, il existe une dépendance forte entre le processus de planification d'un projet de conception et celui de conception de produits ou systèmes lui-même. En effet, la connaissance de l'architecture de ce que l'on cherche à concevoir permet de réaliser la planification mais également, la connaissance des activités à planifier dépend du travail de conception lui-même. C'est cette interdépendance que nous nommons « couplage » dans le mémoire et que nous proposons d'étudier.

Nos travaux se positionnent dans le cadre du projet ATLAS (Aides et assisTances pour la conception, la conduite et leur coupLage par les connAissanceS). Ce projet ANR-RNTL 2007, labellisé par le pôle de compétitivité mondial Aerospace Valley, dispose d'un budget de 1,8 millions d'euros et le consortium constitué pour ce dernier implique cinq laboratoires de recherche : le LATTIS de l'INSA de Toulouse, le LAAS-CNRS de Toulouse, le LGP de l'ENI de Tarbes, le CGI de l'Ecole des Mines d'Albi-Carmaux et le Laboratoire IMS de l'Université Bordeaux 1. Deux entreprises sont également parties prenantes dans le consortium : Sigma Plus et Pulsar Innovation, toutes deux basées à Toulouse. Les objectifs de ce projet sont de recueillir les compétences, les savoir-faire et les pratiques industrielles puis, de proposer une méthodologie tenant compte à la fois de la conception de systèmes, de la planification des projets de conception associés et des interactions entre les deux. La finalité du projet est de formaliser des modèles, des processus et de développer une plateforme de démonstration logicielle supportant le couplage.

C'est dans ce cadre et, plus particulièrement celui d'une convention CIFRE entre la société Pulsar Innovation et les laboratoires LGP et CGI, que s'inscrivent nos travaux de recherche dont l'objectif global est d'identifier, de caractériser et d'outiller différentes formes de couplage entre la planification des projets de conception et la conception des systèmes associés.

D'un point de vue industriel, le travail a débuté par la réalisation d'une enquête en collaboration avec un stagiaire de l'Institut de Cognitique de Bordeaux (IdC), auprès d'une quinzaine d'entreprises industrielles du pôle de compétitivité Aerospace Valley intervenant dans le domaine de l'ingénierie et des services [**Abeille et al., 2009**]. Il ressort de l'enquête que toutes les entreprises réalisent un couplage de la conception de produit et de la planification du projet de conception, cependant 18% des responsables interrogés n'en sont pas pleinement conscients. De plus, il se dégage que, dans la plupart des entreprises (82%), l'un des deux processus domine l'autre. Les décisions sont majoritairement prises par les acteurs d'un des deux processus, l'autre en subissant les effets et les conséquences, bien souvent trop tardivement. D'autre part, nous avons proposé une cartographie des processus de conception et de planification et nous avons identifié trois phases où l'utilisation d'outils d'aide au couplage est importante :

- lors de la phase d'avant projet afin de garantir l'adéquation entre la conception à réaliser et le planning prévisionnel à mettre en place,
- lors de la phase de conception détaillée, la conception et les plannings sont alors affinés et leur cohérence doit être suffisante pour poursuivre le projet,
- lors de la phase de suivi du projet, où il est important de pouvoir présenter des informations sur la réalisation des tâches de même que sur l'état des artefacts à concevoir.

Peu d'entreprises (22%) ont mené leur réflexion jusqu'à mettre en place des outils de travail collaboratifs. Les autres entreprises se limitent à une formalisation sous forme de procédures (45%) ou, plus simplement, font confiance aux acteurs humains (33%). D'un point de vue plus concret, les solutions existantes assurant un couplage de la conception et de la planification sont, pour la moitié des entreprises interrogées, des réunions de type revues de projet, revues de conception, vérification de conceptions, etc. Le quart des entreprises se base sur des standards normatifs tels que l'EIA-632. Enfin, des solutions plus « évoluées » sont évoquées comme l'utilisation de scénarios de référence ou de bases de données. En complément de cette analyse, le questionnaire et la fiche d'entretien ayant servi de cadre aux entretiens ainsi que le tableau synthétique des entretiens sont proposés dans l'annexe 1.

Ainsi, à partir de cette enquête, les besoins de couplage que nous avons identifiés concernent, de manière très générale :

- la mise en relation directe des acteurs de la conception et de la planification afin de leur permettre de partager les informations nécessaires à la prise de décision,
- l'offre de processus intégrés permettant une bonne synchronisation et,
- la formalisation, la capitalisation et l'exploitation de connaissances liant conception et planification.

À un niveau macroscopique, les processus de planification de projet de conception et de conception de systèmes, les interactions entre ces deux processus organisées par le couplage et l'apport de connaissances sur les processus mais également sur le couplage lui-même sont illustrés sur la figure 1.

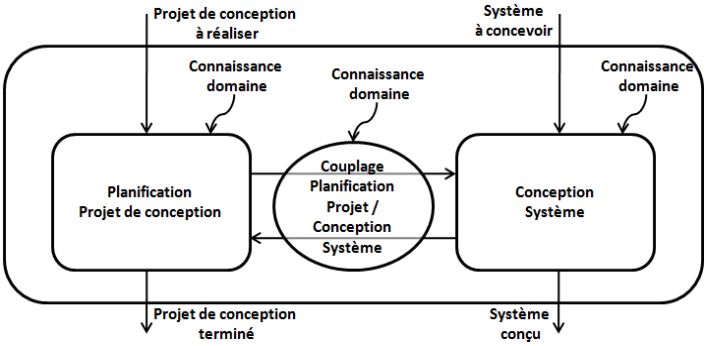


Figure 1 : Couplage Conception Système / Planification Projet

Le mémoire est organisé en sept chapitres.

Dans le **chapitre 1**, nous proposons un état de l'art concernant les travaux sur le couplage des processus de conception de systèmes (ou de produits) et de planification des projets associés. Après analyse des différentes approches, nous exposons la problématique de nos travaux de manière détaillée.

Ensuite, le mémoire est composée de deux parties de trois chapitres chacune. La première partie concerne la description des types de couplage que nous avons identifiés reposant uniquement sur des connaissances méthodologiques, c'est-à-dire sur la manière de structurer et capitaliser l'information de manière claire et non ambigüe, au niveau des projets, des systèmes et de leur association. Cette partie de décline en trois chapitres.

Le **chapitre 2** aborde le couplage structurel. Nous nous intéressons à la définition des entités de conception et des entités de planification, et formalisons les différentes possibilités d'association des unes avec les autres. Après analyse de ces possibilités, nous proposons un modèle pour le couplage ainsi qu'un processus de conception / planification intégré garantissant que les informations sur les entités couplées sont renseignées au moment opportun et de manière cohérente en permettant aux décideurs de réaliser des choix synchronisés.

Le **chapitre 3** présente le couplage informationnel. Ce couplage consiste à synchroniser l'état de chacune des entités couplées en réalisant des analyses de faisabilité et de vérification. Ensuite, nous formalisons des règles garantissant la cohérence du changement des caractéristiques de faisabilité et de vérification des entités couplées structurellement. Ces règles constituent des éléments de connaissances méthodologiques sur le couplage. Le modèle et le processus de conception / planification sont complétés pour supporter ce couplage informationnel.

Le **chapitre 4** expose le couplage décisionnel. Ce couplage est lié à l'utilisation par les décideurs, en conception et en planification, d'outils d'aide à la décision de type tableaux de bord. Nous présentons un processus de constitution de tableau de bord.

Dans la deuxième partie, outre les connaissances méthodologiques présentées dans la partie précédente, nous formalisons et exploitons des connaissances métier pour le couplage, c'est-à-dire des connaissances propres aux domaines d'application. Dans cette optique, cette partie se décompose en trois chapitres.

Le **chapitre 5** propose de formaliser la connaissance métier en conception et en planification grâce aux notions de concept et d'ontologie. Nous proposons un modèle d'ontologie et décrivons comment il peut être utilisé pour guider le développement des entités de conception et de planification.

Ensuite, nous faisons évoluer le modèle de la partie précédente afin de le rendre apte à supporter la connaissance métier.

Le **chapitre 6** permet de formaliser un processus intégré de planification / conception à base de cas. Le processus présenté dans les chapitres précédents est enrichi par la possibilité de réutiliser des acquis de planification et de conception pour aider le développement de nouveaux systèmes. Un processus de Raisonnement à Partir de Cas qui s'intègre dans les propositions précédentes est proposé.

Le **chapitre 7** est basé sur l'utilisation de problèmes de satisfaction de contraintes (Constraint Satisfaction Problems - CSP). Nous montrons l'utilisation de connaissances en conception et en planification grâce à des CSP puis nous étendons cela afin de formaliser des connaissances sur le couplage. Des contraintes sont ainsi définies entre les entités de planification et de conception. Des techniques de filtrage sont ensuite utilisées afin de propager l'effet de décisions prises dans un domaine vers l'autre et réciproquement. Nous modifions le modèle afin qu'il puisse supporter ce type de couplage et positionnons nos propositions dans le processus intégré de conception / planification.

Chaque chapitre est illustré par un exemple académique du domaine aéronautique.

Chapitre 1 : Etat de l’art, contexte et problématique

1 Etat de l'art, contexte et problématique

Le domaine de la conception de systèmes et celui de la planification de projets constituent des objets de recherche récurrents depuis plusieurs années et de nombreux travaux ont été menés dans ces deux champs d'investigation. L'objectif de ce chapitre est de présenter un panorama des approches, méthodes, outils et normes proposés en conception de systèmes et en planification de projets en focalisant sur l'idée d'**interaction** entre ces deux domaines. Ainsi nous présentons, dans un premier temps, plusieurs approches focalisées sur le processus de conception de système (section 1.1) puis des approches traitant davantage de l'aspect planification de projet (section 1.2). Ensuite, nous abordons les travaux associant ces deux domaines (section 1.3). Enfin, à partir des enseignements de ces travaux, nous précisons la problématique étudiée dans ce mémoire (section 1.4).

1.1 Conception de Système

Dans [**Finkelstein et Finkelstein, 1995**], le processus de conception de système est défini comme un processus de création qui débute par le recensement d'un besoin et s'achève par la définition d'un système qui répond au besoin et d'une méthode permettant sa réalisation (voir figure 1.1). De nombreuses définitions ont été proposées dans la littérature (notamment dans le champ de la systémique) pour caractériser la notion de système. Nous retiendrons, pour nos travaux, la définition suivante : un système est un ensemble de composants interdépendants qui interagissent de façon organisée dans un but commun. Les composants d'un système peuvent être divers et se composer de personnes, d'organismes, de procédures, de logiciels, d'équipements ou d'installations. Cette définition est globalement en accord avec celle proposée dans la norme ISO 15288 qui définit un système comme un ensemble d'éléments en interaction, organisés pour atteindre un ou plusieurs résultats déclarés. Les systèmes sont de natures très diverses et différentes typologies ont été proposées pour les classifier (systèmes technologiques, systèmes biologiques, systèmes socio-techniques...). Les travaux présentés dans ce mémoire s'adressent a priori aux systèmes dont les constituants sont des composants technologiques en interaction, à l'instar des systèmes abordés dans le cadre de l'ingénierie des systèmes.

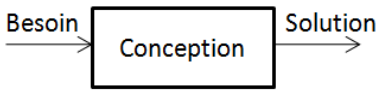


Figure 1.1 : Activité de conception

La démarche de conception de tels systèmes est naturellement très largement conditionnée par la connaissance dont dispose le concepteur (ou plus généralement l'équipe de conception). Dans [**Sriram et al., 1989**], une classification de la conception, en fonction de cette connaissance, est proposée. Elle distingue :

- **la conception routinière** : un plan permettant d'obtenir la solution existe a priori. La manière de décomposer le problème en sous problèmes plus simples et les différentes réponses possibles (alternatives) sont connues à l'avance. Elles sont issues d'une conception antérieure. La solution consiste principalement à effectuer les bons choix permettant de sélectionner l'alternative la mieux appropriée pour chaque sous-problème ;
- **la re-conception** : une conception existante est modifiée pour satisfaire de nouvelles fonctionnalités. La re-conception peut concerner des modifications à apporter à un produit commercialisé, nécessitées par de nouvelles exigences ou par un manque de performances. Il peut s'agir également d'une mise à jour, planifiée dans le cycle de vie

du produit avant son introduction sur le marché, afin qu'il reste compétitif [**Dieter, 2000**] ;

- **la conception innovante** : la décomposition du problème en sous problèmes est connue, mais les solutions pour chacun des sous problèmes n'existent pas forcément et doivent éventuellement être élaborées. Le concepteur utilise des principes fondamentaux du domaine pour développer les solutions ;
- **la conception créative** : il n'existe pas a priori de décomposition abstraite du problème en sous problèmes plus simples pour résoudre le problème de conception.

Signalons que [**Brown et Chandrasekaran, 1985**] propose également une classification de la conception en fonction de la connaissance en conception routinière, innovante et créative sans toutefois faire mention de la re-conception. Dans le cadre de nos travaux, nous proposons de traiter l'ensemble de ces classes et de proposer des mécanismes adaptés aux différentes situations pouvant être rencontrée.

La classification des approches de conception met aussi en évidence la notion d'alternative. En effet, la conception peut conduire à étudier plusieurs solutions potentielles répondant à un même besoin afin, par exemple, de limiter les risques de ne pas pouvoir produire de solution réaliste. Ces différentes alternatives, bien que n'ayant pas été retenues lors de la conception d'un système donné, peuvent être conservées afin de les réutiliser pour répondre à de futurs besoins dans un contexte différent (par exemple lorsque la maturité du marché est suffisante) [**Eynard, 2005**].

La conception étant une activité complexe, il existe de nombreuses approches offrant un cadre méthodologique permettant de la structurer. [**Bonjour, 2008**] mentionne entre autres les approches suivantes :

- la théorie C-K (Concept-Knowledge) de [**Hatchuel et Weil, 2002**] qui considère le raisonnement de conception comme une co-évolution par interactions d'un espace de concepts avec un espace de connaissances ;
- la conception systématique [**Pahl et Beitz, 1996**] qui comporte quatre phases utilisées comme modèle de référence dans de nombreux travaux. Ces phases sont la clarification des tâches (spécification de l'information), la conception conceptuelle (spécification des principes de la solution), la conception architecturale (spécification de la structure) et la conception détaillée (pour aboutir à une solution réalisable complètement décrite) ;
- le modèle Fonction Comportement Structure (ou FBS : *Functionnal Behaviour Structure*) [**Gero, 1990**] suggère une démarche de conception constituée de huit étapes élémentaires organisées autour de trois domaines [**Vermaas, 2007**] : le domaine fonctionnel (représentant les besoins du client), le domaine comportemental (permettant de décrire le fonctionnement du système à concevoir) et le domaine structurel (artefact physique devant réaliser les fonctions) ;
- la conception axiomatique [**Suh, 1990**] est basée sur le principe que la conception agit sur quatre domaines : le domaine clients (représenté par les besoins client CN), le domaine fonctionnel (représenté par des exigences fonctionnelles FR), le domaine physique (représenté par des paramètres de conception DP) et le domaine processus (représenté par des variables de processus PV).

En appui des démarches et méthodologies de conception précédemment listées, il existe de nombreux outils et/ou référentiels permettant de mettre en œuvre la conception de systèmes, c'est-à-dire d'aider le concepteur dans son activité de conception. Citons par exemple : les outils de gestion et

de traçabilité des exigences, de gestion de la configuration, de gestion des données techniques et des versions (SGDT), de conception assistée par ordinateur (CAO) ou, plus globalement, de gestion du cycle de vie des produits (PLM). Ces outils s'appuient sur différents modèles du produit et du processus de conception (fonctionnel, structurel, comportemental, géométrique, …).

En outre, et dans une vision plus globale, différents référentiels normatifs ont été proposés notamment dans le cadre de l'ingénierie système. Le travail des associations d'ingénierie système (INCOSE, AFIS) mené en collaboration avec de grands organismes de normalisation comme l'AFNOR, l'ANSI ou l'IEEE a conduit à la rédaction des grandes normes d'ingénierie système (IS).

Actuellement, trois normes font référence :

- l'IEEE 1220 qui se focalise sur les processus techniques d'ingénierie système allant de l'analyse des exigences jusqu'à la définition physique du système. La portée de cette norme dans le cycle de vie du système est limitée à la phase de définition du système ;
- l'EIA-632 dans laquelle les processus de conception technique sont associés aux processus d'acquisition et de fourniture. Processus techniques et processus contractuels sont pris en compte. Le cycle de vie du système est couvert explicitement par la norme qui s'étend de la définition du système à son transfert vers l'exploitation ;
- l'ISO 12588 qui s'adresse à l'ensemble du cycle de vie d'un système industriel. Il inclut les processus d'exploitation, de maintien en condition opérationnelle et de retrait de service.

Ces différents référentiels d'Ingénierie Système (IEEE 1220, EIA-632 et ISO 15288) proposent des modèles de processus. Ces modèles sont plus ou moins détaillés en fonction de la couverture du cycle de vie à laquelle les normes se rapportent. Ceci est illustré dans la figure 1.2.

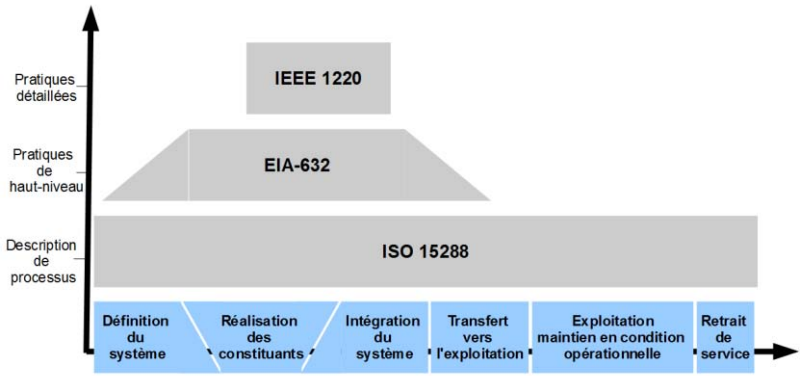


Figure 1.2 : Couverture du cycle de vie système par les normes d'IS [Rochet, 2007**]**

Dans le cadre de nos travaux, nous nous sommes concentrés sur la conception de système. La norme qui couvre le plus justement cette conception de système (définition, réalisation, intégration, transfert vers l'exploitation) est la norme EIA-632 [**Electronic Industries Alliance, 1999**]. La norme EIA-632 considère un système comme un ensemble de produits finaux (les produits qui vont réaliser les fonctions opérationnelles du système) et de produits contributeurs (les systèmes permettant aux produits finaux d'être développés, testés, mis à disposition, produits, déployés, supportés ou encore utilisés pour la formation du personnel). Dans le cadre de cette norme, chaque système est composé de sous-systèmes pouvant eux-mêmes inclure leurs propres sous-systèmes et ainsi de suite sans limite pré-établie dans la décomposition du système initial.

La norme EIA-632 propose, entre autres, treize processus et les résultats attendus de ces processus, ainsi que trente trois exigences. Ces processus sont classés en cinq groupes représentés sur la figure 1.3. Ces groupes de processus sont :

- le groupe *« management technique »* permettant de planifier, d'évaluer et de contrôler les efforts de travail technique exigés pour satisfaire l'accord établi,
- le groupe *« acquisition et fournitures »* qui est utilisé par les développeurs pour obtenir un accord avec une autre partie dans le but d'accomplir un travail spécifique ou de livrer les produits désirés,
- le groupe *« conception système »* qui est utilisé pour définir les exigences acceptées du client et, à partir de ces exigences, définir une solution (ensemble de produits réalisables satisfaisant le client et les autres parties prenantes),
- le groupe *« réalisation des produits »* permettant de : (1) convertir les caractérisations de solutions soit en un produit final vérifié, soit en un ensemble de produits finaux conforme au contrat et aux exigences des parties prenantes, (2) les livrer au client, (3) les installer au site opérationnel désigné et (4) fournir l'assistance de mise en service conformément au contrat,
- le groupe *« évaluation technique »* qui est destiné à être invoqué par un des autres processus pour réaliser un système. Il comporte notamment des processus d'analyse du système, de validation des exigences, de vérification du système et de validation des systèmes finaux.

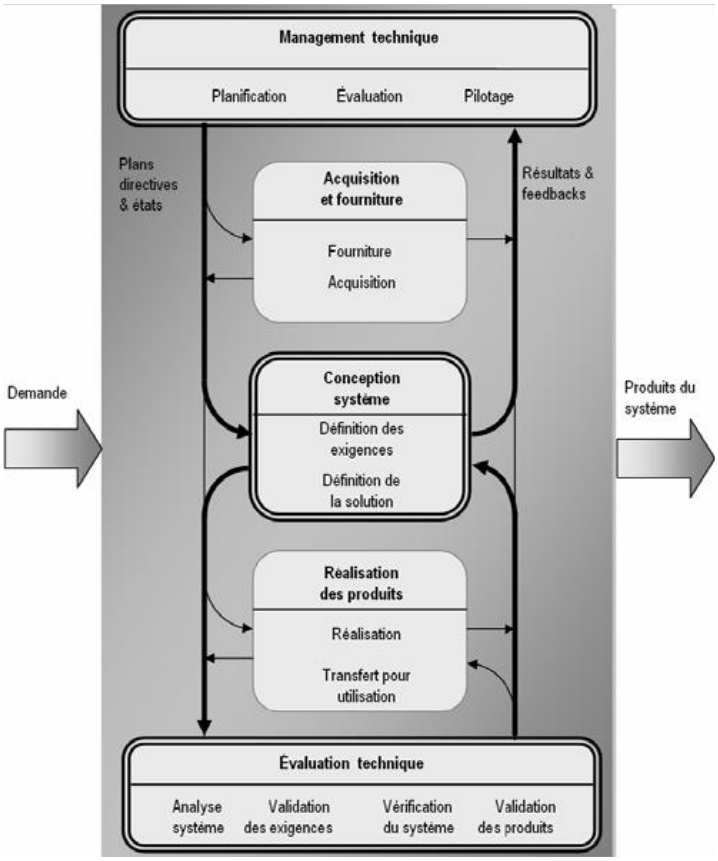


Figure 1.3 : Les processus de l'EIA-632 [Electronic Industries Alliance, 1999**]**

Nous pouvons remarquer que la norme EIA-632 associe clairement le système au projet de conception. En effet, cette norme propose un modèle de décomposition de système en produits finaux et contributeurs et y associe les processus nécessaires à la conception du système et donc de ses

composants. Le système est donc considéré comme un livrable du processus de conception. Ce standard n'offre cependant aucun outil ni méthode pour résoudre les problèmes ou développer des applications d'aide à la décision. Son aspect structurant est tout de même un point positif de même que son acceptation par les entreprises du secteur aéronautique pour construire leurs propres processus [**Rochet et Baron, 2008**].

1.2 Planification de Projet

Un projet peut être défini comme « une démarche spécifique qui permet de structurer méthodiquement et progressivement une réalité à venir » [**AFITEP-AFNOR, 2000**]. En complément de cette définition, il est précisé que « cette démarche est mise en œuvre pour élaborer une réponse au besoin d'un utilisateur, d'un client ou d'une clientèle et implique un objectif et des actions à entreprendre avec des ressources données ». Le concept de projet englobe ici l'idée d'un objectif à atteindre qui a été préalablement défini et la détermination d'un chemin et des actions à entreprendre pour l'atteindre ainsi que la spécification des moyens nécessaires [**Gourc, 2005**].

Dans le cadre de ce mémoire, nous faisons le choix de dissocier clairement la finalité du projet (le système à concevoir) et la détermination d'un chemin et des actions à entreprendre ainsi que la spécification des moyens nécessaires (la planification du projet de conception du système).

C'est ce deuxième point que nous allons documenter dans cette section, en nous appuyant sur le PMBOK Guide (Project Management Body Of Knowledge) [**Project Management Institute, 2004**]. Le PMBOK est en effet un guide de la gestion de projet et un standard reconnu internationalement. Il décrit les principes fondamentaux de la gestion de projet. L'objectif de ce guide est de proposer et de promouvoir un vocabulaire et un référentiel commun concernant la gestion de projet. Il est basé sur la notion de processus, ce qui signifie qu'il décrit le travail comme étant accompli suivant un ensemble de processus. Ces processus se déroulent la plupart du temps en parallèle en interagissant. Ils sont décrits par :

- leurs entrées (documents, plannings, …),
- les mécanismes appliqués aux entrées pour élaborer les sorties (outils et techniques),
- leurs sorties (documents, plannings, …).

Le PMBOK décrit la gestion de projet selon quarante quatre processus agrégés en cinq groupes de processus et concernant neuf domaines de connaissances communs à la plupart des projets. Les cinq groupes de processus sont l'initialisation, la planification, la réalisation, le pilotage et contrôle et la clôture du projet. Ils sont représentés dans la figure 1.4. Les neuf domaines de connaissances sur lesquels le management de projet s'étend sont : la gestion de l'intégration, du contenu (*scope*), du temps, des coûts, de la qualité, des ressources humaines, de la communication, du risque et des approvisionnements.

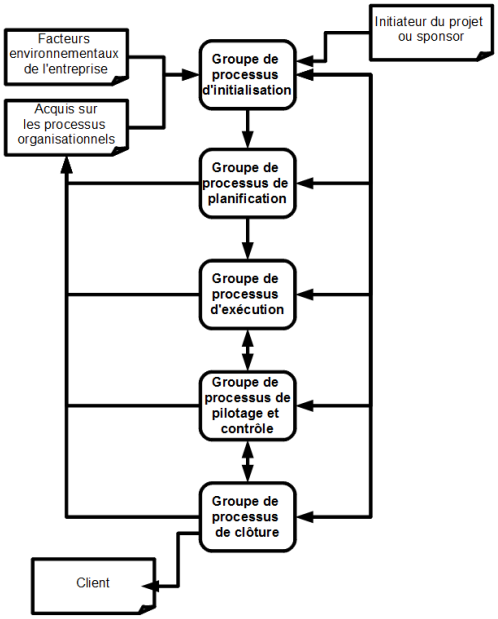


Figure 1.4 : Interactions entre les groupes de processus

Selon le Project Management Institute, à l'origine du PMBOK, la structure de découpage du projet (Work Breakdown Structure, WBS) est un modèle permettant une décomposition du travail, axée sur les tâches et activités, que l'équipe de projet doit exécuter pour atteindre les objectifs du projet et produire les livrables voulus. Un exemple de WBS est proposé dans la figure 1.5. Le WBS est à rapprocher du PBS (Product Breakdown Structure) qui est également une décomposition hiérarchique mais appliquée au système à concevoir, c'est à dire au résultat du projet. Enfin, il est possible de préciser la structure organisationnelle du projet (Organizational Breakdown Structure, OBS) en représentant les responsabilités de chaque membre pour chaque tâche du projet.

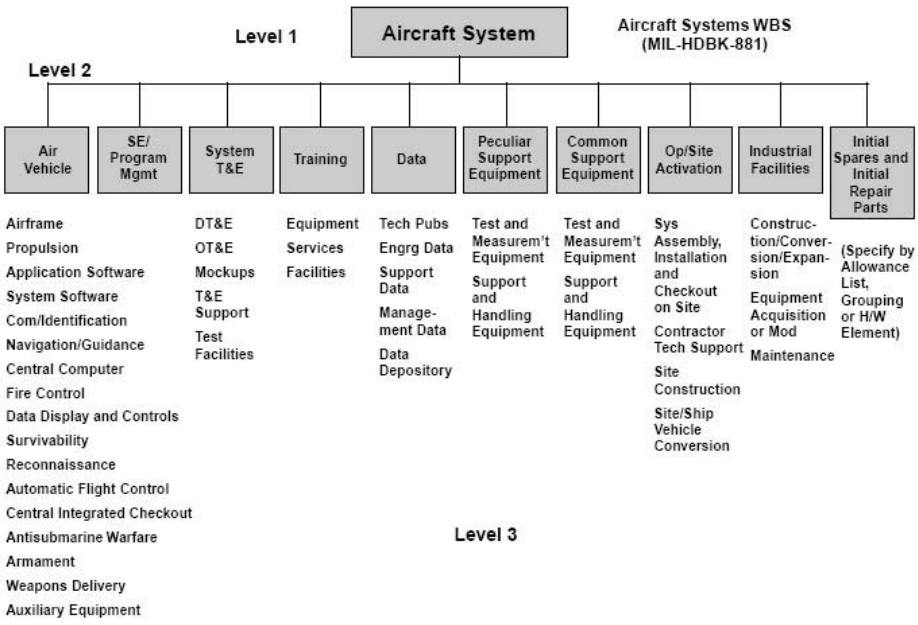


Figure 1.5 : Exemple illustrant les trois premiers niveaux de WBS pour un système avion
[Defense Acquisition University, 2001]

1.3 Travaux associant Conception et Planification

Nous venons de présenter rapidement plusieurs éléments qui nous semblent importants pour positionner la conception de système et la planification de projet dans une perspective de couplage. Nous allons maintenant nous intéresser plus spécifiquement aux travaux traitant de l'association de ces deux domaines. Nous distinguerons les approches :

- basées sur la conception axiomatique,
- à base de DSM et DMM,
- à base de contraintes,
- Project-Product Lifecycle Management,
- associant nomenclature du produit et ressources,
- associant processus de conception et diagramme de Gantt,
- par sélection de scénarios.

1.3.1 Approches basées sur la conception axiomatique

La première approche de couplage que nous décrivons est basée sur la conception axiomatique. Rappelons que la conception axiomatique [**Suh, 1990**] décrit le processus de conception comme un cheminement entre quatre domaines : le domaine clients (besoins client CN), le domaine fonctionnel (exigences fonctionnelles FR), le domaine physique (paramètres de conception DP) et le domaine processus (variables de processus PV). La conception axiomatique fait ainsi explicitement intervenir le domaine fonctionnel à l'interface entre le domaine client (les besoins) et le domaine physique (la solution). Le concepteur doit d'abord définir les exigences fonctionnelles qui permettront de répondre aux besoins du client avant de déterminer une solution physique pouvant satisfaire ces exigences et, au delà, les processus permettant de réaliser les solutions physiques.

[**Gonçalves-Coelho et al., 2005**] propose d'utiliser les quatre domaines définis en conception axiomatique (cf. section 1.1) et de formaliser des liens entre ces domaines à l'aide de matrices de type « QFD » (Quality Function Deployment). Les associations pouvant être représentées sont alors [CN,FR], [FR,DP] et [DP,PV]. Le premier terme de l'association [X,Y] est représenté en lignes alors que le deuxième terme est associé aux colonnes. Ceci aboutit à une cartographie des liens entre les domaines.

Globalement, cette cartographie associe la conception de produit (domaines client, fonctionnel et physique) à la planification de projet (domaine processus). [**Gonçalves-Coelho et al., 2005**] introduit également la notion de « zigzag » : en effet, d'après la conception axiomatique, le processus de conception doit être développé avec une approche descendante (top-down : du système vers ses composants) de manière itérative jusqu'à ce que la conception soit définie avec suffisamment de détails. De plus, pour concevoir un système, il faut mettre en œuvre des échanges d'informations entre domaines comme illustré sur la figure 1.6.

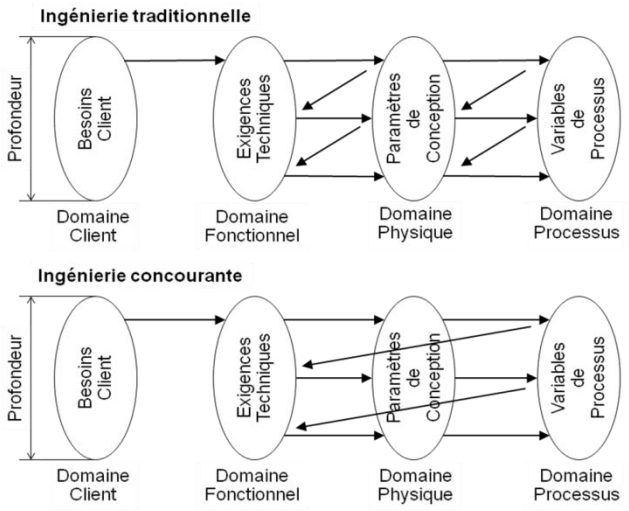


Figure 1.6 : Décomposition des processus en ingénierie traditionnelle et en ingénierie concourante [Gonçalves-Coelho et al., 2005]

A. Gonçalves-Coelho distingue deux cas pour organiser un processus de conception :

- l'ingénierie traditionnelle qui consiste à réaliser des zigzags entre le domaine fonctionnel et le domaine physique jusqu'à ce que le système soit décomposé avec un niveau de complexité et de détail acceptables puis de zigzaguer entre les domaines physiques et processus afin d'associer des activités aux différents éléments physiques dont la structure est entièrement connue,
- l'ingénierie concourante pour laquelle les zigzags englobent la totalité de la conception, c'est-à-dire qu'à un niveau de décomposition donné par le domaine physique, une activité du domaine processus doit être associée.

Ainsi cette étude suggère de définir successivement ou de manière simultanée les domaines spécifiques du système (domaine physique) et du projet (domaine processus) et de formaliser les dépendances entre ces deux domaines.

1.3.2 Approches à base de DSM et de DMM

Un travail assez similaire a été mené par U. Lindemann [**Lindemann, 2007**] en utilisant les matrices DSM (Design Structure Matrix) et les matrices DMM (Domain Mapping Matrice). Une matrice DSM présente les relations existant entre les éléments d'un système au sein d'un même domaine. C'est une matrice carrée notamment utilisée pour matérialiser les interactions entre les éléments d'un système, indépendamment du temps [**Bonjour, 2008**]. Les matrices DSM permettent ainsi de représenter les interactions entre les éléments d'un même domaine (par exemple les interactions entre deux composants).

Il est également possible de coupler deux matrices DSM grâce à une matrice DMM mettant en évidence les relations entre plusieurs domaines. Les matrices DMM permettent ainsi de modéliser les interactions entre les éléments de deux domaines (par exemple, les relations entre les fonctions et les composants). Dans [**Lindemann, 2007**], quatre domaines (fonctions, composants, processus et ressources) sont identifiés et les dépendances entre ces domaines sont formalisées comme illustré sur la figure 1.7.

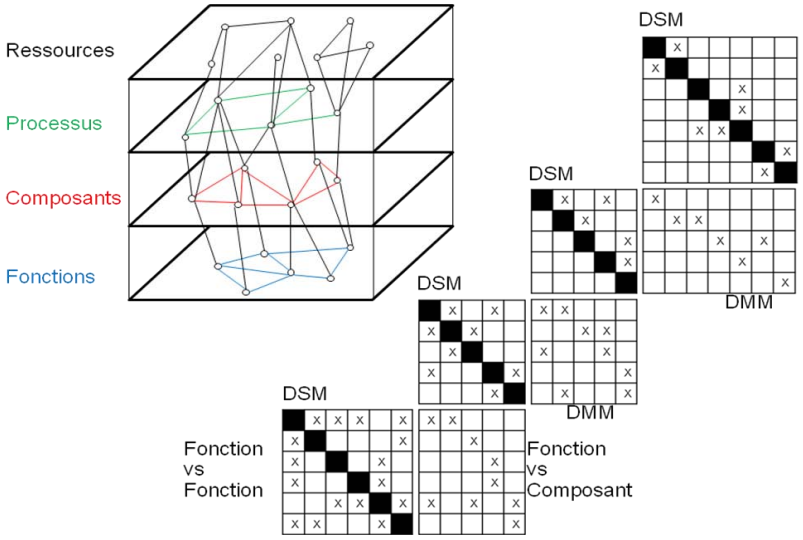


Figure 1.7 : Représentation d'une MDM [Lindemann, 2007**]**

Signalons que dans [**Danilovic et Browning, 2007**], les auteurs proposent une approche similaire en utilisant les matrices DSM et DMM. Les domaines sont interdépendants et peuvent être pris en compte ou non dans l'analyse en fonction des besoins de la conception.

Ces approches présentent toutefois des faiblesses lorsqu'il s'agit de séquencer les tâches d'un projet car elles ne tiennent pas compte explicitement de la notion de flux d'information devant circuler entre les tâches. Pour combler ce déficit, les approches proposées dans [**Chen et al., 2003**] ou [**Maheswari et Varghese, 2005**] utilisent des matrices de type DSM pour l'ordonnancement des tâches et la planification. Il s'agit de représenter les précédences (au sens des flux d'information requis pour passer d'une tâche à l'autre) entre les tâches sous forme matricielle puis d'organiser la séquence de tâches en fonction de leurs précédences en évitant, autant que possible, les itérations. Il est possible de représenter des précédences simples (matrices binaires) ou des précédences plus complexes comme par exemple des temps de communication entre les tâches.

Globalement, ces approches à base de DSM et DMM sont intéressantes puisqu'elles permettent de mettre en évidence les liens existants entre les domaines associés à la conception du système (fonctions et composants) et ceux associés à la planification du projet (processus et ressources). Elles restent toutefois inscrites dans une vision relativement statique et ne permettent qu'une prise en compte assez restreinte de la dynamique des processus de conception.

1.3.3 Approches à base de contraintes

Dans [**Aldanondo et al., 2007**], les auteurs proposent d'intégrer un CSP correspondant au système et un CSP correspondant au projet (exemple figure 1.8). Un problème de satisfaction de contraintes (Constraint Satisfaction Problem, CSP) est un problème représenté formellement où l'on cherche des états ou des objets satisfaisant un certain nombre de contraintes ou de critères. D'un point de vue général, un problème de satisfaction de contraintes est défini à partir d'un ensemble de variables, chacune prenant valeur dans un domaine, et d'un ensemble de contraintes qui restreignent les valeurs que les variables peuvent prendre simultanément. La résolution consiste à assigner une valeur à chaque variable en satisfaisant toutes les contraintes [**Tsang, 1993**] [**Dechter, 2003**] [**Barták et al., 2010**].

L'intégration entre un CSP représentant le système et un CSP représentant le projet impose de définir des contraintes supplémentaires entre les éléments des deux CSP, associant ainsi les deux domaines. Cette approche a été illustrée sur deux exemples : le premier est appliqué à la conception d'un bateau [**Vareilles et al., 2008**] et le second à la conception d'une grue [**Aldanondo et al., 2009**]. Le premier exemple est représenté sur la figure 1.8.

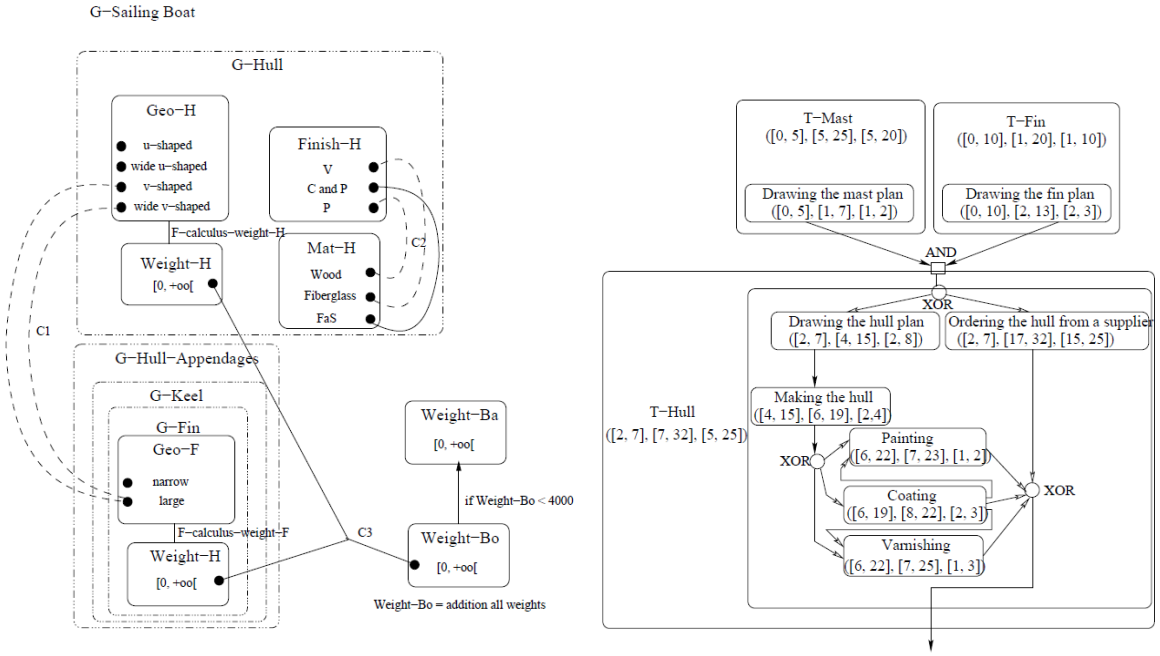


Figure 1.8 : Exemple de modèle CSP d'un navire et du projet associé

Cette approche a pour intérêt de formaliser au sein des modèles CSP, les liens existants et de pouvoir ainsi répercuter automatiquement une décision d'un domaine vers un autre en résolvant le CSP. Cependant, avant de pouvoir utiliser le CSP, il est nécessaire de procéder à l'extraction, l'interprétation et la formulation de la connaissance sous forme de contraintes.

1.3.4 L'approche PPLM (Project Product Lifecycle Management)

[**Sharon et al., 2008**] s'est intéressé au pilotage de projets pour la réalisation de systèmes complexes. Une étude a été réalisée sur les méthodes de management de projet et leur aptitude à produire des résultats sur les dimensions projet, produit et sur l'ensemble projet-produit. La méthodologie OPM (Objet-Processus) est, dans cette étude considérée comme la mieux adaptée à l'ensemble des dimensions. Les travaux présentés ci-dessous sont basés sur cette méthodologie.

Ces travaux s'inscrivent dans le cadre de l'approche Project-Product Lifecycle Management (PPLM). Cette approche intègre le domaine projet et le domaine produit (ou système) grâce à une ontologie associant les entités du projet aux entités du produit proposant ainsi un cadre de référence unifié. Cette approche à pour but de fusionner le produit qui doit être développé avec le projet tel qu'il doit être réalisé dans une entreprise donnée [**Sharon et al., 2008**].

Ainsi, un cadre est proposé pour l'approche PPLM. Le pilotage du PPLM requiert à la fois des données sur le projet et sur le produit qui sont respectivement obtenues des processus de pilotage du produit et de pilotage du projet. Chacun de ces processus est réalisé par une équipe de pilotage adaptée incluse dans l'équipe de pilotage global et disposant d'un ensemble d'outils de pilotage propres. Pour le pilotage projet, les outils proposés sont : un outil de management financier, un outil de gestion des

plannings et des ressources, un outil WBS et un outil de gestion des risques. Pour le pilotage produit, les outils utilisés permettent la gestion des exigences, la gestion des fonctionnalités, la gestion d'architecture, la gestion des interfaces, l'analyse de comportement du système et l'intégration et tests. La figure 1.9 illustre ce cadre pour l'approche PPLM à haut niveau d'abstraction.

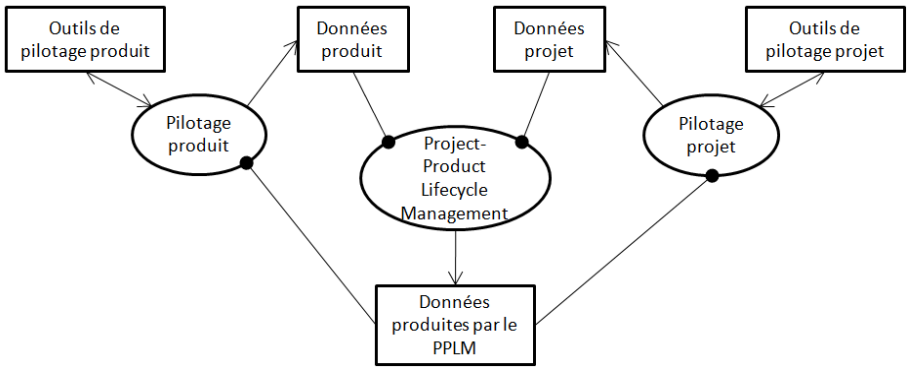


Figure 1.9 : Cadre du Project-Product Lifecycle Management [Sharon et al., 2008]

Cette approche est incrémentale, c'est-à-dire que le développement est réalisé pas à pas en intégrant successivement l'affectation des ressources. L'approche PPLM est également récursive, c'est-à-dire que le cadre présenté dans la figure 1.9 s'applique à l'ensemble projet/produit mais également aux sous-ensembles (sous-projet/sous-systèmes) [**Sharon et al., 2009**].

L'intérêt de la démarche proposée réside à nos yeux d'une part dans l'idée d'exploiter une ontologie support du couplage entre le domaine système et le domaine projet et, d'autre part, de positionner la démarche de couplage dans une perspective opérationnelle (i.e. possible à instrumenter) avec la notion de PPLM.

1.3.5 Approches associant nomenclature du produit et ressources

Les travaux exposés par B. Oosterman [**Oosterman, 2001**] concernent l'organisation des tâches autour de la notion de « building-block », entité représentant un composant du produit final. Les travaux s'appuient sur deux principes :

- les interactions entre les *building blocks* sont moins importantes que celles à l'intérieur même d'un *building block*,
- l'organisation classique du projet consiste à structurer les tâches élémentaires en groupes de tâches indépendants les uns des autres dans le but de minimiser les efforts de coordination entre les tâches et d'augmenter la vitesse d'exécution du projet.

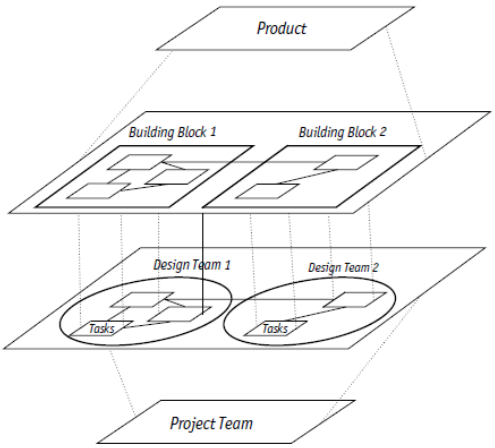


Figure 1.10 : Organisation du travail reflétant l'architecture du produit.

L'auteur déduit de cette observation qu'il est possible d'allouer la réalisation d'un *building-block* à une unité organisationnelle afin d'obtenir des projets de conception efficaces. Il aboutit à une organisation du travail reflétant l'architecture du produit à concevoir comme illustré sur la figure 1.10. L'approche de B. Oosterman consiste, à partir des processus de conception actuels, à représenter l'architecture du produit puis déterminer les besoins de coordination entre les équipes de conception.

Dans ce même ordre d'idée, S. Fixson, dans [**Fixson, 2005**], montre que l'architecture du produit, ses processus de réalisation et la chaîne logistique sont étroitement liés et que des décisions prises dans un des domaines a des conséquences dans les deux autres. Il propose un cadre multidimensionnel consistant à allouer les fonctions attendues à des composants sous forme matricielle puis, à définir les interfaces entre les composants, ce qui permet de constituer l'architecture du produit. Le nombre de composants et le nombre d'interfaces entre composants sont des facteurs importants dans le coût global du produit (et donc le temps passé à le concevoir) mais ils sont nuancés par le degré de réutilisation des composants. Il précise, comme B. Oosterman, que la structure des unités de développement de produit tend à se calquer sur l'architecture du produit dans le cadre des industries « stables ». L'idée développée dans ces travaux est que le choix d'une architecture de produit a un impact important sur le processus permettant de concevoir le produit.

Un des points clés intéressant pour notre propre problématique est que dans ces travaux, un lien unique existe entre un composant du produit global et une équipe de conception.

1.3.6 Approches associant processus de conception et diagramme de Gantt

Dans [**Steward et Tate, 2000**], les auteurs proposent d'utiliser les domaines fonctionnel (exigences fonctionnelles - FR) et physique (paramètres de conception - DP) de la conception axiomatique. Il s'agit ici de convertir les exigences fonctionnelles en paramètres de conception en définissant une matrice de conception pour coupler les deux domaines. Cette matrice contient des valeurs numériques caractérisant l'impact de chaque exigence technique sur les paramètres de conception. Les paramètres de conception définis sont ensuite considérés comme des tâches à réaliser et deviennent les tâches d'un diagramme de Gantt, les liens entre les tâches étant définis par la matrice de conception.

Cette démarche a été outillée grâce à un développement ad hoc couplé au progiciel Microsoft Project et testée dans le cadre d'une entreprise d'ingénierie logicielle. L'approche montre l'intérêt de

construire un diagramme de Gantt (et donc de planifier) au plus tôt dans le processus de conception ainsi que de développer des outils d'aide à la décision adaptés.

Un autre travail, présenté dans [**Lu et al., 2008**], propose une démarche permettant d'intégrer le processus de conception de produit et le management des tâches. La structure du projet, définie sous forme de WBS (structure hiérarchique de décomposition du travail), est directement dérivée du PBS (structure de décomposition du produit). Cette dérivation est réalisée en agrégeant, décomposant et associant les deux structures sous forme de matrice (une partie du produit est associée à une partie du projet). Cette démarche est outillée par une plateforme logicielle de démonstration.

1.3.7 Sélection de scénarios

Dans [**Pitiot, 2009**] et [**Pitiot et al., 2010**], les auteurs proposent d'utiliser un algorithme évolutionnaire guidé par l'utilisation d'un mécanisme de retour d'expérience pour la sélection de scénarios en conception préliminaire de produit / projet. Pour cela, un modèle représentant l'ensemble produit / projet sous forme d'un graphe [**Rochet et Baron, 2008**] est utilisé. Ce modèle constitue le lien entre conception et planification puisqu'on retrouve à la fois l'architecture produit avec la représentation de toutes les alternatives envisagées en conception et les tâches du projet. Dans le graphe, les alternatives de conception et de projet (matérialisées par de nœuds OU) sont représentées avec, pour chacune, des indicateurs de coût et de délai. Ces alternatives sont testées par l'algorithme évolutionnaire afin d'optimiser le coût et le délai. Par ailleurs, afin de guider l'algorithme évolutionnaire, un réseau Bayésien permet de capitaliser la connaissance sur l'influence des décisions de conception et de projet sur l'atteinte des objectifs. Le modèle est représenté sur la figure 1.11.

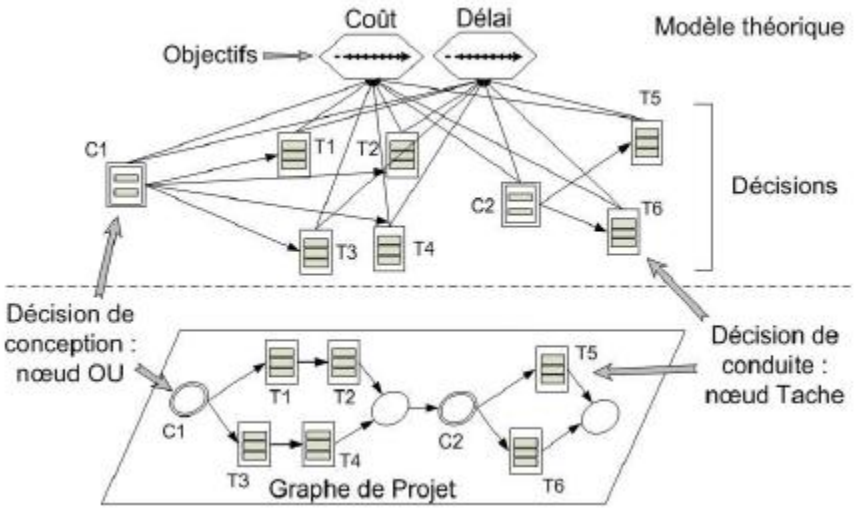


Figure 1.11 : Modèle de capitalisation de cas [Pitiot, 2009**]**

Dans le même ordre d'idée, [**Jiao et al., 2008**] propose d'extraire des règles d'association entre le produit et le processus associé. La démarche consiste, à partir de l'ensemble des variantes d'une famille de produit et de l'ensemble des chemins possibles dans le processus, à réaliser une cartographie de ces deux ensembles et à partir de cela, à définir des modèles génériques de produit et de processus permettant de répondre à l'ensemble des variantes. Il est alors possible de déduire le processus adéquat à partir de la variante de produit désirée.

Ces travaux présentent l'intérêt de prendre en compte plusieurs alternatives de conception et donc de projet et de définir quelle solution (produit et projet) répond le mieux aux fonctions objectif à optimiser.

1.4 Contexte et Problématique

Nous avons présenté dans la section précédente plusieurs approches visant à mettre explicitement en relation la conception de système et la planification de projet. Il apparaît que les travaux associant conception et planification sont relativement peu nombreux, restent essentiellement descriptifs et ne se basent que de manière limitée sur une explicitation et une exploitation des connaissances. Enfin, l'outillage (par exemple à l'aide de logiciels) des propositions de couplage reste assez restreint et ne permet pas de prendre en compte la dynamique du processus de conception. Les travaux associant conception et planification restent donc globalement assez abstraits et sont peu utilisés dans le monde industriel.

Afin de répondre à ces différents problèmes, nous proposons donc de développer :

- un modèle cohérent du couplage entre conception et planification,
- l'identification d'une typologie des situations d'interaction entre conception et planification,
- un outillage logiciel pour l'aide à l'interaction entre conception et planification.

La problématique que nous abordons dans ce travail vise donc à définir précisément le couplage entre conception et planification ainsi que son instrumentation. Le couplage correspond selon nous :

- aux interactions entre un environnement de conception de système et un environnement de planification du projet de conception (par exemple, une prise de décision de conception peut avoir un impact en planification et réciproquement),
- à leur mise en cohérence consistant à garantir que les informations des deux environnements ne sont pas en contradiction.

Il apparaît que ce couplage est fortement lié à la connaissance donc les acteurs disposent. Nous faisons la distinction entre :

- la connaissance méthodologique, c'est-à-dire une connaissance « générique » pouvant être appliquée à n'importe quelle conception ou planification et,
- la connaissance métier du domaine de l'acteur, c'est-à-dire une connaissance plus spécifique à la conception ou la planification. Par exemple, dans le domaine aéronautique, il est possible de définir des architectures de systèmes ou de projets et de définir des paramètres inhérents à chaque famille de produit.

En conséquence, nous proposons un modèle intégrant à la fois la conception de systèmes et la planification des projets associés. Ce modèle doit permettre de mettre en application les types de couplage que nous aurons définis. Le modèle doit également pouvoir être implanté dans un outil d'aide à la décision faisant intervenir à la fois la connaissance méthodologique et la connaissance métier.

Nous avons, dans cet objectif, recensé cinq types de couplage :

- le couplage des entités qui consiste à associer les entités de conception (par exemple, un système) aux entités de planification (par exemple, un projet). Ce couplage est nommé couplage structurel dans la suite de ce mémoire,
- le couplage informationnel dont le but est de synchroniser les évolutions des processus de conception de produit et de planification de projet de développement associé,
- le couplage décisionnel qui doit aider à la décision en cas de conflit entre les concepteurs et les planificateurs ou en cas de choix multiples,

- le couplage par réutilisation qui consiste à stocker les conceptions et les planifications précédemment réalisées puis les réutiliser dans de nouveaux cas similaires,
- le couplage par contraintes qui doit permettre, après la réalisation de plusieurs cas plus ou moins similaires, d'en tirer des enseignements généraux sous forme de contraintes faisant intervenir à la fois des données de conception et de planification.

Dans un premier temps, nous proposerons un modèle permettant de prendre en compte la connaissance méthodologique (Partie I). Dans un second temps, nous nous intéresserons à l'intégration à ce modèle de la connaissance métier (Partie II).

Partie I : Couplage basé sur les connaissances méthodologiques

Partie I : Couplage basé sur les connaissances méthodologiques

Le couplage des activités de conception avec celles de définition et de planification de ces mêmes activités nécessite des modèles adaptés, que ce soit pour capitaliser les informations nécessaires ou pour mettre en œuvre des processus. Ces connaissances, devant être capitalisées et exploitées systématiquement, sont méthodologiques. Les modèles proposés à partir des entités fondamentales nécessaires au couplage sont exposés dans cette section. Nous proposons ainsi une classification des modes de couplages entre conception et planification en trois parties distinctes et complémentaires : le *couplage structurel*, le *couplage informationnel* et le *couplage décisionnel*.

Le premier mode de couplage, le couplage structurel décrit dans le chapitre 2, consiste à définir des entités de conception et des entités de planification simultanément afin de garantir que l'existence des unes est bien liée à l'existence des autres. Ainsi, une structure de système est liée ou couplée à une structure de projet. Afin de garantir ceci, un processus de conception / planification est proposé. Etant donné les multiples possibilités de définition d'entités et d'association de ces entités entre elles, nous débutons par la description des entités nécessaires aux modèles ainsi que leur sémantique et critiquons les différentes manières de les associer. Suite à cela, nous nous basons sur la bijection des structures projet et système et le modèle de couplage structurel est proposé. Une fois ce modèle pertinent pour le couplage obtenu, nous proposons un processus de conception / planification permettant de garantir que les informations sur les entités couplées sont renseignées au bon moment avec une parfaite cohérence en permettant aux décideurs, en conception et en planification, de prendre leurs décisions avec un maximum d'informations pertinentes.

Le second mode de couplage, le couplage informationnel décrit dans le chapitre 3, consiste à définir des règles de cohérence à partir de l'état des informations stockées dans les entités couplées. En effet, afin d'avoir des informations cohérentes et pertinentes, certaines analyses de faisabilité des solutions envisagées ou envisageables et de vérification des solutions obtenues sont nécessaires. Ainsi, dans cette section, nous définissons la sémantique liée à cette notion de cohérence des informations puis nous définissons les règles permettant de garantir celle-ci.

Le dernier mode de couplage envisagé, le couplage décisionnel décrit dans le chapitre 4, est lié à l'utilisation par les responsables en conception et en planification d'outils d'aide à la décision de type tableau de bord de couplage. Les décisions dans un domaine ou dans un autre doivent être prises en considérant l'ensemble des informations couplées. Nous définissons la notion de tableau de bord puis nous proposons un processus pour leur constitution.

Chapitre 2 : Couplage structurel

2 Couplage structurel

Dans ce chapitre, nous présentons dans un premier temps les entités fondamentales du problème de couplage et définissons ce que nous entendons par couplage structurel. Dans un deuxième temps, nous proposons un modèle de classe supportant ce couplage. Dans un troisième temps, nous présentons les processus exploitant le couplage structurel et nous illustrons sa mise en œuvre sur un exemple concret de conception d'un avion d'affaire.

2.1 Entités fondamentales du couplage structurel

Dans un premier temps, nous présentons les entités de conception et de planification en les définissant et en explicitant leur sens. Dans un deuxième temps, nous abordons la décomposition de ces entités. Dans un troisième temps, nous étudions les différentes mises en relation de ces entités. Enfin, nous proposons une définition du couplage structurel.

2.1.1 Définition et sémantique des entités fondamentales

2.1.1.1 *Entités de conception*

Nous rappelons que nous avons défini la conception de système comme étant l'action de transformer des besoins émis par un client en un système conçu, un système étant un ensemble de composants interdépendants qui interagissent entre eux de façon organisée vers un but commun (voir section 1.1).

Notre proposition est de considérer l'entité **Système (S)** comme l'ensemble (voir la partie gauche de la figure 2.1) :

- des besoins et exigences techniques, que nous appellerons entité **Exigences Système (ES),** auxquels il doit répondre et,
- une ou plusieurs solutions, nommées entités **Alternatives Système (AS)** répondant à ces besoins.

Ce choix permet de fédérer, au sein d'une même entité, ce que l'on souhaite que le système réalise (les exigences) et les solutions envisagées pour satisfaire ces exigences (les alternatives). Plusieurs solutions peuvent coexister en parallèle avec des niveaux de maturité, d'avancement et de complexité différents. Par exemple, il peut s'agir d'une solution dont la technologie est éprouvée et d'une autre solution mettant en œuvre une technologie plus récente. De même, une solution peut être en cours de conception alors qu'une autre est totalement conçue. Aussi, une solution peut ne nécessiter que peu de travail de conception alors qu'une autre, plus complexe et/ou plus créative, nécessitera l'intervention de plusieurs équipes lors de la conception.

Nous regroupons sous le vocable **entité de conception** trois entités qui sont l'entité Système S, l'entité Exigences Systèmes ES et l'entité Alternative Système AS. L'ensemble de ces entités est sous la responsabilité d'un responsable de conception.

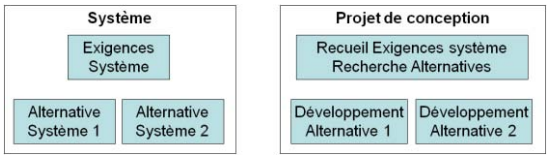


Figure 2.1 : Composition d'un système et d'un projet

2.1.1.2 *Entités de planification*

Nous rappelons également que nous avons défini la planification d'un projet comme étant la recherche du chemin et des actions à entreprendre pour atteindre un objectif. Dans le cas présent, l'objectif est la conception d'un système (voir section 1.2).

Nous proposons donc de considérer l'entité **Projet de conception (P)** comme un ensemble de deux entités tâches prédéfinies, tel qu'illustré dans la partie droite de la figure 2.1. Ces deux tâches sont :

- une entité **Tâche de recueil d'Exigences système et de recherche d'alternatives (TE)** durant laquelle le système est caractérisé, les besoins et exigences techniques du système sont recueillis et les alternatives de conception sont recherchées,
- une ou plusieurs entités **Tâches de Développement d'alternative (TD)** (autant que de solutions potentielles à explorer) qui correspond au management de la définition d'une solution pour une alternative système donnée.

Dans la suite de ce mémoire, nous considérons qu'un projet de conception comprend la définition des besoins et exigences techniques du système ainsi que la recherche et la définition des solutions. Une tâche de développement d'alternative ne comprend donc que la définition d'une solution : cette dernière ne sera donc pas considérée comme un projet de conception à part entière.

Une relation de précédence existe entre la tâche TE et la (les) tâche(s) TD, les besoins communs à toutes les alternatives devant nécessairement être identifiés avant de débuter le travail de conception des solutions. La tâche TE doit donc être réalisée avant toute tâche TD dans un projet de conception. Plusieurs tâches de développement d'alternative TD peuvent coexister avec des niveaux de maturité, d'avancement et de complexité différents. Par exemple, il peut s'agir d'une tâche TD dont le contenu (sous-tâches, délais, coûts…) est éprouvé et d'une autre, menée en parallèle, et mettant en œuvre un contenu plus récent dans lequel une certaine incertitude existe. De même, une tâche TD peut être en cours de réalisation alors qu'une autre est totalement achevée ou encore, elle peut nécessiter peu de ressources alors qu'une autre, plus complexe, nécessitera l'intervention de plusieurs équipes.

Ces deux tâches sont complétées par une tâche de management de projet qui possède les mêmes dates de début et de fin ainsi que la même durée que l'entité projet de conception. Cette tâche est conduite par un responsable de planification. Elle correspond au travail de définition des objectifs et des contraintes du projet ainsi qu'au management du projet de conception dans son intégralité. Cette tâche est active tout au long du projet. Dans l'optique du couplage que nous souhaitons mettre en place, cette tâche n'intervient pas dans les relations entre entités de conception et de planification. Elle n'est donc pas considérée dans la suite de ce mémoire.

Nous regroupons sous le vocable **entité de planification** trois entités qui sont l'entité Projet de conception P, l'entité tâche TE et l'entité tâche TD. L'ensemble de ces entités est sous la responsabilité d'un responsable de planification.

2.1.2 Décomposition des entités fondamentales

2.1.2.1 *Entités de conception*

En complément des notions que nous venons d'aborder, un système peut être jugé trop complexe pour être conçu dans son ensemble. Dans ce cas, le standard d'Ingénierie Système EIA-632 suggère de décomposer le système en sous-systèmes plus « simples » de manière récursive sans limite préétablie (voir section 1.1). Nous proposons donc qu'une entité **alternative système AS** puisse **se décomposer en n entités système S,** nommés sous-systèmes, si la complexité l'exige. La

décomposition s'arrête quand le système à concevoir est suffisamment « simple » pour être conçu localement : il fait alors appel à des composants élémentaires ou des fournitures « sur étagère ». Par exemple, un banc de test peut être décomposé en deux parties distinctes (mécanique et électrique), chacune faisant l'objet d'un sous-système. Dans la partie gauche de la figure 2.2, l'entité Alternative Système 2 est décomposée en deux sous-systèmes nommés Sous-système 1 et Sous-système 2 alors que l'entité Alternative Système 1 ne l'est pas.

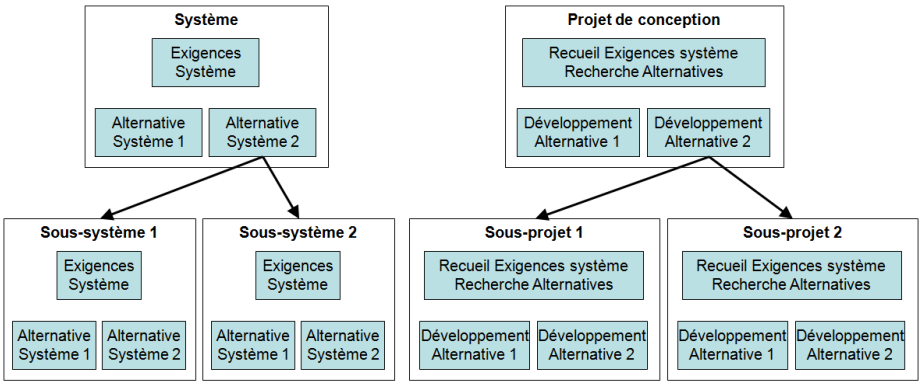


Figure 2.2 : Décomposition d'un système en sous-systèmes et d'un projet en sous-projets

2.1.2.2 *Entités de planification*

De même, un projet peut être jugé trop complexe pour être managé dans son ensemble. De manière identique à la décomposition d'un système, nous proposons donc **qu'une entité tâche de développement d'alternative TD** puisse se **décomposer récursivement en n entités projet de conception P**, nommés sous-projets, sans limite préétablie, si la complexité l'exige (voir partie droite de la figure 2.2). La décomposition s'arrête quand le projet de conception est suffisamment « simple » pour être piloté localement. Par exemple, un projet de conception d'un banc de test, jugé trop complexe à gérer par un seul responsable de planification, peut être décomposé en deux sous-projets : l'un correspondant à la conception de la partie mécanique et le second à la partie électrique. Dans la partie droite de la figure 2.2, l'entité Tâche de Développement d'Alternative 2 est décomposée en deux sous-projets nommés Sous-projet 1 et Sous-projet 2 alors que l'entité Tâche de Développement d'Alternative 1 ne l'est pas.

2.1.3 Mise en relation des entités fondamentales

Ceci étant posé, nous appelons **mise en relation des entités** le fait d'associer directement une ou plusieurs **entités de conception** à une ou plusieurs **entités de planification**. L'analyse des liens possibles entre les entités Système et les entités Projet de conception nous a permis de recenser trois possibilités de mise en relation :

- **Une entité Système est liée à une entité Projet de conception et réciproquement (voir figure 2.3).** Dans cette situation, une entité Système est pilotée par une seule entité Projet de conception et chaque sous-système est piloté par un seul sous-projet. Dans ce cas, un seul responsable de planification est responsable du pilotage de l'entité Projet de conception associée à une entité Système. Le risque d'exigences contradictoires est donc maîtrisé dans la mesure où ces dernières sont caractérisées lors de la réalisation d'une tâche TE spécifique à chaque système ou sous-système. De plus, si la tâche TE associée à un sous-système nécessite une compétence particulière pour être réalisée, le sous-projet associé pourra prendre en compte cette exigence. La décision de décomposition en sous-systèmes pouvant être prise en cours de conception d'une alternative, les sous-projets associés seront créés à ce moment avec

nécessairement un pilotage adapté et notamment grâce à un choix de ressources opportun. Cependant, ceci implique une certaine rigueur dans le déroulement du processus de conception et de planification. En effet, une entité Système, aussi simple soit-elle, étant impérativement pilotée par une entité Projet de conception, il faut prendre garde à ne pas décomposer systèmes et projets de conception avec une granularité trop fine menant ainsi à un trop grand nombre d'entités et donc à un trop fort partitionnement de l'information.

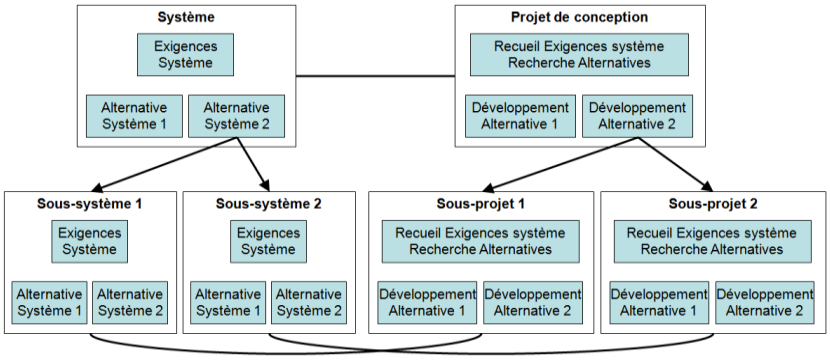


Figure 2.3 : Mise en relation d'une entité Système avec une seule entité Projet de conception

- **Une entité Projet de conception est liée à plusieurs entités Système.**
  - **Sur plusieurs niveaux de décomposition (voir figure 2.4).** Ici, une entité Système et ses sous-systèmes sont pilotés par une seule entité Projet de conception. Cette mise en relation présente l'intérêt de « factoriser » la tâche de management de projet : le responsable de planification est le même pour l'entité Système et ses sous-systèmes. En revanche, les informations de planification relatives à une entité Système en particulier ne pourront pas être facilement extraites dans ce pilotage global. De plus, la tâche TE doit permettre d'identifier les exigences du système et de tous les sous-systèmes simultanément avant de débuter la conception des solutions. Ceci peut poser des problèmes dans la mesure où le travail de caractérisation des exigences d'un sous-système peut necessiter un fort niveau d'expertise pour être réalisé et que le choix de décomposition en sous-systèmes peut s'avérer difficile à anticiper avant même d'avoir débuté le travail de conception.

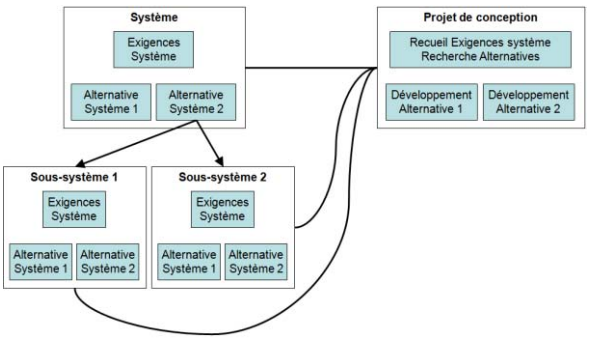


Figure 2.4 : Mise en relation d'une entité Projet de conception avec plusieurs entités Système sur plusieurs niveaux de décomposition

  - **Sur un même niveau de décomposition (voir figure 2.5).** Ici, plusieurs sous-systèmes sont pilotés par un même projet sur un même niveau de décomposition.

Cela signifie que plusieurs sous-systèmes sont gérés par un même responsable de planification. Le risque pour ce dernier est de rencontrer des difficultés dans le management de plusieurs équipes de concepteurs différentes et potentiellement distribuées. En outre, les informations concernant les exigences des sous-systèmes vont être identifiées lors de la réalisation d'une seule et même tâche TE, ce qui peut entraîner des incohérences et des objectifs de conception peu détaillés ou mal définis.

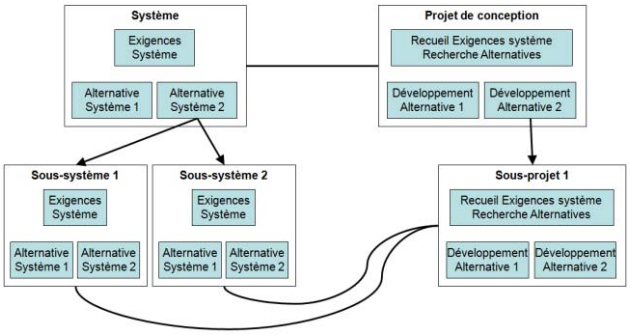


Figure 2.5 : Mise en relation d'une entité Projet de conception avec plusieurs entités Système sur un seul niveau de décomposition

- **Une entité Système est liée à plusieurs entités Projet de conception.**
  - **Sur plusieurs niveaux de décomposition (voir figure 2.6).** Ici, une entité Système est pilotée par une entité Projet de conception et plusieurs sous-projets. Cette mise en relation suggère que le projet de conception est trop complexe pour pouvoir être piloté par une seule tâche de management de projet : le risque est d'avoir des tâches exécutées plusieurs fois par des acteurs différents et surtout, d'obtenir de nombreuses exigences contradictoires ou des objectifs conflictuels.

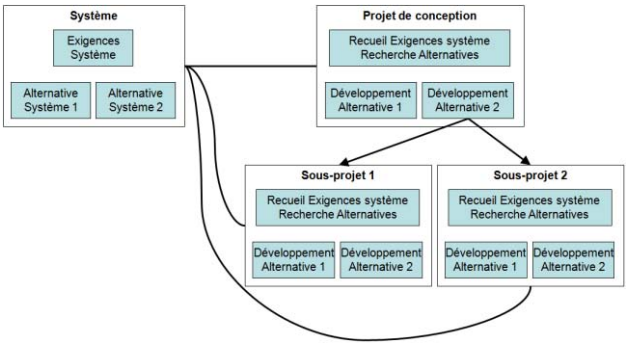


Figure 2.6 : Mise en relation d'une entité Système avec plusieurs entités Projet de conception sur plusieurs niveaux de décomposition

  - **Sur un même niveau de décomposition (voir figure 2.7).** Ici, une entité Système est pilotée par plusieurs entités Projet de conception sur un même niveau de décomposition. Cela suppose, si on tient compte du projet tel qu'il a été décrit précédemment, que la conception d'un système peut être pilotée par plusieurs responsables de planification. Les écueils cités dans le paragraphe précédent sont également présents ici.

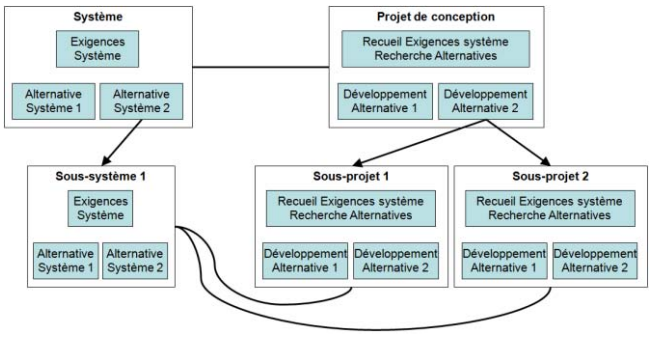


Figure 2.7 : Mise en relation d'une entité Système avec plusieurs entités Projet de conception sur un seul niveau de décomposition

2.1.4 Définition du couplage structurel

Compte tenu des caractéristiques des différents modes de couplages envisagés dans la section précédente, nous proposons, dans la suite de ce mémoire, de développer notre proposition en considérant une **bijection entre les entités de conception et de planification** telle qu'énoncée au premier point de la section 2.1.3. Ce choix nous permet donc de définir le **couplage structurel** comme le fait de mettre en relation une et une seule entité de conception avec une et une seule entité de planification, et inversement, en tenant compte du contenu informationnel de chaque entité. En conséquence, la création d'une entité système S implique la création d'une entité projet de conception P et réciproquement. Dans cette optique, nous proposons de limiter les liens entre entités de conception et de planification à des bijections entre :

- une entité système S et une entité projet de conception P,
- une entité exigences système ES et une entité tâche TE,
- une entité alternative système AS et une entité tâche TD.

Nous avons étudié dans la section 2.1.3 les liens pouvant exister entre une entité système S et une entité projet de conception P. Implicitement, la création d'une entité système S implique la création d'une entité exigences système ES et d'une entité alternative système AS. De même, la création d'une entité projet de conception P implique la création d'une entité tâche TE et d'une entité tâche TD. Nous proposons donc que :

- la création d'une entité système S implique la création d'une entité projet de conception P et réciproquement ainsi que l'association :
  - du système et du projet (1),
  - des exigences système ES et de la tâche TE (2),
  - de l'alternative système AS et de la tâche TD (3) ;
- l'ajout d'une entité alternative système AS implique l'ajout d'une entité tâche TD et réciproquement ainsi que l'association de celles-ci (4).

Ces liens sont représentés sur la figure 2.8.

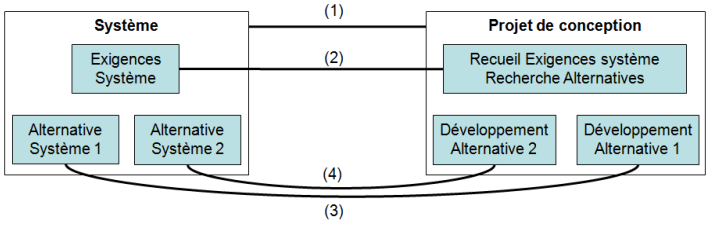


Figure 2.8 : Couplage proposé des entités de conception et de planification

2.2 Modèle de classe supportant le couplage structurel

Nous proposons un modèle de classe, supportant le couplage structurel et reprenant le principe de bijection présenté dans la section précédente. Cette structure comporte trois parties :

- la **représentation des entités de conception,**
- la **représentation des entités de planification,**
- le **couplage des entités et orchestration**.

2.2.1 Représentation des entités de conception

La définition de l'entité système développée dans la section 2.1.1.1 nous permet de proposer le modèle de classe illustré sur la figure 2.9. Un objet de la classe Système est composé d'un objet de la classe Exigences Système et d'un ou plusieurs objets de la classe Alternative Système. Un objet de la classe Exigences Système n'appartient qu'à un seul objet de la classe Système et il en est de même pour un objet de la classe Alternative Système. Dans ce modèle, systèmes et sous-systèmes sont modélisés par la même classe Système.

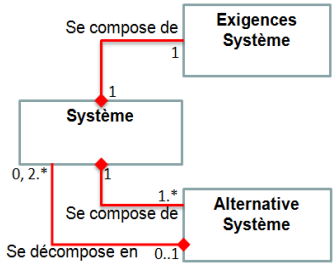


Figure 2.9 : Diagramme de classe représentant les entités de conception

Avant de détailler chacune de ces classes, nous abordons la notion de **variables de conception**. Ces dernières permettent de caractériser les exigences systèmes ainsi que les solutions potentielles décrites dans cette section. Une variable est décrite au moyen de son identifiant sous forme de texte (par exemple : « Masse »), de son unité (par exemple « Grammes ») et de son domaine de validité (par exemple : réels positifs). Au cours de la conception, ces variables sont associées à des valeurs qui peuvent être soit des valeurs discrètes (par exemple : Masse = 15g), soit des intervalles de valeurs (par exemple : Masse comprise dans l'intervalle [10g, 20g]), soit des valeurs symboliques (par exemple : Forme = U).

De manière plus détaillée, la classe **Exigences système** est composée de **besoins** et d'**exigences techniques**. Les **besoins** regroupent les attentes et contraintes des clients, des parties prenantes et/ou émanent de l'entité système supérieure si elle existe, exprimées qualitativement sous forme textuelle. Par exemple, *« le composant C doit être aussi léger que possible »* est un besoin. Les **exigences techniques** sont une transcription des besoins par les concepteurs sous forme de variables de conception sur lesquelles peuvent s'appliquer des contraintes. Les valeurs données à ces variables peuvent être soit des valeurs si l'exigence est stricte, soit des intervalles ou des ensembles de valeurs si l'exigence laisse une marge de manœuvre au concepteur. Par exemple, le besoin précédent peut être transcrit par l'exigence technique suivante : *« R1 : Masse de C < 20g ».*

Un objet de la classe **Alternative système** représente une solution satisfaisant les exigences système. Elle est composée d'une solution logique et d'une solution physique. La **solution logique** permet de décrire les principes de fonctionnement du système associé ainsi que son architecture. Une solution logique peut comporter des documents comme des textes, des dessins et/ou, des notes de calcul ayant conduits à cette solution.

Elle précise des concepts (par exemple, représentation schématique), des technologies à employer, certaines valeurs de variables comme des dimensions, des écarts, etc. Si le système est considéré comme suffisamment simple pour être conçu sans décomposition, la solution logique ne comporte pas de sous-système. Quand une décomposition est décidée par le responsable de conception, la solution logique se compose d'au moins deux sous-systèmes (voir figure 2.9, cardinalité 0, 2.* du lien de composition entre Alternative système et Système). Elle comprend alors les documents suivants :

- une liste des sous-systèmes,
- les attentes du responsable de conception pour chacun de ces sous-systèmes et,
- l'architecture des interfaces entre les sous-systèmes.

Le recueil des besoins et/ou exigences techniques de chaque sous-système aura pour entrée ces attentes. Le choix des principes de fonctionnement peut entraîner l'apparition de variables et/ou de contraintes supplémentaires par rapport aux exigences système initiales. Par exemple, suite à un calcul par éléments finis, l'exemple précédent est réduit à *« R1 : Masse de C comprise dans l'intervalle [10g, 18g] »* et l'exigence *« R2 : Module d'Young du matériau de C > 200000MPa »* est créée.

La **solution physique** permet la description des composants et/ou interfaces physiques utilisés dans l'alternative. Elle est définie par :

- une liste de couples (variable, valeur) décrivant la solution, et
- une liste de composants pouvant être fabriqués, approvisionnés ou sous-traités.

Une solution physique peut également comporter des documents comme du texte, des dessins et/ou, des notes de calcul ayant conduits à cette solution. Si le système est considéré comme suffisamment simple pour être conçu sans décomposition, la solution physique comprendra :

- un mode opératoire indiquant les procédés de fabrication et d'assemblage si nécessaire,
- une liste des fournitures nécessaires.

Quand une décomposition est décidée par le responsable de conception, la solution physique comprendra un mode opératoire correspondant à l'assemblage des sous-systèmes.

Les variables découlent des exigences système et de la solution logique retenue. Les valeurs données à ces variables peuvent être soit des singletons si la solution est certaine et totalement connue, soit des intervalles ou des ensembles de valeurs si la solution n'est pas complète, incertaine ou imprécise. Par exemple, le composant C peut avoir un poids compris entre 10 et 12 grammes en fonction des composants des sous-systèmes qui ne sont pas encore totalement définis : *« S1 : Poids de C compris entre [10g, 12g] ».*

2.2.2 Représentation des entités de planification

La définition de l'entité projet de conception développée dans la section 2.1.1.2 nous permet de proposer le modèle de classe illustré sur la figure 2.10. Un objet de la classe Projet de conception est composé d'un objet de la classe Tâche de recueil des exigences et recherche d'alternatives (tâche TE) et d'un ou plusieurs objets de la classe Tâches de développement d'alternative (tâche TD). Un objet de la classe tâche TE n'appartient qu'à un seul objet de la classe Projet de conception et il en est de même pour un objet de la classe tâche TD. Dans ce modèle, projets et sous-projets de conception sont modélisés par la même classe Projet de conception.

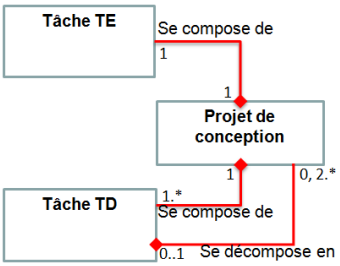


Figure 2.10 : Diagramme de classe représentant les entités de planification

Avant de détailler ces classes, soulignons que ces dernières sont des **tâches** et qu'en conséquent, elles sont caractérisées par des attributs spécifiques. Il s'agit de durée, date de début, date de fin, de type et quantité de ressources et divers indicateurs : coût, risques... Ces attributs sont, de manière identique aux variables de conception, définis soit par des valeurs discrètes (par exemple, *Nombre_semaines = 2*), soit par des intervalles de valeurs (par exemple, *Durée comprise dans l'intervalle [10 mois, 12 mois]*), soit des valeurs symboliques (par exemple, Ressource = M. Smith).

De manière plus détaillée, la **tâche de recueil des exigences et recherche d'alternatives, soit la tâche TE,** correspond au recueil des besoins et exigences techniques et à la recherche des alternatives de conception. Au cours du recueil des besoins et exigences techniques, le responsable de conception interagit avec le client et les autres parties prenantes pour définir leurs attentes, c'est-à-dire leurs besoins et/ou exigences techniques. Suivant le fonctionnement de l'entreprise, cette tâche pourra être accompagnée d'une recherche d'alternatives, c'est-à-dire d'une recherche des concepts correspondant à chaque alternative à étudier lors de la réalisation ultérieure des tâches TD.

La **tâche de développement d'alternative, soit la tâche TD,** a pour objet le pilotage du développement d'une solution. Elle est composée de trois tâches réalisées séquentiellement :

- une **tâche de conception du système** qui correspond à la réalisation d'une partie des documents qui vont être recensés dans la solution logique de l'alternative système (à l'exception de la liste des sous-systèmes et de la formalisation textuelle et qualitative des attentes du concepteur pour chacun de ces sous-systèmes). Cette tâche n'est pas un projet de conception puisqu'il ne s'agit pas, au cours de celle-ci, de recueillir des besoins et/ou exigences techniques. Au terme de cette tâche, la décision soit de décomposer l'alternative système (conception modulaire) en sous-systèmes, soit de ne pas la décomposer (conception intégrée) doit être prise ;
- une **tâche de conception des composants** qui correspond à la définition des informations relatives à la solution. Le livrable de cette tâche est différent si la décision de décomposer en sous-systèmes a été prise ou non lors de la conception de la solution logique :
    - si le système est considéré comme suffisamment simple pour être conçu sans décomposition, les documents nécessaires à la définition de la solution physique sont réalisés (voir section 2.1.2.1) ;
    - si le système nécessite une décomposition en sous-systèmes, la liste des sous-systèmes et les attentes du concepteur pour chacun de ces sous-systèmes sont produites. Les projets de conception associés aux sous-systèmes peuvent alors débuter (voir figure 2.10, cardinalité 0, 2.* du lien de composition entre Tâche TD et Projet). À leur issue, les documents nécessaires à la définition de la solution physiques sont produits ;

- une **tâche d'intégration et de vérification** permettant, en cas de décomposition, de réaliser l'intégration dans la solution physique des sous-systèmes et, dans tous les cas, de s'assurer que l'alternative système satisfait l'ensemble des exigences système ;
- une **tâche de production** qui, dans le cas où le projet inclut la réalisation physique de l'alternative, permet la réalisation physique du système.

2.2.3 Couplage des entités et orchestrateur

Le couplage des entités est assuré par une entité spécifique nommée entité Couplage. Le rôle de l'entité Couplage est d'assurer la cohérence du projet global de développement d'un système en permettant de répercuter les décisions prises d'un environnement à un autre. Cette classe orchestre donc l'échange de données entre les environnements de conception de système et celui de planification de projet de conception. Pour cela, l'orchestrateur ainsi défini réalise deux fonctions :

- assurer une bijection entre les entités de conception et de planification : chaque système doit être associé à un projet de conception, chaque alternative de système à une alternative de projet de conception, et inversement. L'orchestrateur doit donc associer :
  - un identifiant système (IdS) avec un identifiant projet de conception (IdP),
  - un identifiant d'exigence système ES (IdES) avec un identifiant de tâche TE (IdTE) et,
  - un identifiant d'alternative système AS (IdAS) avec un identifiant de tâche TD (IdTD) ;
- assurer la cohérence des bijections : l'orchestrateur doit affilier un identifiant unique à chaque entité, réaliser l'appariement des entités et retrouver le pendant d'une entité.

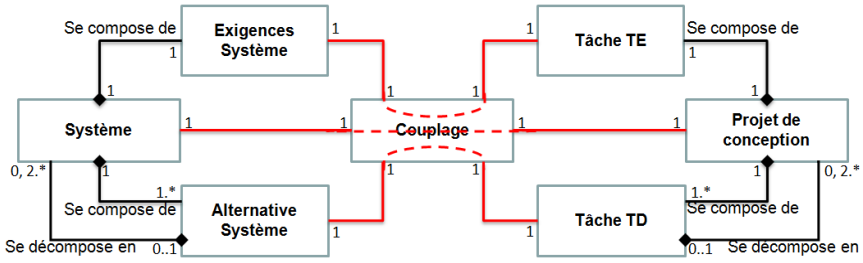


Figure 2.11 : Diagramme de classe intégrant le couplage des entités

Un objet de la classe Couplage est associé à :

- un couple composé d'un seul objet de la classe Système S et d'un seul objet de la classe Projet de conception P ou bien à,
- un couple composé d'un seul objet de la classe Exigences Système ES et d'un seul objet de la classe Tâche TE ou encore à,
- un couple composé d'un seul objet de la classe Alternative Système AS et d'un seul objet de la classe Tâche TD.

Chacun de ces objets n'est lié qu'à un seul objet de la classe Couplage comme illustré dans la figure 2.11.

2.3 Processus de mise en œuvre du couplage structurel

Nous proposons dans cette section de décrire le processus intégré de conception et de planification permettant la mise en œuvre du couplage structurel. Dans un premier temps, les acteurs

du processus sont décrits. Puis, le processus est détaillé dans la section 2.3.2 et deux sous-processus sont précisés dans les sections 2.3.3 et 2.3.4.

2.3.1 Rôles des acteurs

Avant de détailler ces processus, nous proposons de définir trois acteurs dont les rôles sont les suivants (voir figure 2.12) :

- le **responsable de planification** a la responsabilité des entités de planification pour un niveau de décomposition donné. Il est responsable de l'équipe de planification. Son rôle est de définir un projet de conception P correspondant à son niveau, c'est-à-dire de définir les tâches TE et TD (dates, délais, coûts, etc.) correspondant au système à concevoir et de proposer des plannings pour leur réalisation ;
- le **responsable de conception** a la responsabilité des entités de conception pour un niveau de décomposition donné. Il est également responsable de l'équipe de conception. Son rôle est de lancer et gérer la réalisation des tâches TE et TD selon les plannings fournis par le responsable de planification. Pour y parvenir, il est en lien étroit avec le responsable de planification afin de définir puis réaliser un projet correspondant au système qu'il a à concevoir en tenant compte au plus tôt des orientations reçues du directeur de programme et des contraintes inhérentes au projet et au système ;
- le **directeur de programme** nomme les responsables de planification et de conception, réalise les arbitrages si nécessaire et définit les orientations du projet et du système à concevoir dans ce projet. Eventuellement, il peut être en relation avec le client duquel il obtient les besoins que le futur système doit satisfaire. Dans certains cas, il transmettra directement les exigences techniques à son responsable de conception.

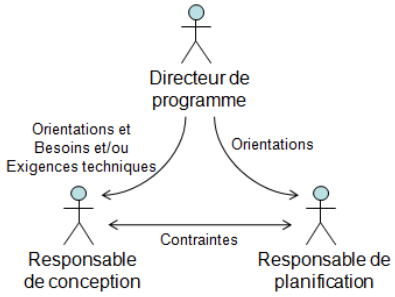


Figure 2.12 : Interactions entre les acteurs de la conception

2.3.2 Processus de création et de renseignement des entités système et projet de conception

Nous proposons dans cette section de décrire le processus de création et de renseignement des entités système et projet de conception mettant en œuvre les principes de couplage structurel décrits dans les sections précédentes. Le processus général, nommé « Processus intégré de planification / conception » et représenté grâce au le formalisme BPMN (*Business Process Modeling Notation*), est illustré sur la figure 2.13. Il fait référence à deux sous-processus représentés sur les figures 2.14 et 2.15. Les principes du formalisme BPMN sont représentés dans l'annexe 2. Ce processus est instancié de manière propre à un niveau de décomposition. Il est récursif, ce qui signifie qu'une décomposition entraîne de nouvelles instances à un niveau inférieur.

Les participants : le processus général implique quatre « participants » (chaque participant est représenté dans un « bassin » particulier). Trois d'entre eux sont les acteurs décrits dans la section 2.3.1, le dernier est l'environnement support des entités, c'est-à-dire la plateforme logicielle. Cet

environnement doit permettre la création des entités (en garantissant qu'elles sont correctement couplées entre elles) ainsi que leur renseignement, c'est-à-dire la saisie des informations inhérentes à ces entités. Nous considérons que cet environnement intègre trois sous-parties : la conception, la planification et le couplage. La participation de chacune dans le processus est représentée par une « ligne » dans le bassin correspondant. La communication entre ces quatre participants est représentée par des flèches pointillées allant d'un bassin vers un autre. Les tâches du processus inhérentes à l'environnement intégré sont des tâches automatisées alors que les autres sont manuelles, assistées par l'environnement intégré.

Principe général du processus : le directeur de programme initialise le processus et, ensuite, les responsables de planification et de conception réalisent leurs tâches. L'un ou l'autre peut alors prendre ses décisions d'exploration de nouvelles alternatives et/ou de décomposition en entités plus détaillées. Le processus doit alors prendre en considération les décisions prises par l'un afin que l'autre réalise ses tâches en conséquence, c'est-à-dire crée et renseigne ses entités en respectant les principes du couplage structurel. La communication entre responsable de conception et responsable de planification se fait nécessairement au travers de l'environnement intégré (même si des discussions informelles peuvent avoir lieu dans le but de faciliter les prises de décisions).

Synchronisation des tâches : l'analyse des tâches de planification et de conception nous permet de recenser trois possibilités de séquence :

- **le système est renseigné avant le projet de conception.** Cette séquence de tâches suppose qu'il n'y a pas de pilotage de la conception lors du renseignement du système, le projet de conception n'étant pas défini. Cela signifie que le système est conçu avant de définir le projet de conception et donc sans connaissance des objectifs et contraintes liées au projet (coût, délais, ressources, etc.) ;
- **le projet de conception est entièrement renseigné avant le système.** Ici, un pilotage de projet est mis en place avant le début effectif de la conception : un planning complet est établi en tenant compte des contraintes de coût et de ressources. Le responsable de conception dispose donc d'un *cadre* avant de débuter le recueil des exigences systèmes puis le développement des alternatives. En revanche, le planning est établi sans connaissance des données techniques concernant le système à concevoir (exigences, ressources particulières, délais, etc.) ;
- **le système et le projet de conception sont renseignés simultanément.** Cette possibilité permet de recueillir des exigences système avant de choisir les alternatives à étudier et donc, juste avant leur planification. Cela permet ainsi de définir et planifier le projet en plusieurs étapes : d'abord, la tâche de recueil des exigences système TE et ensuite, une fois celles-ci récoltées, les tâches de développement d'alternative TD en ayant déjà une vision assez détaillée des besoins du client ES et des solutions potentielles à explorer AS. Pour cela, une synchronisation des deux tâches est nécessaire. En effet, la planification devant permettre de piloter ensuite les tâches de développement d'alternative, les tâches de renseignement, puis de planification de ces dernières doivent nécessairement avoir lieu avant leur réalisation par la conception.

Parmi ces trois possibilités, celle permettant d'obtenir un couplage le plus étroit possible entre conception et planification est clairement la troisième. Dans cette situation, chaque tâche TE ou TD est cadrée par un planning établi préalablement en ayant un niveau de connaissance maximum sur le problème à résoudre. Le processus proposé concerne donc cette troisième possibilité.

Description du processus : le directeur de programme initialise le projet général de conception. Tout d'abord, les responsables de conception et de planification sont désignés et les orientations du projet sont fixées. Les orientations concernent à la fois la conception et la planification. Il peut s'agir du délai de réalisation du projet, du budget alloué, de recommandations sur les ressources et la manière de les utiliser, les technologies à privilégier, les composants à utiliser (ou réutiliser), etc. Elles se déclinent en objectifs (par exemple, *minimiser le coût du système*), en contraintes (*le projet doit être terminé dans 18 mois*) et en critères (par exemple, *privilégier les sous-traitants européens*).

L'environnement de couplage doit alors créer trois couples d'identifiants (message m10) :

- un couple (IdS, IdP) représentant l'association de l'identifiant système IdS et de l'identifiant IdP du projet de conception de ce système ;
- un couple (IdES, IdTE) représentant l'association de l'identifiant IdES d'une entité Exigences système ES avec l'identifiant IdTE d'une entité Tâche TE ;
- un couple ($IdAS_1$, $IdTD_1$) représentant l'association de l'identifiant $IdAS_1$ d'une alternative système AS_1 et de l'identifiant $IdTD_1$ de la tâche TD_1 associée.

Ensuite, le processus se sépare en deux parties distinctes devant être réalisées en parallèle :

- l'environnement de planification crée un projet vierge (un objet Projet de conception d'identifiant IdP, un objet Tâche TE d'identifiant IdTE et un objet Tâche TD_1 d'identifiant Id TD_1) et transmet une demande (m20) au responsable de planification de saisie des informations de la tâche TE et de planifier cette tâche ;
- l'environnement de conception crée un système vierge (un objet Système d'identifiant IdS, un objet Exigences Système d'identifiant IdES et un objet Alternative Système AS_1 d'identifiant $IdAS_1$) et transmet une demande de réalisation de la tâche TE de recueil des exigences au responsable de conception (m30). Ce dernier ne pourra débuter son activité qu'après avoir reçu son planning.

Nous allons décrire parallèlement ces deux parties du processus : celle concernant la planification et celle concernant la conception. Dans la mesure où il faut nécessairement que le responsable de conception ait un planning de travail, il doit attendre la réception du planning de la tâche TE réalisé par le responsable de planification pour réaliser cette tâche (messages m40). Ainsi, le responsable de planification doit saisir les informations sur le projet de conception et sur la tâche TE, c'est-à-dire définir les durées des tâches, les ressources allouées et planifier le projet avec sa tâche TE. La saisie des informations sur la tâche TD_1 et sa planification n'interviennent que plus tard dans le processus lorsque le nombre d'alternatives à étudier est connu. Bien que non représenté sur le processus de la figure 2.13, la tâche de planification peut faire appel à un outil d'aide à la planification intégré dans l'environnement de planification.

Le planning de la tâche TE étant réalisé, un message (m40) est envoyé au responsable de conception (par l'intermédiaire de l'environnement intégré). Dès réception de ce message, le responsable de conception peut lancer l'exécution de la tâche TE en fonction du planning reçu. Afin de réaliser le suivi de la tâche TE par le responsable de planification, un message est envoyé au responsable de planification via l'environnement intégré (message m50). Son contenu concerne les informations de suivi (dates de début et de fin effectives, ressources effectivement utilisées) ainsi que la liste des exigences techniques capitalisées lors du déroulement de la tâche TE. Dès réception du message m50, le responsable de planification réalise sa tâche de suivi de la tâche TE et transmet les informations au directeur de programme (message m55) pour qu'il réalise le suivi global du projet (remarque : bien que non représenté sur le processus de la figure 2.13, cet échange entre responsable de planification et directeur de programme se fait également au travers de l'environnement intégré).

A ce stade, le responsable de planification ou le responsable de conception peut décider d'étudier plusieurs alternatives ou bien, au contraire, de n'en étudier qu'une seule (celle prévue initialement). Une concertation informelle entre responsables doit avoir lieu afin qu'il n'y ait qu'un seul des deux qui prenne la décision d'explorer un nombre m de nouvelles alternatives. Ainsi, quatre possibilités sont proposées sur le processus représenté sur la figure 2.13 :

- le responsable de planification prend la décision de n'explorer qu'une seule alternative. Si le responsable de conception a pris la même décision, le responsable de planification peut alors commencer la saisie puis la planification de la tâche TD_1 au sein du projet ;
- le responsable de conception prend la décision de n'explorer qu'une seule alternative : si le responsable de planification a pris la même décision, le responsable de conception doit attendre de recevoir le planning de la tâche TD_1 avant de lancer le travail de conception (message m100) ;
- le responsable de planification prend la décision d'explorer m alternatives supplémentaires : une demande (message m60) est envoyé à l'environnement de couplage de créer m nouvelles tâches TD_i ($i=\{2, 3, \ldots, m+1\}$) et le responsable attend de recevoir de l'environnement les m tâches TD_i vierges afin d'en saisir les informations et de les planifier. Le responsable de conception doit attendre de recevoir un message de l'environnement de couplage l'avertissant de la création de m alternatives système vierges AS_i ($i=\{2, 3, \ldots, m+1\}$) (message m90) ;
- le responsable de conception prend la décision d'explorer m alternatives supplémentaires : un message (m70) est envoyé à l'environnement de couplage de créer m nouvelles alternatives vierges et le responsable de conception attend ensuite de recevoir les plannings des m+1 taches TD avant de lancer le travail de conception de chacune d'elles (m100). Le responsable de planification doit attendre de recevoir un message de l'environnement de couplage (m80) contenant les m tâches TD_i ($i=\{2, 3, \ldots, m+1\}$) vierges afin de saisir leurs information respectives et de les planifier.

Du point de vue de l'environnement de couplage, la réception des messages m60 ou (exclusif) m70 mène à la création de m nouveaux couples ($IdAS_i$, $IdTD_i$). $IdAS_i$ et $IdTD_i$ représentent respectivement l'identifiant d'une nouvelle alternative système AS_i (liée au système d'identifiant IdS) et l'identifiant d'une nouvelle tâche de développement d'alternative TD_i (liée au projet d'identifiant IdP).

Pour le responsable de planification, la suite consiste à saisir les informations liées aux m+1 tâches TD et à les planifier (avec éventuellement m = 0). La tâche correspondant à cette activité pour une tâche TD particulière correspond au sous-processus illustré sur la figure 2.14. Ce processus, intitulé « **Processus de saisie des informations / Planification tâche TD** » est réitéré m+1 fois. Il est décrit dans la section 2.3.3. Lorsque les m+1 tâches TD sont renseignées et planifiées, le responsable de planification envoie un message (m100) au responsable de conception au travers de l'environnement intégré. Le contenu du message concerne les m+1 plannings des m+1 tâches TD.

Dès réception de ce message, le responsable de conception peut lancer le développement des m+1 alternatives système AS_i en parallèle selon les plannings qu'il a reçu. Le développement d'une alternative est décrit par le sous-processus intitulé « **Processus de réalisation tâche TD** » et illustré sur la figure 2.15. Il est décrit dans la section 2.3.4. Le responsable de conception envoie des

informations de suivi sur la réalisation des tâches TD au responsable de planification (message m110). Celui-ci en réalise le suivi et communique ses informations au directeur de programme (m115).

Une fois que les m+1 tâches TD sont terminées, que le responsable de planification a terminé leur suivi de même que le directeur de programme, ce dernier peut clore le projet. Le processus de conception du système avec ses m+1 alternatives est donc terminé.

Nous pouvons remarquer que ce processus est dédié à la conception d'un système à un niveau particulier. Il ne fait aucune référence à la possibilité de décomposer les entités (projets en sous-projets, systèmes en sous-systèmes). Les sous-processus de saisie des informations et de réalisation des tâches TD sont décrits dans les sections suivantes et abordent la possibilité de décomposition.

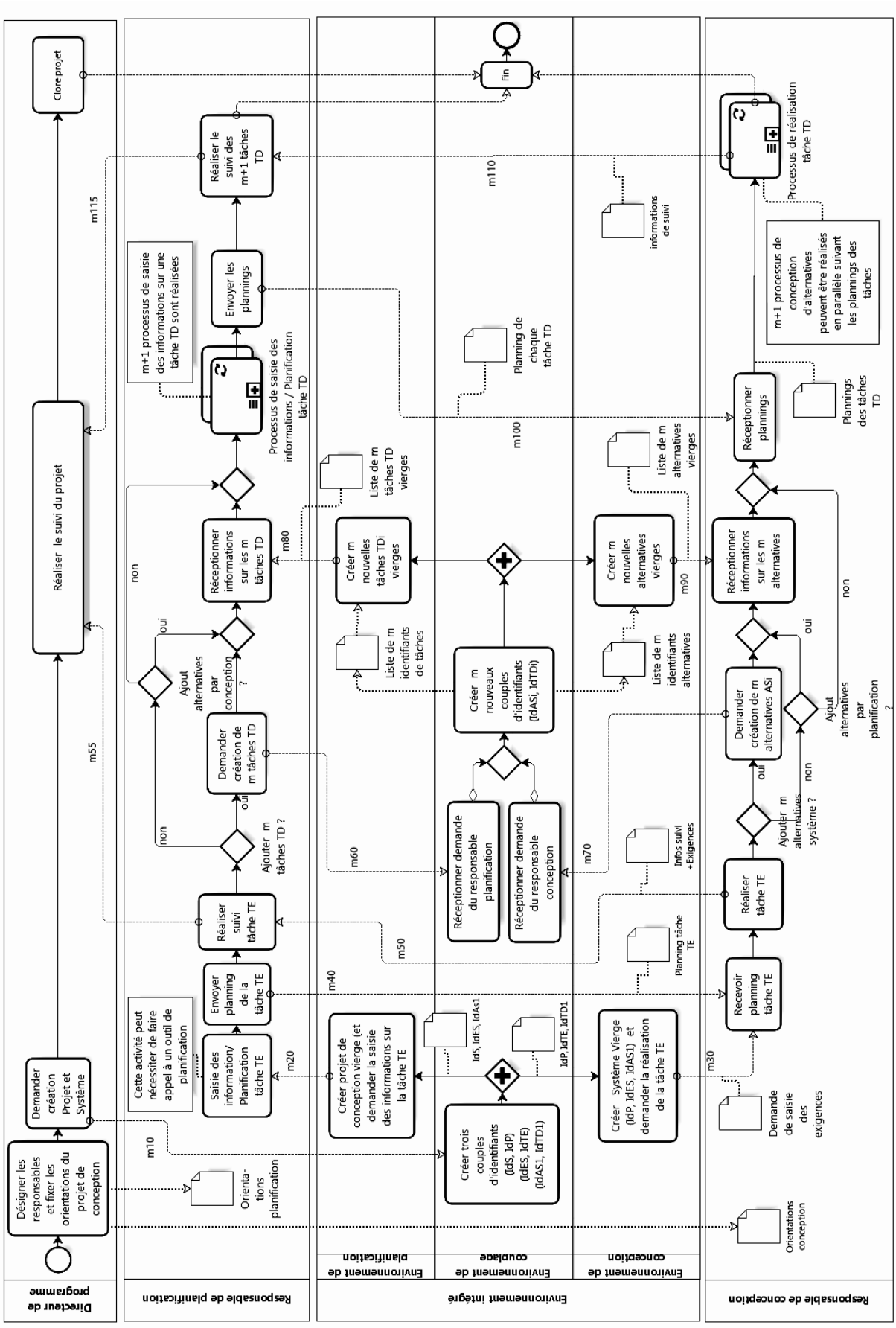


Figure 2.13 : Processus intégré de planification et de conception

2.3.3 Processus de saisie des informations et de planification des tâche TD

Le processus général décrit dans la section 2.3.2 fait référence au sous-processus intitulé « Processus de saisie des informations / Planification tâche TD ». Ce sous-processus est décrit dans cette section et illustré sur la figure 2.14 ci-dessous.

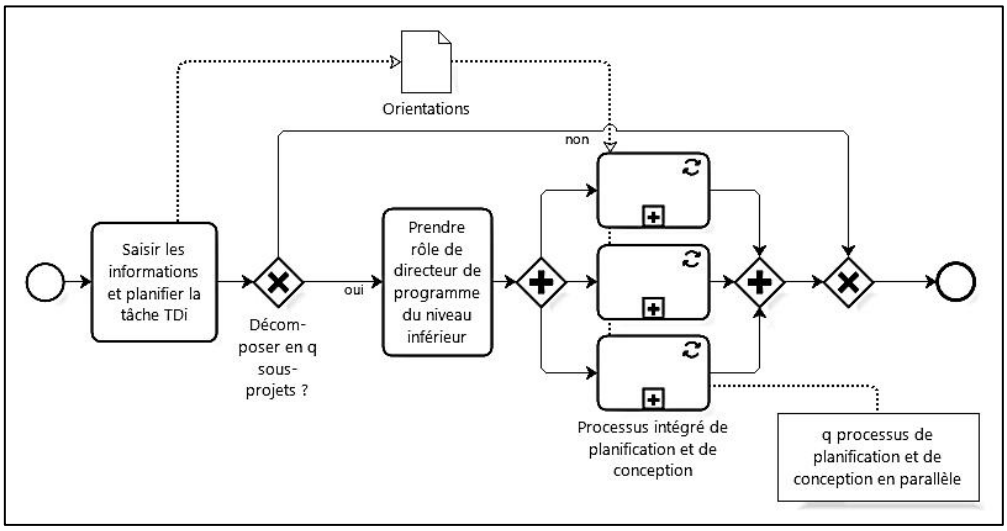


Figure 2.14 : Processus de saisie des informations / Planification tâche TD

La première tâche de ce sous-processus, réalisée par un responsable de planification de niveau n, consiste à effectuer la saisie des informations d'une tâche TD. Les dates, durées et ressources sont donc renseignées et la tâche est planifiée. Comme pour la section précédente, cette tâche de planification peut nécessiter l'utilisation d'un outil de planification, cette dernière n'est cependant pas représentée ici.

Le responsable de planification peut prendre la décision de décomposer son projet en un nombre q de sous-projets s'il juge la complexité trop importante. Cette décision doit être le fruit d'une concertation avec le responsable de conception car un seul des deux doit mettre en œuvre la décision et engager le processus permettant la création des entités couplées correspondantes.

Si la décision est prise de ne pas décomposer, ce sous-processus est terminé et le processus principal reprend son cours. Si la décision est prise de décomposer en q sous-projets, le directeur de programme du niveau inférieur doit être désigné. Il peut s'agir d'un nouvel acteur ou de l'un ou l'autre des responsables de conception et de planification. Dans notre proposition, nous considérons cette seconde alternative afin de simplifier les processus. En effet, si le directeur de programme du niveau inférieur est un autre acteur que l'un des responsables de conception ou de planification, le processus devra préciser les échanges entre ce nouvel acteur et le niveau supérieur. Ainsi, le responsable de planification, qui est à l'initiative de la décomposition, prend le rôle de directeur de programme de l'ensemble des q sous-projets du niveau inférieur. De manière récursive, le processus principal « Processus intégré de planification et de conception » est remis en œuvre q fois en parallèle avec, à chaque fois, un nouveau directeur de programme de niveau n-1 qui est le responsable de planification du niveau n (Remarque : sur la figure 2.14, seuls trois processus sont représentés en parallèle).

Une fois les q sous-projets renseignés, planifiés et exécutés, les q sous-processus sont terminés et le processus principal du niveau n reprend son cours (rappel : le message m100 sera envoyé au responsable de conception afin de lancer le travail de conception du niveau n).

2.3.4 Processus de réalisation d'une tâche TD

Le processus général décrit dans la section 2.3.2 fait référence au sous-processus intitulé « Processus de réalisation tâche TD ». Ce sous-processus est décrit dans cette section et illustré sur la figure 2.15 ci-dessous.

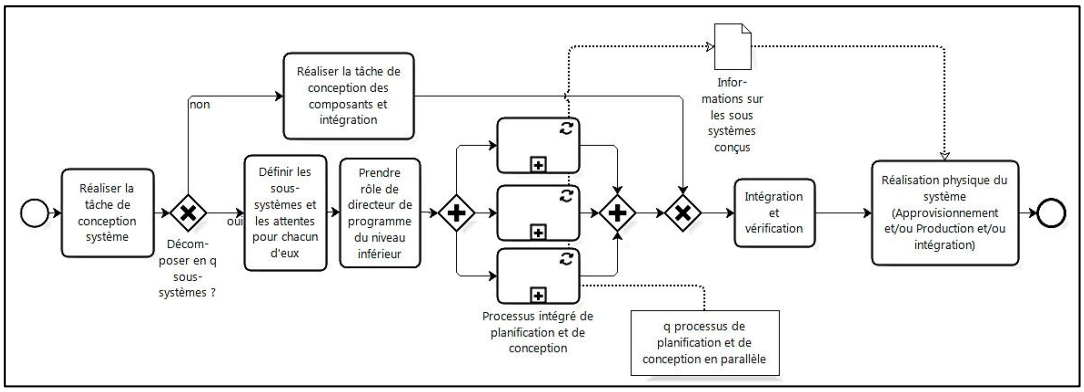


Figure 2.15 : Processus de réalisation d'une tâche TD

La première tâche consiste à réaliser la conception du système, c'est-à-dire à effectuer la saisie, par l'équipe de concepteurs, d'une partie des informations sur la solution logique. Le concept qui sera développé, les principes de fonctionnements, des modèles théoriques ou des calculs seront formalisés. La décision de décomposer en sous-système doit être prise en concertation, ici également, avec le responsable de planification afin de n'avoir qu'une seule mise en œuvre du processus de décomposition pour ce niveau.

Si la décision est prise de ne pas décomposer en sous-systèmes, la tâche suivante est celle de conception des composants nécessaires ainsi que de la spécification de leur intégration. Cette tâche de développement des composants peut s'avérer très simple si ce sont des composants existants approvisionnés et intégrés ainsi. En revanche, si au sein du projet il est envisagé de les produire en interne, une tâche de production est nécessaire et le procédé de fabrication doit alors être développé.

Si au contraire une décision de décomposition en q sous-systèmes a été prise, il est nécessaire de définir quels sont ces sous-systèmes et de préciser quelles sont les attentes vis-à-vis de leur conception. Ensuite, un directeur de programme pour le niveau inférieur doit être désigné (cf. section 2.3.3.). Ainsi, le responsable de conception, qui est ici à l'initiative de la décomposition, prend le rôle de directeur de programme pour les q sous-systèmes. De manière récursive, le processus principal « Processus intégré de planification et de conception » est remis en œuvre q fois en parallèle avec, à chaque fois, un nouveau directeur de programme de niveau inférieur n-1, rôle pris par le responsable de conception du niveau n. Ainsi, chaque sous-système sera l'objet d'un sous-projet propre sous la responsabilité générale du responsable de conception qui pourra définir les orientations de chacun d'entre eux.

Une fois les q sous-systèmes conçus (ou les composants développés s'il n'y a pas décomposition), nous proposons une tâche d'intégration et de vérification puis une tâche optionnelle permettant la « réalisation physique » du système, c'est-à-dire :

- l'approvisionnement des composants et/ou,
- la fabrication des composants et/ou,
- l'intégration physique des sous-systèmes qui ont été physiquement réalisés au niveau inférieur.

Nous considérons cette tâche optionnelle dans la mesure où la conception peut s'arrêter lorsque les spécifications de la solution correspondant à l'alternative existent. Une fois ces composants ou ces q sous-systèmes conçus, éventuellement approvisionnés et/ou fabriqués puis intégrés, la tâche TD est terminée. Le sous-processus de « réalisation d'une tâche TD » est terminé et le processus général de la figure 2.13 reprend son cours.

2.4 Illustration du couplage structurel

Cette section illustre l'application des processus proposés dans la section 2.3 sur la conception d'un avion d'affaire en se focalisant sur les tâches menées par chaque acteur. Le scénario illustré se décompose en trois parties : la création d'un système S et du projet de conception P associé, l'ajout d'une alternative système AS et de la tâche de développement TD associée et enfin, la décomposition d'une alternative système AS en sous-systèmes et de la tâche TD associée en sous-projets. Le scénario comprend également le renseignement des entités.

2.4.1 Création des entités système et projet de conception

La décision d'initialiser la conception d'un avion d'affaire (AL001) est prise par M. DiProg qui en devient le directeur de programme. Il désigne alors un responsable de planification (M. RespP) et un responsable de conception (M. RespC) et mentionne les orientations qu'ils vont devoir respecter (un exemple d'orientations est proposé sur la figure 2.16). Ces deux derniers sont informés de leur nomination ainsi que des orientations.

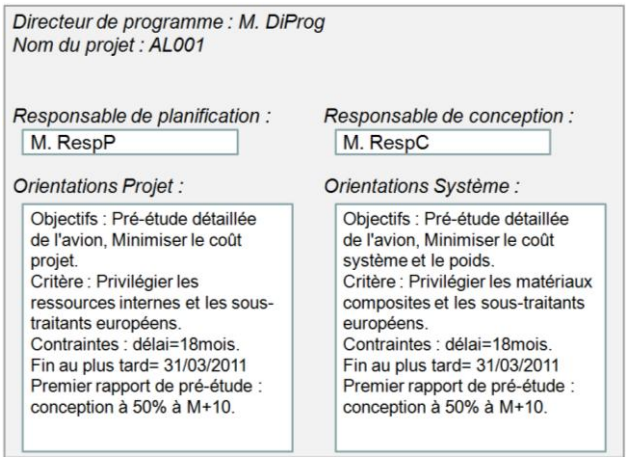


Figure 2.16 : Initialisation de la conception d'un avion AL001

Un projet de conception vierge (incluant une tâche TE et une tâche TD1) est créé ainsi que, parallèlement, un système vierge (incluant une entité ES et une entité AS1). M. RespP, le responsable de planification, renseigne alors le projet de conception AL001 ainsi que la tâche TE, en définissant la durée et les ressources allouées et planifie la tâche TE au sein du projet. Dans la partie supérieure de la figure 2.17, le projet et la tâche TE sont représentés grâce aux attributs spécifiques des tâches. Le planning de la tâche TE est envoyé à M. RespC.

M. RespC, le responsable de conception, en suivant son planning, renseigne les exigences système (partie inférieure de la figure 2.17). Dans un premier temps, il définit les besoins puis les transcrit en exigences techniques grâce aux variables de conception. Une fois ce travail effectué, MM. RespC et RespP envisagent ensemble le nombre d'alternatives supplémentaires à explorer au regard des exigences à satisfaire. Ici, ils considèrent qu'une seule alternative est suffisante. Il est donc inutile de créer de nouvelles entités.

M. RespP saisit les informations de la tâche TD1 et termine la planification du projet AL001 en définissant dates, durées et ressources affectées à la tâche TD1. Dès réception de son planning, l'équipe de concepteurs commence son travail de conception dans lequel les solutions logique et physique pour l'alternative système AS1 sont définies (voir figure 2.17). Une fois ce travail réalisé, la conception est terminée et le projet est clos par le directeur de programme.

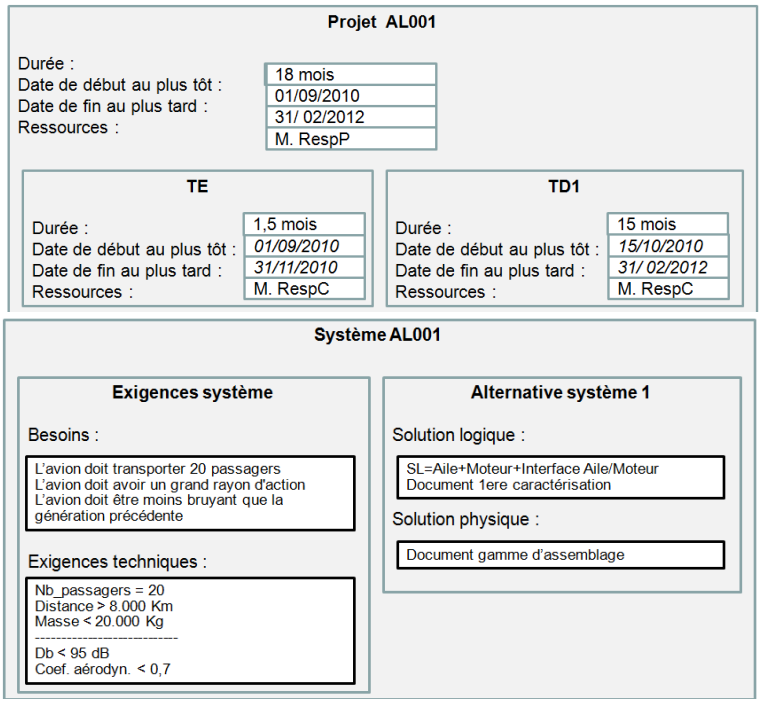


Figure 2.17 : Entité Projet de conception et entité Système de l'avion AL001 renseignées

2.4.2 Processus d'ajout d'une nouvelle alternative système et d'une nouvelle tâche TD

Nous poursuivons le scénario en considérant que MM. RespC et RespP, lors de leur concertation pour décider du nombre d'alternatives supplémentaires à étudier choisissent d'en explorer une seconde. Il est décidé également que ce sera M. RespC qui lancera le processus de création des entités nécessaires.

M. RespC demande la création d'une seconde alternative système AS2 associée à une seconde tache TD2. Une fois ces deux entités vierges créées, M. RespP effectue la saisie des informations des tâches TD1 et TD2 et les planifie au sein du projet AL001. Les informations saisies sont représentées sur la figure 2.18. M. RespP transmet les deux plannings de tâches à M. RespC qui lance le travail de conception des deux alternatives en suivant les plannings reçus.

Dans cet exemple, la seconde alternative correspond à une solution existante, déjà utilisée par le passé : une solution achetée sur catalogue à un fournisseur. Le travail de conception se résume à la recherche d'un système existant pouvant remplir les exigences système ES identifiées et à sa réutilisation (cf. section 5).

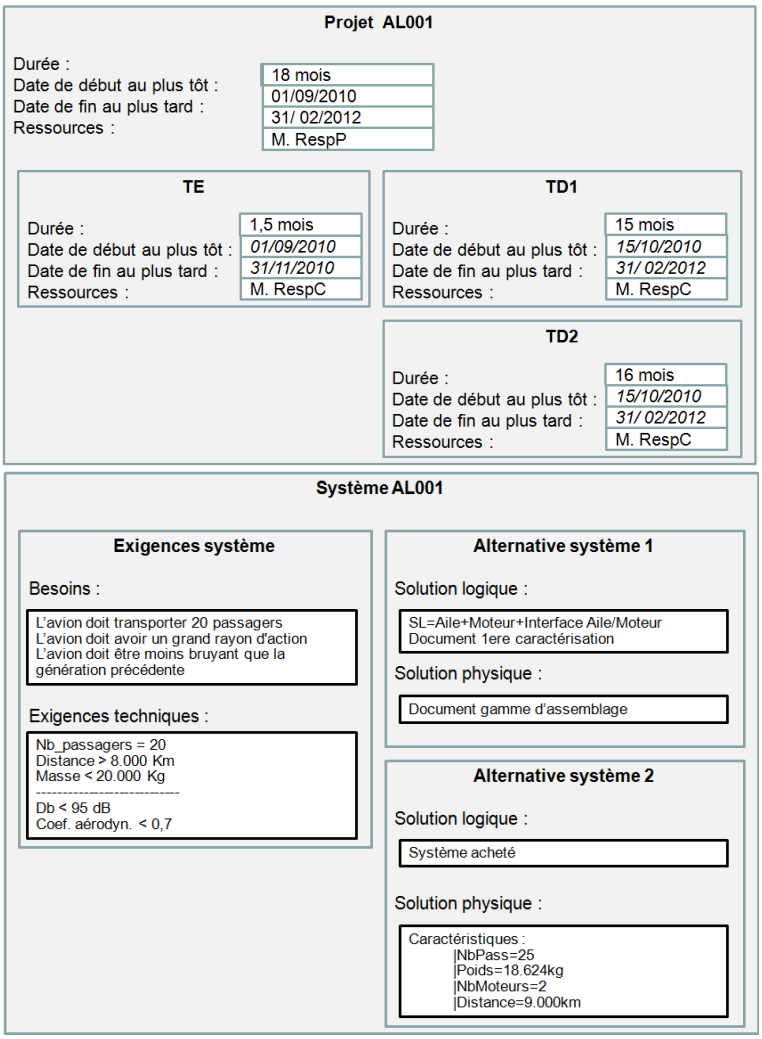


Figure 2.18 : Entité Projet de conception et entité Système de l'avion AL001 avec deux alternatives de conception

2.4.3 Processus de décomposition

Nous considérons dans la suite de ce scénario qu'une alternative système AS1 est trop complexe pour être conçue sur un seul niveau sans partitionner le problème de conception (voir figure 2.19). Le responsable de conception, M. RespC, décide donc de la décomposer en plusieurs sous-systèmes.

Le scénario est donc repris jusqu'à l'étape où l'équipe de concepteurs réalise le travail de définition des solutions logique et physique. Après que l'équipe ait travaillé sur la solution logique, le responsable de conception, M. RespC, informe le responsable de planification, M. RespP, de son intention de décomposer son système en deux sous-systèmes, l'un pour la structure, l'autre pour la propulsion. M. RespP analyse les contraintes liées au projet (durée et disponibilité des ressources). Une discussion entre le responsable de planification et le responsable de conception, sous le contrôle du directeur de programme en cas de désaccord, est nécessaire pour prendre la décision de décomposition. Pour notre scénario, M. RespC prend le rôle de directeur de programme pour chacun des deux sous-systèmes et le processus de la figure 2.13 (cf. section 2.3.2.) est mis en œuvre deux fois :

- M. RespC désigne les responsables de conception (M. RespC1 et M. RespC2) pour chacun des sous-systèmes et les responsables de planification (M. RespP1 et M. RespP2) pour chaque sous-projet ;
- il fixe les orientations pour chaque sous-projet et sous-système ;
- il demande la création des sous-systèmes et des sous-projets de conception.

Pour le processus de conception du sous-système « Structure », M. RespP1, le responsable de planification, définit le sous-projet et la tâche TE1 (dates, durées), planifie la tâche TE1 au sein du sous-projet et envoie son planning à M. RespC1, le responsable de conception. Celui-ci débute la tâche TE1 de recueil des exigences système. Ces exigences sont obtenues de M. RespC, responsable de conception au niveau supérieur. Ceci fait, la décision est prise de n'explorer qu'une seule alternative pour le sous-système « Structure » (voir figure 2.20). M. RespP1 définit et planifie la tâche TD11 de développement de l'alternative AS11 puis envoie son planning à M. RespC1 qui lance le travail de conception. Une fois ce travail terminé M. RespP1 finalise son travail de suivi et informe son directeur de programme M. RespC qui clos le sous-projet de conception de la structure. Le sous-projet de conception de la propulsion est mené de la même manière avec ses propres responsables sous la direction de M. RespC. Les informations relatives à ce sous-système et au sous-projet associé ne sont pas illustrées ici.

Une fois les deux sous-projets clos par M. RespC, les deux sous-systèmes étant conçus, le projet AL001 est repris. La définition de la solution physique est mise en œuvre à partir de toutes les informations et documents obtenus du niveau de conception inférieur. Le travail de définition de la solution physique consiste à intégrer les sous-systèmes « Structure » et « Propulsion ». Une fois l'intégration terminée, M. RespP, le responsable de planification termine sa tâche de suivi et en informe le directeur de programme M. DiProg qui peut clore le projet AL001. La conception du système est terminée.

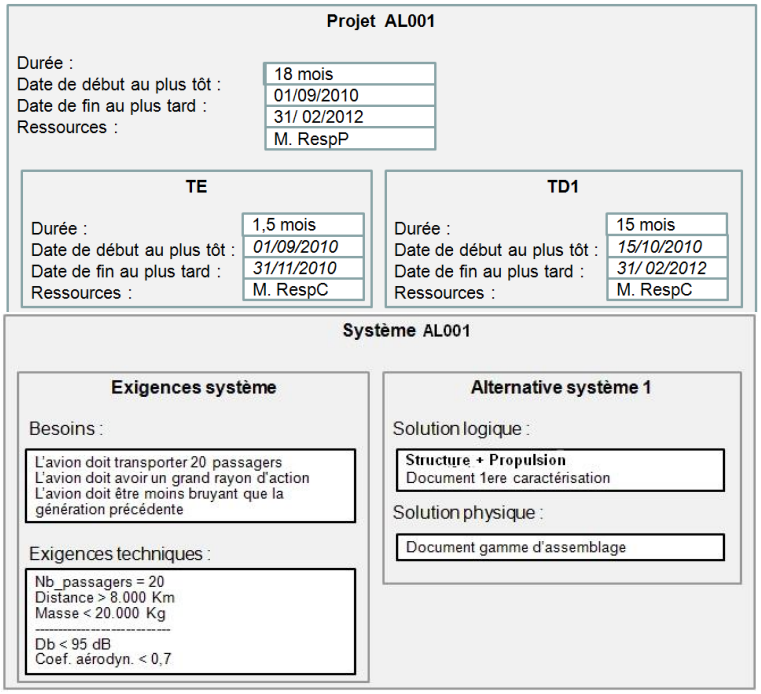


Figure 2.19 : Entité système et entité projet de conception de l'avion AL001

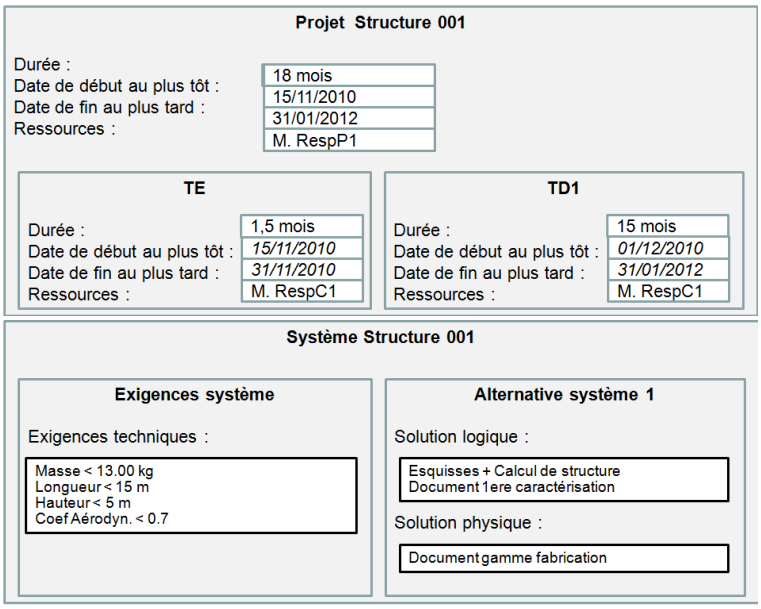


Figure 2.20 : Entité système et entité projet de conception de la structure 001

2.5 Conclusion sur le couplage structurel

Nous avons proposé dans cette section de définir le couplage structurel entre les entités de conception et les entités de planification d'un projet de conception. Dans un premier temps, la sémantique du couplage structurel a été définie afin d'envisager les différentes possibilités de couplage et déterminer les plus intéressantes qui sont retenues dans ce mémoire. Ensuite, un modèle de classes a été proposé afin de formaliser l'ensemble des entités nécessaires. Enfin, un processus générique de mise en œuvre du couplage structurel a été proposé puis illustré par un exemple de conception d'un avion d'affaire.

Les modèles proposés servent de support aux autres types de couplage que nous allons décrire dans les sections suivantes. En effet, nous avons vu que le couplage structurel s'attachait surtout à garantir que :

- les entités de conception et de planification étaient créées au bon moment et correctement associées,
- les informations inhérentes à ces entités étaient saisies par les bons acteurs aux instants les mieux adaptés en considérant que chaque responsable doit avoir le maximum d'informations pour prendre ses décisions de manière pertinente,
- les tâches de conception étaient correctement pilotées, encadrées par une structure projet adéquate fournissant des plannings de travail adaptés aux contraintes temporelles, aux ressources et aux objectifs à atteindre,
- une structure hiérarchique de systèmes et sous-systèmes, associés à leurs projets et sous-projets permettait d'appréhender les difficultés de conception en décomposant progressivement en entités de plus en plus simples.

Cependant, nous constatons qu'il est nécessaire de définir en outre d'autres formes de couplage afin de :

- garantir la cohérence des informations saisies dans les entités,
- détecter au plus tôt les non faisabilités afin d'être en mesure d'apporter des corrections et atteindre les objectifs au sein de la structure projet,

- offrir aux décideurs des outils d'aide à la décision de type tableaux de bord afin d'avoir un suivi efficace des tâches de conception.

L'ensemble de ces propositions a fait l'objet d'une communication dans le congrès Complex System Design and Management, CSDM 2010 [**Abeille et al., 2010**].

Ainsi, dans la suite de cette partie, nous proposons de décrire le couplage informationnel (chapitre 3) permettant d'assurer la cohérence de l'avancement des entités de conception et des entités de planification et le couplage décisionnel (chapitre 4) proposant des tableaux de bords adaptés permettant aux acteurs de prendre la ou les meilleures décisions en tenant compte des informations provenant des deux environnements.

Chapitre 3 : Couplage informationnel

3 Couplage informationnel

Dans cette section, nous présentons dans un premier temps les entités fondamentales du problème de couplage et définissons ce que nous entendons par couplage informationnel. Dans un deuxième temps, nous identifions les éléments à ajouter au modèle de classe établi dans la section 2.1.2 afin de supporter ce nouveau couplage. Dans un troisième temps, nous présentons les processus liés à ce nouveau couplage lors du déroulement du projet de conception. Enfin, nous illustrons sa mise en œuvre sur l'exemple concret de conception d'un avion d'affaire.

3.1 Entités fondamentales du couplage informationnel

Dans un premier temps, nous définissons deux notions sur lesquelles reposent le couplage informationnel : la faisabilité et la vérification. Ensuite, nous abordons ce que ces notions impliquent sur les entités proposées dans la section 2.2.3 et sur leur décomposition. Dans un troisième temps, nous proposons une définition du couplage informationnel.

3.1.1 Définitions générales de la faisabilité et de la vérification

3.1.1.1 *Faisabilité*

La **faisabilité** [**Froman et Gourdon, 2003**] est l'aptitude d'un produit, processus ou service étudié à être élaboré techniquement et dans des conditions économiques satisfaisantes.

Cette définition permet de porter un premier jugement sur la conception d'un système avant même son commencement. Il s'agit ici pour le responsable de conception et le responsable de planification de se prononcer sur la capacité de leur équipe à réaliser la conception d'un système ou d'un projet donné et des différentes alternatives de conception connaissant les besoins et exigences attendues par le client mais également l'ensemble des attentes et des contraintes émises par les autres parties prenantes. Il est en effet nécessaire d'évaluer cette faisabilité en amont afin de ne pas démarrer une conception pour laquelle on sait à l'avance qu'elle est vouée à l'échec.

Retenant cette définition, nous proposons que la faisabilité caractérise les entités de conception et celles de planification de manière descendante. Nous intégrons à chacune des entités E définies dans la section 2.1 un attribut de faisabilité. Cet attribut sera noté E.Fa dans la suite de ce mémoire.

Les états possibles pour l'attribut de faisabilité Fa d'une entité E quelconque sont :

- *non déterminé* noté « E.Fa = ND » état par défaut : cet état signifie que l'analyse de faisabilité de l'entité E n'a pas encore été réalisée ou n'est pas encore terminée ;
- *faisable* noté « E.Fa = OK » : cet état signifie que l'entité E a été analysée comme étant réalisable. Cet état peut éventuellement être remis en cause en cas d'évolutions des exigences par exemple ;
- *infaisable* noté « E.Fa = KO » : cet état signifie que l'entité E a été analysée comme étant irréalisable. Cet état peut cependant être réévalué par la suite en fonction de l'évolution des exigences.

Décrivons maintenant les évolutions possibles de cet attribut. Le diagramme états-transitions de la figure 3.1 et, plus particulièrement les états 1, 2 et 3, illustrent ces évolutions. À sa création, une entité E n'ayant pas encore vu sa faisabilité analysée, son attribut est fixé à *non déterminé* E.Fa = ND*,* ce qui correspond à l'état 1. Celle-ci peut ensuite passer dans l'état :

- *faisable* E.Fa = OK (état 2) si, suite à une analyse, l'entité peut être élaborée dans des conditions économiques et/ou techniques satisfaisantes (transition a) ;

- *infaisable* E.Fa = KO (état 3) dans le cas contraire (transition b).

Lorsque l'attribut de faisabilité d'une entité E est dans l'état *infaisable* E.Fa = KO, des négociations peuvent être menées entre protagonistes (client, responsable de conception, responsable de planification, directeur de programme, autres parties prenantes, etc.) afin de faire évoluer les exigences, les besoins ou encore les orientations du projet et permettre ainsi de lever l'infaisabilité [**Antón et Potts, 2003**] [**Gorschek et Davis, 2008**]. Dans ce cas, l'attribut de faisabilité de l'entité E repasse dans l'état *non déterminé* E.Fa = ND (transition c_1) afin d'être à nouveau analysé.

Dans de nombreux cas, certaines exigences ou besoins ne sont connus ou précisés qu'en cours de conception [**Gómez de Silva Garza et Maher, 1996**] [**Ramesh et al., 2010**]. Ainsi, lorsque l'attribut de faisabilité d'une entité E est dans l'état *faisable* E.Fa = OK, en cas d'évolution des exigences, des besoins ou des orientations de projet, il faut alors à nouveau analyser la faisabilité de l'entité concernée par rapport aux nouvelles exigences. L'attribut de faisabilité de l'entité doit retourner dans l'état *non déterminé* E.Fa = ND (transition c_2).

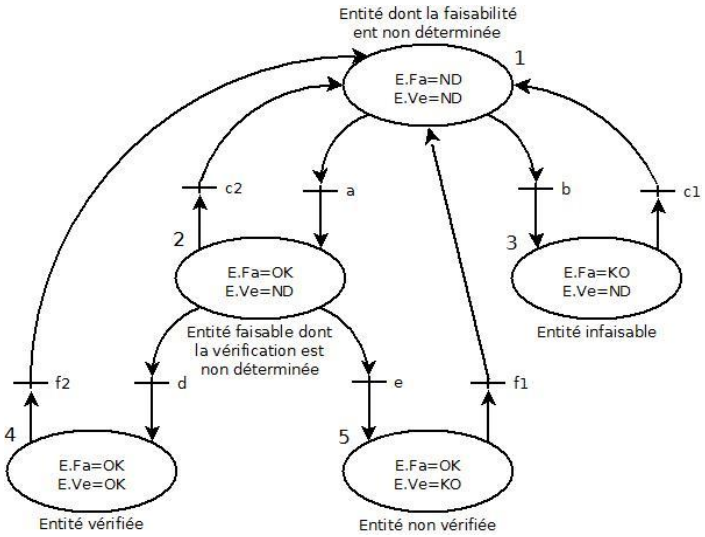


Figure 3.1 : Diagramme d'états-transitions des attributs de faisabilité et de vérification

3.1.1.2 *Vérification*

La **vérification** [**AFIS, 2004**] est la confirmation par des preuves tangibles que les exigences spécifiées ont été satisfaites.

Un système conçu doit nécessairement être analysé afin de s'assurer a posteriori qu'il satisfait toutes les exigences, contraintes et attentes du client et des autres parties prenantes : cette analyse est nommée vérification. De manière plus générale, lorsqu'une entité E quelconque a été créée, définie et que toutes ses informations sont renseignées, une vérification doit être mise en œuvre afin de s'assurer que cette entité satisfait bien les exigences ayant mené à sa création.

Retenant cette définition, nous proposons que la vérification caractérise la conception et le projet de manière ascendante et soit mise à jour lors de la consolidation du système et du projet. Nous associons donc à chaque entité E de conception et de planification un attribut supplémentaire permettant de qualifier sa vérification. Cet attribut est noté E.Ve dans la suite de ce mémoire.

Les états possibles pour l'attribut de vérification Ve d'une entité E quelconque sont :

- *non déterminé* noté « E.Ve = ND » état par défaut : cet état signifie que l'analyse de vérification de l'entité E n'a pas encore été réalisée ou n'est pas encore terminée ;
- *vérifiée* noté « E.Ve = OK » : cet état signifie que l'entité E a été conçue et satisfait les contraintes qui lui étaient assignées. Cet état peut éventuellement être remis en cause en cas d'évolutions des exigences par exemple ;
- *non vérifiée* noté « E.Ve = KO » : cet état signifie que l'entité E ne satisfait pas les contraintes qui lui étaient assignées. Cet état peut cependant être réévalué par la suite en fonction de l'évolution des exigences par exemple.

Suite à ces définitions, nous proposons de formaliser une première connaissance méthodologique liée aux évolutions possibles des attributs de faisabilité et de vérification. Nous proposons la règle suivante : si une entité E n'est pas, au préalable, qualifiée de *faisable*, son attribut de vérification ne peut pas évoluer : il reste dans l'état *non déterminé* (Règle 1).

Règle 1 : Vérification d'une entité en fonction de sa faisabilité

$$(E.Fa \neq OK) \Rightarrow (E.Ve = ND)$$

En effet, une entité qualifiée d'*infaisable* (état 3, fig. 3.1) ne sera pas étudiée et donc sa vérification n'aura pas lieu d'être. De même, une entité dont la faisabilité n'est pas encore déterminée (état 1, Fig. 3.1) n'est pas totalement étudiée (voire pas du tout) et ne peut donc pas être vérifiée.

Comme pour l'attribut de faisabilité, décrivons maintenant les transitions possibles entre états compte tenu de la règle précédente. Les états de l'attribut de vérification ainsi que les transitions entre états sont illustrés sur la figure 3.1. Dès sa création, une entité E ayant son attribut de faisabilité dans l'état *non déterminé*, l'application de la règle 1 permet de préciser que son attribut de vérification est nécessairement dans l'état *non déterminé* E.Fa = ND ∧ E.Ve = ND. Il en est de même pour l'état 3 où l'entité est *infaisable* E.Fa = KO ∧ E.Ve = ND et, par conséquent, non vérifiable.

Une fois dans l'état 2, état où l'entité est faisable, l'attribut de vérification peut évoluer de l'état *Non Déterminé* :

- soit à l'état *vérifié* E.Fa = OK ∧ E.Ve = OK (état 4, fig. 3.1) si les informations saisies pour renseigner l'entité correspondent à ce qui était attendu lors de sa création, c'est-à-dire aux exigences concernant cette entité (transition d) ;
- soit à l'état *non vérifié* E.Fa = OK ∧ E.Ve = KO (état 5, fig. 3.1) dans le cas contraire (transition e).

Si l'entité est qualifiée de *vérifiée* (E.Ve = OK) ou de *non vérifiée* (E.Ve = KO) (état 4 ou 5, fig. 3.1), le client ou toute autre partie prenante peut remettre en cause les exigences ou contraintes liées à l'entité (transitions f_1 et f_2), les deux attributs (faisabilité et vérification) reviendront à l'état initial *non déterminés* E.Fa = ND ∧ E.Ve = ND (état 1, fig. 3.1). Il faut en effet tout d'abord analyser la faisabilité de l'entité par rapport à ces nouvelles exigences puis sa vérification.

Suite à cette présentation générale des attributs de faisabilité et de vérification de toute entité E générique ainsi que de leurs évolutions, nous proposons, dans la section suivante, de spécialiser la faisabilité et la vérification pour les trois entités de conception et pour les trois entités de planification définies dans la section 2.1. Le tableau 1 présente le plan de lecture de ces spécialisations.

	Entités de conception			Entités de planification		
	Exigences système	Alternative système	Système	Tâche TE	Tâche TD	Projet
Faisabilité	1	2	3	7	8	9
Vérification	4	5	6	10	11	12

Tableau 1 : Spécialisations de la faisabilité et de la vérification sur les entités de conception et de planification

3.1.2 Attributs de faisabilité et vérification des entités de conception

3.1.2.1 *Faisabilité des entités de conception*

(1) Faisabilité des exigences système : ES.Fa

En se basant sur la définition générale de la faisabilité, tant que les besoins et exigences techniques n'ont pas été totalement recueillis et que le responsable de conception ne s'est pas prononcé sur la capacité de l'équipe de concepteurs à concevoir un système répondant aux exigences système, la faisabilité de l'entité exigences système ES est considérée comme *non déterminée*, (ES.Fa=ND - état initial 1, fig. 3.2).

Après le recueil des besoins et exigences techniques, le responsable de conception peut faire évoluer l'attribut de faisabilité :

- de l'état *non déterminé* (état initial 1, fig. 3.2) à l'état *faisable* (état 2, fig. 3.2) si les besoins et exigences techniques ont été recueillis et qu'il pense être capable de développer une (ou plusieurs) solution(s) au regard des exigences spécifiées (transition a). Cette analyse peut consister à contrôler que les exigences système recueillies sont cohérentes, complètes et ne sont pas contradictoires et être complétée par l'expérience du responsable de conception ;
- de l'état *non déterminé* (état initial 1, fig. 3.2) à l'état *infaisable* (état 3, fig. 3.2) si les besoins et exigences techniques ont été recueillis et qu'il pense ne pas être en mesure de développer au moins une solution au regard des exigences spécifiées (transition b). Ce choix peut être guidé par la détection d'une incohérence dans les exigences techniques ou peut également être lié à l'expérience du responsable de conception. Par exemple, après l'analyse des besoins et exigences techniques, le responsable de conception sait, d'après son expérience, que le grand rayon d'action souhaité par le client (RA > 8 000km) n'est pas compatible avec l'exigence sur la masse totale (M < 20 000kg) : il n'est pas possible a priori pour lui de développer un système répondant à de telles exigences.

Dans le cas où les exigences système ES ont été analysées comme *faisables* (état 2 – transition a, fig. 3.2) ou *infaisables* (état 2 – transition b, fig. 3.2), certaines d'entre-elles peuvent être remises en cause ou négociées. Dans ce cas, l'analyse de faisabilité doit être réitérée et l'entité ES retourne dans l'état *non déterminé* (état 1 – transitions c_1 et c_2, fig. 3.2). Cette situation signifie que certaines exigences systèmes sont modifiées à la demande du client, des autres parties prenantes ou à la demande du responsable de conception ne parvenant pas à trouver de solution satisfaisant les exigences incriminées. Dans ce cas, une demande de dérogation peut être émise afin de relaxer certaines contraintes et trouver des solutions à ces infaisabilités.

La figure 3.2 présente le diagramme d'état-transition pour la faisabilité d'une entité exigences système : ES.Fa.

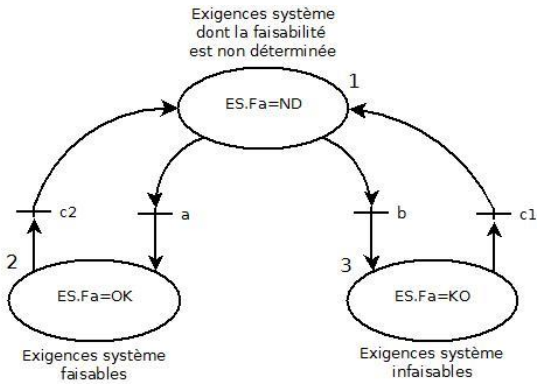


Figure 3.2 : Diagramme d'états-transitions de la faisabilité de l'entité exigences système

(2) Faisabilité d'une alternative système : AS.Fa

Les exigences système doivent être jugées *faisables* pour pouvoir envisager de débuter la conception d'une ou plusieurs solutions (alternatives système AS_i). Nous proposons donc de formaliser une seconde connaissance méthodologique précisant que l'attribut de faisabilité d'une entité alternative système AS doit être obligatoirement dans l'état *non déterminé* si les exigences systèmes ES ne sont pas qualifiées de *faisables* au préalable. Dans la suite de ce mémoire, nous nommons ***n*** le nombre d'alternatives étudiées. Plusieurs alternatives système AS pouvant être rattachées à un même système, la règle 2 est donnée ci-dessous.

Règle 2 : Faisabilité des alternatives système (AS_i) en fonction de la faisabilité des exigences système (ES)

$$\forall i \in [1..n], (ES.Fa \neq OK) \Rightarrow (\forall AS_i.Fa = ND)$$

De plus, tant que le responsable de conception ne s'est pas prononcé sur la capacité de son équipe à aboutir ou non à une solution physique valable au regard des exigences système, une alternative système ne peut être qualifiée de *faisable* ou d'*infaisable*. Comme indiqué dans la section 3.1.1.1, lorsqu'une alternative système est créée, la faisabilité de cette dernière est *non déterminée* (AS.Fa=ND).

Une alternative système AS peut être qualifiée de *faisable* (AS.Fa = OK - état 4, fig.3.3) suite à l'analyse de sa solution logique si :

- dans le cas d'une conception intégrée (donc sans décomposition en sous-systèmes), les principes de fonctionnement ont été conceptualisés et une liste de spécifications de fournitures ainsi que la spécification de leurs interfaces ont été produites. Ainsi, la solution physique n'existe pas encore mais, suite à l'analyse de sa conceptualisation dans la solution logique, le responsable de conception peut juger de sa faisabilité, c'est-à-dire de son aptitude à être transformée en solution physique. Des simulations réalisées à partir des modèles de la solution logique peuvent aider le responsable de conception à qualifier l'attribut de faisabilité de l'alternative. Prenons par exemple une alternative d'aile d'avion dont la conception est sous traitée : il n'y aura donc qu'une seule fourniture (et donc aucune interface) dont la seule spécification est un poids inférieur à 1 000 kg. La fourniture étant spécifiée, l'alternative système est qualifiée de *faisable* ;
- dans le cas d'une conception modulaire (une décision a été prise de concevoir le système en le décomposant en sous-systèmes puis, une fois ces derniers conçus, en les intégrant), la solution logique a été conçue et une liste de spécifications des sous-

systèmes ainsi que les spécifications de leurs interfaces ont été produites. Ainsi, la conceptualisation de la solution représentée dans la solution logique a permis de donner la liste des sous-systèmes et de leurs interfaces en précisant ce qui est attendu de chacun de ces sous-systèmes. Ultérieurement, au niveau inférieur, lorsque la conception de chaque sous-système débutera, ces spécifications serviront de base pour la définition des exigences système des sous-systèmes. Reprenons l'exemple de l'aile d'avion avec une alternative conçue en interne, dont la masse totale doit être inférieure à 1 000 kg. La solution logique précise que cette dernière sera composée de deux sous-systèmes : une partie fixe et une partie mobile. Ces sous-systèmes devront respectivement peser moins de 800 kg et 150 kg, les 50 kg restants étant réservés à l'interface entre les deux sous-systèmes. Suite à ce travail de conceptualisation de la solution en deux parties et de répartition des exigences concernant la masse maximale de l'avion, le responsable de conception peut faire évoluer l'attribut de faisabilité de l'alternative vers *faisable*. Nous remarquons cependant que la faisabilité de l'alternative système ne préjuge en rien de la faisabilité des sous-systèmes, mais il s'agit, pour le responsable de conception, de proposer une spécification et des interfaces entre les sous-systèmes.

Une alternative système AS est qualifiée d'*infaisable* (AS.Fa = KO- état 5, fig.3.3) si :

- dans le cas d'une solution intégrée, la solution logique conçue ne permet pas de proposer des spécifications de fournitures et des spécifications d'interfaces satisfaisantes, c'est-à-dire que certaines spécifications de fournitures paraissent irréalisables au responsable de conception. Reprenons l'exemple d'une alternative d'aile d'avion dont la conception est sous traitée et dont la masse doit être inférieure à 1 000 kg. Lors du renseignement de la solution logique, le responsable de conception précise que l'aile doit être composée d'une partie fixe et d'une partie mobile. Cependant, il sait par expérience que les masses respectives de chaque partie seront supérieures à 900 kg et 150 kg auxquelles il faut encore ajouter 50 kg pour l'interface entre les deux parties : la masse estimée est donc a priori supérieure à l'exigence technique. Cette alternative est donc qualifiée d'*infaisable* ;
- dans le cas d'une solution modulaire, la solution logique conçue ne permet pas de proposer des spécifications de sous-systèmes et des spécifications d'interfaces satisfaisantes. Le responsable de conception n'est pas en mesure de spécifier les sous-systèmes en accord avec les exigences système. Reprenons l'exemple de l'aile d'avion avec une deuxième alternative conçue en interne, dont la masse doit être inférieure à 1 000 kg. Cette dernière sera ici composée de deux sous-systèmes : le premier pour la partie fixe et le second pour la partie mobile. Par expérience, le responsable de conception sait que ces sous-systèmes ne pourront peser respectivement moins de 900 kg et 150 kg auxquels il faudra encore ajouter 50 kg pour l'interface entre les deux parties. La masse totale estimée ne satisfait donc pas, a priori, l'exigence technique. Cette alternative est donc qualifiée d'*infaisable.*

La figure 3.3 présente le diagramme états-transitions pour les attributs de faisabilité des exigences système ES et d'une alternative système AS. Partant de l'état 2 de la figure 3.3 correspondant à des exigences système *faisables* et une alternative système AS *non déterminée*, il est possible d'atteindre les états 4 ou 5. L'état 4 est celui où l'alternative système est jugée *faisable*, l'état 5, celui où elle est jugée *infaisable*.

À partir de l'état où l'alternative est *faisable* (état 4) ou *infaisable* (état 5), les exigences système peuvent évoluer, être remises en cause ou négociées (transitions f_1 et f_2). Dans ce cas, les alternatives systèmes ne répondent potentiellement plus aux exigences modifiées et au-delà, les exigences système modifiées peuvent ne plus être satisfaites, les attributs de faisabilité de l'entité exigences système ainsi que des entités alternatives système redeviennent donc à l'état *non déterminé* (état 1 – transitions f_1 ou f_2).

À partir de l'état 5 où l'alternative système est jugée *infaisable*, le travail de conception système peut être poursuivi afin de la rendre *faisable*. Dans ce cas, l'alternative système est jugée comme techniquement faisable, mais pas dans le cadre des contraintes reçues de la planification (délais trop coûts, budget insuffisant par exemple). Une demande de relaxation des contraintes projet inhérentes à la tâche TD associée doit être envoyée au responsable de planification (état 2 – transition e1). Le responsable de conception est alors en attente du nouveau planning de la tâche TD.

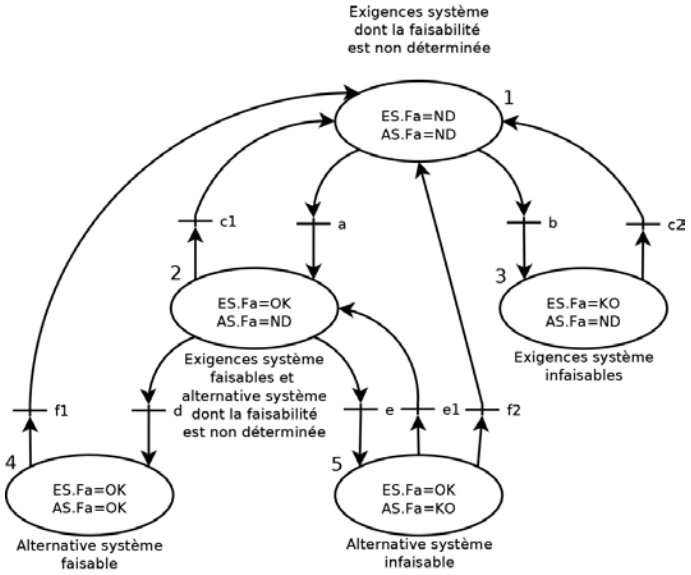


Figure 3.3 : Diagramme d'états-transitions pour les attributs de faisabilité des entités exigences système et alternative système

(3) Faisabilité d'une entité Système : S.Fa

D'après la section 2.1.1, une entité Système étant composée d'une entité d'Exigences Système et d'une ou plusieurs Alternatives Système, nous considérons que la faisabilité d'un système (S.Fa) peut être déduite de la faisabilité des exigences système et des alternatives système.

Au début de la conception, l'attribut de faisabilité des exigences système ES et des alternatives systèmes AS étant dans l'état *non déterminé,* la faisabilité du système est a fortiori également *non déterminée*. Ceci permet de formaliser une connaissance méthodologique avec la règle 3 ci-dessous.

Règle 3 : Faisabilité d'un système à sa création

$$\forall i \in [1..n], (ES.Fa = ND) \wedge (\forall AS_i.Fa = ND) \Rightarrow (S.Fa = ND).$$

L'attribut de faisabilité de l'entité système reste dans l'état *non déterminé* (états 1 et 2, fig. 3.4) tant que :

- les exigences système ES ne sont pas qualifiées d'*infaisables* car en pareil cas, il n'est pas, a priori, possible de concevoir un système répondant aux exigences exprimées : $(ES.Fa \neq KO) \Longrightarrow (S.Fa = ND)$ ou,
- au moins une alternative système AS n'est pas qualifiée de *faisable*, c'est-à-dire qu'aucune solution logique satisfaisant les exigences système n'a pu être conçue pour l'instant, et que toutes les alternatives système AS ne sont pas qualifiées d'*infaisables,* c'est-à-dire qu'il existe encore une solution logique qui n'est pas totalement conçue et pouvant potentiellement répondre aux exigences système : $\forall i \in [1..n], \big((\forall AS_i.Fa \neq OK) \wedge (\exists AS_i.Fa = ND)\big) \Rightarrow (S.Fa = ND)$.

Les deux propositions présentées ci-dessus sont reprises dans la règle 4.

Règle 4 : Faisabilité d'un système *non déterminée*

$$\forall i \in [1..n], \big((ES.Fa \neq KO) \vee ((\forall AS_i.Fa \neq OK) \wedge (\exists AS_i.Fa = ND))\big) \Rightarrow (S.Fa = ND)$$

Le diagramme états-transitions de la figure 3.3 est complété en considérant les évolutions possibles de l'attribut de faisabilité du système. Un système est qualifié de *faisable* (état 4 – transition d, fig. 3.4) si, d'une part, l'équipe de conception est à priori capable de concevoir une solution correspondant aux exigences système, et d'autre part, si la ou les alternatives système, dont la solution logique est conçue, satisfont les exigences système. Un système est donc qualifié de faisable si les exigences systèmes sont qualifiées de faisables $(ES.Fa = OK)$ et, au moins une alternative système est qualifiée de *faisable* $\forall i \in [1..n], (\exists AS_i.Fa = OK)$. Ceci nous permet de formaliser la règle 5 ci-dessous.

Règle 5 : Faisabilité d'un système *faisable*

$$\forall i \in [1..n], ((ES.Fa = OK) \wedge (\exists AS_i.Fa = OK)) \Rightarrow (S.Fa = OK)$$

Un système est qualifié d'*infaisable* (état 3 ou 5, fig. 3.4) si, d'une part, l'équipe de conception n'est a priori pas en mesure de concevoir une solution satisfaisant les exigences système, ou d'autre part, si la ou les alternatives système, dont la solution logique est déjà conçue, ne permettent pas de répondre de façon satisfaisante aux exigences système. Un système est donc qualifié d'*infaisable* si les exigences systèmes sont qualifiées d'*infaisables* $(ES.Fa = KO)$ ou si toutes les alternatives système sont qualifiées d'*infaisables* $\forall i \in [1..n], (\forall AS_i.Fa = KO)$. Nous formalisons ainsi la règle 6.

Règle 6 : Faisabilité d'un système *infaisable*

$$\forall i \in [1..n], ((ES.Fa = KO) \vee (\forall AS_i.Fa = KO)) \Rightarrow (S.Fa = KO)$$

La figure 3.4 ci-dessous présente, à partir du diagramme états-transitions de la figure 3.3, les évolutions possibles de l'attribut de faisabilité du système S.Fa en fonction des états des attributs de faisabilité des exigences système ES.Fa et d'une unique alternative système AS.Fa.

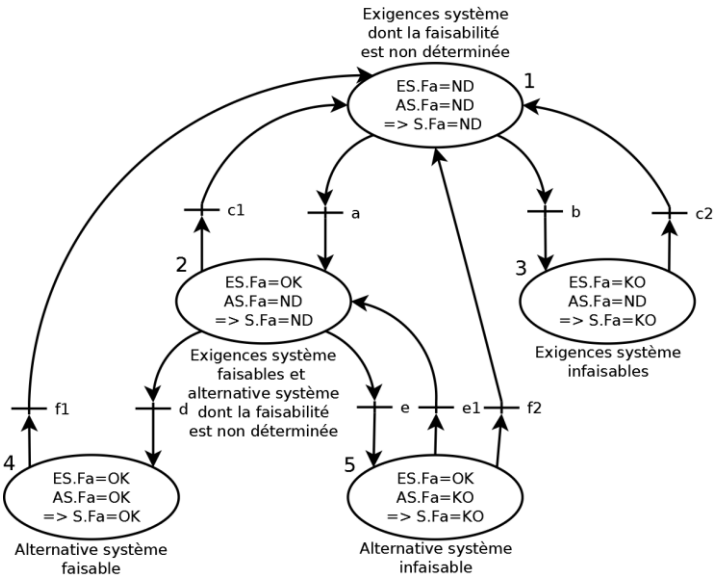


Figure 3.4 : Diagramme d'états-transitions pour l'attribut de faisabilité d'un système

3.1.2.2 *Vérification des entités de conception*

(4) Vérification des exigences système : ES.Ve

En se basant sur la définition de la vérification que nous avons posée dans la section 3.1.1, la vérification d'une entité exigences systèmes ES consiste à s'assurer que les exigences techniques traduisent correctement les besoins du client. Dans ce mémoire, nous ne considérons pas la vérification des exigences : l'entité exigences système n'aura donc pas d'attribut spécifique de vérification, l'attribut de faisabilité tenant ce rôle.

(5) Vérification d'une alternative système : AS.Ve

En se basant sur les définitions de l'entité alternative système (section 2.1.1.1) et de la vérification (section 3.1.1), l'équipe de conception doit avoir conçu une solution physique avant de pouvoir envisager la vérification d'une alternative système. Il s'agit alors de vérifier que cette solution physique satisfait les exigences système. Un attribut spécifique à la vérification, noté AS.Ve, est donc lié à chaque alternative système.

De plus, tant que l'alternative système est en cours de conception, la vérification de cette dernière est considérée comme *non déterminée*. En se basant sur la règle 1, si une entité alternative système AS n'est pas, au préalable, qualifiée de *faisable*, l'attribut de vérification de cette entité ne peux pas se trouver dans un état autre que *non déterminé.* Cette connaissance méthodologique est instanciée par la règle 7.

Règle 7 : Vérification d'une alternative système en fonction de sa faisabilité

$$(AS.Fa \neq OK) \Rightarrow (AS.Ve = ND)$$

Une alternative système AS est qualifiée de *vérifiée* (AS.Ve = OK) si la solution physique est conçue et :

- dans le cas d'une conception intégrée, cette dernière satisfait les exigences techniques du système. Par exemple, une aile d'avion, dont la seule exigence technique est un masse inférieure à 1 000 kg, a une masse réelle de 960 kg. L'exigence technique est donc respectée et l'alternative système est *vérifiée* (AS.Ve=OK),

- en cas de conception modulaire, cette dernière doit également satisfaire les exigences techniques du système mais il faut ajouter la condition suivante : tous les sous-systèmes doivent être qualifiés de *vérifiés* et après l'intégration des différents sous-systèmes, l'alternative doit satisfaire les exigences système. Par exemple, une autre aile d'avion, dont la masse doit être inférieure à 1 000 kg, et composée de deux sous-systèmes (une partie fixe et une partie mobile) devant respectivement peser moins de 800kg et 150kg, pèse réellement 950 kg et les sous systèmes respectivement 790 kg et 140 kg, les 20kg restants étant liés à l'interface entre ces deux parties. L'exigence système est respectée, les sous systèmes sont vérifiés, et leur intégration vérifie les exigences exprimées ; l'alternative est donc *vérifiée* (AS.Ve=OK).

Une alternative système AS est qualifiée de *non vérifiée* (AS.Ve = KO) si la solution physique est conçue et :

- dans le cas d'une non décomposition, cette dernière ne respecte pas toutes les exigences techniques du système. Par exemple, une aile d'avion, dont la seule exigence technique est une masse inférieure à 1 000 kg, a une masse réelle de 1 030 kg. L'exigence technique n'est donc pas respectée et l'alternative système est *non vérifiée* (AS.Ve = KO),
- s'il y a décomposition en p sous-systèmes :
  - tous les sous-systèmes sont qualifiés de *vérifiés* mais l'alternative ne respecte pas toutes les exigences système. Par exemple, une autre alternative d'aile d'avion, dont la masse doit être inférieure à 1 000 kg, et composée de deux sous-systèmes (une partie fixe et une partie mobile) devant respectivement peser moins de 800kg et 150kg, pèse réellement 1 020 kg et les sous systèmes respectivement 790 kg et 140 kg, les 70kg restants étant liés à l'interface entre ces deux parties. Les sous-systèmes sont bien *vérifiés* mais l'exigence système de l'alternative n'est pas respectée, l'alternative est donc *non vérifiée* (AS.Ve=KO), ou,
  - certains sous-systèmes sont qualifiés de *non vérifiés* (règle 8). Par exemple, une autre alternative d'aile d'avion, dont la masse doit être inférieure à 1 000 kg, et composée de deux sous-systèmes SS_1 et SS_2 (une partie fixe et une partie mobile) devant respectivement peser moins de 800kg et 150kg, pèse réellement 980 kg et les sous systèmes respectivement 820 kg et 140 kg, les 20kg restants étant liés à l'interface entre ces deux parties. L'exigence technique est bien respectée mais un des sous systèmes n'est pas vérifié (SS_1), l'alternative est donc *non vérifiée* : AS.Ve=KO. Nous précisons toutefois que certains sous-systèmes peuvent être qualifiés de *non vérifiés* tout en présentant une configuration telle que les exigences système sont respectées. Par exemple, une alternative d'aile d'avion, dont le poids doit être inférieur à 1 000 kg, et composée de deux sous-systèmes (une partie fixe et une partie mobile) devant respectivement peser moins de 800kg et 150kg, pèse réellement 970 kg et les sous systèmes respectivement 820 kg et 130 kg, les 20kg restants étant liés à l'interface entre ces deux parties. Le premier sous-système est qualifié de *non vérifié* et pourtant, les exigences systèmes de l'alternative d'aile sont satisfaites. Nous considérons, dans ce cas, que la solution logique, et plus particulièrement les spécifications des sous-systèmes, a été mal définie et doit donc être remise en cause, l'alternative est donc qualifiée de *non vérifiée* tant que la solution logique n'est pas revue.

Règle 8 : Vérification d'une alternative système en fonction de la vérification de ses sous-systèmes

$$\forall i \in [1..p], (\exists SS_i.Ve = KO) \Rightarrow (\text{AS.Ve} = \text{KO})$$

La figure 3.5 présente le diagramme états-transitions pour les attributs de vérification d'une alternative système AS. Partant de l'état 4 correspondant à des exigences système *faisables* (ES.Fa = OK ∧ AS.Ve = OK) et une alternative système AS *faisable*, il est possible d'atteindre les états 6 ou 7. L'état 6 est celui où l'alternative système est jugée *vérifiée* (AS.Ve = OK), l'état 7, celui où elle est jugée *non vérifiée* (AS.Ve = KO).

À partir de l'état où l'alternative est *vérifiée* (état 6) ou *non vérifiée* (état 7), les exigences système peuvent évoluer, être remises en cause ou négociées (transitions i_1 et i_2). Dans ce cas, les alternatives systèmes ne répondent potentiellement plus aux exigences modifiées et au-delà, les exigences système modifiées peuvent ne plus être satisfaites, l'attribut de vérification de l'alternative système, les attributs de faisabilité de l'entité exigences systèmes ainsi que des entités alternatives système reviennent donc à l'état *non déterminé* (état 1 – transitions i_1 ou i_2).

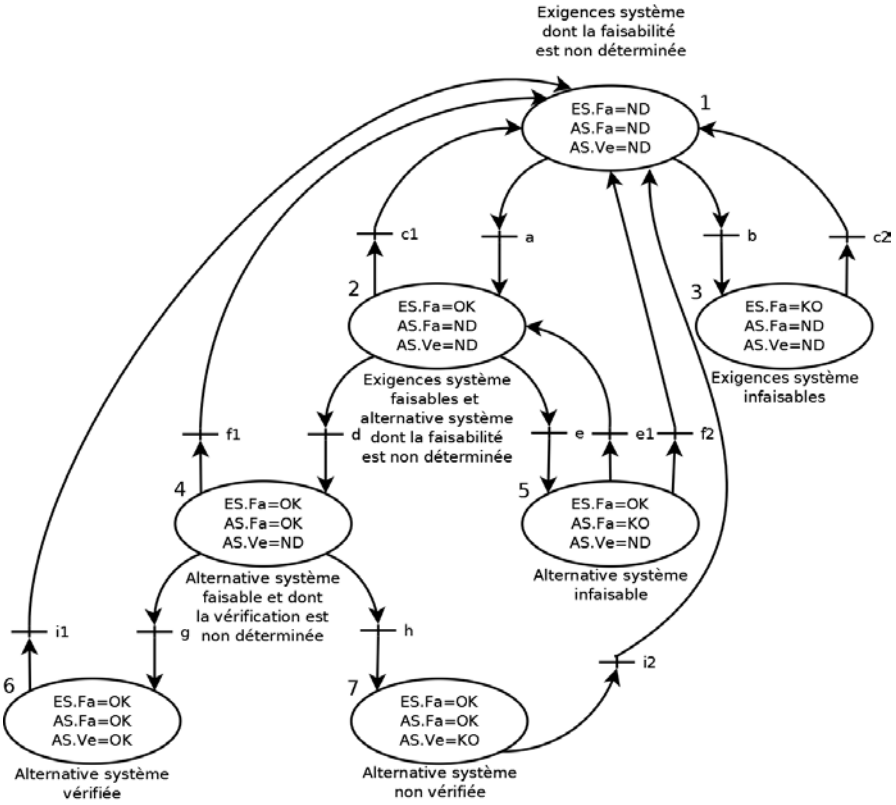


Figure 3.5 : Diagramme d'états-transitions pour l'attribut de vérification d'une alternative système

(6) Vérification des entités système : S.Ve

D'après la section 2.1, une entité Système est composée d'une entité d'exigences système et d'une ou plusieurs alternatives système. Or, nous avons défini dans cette section qu'il n'y avait pas d'attribut de vérification pour les exigences système, nous considérons donc que la faisabilité d'un système S.Ve est uniquement déduite à partir des états de la faisabilité des alternatives système AS.Ve.

Dès le début de la conception, les alternatives systèmes AS ont leur attribut de vérification à l'état *non déterminé,* la vérification du système est donc également *non déterminée* (règle 9) et cet attribut reste dans cet état tant qu'il n'existe pas d'alternative système qualifiée de *vérifiée* et que

toutes les alternatives système ne sont pas qualifiées de *non vérifiées*, c'est-à-dire qu'aucune solution n'est en mesure de répondre aux attentes du client. L'attribut de vérification d'un système est donc, dès la création de ce dernier, *non déterminé*.

Règle 9 : Vérification d'un système à sa création

$$\forall i \in [1..n], (\forall AS_i.Ve = ND) \Rightarrow (\mathrm{S.Ve} = \mathrm{ND})$$

L'attribut de vérification de l'entité système reste dans l'état *non déterminé* tant que :

- il n'existe pas d'alternative système qualifiée de *vérifiée.* Dans ce cas, il n'existe pas encore une solution physique satisfaisant les exigences système ES : $\forall i \in [1..n], (\nexists AS_i.Ve = OK)$ et,
- toutes les alternatives système ne sont pas qualifiées de *non vérifiées*, c'est-à-dire qu'il existe au moins une solution physique pouvant potentiellement satisfaire les exigences système ES : $\forall i \in [1..n], (\exists AS_i.Ve \neq KO)$.

Les deux propositions présentées ci-dessus sont reprises dans la règle 10.

Règle 10 : Vérification d'un système *non déterminé*

$$\forall i \in [1..n], (\nexists AS_i.Ve = OK) \wedge (\exists AS_i.Ve \neq KO) \Rightarrow (S.Ve = ND)$$

Une alternative système qualifiée de *vérifiée* signifie qu'il existe une solution physique conçue qui satisfait les exigences système. Un système est donc qualifié de *vérifié* si au moins une alternative système est qualifiée de *vérifiée.* Ceci nous permet de formaliser la règle 11.

Règle 11 : Vérification d'un système *vérifié*

$$\forall i \in [1..n], (\exists AS_i.Ve = OK) \Rightarrow (S.Ve = OK)$$

S'il existe une alternative système qualifiée de *vérifiée*, le système est *vérifié* (règle 11) et s'il existe une alternative qualifiée de *non déterminée*, cela signifie qu'elle n'est pas encore totalement conçue et peut potentiellement être qualifiée de vérifiée à l'issue de la conception (règle 10). Un système ne peut donc être *non vérifié* que si toutes ses alternatives système sont qualifiées de *non vérifiées* au préalable. Cette connaissance méthodologique est formalisée dans la règle 12.

Règle 12 : Vérification d'un système *non vérifié*

$$\forall i \in [1..n], (\forall AS_i.Ve = KO) \Rightarrow (S.Ve = KO)$$

Ceci étant posé, la figure 3.6 présente, sur le diagramme d'états-transitions de la figure 3.5, la faisabilité et la vérification du système S en fonction des états de faisabilité et de vérification des exigences système ES et d'une alternative système AS.

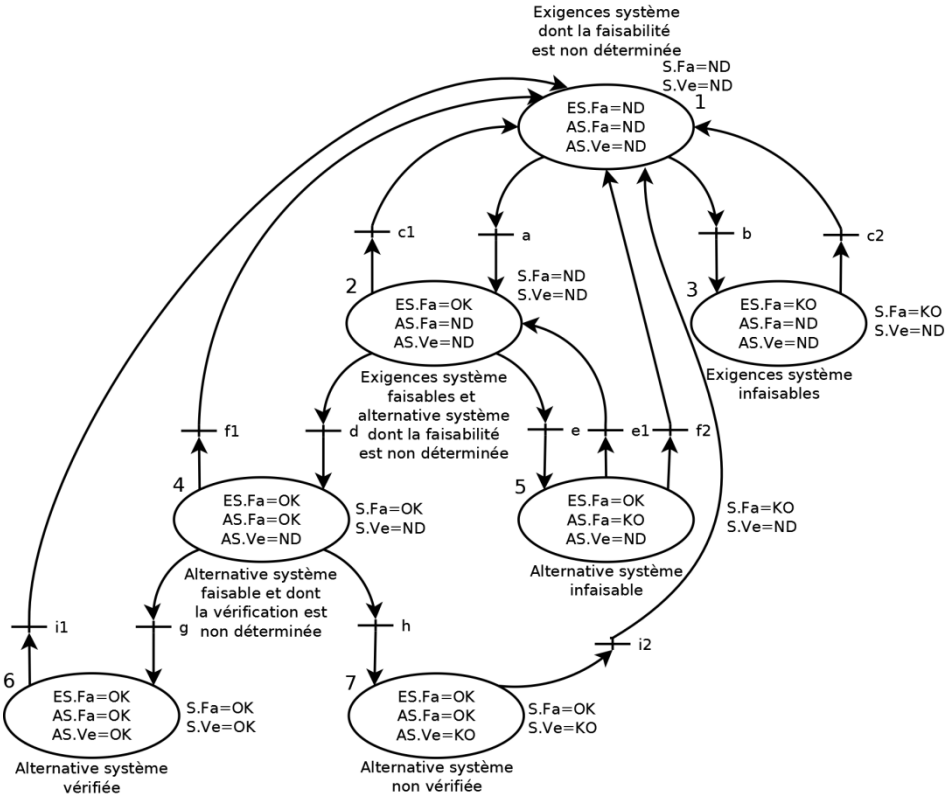


Figure 3.6 : Diagramme d'états-transitions pour l'attribut de vérification d'un système

3.1.3 Faisabilité et vérification des entités de planification

3.1.3.1 *Faisabilité des entités de planification*

(7) Faisabilité d'une tâche TE : TE.Fa

En se basant sur la définition générale de la faisabilité, définition 1 section 3.1.1, tant que le responsable de planification n'a pas renseigné puis planifié la tâche de recueil des exigences système TE et qu'il ne s'est pas prononcé sur la capacité des ressources à satisfaire les contraintes liés à cette tâche, la faisabilité de la tâche TE est considérée comme *non déterminée* (TE.Fa=ND - état initial 1, fig. 3.7).

Après la planification de la tâche TE, le responsable de planification peut faire passer l'attribut de faisabilité :

- de l'état *non déterminé* (état initial 1, fig. 3.7) à l'état *faisable* (état 2, fig. 3.7) si les ressources sont affectées, une durée donnée, l'ensemble des contraintes projet est renseigné et les dates de début et de fin estimées et s'il pense que cette tâche est réalisable au regard des contraintes projet spécifiées (transition a). Par exemple, après l'analyse de la planification de la tâche présentée dans la figure 2.17 (cf. section 2.4.1), le responsable de planification, M. RespP, d'après son expérience, sait qu'il est possible, a priori, de réaliser la tâche en respectant les contraintes,
- de l'état *non déterminé* (état initial 1, fig. 3.7) à l'état *infaisable* (état 3, fig. 3.7) s'il ne pense pas, au regard des résultats de planification, que les contraintes projet puissent toutes être respectées (transition b). Par exemple, toujours après l'analyse de la planification de la tâche présentée sur la figure 2.17 (cf. section 2.4.1), le responsable de planification sait, d'après son expérience, que la durée de la tâche TE (1,5 mois) n'est pas compatible avec les ressources allouées (M. RespC) : il n'est pas possible a priori pour lui

de proposer une planification de la tâche TE répondant aux contraintes de planification. La faisabilité de la tâche TE passe de *non déterminé* à *infaisable* (TE.Fa = KO).

Dans le cas où la tâche TE est analysée comme *faisable* (état 2, fig. 3.7) ou *infaisable* (état 3, fig. 3.7), sa faisabilité peut être remise en cause ou négociée. Dans ce cas, l'analyse de faisabilité doit être réitérée et la tâche TE retourne dans l'état *non déterminé* (état 1 – transition c_1 ou c_2, fig. 3.7). Cette situation signifie que certaines contraintes projet sont modifiées à la demande du client, des autres parties prenantes ou encore, du responsable de planification ne parvenant pas à trouver de solution satisfaisant ces contraintes. Dans ce cas, une demande de dérogation peut être émise afin de relaxer certaines contraintes et trouver des solutions à ces infaisabilités.

La figure 3.7 montre le diagramme d'états transitions pour la faisabilité d'une tâche de recueil d'exigences systèmes et de recherche d'alternatives TE.Fa.

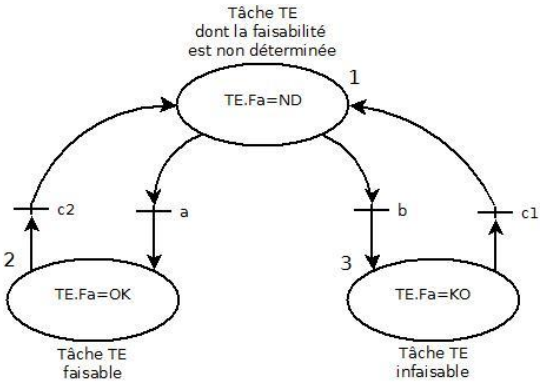


Figure 3.7 : Diagramme d'états-transitions pour la faisabilité d'une tâche TE

(8) Faisabilité d'une tâche TD : TD.Fa

De manière similaire aux entités de conception, la tâche TE doit être évaluée *faisable* pour pouvoir envisager une tâche de développement d'alternative (tâche TD) car le responsable de planification doit disposer des informations recueillies durant la tâche TE pour créer et renseigner la tâche TD. Nous proposons donc de formaliser cette connaissance méthodologique en précisant que la faisabilité d'une tâche de développement d'alternative TD doit être obligatoirement dans l'état *non déterminé* si la tâche TE n'est pas qualifiée de *faisable* au préalable. Dans la suite de ce mémoire, **n** est le nombre de tâches TD étudiées. Plusieurs tâches TD pouvant être rattachées à un même projet, cette connaissance méthodologique se décline selon la règle 13.

Règle 13 : Faisabilité de la tâche TD en fonction de la faisabilité de la tâche TE

$$\forall i \in [1..n], (TE.Fa \neq OK) \Rightarrow (\forall TD_i.Fa = ND)$$

De plus, tant que le responsable de planification n'a pas encore planifié la tâche TD ou qu'il ne s'est pas encore prononcé sur la capacité des ressources à satisfaire les contraintes projet associées à la tâche, une tâche TD ne peut pas être qualifiée de *faisable* ou *d'infaisable.* Comme indiqué en section 3.1.1.1 lorsqu'une tâche TD est créée, sa faisabilité est *non déterminée* (TD.Fa=ND).

Dans le cas d'une non décomposition (donc sans décomposition en sous-projets), l'analyse de faisabilité d'une tâche de développement d'alternative TD est similaire à celle d'une tâche TE. Dans le cas d'une décomposition (une décision a été prise de planifier le projet en le décomposant en sous-projets puis, une fois ces derniers planifiés, en les intégrant) :

- un tâche TD peut être qualifiée de *faisable* (TD.Fa = OK - état 4, fig. 3.8) si, suite à la planification de la tâche TD, le responsable de planification pense que cette tâche est réalisable dans une telle configuration. En outre, une liste de spécifications pour chaque sous-projet est proposée. Au niveau inférieur, lorsque la planification de chaque sous-projet débutera, ces spécifications serviront de base pour la définition des contraintes projet des sous-projets. Prenons l'exemple d'une tâche de développement d'une alternative d'aile d'avion dont la conception est réalisée en interne et dont la durée totale doit être inférieure à 6 mois. En analysant la complexité de la tâche, le responsable de planification, M. RespP, décide de décomposer la tâche TD en deux sous-projets : un premier dédié à la conception de la partie fixe et un second pour la partie mobile. Ces sous-projets devront respectivement durer moins de 2 et 1 mois, les 3 mois restants étant alloués à la conception du système dans son intégralité ainsi qu'à l'intégration des composants et à la vérification système. Suite à ce travail de conceptualisation de la planification en deux parties et de répartition des contraintes concernant la durée du projet, le responsable de planification peut faire évoluer l'attribut de faisabilité de la tâche TD vers *faisable*. Nous remarquons cependant que la faisabilité de la tâche TD ne préjuge en rien de la faisabilité des sous-projets, mais il s'agit, pour le responsable de planification, de proposer une répartition réalisable des contraintes du projet incluant les sous-projets.
- un tâche TD peut être qualifiée d'*infaisable* (TD.Fa = KO - état 5, fig. 3.8) si, suite à la planification de la tâche TD, le responsable de planification pense que les contraintes projet ne pourront pas toutes être respectées ou qu'il ne parvient pas à proposer une liste de spécifications pour chaque sous-projet (transition e, fig. 3.8). Reprenons l'exemple d'une deuxième tâche de développement d'une aile d'avion avec une deuxième alternative conçue en interne et dont la durée ne doit pas excéder 6 mois. Cette dernière sera également composée de deux sous-projets : le premier dédié à la conception de la partie fixe et le second pour la partie mobile. Par expérience, le responsable de planification sait que ces sous-projets ne pourront durer respectivement moins de 4 et 3 mois auxquels il faudra encore ajouter 3 mois alloués à la conception du système dans son intégralité ainsi qu'à l'intégration des composants et à la vérification système. Il est possible de paralléliser les deux sous-projets, cependant, la durée totale sera au minimum de 7 mois : la durée totale estimée de la tâche TD ne satisfait donc pas, a priori, les contraintes projet. Cette tâche TD est donc qualifiée d'*infaisable.*

À partir de l'état où la tâche TD est *faisable* (état 4, fig. 3.8) ou *infaisable* (état 5, fig. 3.8), les contraintes projet peuvent évoluer, être remises en cause ou négociées (transitions f_1, fig. 3.8). Dans ce cas, les tâches TD ne répondent potentiellement plus aux contraintes modifiées et au-delà, la tâche TE est replanifiée afin de prendre en compte les contraintes modifiées, les attributs de faisabilité de la tâche TE ainsi que des tâches TD reviennent donc à l'état *non déterminé* (état 1 – transition f_1 et f_2, fig. 3.8).

La figure 3.8 présente le diagramme d'états-transitions pour les attributs de faisabilité d'une tâche TE et d'une tâche TD. Partant de l'état 2, correspondant à une tâche TE *faisable* et une tâche TD dont la faisabilité est *non déterminée*, il est possible d'atteindre les états 4 ou 5. L'état 4 est celui où la tâche TD est jugée *faisable*, l'état 5, celui où elle est jugée *infaisable*. Lorsque la tâche TD est jugée *infaisable* (état 5), il est possible de relaxer certaines contraintes uniquement sur la tâche TD sans affecter la tâche TE et de la replanifier afin de la rendre *faisable.* Dans ce cas, la faisabilité de la tâche

TD retourne à l'état *non déterminé* (état 2 – transition e1). Supposons un projet P comportant une tâche TE et une seule tâche TD. Le projet P doit durer moins de 5 jours. La tâche TE avait une durée prévisionnelle de 2 jours et a été réalisée effectivement en 2 jours. La contrainte initiale sur la durée de la tâche TD est de 2 jours permettant de garder une marge d'un jour sur le projet global. Suite à la planification, la charge de travail à réaliser dans la tâche TD s'avère trop importante pour être réalisée en 2 jours (TD.Fa = KO), le responsable de planification décide d'utiliser la marge et de replanifier avec une contrainte sur la durée de la tâche TD de 3 jours (TD.Fa = ND). La planification s'avérant alors réalisable, la faisabilité de la tâche TD passe à l'état *faisable* (TD.Fa = OK).

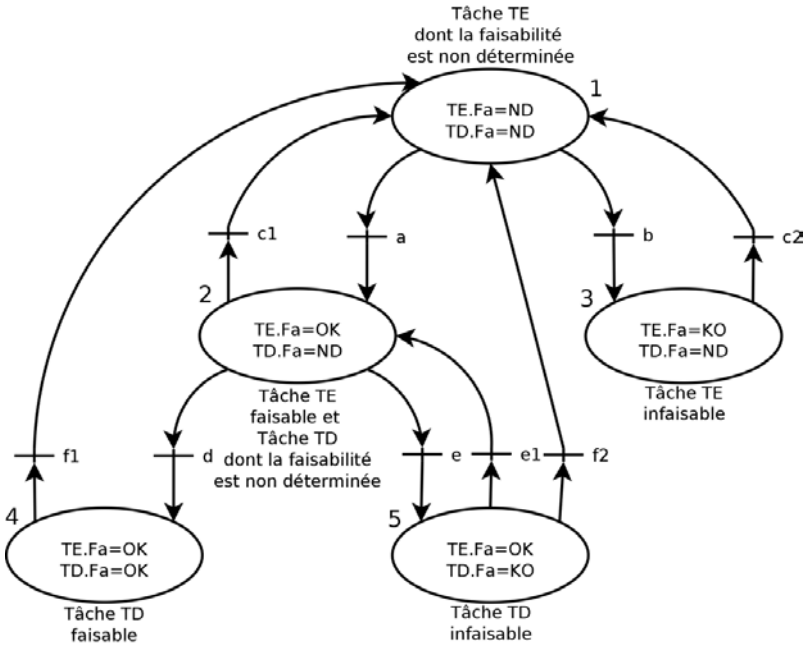


Figure 3.8 : Diagramme d'états-transitions pour les attributs de faisabilité d'une tâche TE et d'une tâche TD

(9) Faisabilité d'une entité Projet de conception: P.Fa

Similairement à une entité Système, une entité Projet de conception étant composée d'une entité tâche TE et d'une ou plusieurs entités tâches TD, nous considérons que la faisabilité d'un projet de conception P.Fa peut être déduite de la faisabilité des entités tâches TE et TD.

Au début de la planification du projet, l'attribut de faisabilité de la tâche TE et des tâches TD étant dans l'état *non déterminé,* la faisabilité du projet de conception est a fortiori également *non déterminée*. Ceci permet de formaliser une connaissance méthodologique avec la règle 14 ci-dessous.

Règle 14 : Faisabilité d'un projet de conception lors de sa création

$$\forall i \in [1..n], \big((TE.Fa = ND) \wedge (\forall TD_i.Fa = ND)\big) \Rightarrow (P.Fa = ND)$$

L'attribut de faisabilité de l'entité Projet de conception reste dans l'état *non déterminé* tant que :

- la tâche TE est qualifiée de *non déterminée* car en pareil cas, la planification de la tâche TE n'a pas encore été réalisée et par conséquent, la planification du projet non plus : $(TE.Fa = ND) \Rightarrow (P.Fa = ND)$ ou,
- la ou les tâches TD sont soit en cours d'analyse (affectation des ressources, planification, renseignement), soit sont qualifiées d'*infaisables*. Dans ce cas, il est

nécessaires que toutes les solutions n'aient pas encore été étudiées (qualifiées de *non déterminées*) et que celles qui l'ont été soient qualifiées d'*infaisables* : $\forall i \in [1..n], \big((\forall TD_i.Fa \neq OK) \wedge (\exists TD_i.Fa = ND)\big) \Rightarrow (P.Fa = ND)$

Les deux propositions présentées ci-dessus sont reprises dans la règle 15.

Règle 15 : Faisabilité d'un projet de conception *non déterminée*

$$\forall i \in [1..n], \big((TE.Fa = ND) \vee ((\forall TD_i.Fa \neq OK) \wedge (\exists TD_i.Fa = ND))\big) \Rightarrow (P.Fa = ND)$$

Le diagramme états-transitions de la figure 3.8 est complété en considérant les évolutions possibles de l'attribut de faisabilité du projet de conception. En partant de l'état 2 où seule la tâche TE est faisable, si la ou les tâches TD satisfont également les contraintes projet, alors le projet de conception est qualifié de *faisable* (état 4 – transition d). Un projet de conception est donc qualifié de *faisable* si la tâche TE est qualifiée de *faisable* $(TE.Fa = OK)$ et, au moins une tâche TD est qualifiée de *faisable* $\forall i \in [1..n], (\exists TD_i.Fa = OK)$. Ceci nous permet de formaliser la règle 16 ci-dessous.

Règle 16 : Faisabilité d'un projet de conception *faisable*

$$\forall i \in [1..n], ((TE.Fa = OK) \wedge (\exists TD_i.Fa = OK)) \Rightarrow (P.Fa = OK)$$

Un projet de conception est qualifié d'*infaisable* (état 3 ou 5) si d'une part, le responsable de planification n'est pas, a priori, en mesure de proposer une planification de la tâche TE répondant aux contraintes et indicateurs du projet, ou d'autre part, si il n'est pas, a priori, en mesure de proposer une planification de l'ensemble des tâches TD. Un projet de conception est donc qualifié d'*infaisable* si la tâche TE est qualifiée d'*infaisable* $(TE.Fa = KO)$ ou si toutes les tâche TD sont qualifiées d'*infaisables* $\forall i \in [1..n], (\forall TD_i.Fa = KO)$. Nous formalisons ainsi la règle 17.

Règle 17 : Faisabilité d'un projet de conception *infaisable*

$$\forall i \in [1..n], ((TE.Fa = KO) \vee (\forall TD_i.Fa = KO)) \Rightarrow (P.Fa = KO)$$

La figure 3.9 présente, à partir du diagramme états-transitions de la figure 3.8, les évolutions possibles de l'attribut de faisabilité du projet de conception P.Fa en fonction des états de faisabilité de la tache TE (TE.Fa) et d'une unique tâche TD (TD.Fa).

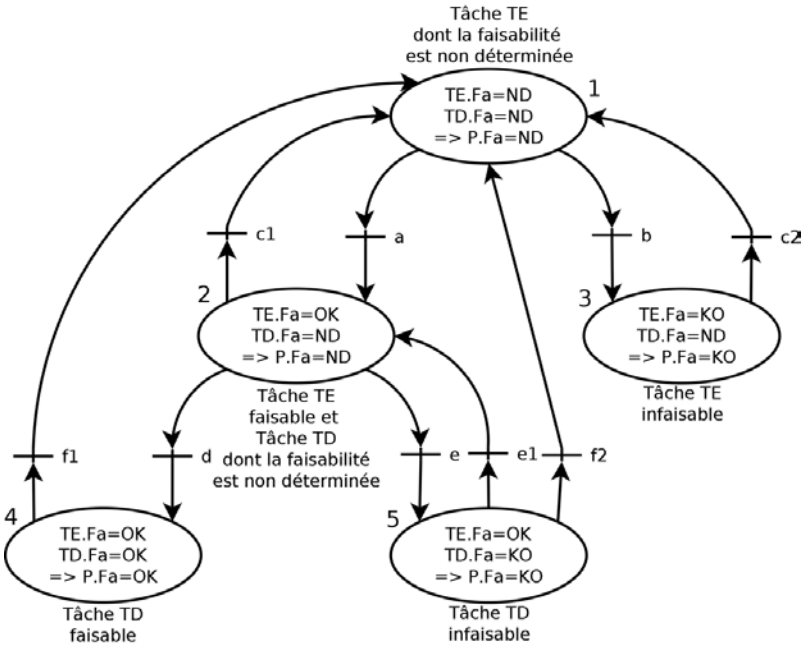


Figure 3.9 : Diagramme d'états-transitions pour l'attribut de faisabilité d'un projet

3.1.3.2 *Vérification des entités de planification*

(10) Vérification d'une tâche TE : TE.Ve

En se basant sur les définitions de la tâche de recueil des exigences système TE (section 2.1.1.2) et de la vérification (section 3.1.1), la tâche TE doit être réalisée avant de pouvoir être vérifiée par le responsable de planification. Il s'agit alors de vérifier que cette dernière satisfait les contraintes projet qui lui sont assignées. Nous proposons qu'un attribut spécifique à la vérification, noté TE.Ve, soit lié à chaque tâche TE.

Tant que la tâche TE est en cours de réalisation, la vérification de cette dernière est considérée comme *non déterminée*. En se basant sur la règle 1, si une tâche TE n'est pas, au préalable, qualifiée de *faisable*, l'attribut de vérification de cette entité ne peut pas se trouver dans un état autre que *non déterminé.* Cette connaissance méthodologique est déclinée par la règle 18.

Règle 18 : Vérification d'une tâche TE en fonction de sa faisabilité

$$(TE.Fa \neq OK) \Rightarrow (TE.Ve = ND)$$

Si la tâche TE est terminée :

- et si les contraintes projet qui lui ont été assignées ont été respectées, la tâche TE est qualifiée de *vérifiée* (TE.Ve = OK - état 4, fig. 3.10). Par exemple, si la tâche est planifiée avec une durée de 2 mois et que cette dernière a duré réellement 1,5 mois, elle est qualifiée de *vérifiée*,
- si les contraintes projet qui lui ont été assignées n'ont pas été respectées, la tâche TE est qualifiée de *non vérifiée* (TE.Ve = KO - état 5, fig. 3.10). Par exemple, si la tâche est planifiée avec une durée de 2 mois et que cette dernière a duré effectivement 2,5 mois, elle est qualifiée de *non vérifiée*.

La figure 3.10 présente le diagramme d'états-transitions pour l'attribut de vérification d'une tâche TE. Partant de l'état 2, correspondant à une tâche TE *faisable*, il est possible d'atteindre les états 4 et 5. L'état 4 est celui où la tâche TE est jugée *vérifiée*, l'état 5, celui où elle est jugée *non vérifiée.*

À partir de l'état où la tâche TE est *vérifiée* (état 4, fig. 3.10) ou *non vérifiée* (état 5, fig. 3.10), les contraintes projet peuvent évoluer, être remises en cause ou négociées (transitions f_1 et f_2, fig. 3.10). Dans ce cas, la tâche TE ne satisfait plus les contraintes. L'attribut de vérification de la tâche TE ainsi que sa faisabilité repassent donc dans l'état *non déterminé* (état 1 – transitions f_1 et f_2, fig. 3.10).

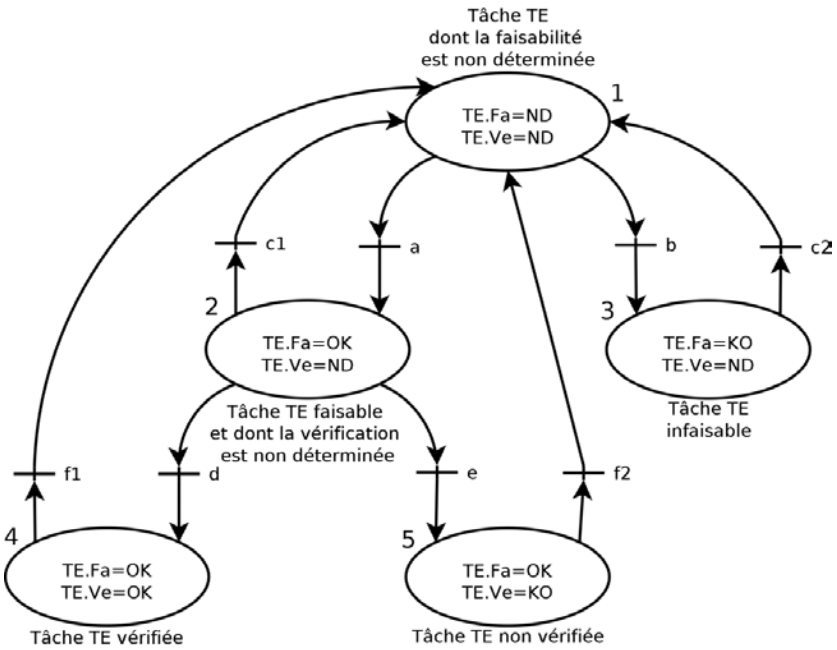


Figure 3.10 : Diagramme d'états-transitions pour l'attribut de vérification d'une tâche TE

(11) Vérification d'une tâche TD : TD.Ve

La vérification d'une tâche de développement d'alternative TD ressemble à la vérification d'une tâche TE. En cas de non-décomposition, la vérification est en tout point similaire à la vérification de la tâche TE. En cas de décomposition, le responsable de planification doit tenir compte des propositions suivantes.

Une tâche TD ne peut être qualifiée de *vérifiée* que si l'ensemble des p sous-projets de conception rattachés à cette tâche sont qualifiés de *vérifiés*. Cette proposition est nécessaire mais non suffisante pour qualifier de *vérifiée* une tâche TD puisque cette dernière se compose, outre les sous-projets précédemment évoqués, d'autres tâches dont dépend sa vérification.

Une tâche TD est qualifiée de *non vérifiée* si au moins un sous-projet de conception est *non vérifié* comme présenté par la règle 19. Nous précisons toutefois que certains sous-projets peuvent être qualifiés de *non vérifiés* alors que les contraintes globales de la tâche TD sont respectées. Dans ce cas, la répartition des contraintes entre chaque sous-projet a été mal réalisée et il est possible de se ramener à une tâche TD vérifiée en modifiant les contraintes de chaque sous-projet. Par exemple, une tâche TD a un coût prévisionnel de 20k€ et ses sous-projets SP1 et SP2 ont respectivement un coût prévisionnel de 15k€ et 5k€ (contraintes projet). Les coûts engagés réellement pour chaque sous-projets sont respectivement de 17k€ (SP1.Ve=KO) et 3k€ (SP2.Ve=OK). La tâche TD a donc effectivement couté

20k€ et peut être qualifiée de *vérifiée* si les contraintes projet inhérentes aux sous-projets sont modifiées.

Règle 19 : Non vérification d'une tâche TD en fonction de la vérification de ses sous-projets

$$\forall i \in [1..p], (\exists SP_i.Ve = KO) \Rightarrow (TD.Ve = KO)$$

Tant que la tâche TE n'est pas terminée, nous considérons qu'il est impossible de planifier la tâche TD en raison du manque d'informations concernant les exigences système. Par conséquent, si la tâche TE est dans l'état *non déterminé*, la faisabilité de la tâche TD reste à l'état *non déterminé*. Ceci permet de formaliser la règle 20.

Règle 20 : Faisabilité d'une tâche TD en fonction de la vérification de la tâche TE

$$(TE.Ve = ND) \Rightarrow (TD.Fa = ND)$$

La figure 3.11 présente le diagramme états-transitions pour l'attribut de vérification d'une tâche TD. L'état initial 4/5 correspond aux états 4 ou 5 de la figure 3.10. Nous représentons également l'état 1 correspondant l'état 1 de la figure 3.10 et pour lequel l'ensemble des attributs des tâches TE et TD sont dans l'état *non déterminé*. Ces états correspondent à une tâche TE *faisable* (TE.Fa = OK), *vérifiée* (TE.Ve = OK) ou *non vérifiée* (TE.Ve = KO). Partant de l'état 4/5, suite à la planification de la tâche TD, celle-ci peut être jugée *faisable* (transition g, état 6, fig. 3.11) ou *infaisable* (transition h, état 7, fig. 3.11) selon que les contraintes projet inhérentes à la tâche TD sont respectées ou non. Partant de l'état 6, suite à la réalisation de la tâche TD, celle-ci peut passer dans l'état *vérifié* (transition j, état 8, fig. 3.11) ou *non vérifié* (transition k, état 9, fig. 3.11). Partant de l'état 7 où la tâche TD est *infaisable*, il est possible :

- soit de remettre en cause localement les contraintes inhérentes à la tâche TD (transition i1, état 4/5, fig. 3.11). La faisabilité de la tâche TD repasse à l'état *non déterminé* ;
- soit de remettre en cause les contraintes projet au niveau global (transition i2, état 1, fig. 3.11). L'ensemble des attributs de la tâche TD et ceux de la tâche TE repassent à l'état *non déterminé* (état 1, fig. 3.11).

À partir de l'état 9 où la tâche TD est jugée *non vérifiée*, il est possible :

- soit de remettre en cause localement les contraintes inhérentes à la tâche TD (transition l1, fig. 3.11). Les attributs de faisabilité et de vérification de la tâche TD repassent à l'état *non déterminé* (état 4/5, fig. 3.11). Une replanification, permettant par exemple d'allonger la durée de la tâche TD, pourra être engagée par le responsable de planification ;
- soit de remettre en cause les contraintes projet au niveau global (transition l2, fig. 3.11). L'ensemble des attributs de la tâche TD et de la tâche TE repassent à l'état *non déterminé* (état 1, fig. 3.11).

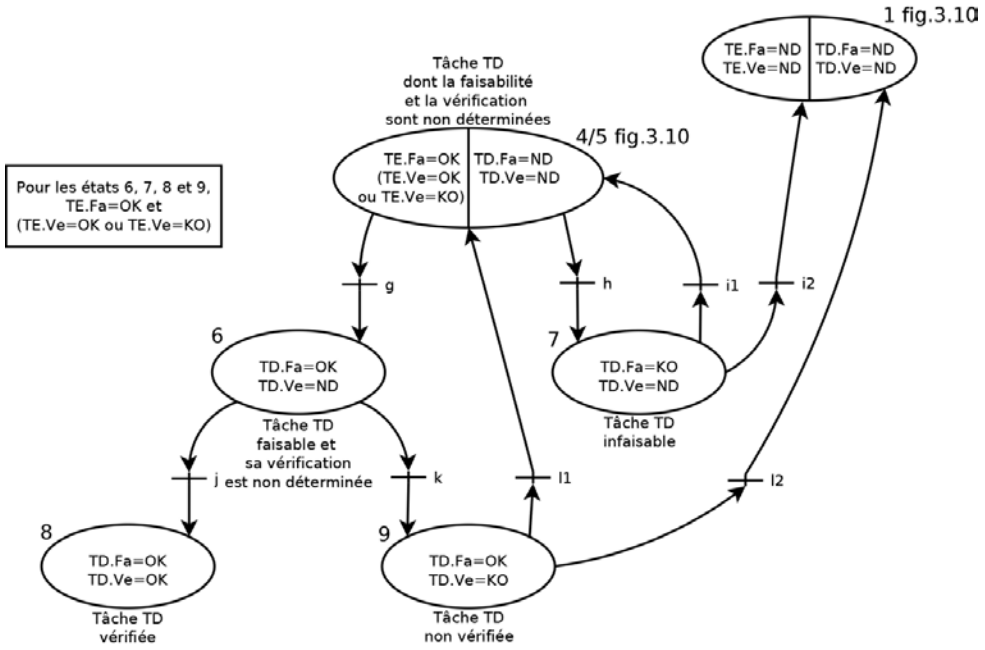


Figure 3.11 : Diagramme d'états-transitions pour l'attribut de vérification d'une tâche TD

(12) Vérification des entités projet de conception : P.Ve

D'après la section 2.1, une entité Projet de conception est composée d'une tâche TE et d'une ou plusieurs tâches TD. Nous considérons donc que la vérification d'un projet de conception P.Ve est déduite à partir des états de la vérification des tâches TE et TD qui le composent.

Dès le début de la conception, les tâches TE et TD ont leur attribut de vérification dans l'état *non déterminé,* la vérification du projet de conception est donc également *non déterminée* (règle 21). Tant qu'il n'existe pas de tâches TE et TD qualifiées de *vérifiées* et que toutes les tâches ne sont pas qualifiées de *non vérifiées*, aucune solution n'est en mesure de répondre aux contraintes de planification. L'attribut de vérification d'un projet de conception est donc, dès la création de ce dernier, *non déterminé*.

Règle 21 : Vérification d'un projet de conception à sa création

$$\forall i \in [1..n], (TE.Ve = ND) \wedge (\forall TD_i.Ve = ND) \Rightarrow (\mathrm{P.Ve} = \mathrm{ND})$$

L'attribut de vérification de l'entité Projet de conception reste dans l'état *non déterminé* tant que :

- la vérification de la tâche TE est qualifiée de *non déterminée* $(TE.Ve = ND)$ et,
- aucune tâche TD n'est qualifiée de *vérifiée* et toutes les tâches TD ne sont pas qualifiées de *non vérifiées*, c'est-à-dire qu'aucun planning correct n'a pour l'instant été produit, mais rien ne permet d'affirmer qu'il n'en existe pas : $\forall i \in [1..n], (\nexists TD_i.Ve = OK) \wedge (\exists TD_i.Ve \neq KO)$.

Les deux propositions présentées ci-dessus sont reprises dans la règle 22.

Règle 22 : Vérification d'un Projet de conception *non déterminé*

$$\forall i \in [1..n], (TE.Ve = ND) \wedge (\nexists TD_i.Ve = OK) \wedge (\exists TD_i.Ve \neq KO) \Rightarrow (P.Ve = ND)$$

Un projet de conception qualifié de *vérifié* signifie que le projet de conception est terminé et que la tâche TE et au moins une tâche TD ont respecté les contraintes qui leur étaient assignées. Un projet de conception est donc vérifié si :

- la tâche TE est qualifiée de *vérifiée* : $(TE.Ve = OK)$ et,
- au moins une tâche TD est qualifiée de *vérifiée* : $\forall i \in [1..n], (\exists TD_i.Ve = OK)$.

Ceci nous permet de formaliser la règle 23.

Règle 23 : Vérification d'un projet de conception *vérifié*

$$\forall i \in [1..n], (TE.Ve = OK) \wedge (\exists TD_i.Ve = OK) \Rightarrow (P.Ve = OK)$$

A contrario, nous considérons qu'un projet de conception est qualifié de *non vérifié* si l'ensemble de ses tâches est qualifié de *non vérifié.* Un projet de conception est donc qualifié de *non vérifié* si :

- la tâche TE est qualifiée de *non vérifiée* : $(TE.Ve = KO)$ et,
- toutes les tâches TD sont qualifiées de *non vérifiées* : $\forall i \in [1..n], (\forall TD_i.Ve = KO)$.

Cette connaissance méthodologique est formalisée dans la règle 24.

Règle 24 : Vérification d'un projet de conception *non vérifié*

$$\forall i \in [1..n], (TE.Ve = KO) \wedge (\forall TD_i.Ve = KO) \Rightarrow (P.Ve = KO)$$

Nous précisons toutefois qu'il est possible d'envisager que certaines tâches puissent être qualifiées de *non vérifiées* tout en présentant une configuration de projet de conception telle que les contraintes globales du projet de conception soient respectées, dans ce cas, la répartition des contraintes entre chaque tâche a été mal réalisée et il est possible de se ramener à un projet de conception vérifié en modifiant les contraintes de chaque tâche. Par exemple, considérons un projet de conception dont le coût global ne doit pas dépasser 30k€. Ce projet de conception se compose d'une tâche TE et d'une tâche TD qui ont un coût prévisionnel de 20k€ et 10k€ respectivement. Les coûts engagés réellement pour chaque tâche sont respectivement de 21k€ (TE.Ve=KO) et 9k€ (TD.Ve=OK). Selon la règle 24, le projet de conception est donc qualifié de *non vérifié.* Cependant, le projet de conception a effectivement couté globalement 30k€, il satisfait donc la contrainte de coût qui lui était assignée et peut être qualifié de *vérifié* si les contraintes projet des tâches TE et TD sont remises en cause. C'est-à-dire que la contrainte sur le coût de la tâche TE passe à 21k€ et celle de la tâche TD à 9k€.

3.1.4 Synthèse sur la faisabilité et la vérification des entités de conception et de planification

Nous venons ainsi de proposer des définitions de la faisabilité et de la vérification et de les décliner selon les entités de conception (Système, Exigences système, Alternative système) et les entités de planification (Projet de conception, Tâche TE, Tâche TD). Des connaissances méthodologiques ont ainsi pu être déterminées sous forme de règles sur l'évolution des états des attributs de faisabilité et de vérification. L'évolution des attributs au sein de chaque environnement est illustrée sur la figure 3.12. Cependant, le couplage des entités de conception/entités de planification doit également être abordé. Nous proposons de formaliser les **connaissances méthodologiques de couplage** dans la section suivante.

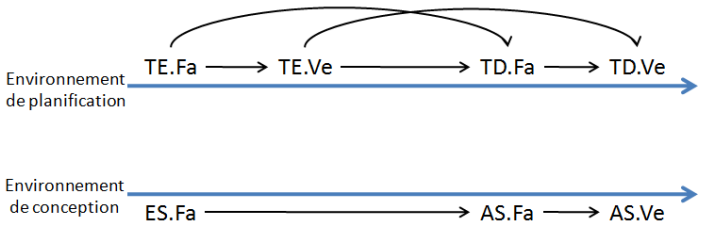


Figure 3.12 : Evolution de chaque attribut au sein d'un même environnement

3.1.5 Synchronisation des attributs

Nous avons vu, dans les sections précédentes, que chaque entité devait être analysée en termes de faisabilité et de vérification. Des règles ont été proposées afin de garantir la cohérence dans la définition des différentes entités de conception et de planification. Ces règles sont, soit propres à la conception, soit propres au projet.

Toute entité de planification couplée avec une entité de conception peut être concernée par le couplage informationnel. Considérons un projet P, une tâche TE et une tâche de développement d'une alternative système TD, couplés respectivement à un système S, des exigences système ES et une alternative système AS. Chaque entité disposant de ces deux attributs de faisabilité et de vérification (à l'exception des exigences système qui ne disposent que de l'attribut de faisabilité), cinq attributs sur les entités de conception et six attributs sur les entités de planification offrent 30 possibilités de couplage, c'est-à-dire trente possibilités de synchronisation de leur changement d'état. Chaque attribut ayant trois états possibles et quatre possibilités de changement d'état, le nombre de synchronisations dans les changements d'états devient très important. Face à cette complexité, nous proposons d'étudier le changement d'états des attributs de faisabilité et de vérification des tâches TE et TD d'un projet et ceux des attributs de faisabilité des exigences système ES et des attributs de faisabilité et de vérification d'une alternative système AS. La synchronisation étudiée concerne :

- le changement d'état de l'attribut de faisabilité des exigences système (ES.Fa) avec le changement d'état des attributs de faisabilité (TE.Fa) et de vérification de la tâche TE (TE.Ve),
- le changement d'état des attributs de faisabilité (AS.Fa) et de vérification (AS.Ve) d'une alternative système avec le changement d'état des attributs de faisabilité (TD.Fa) et de vérification (TD.Ve) de la tâche TD.

Les autres possibilités de couplage informationnel ne nécessitent pas nécessairement de formalisation dans la mesure où, dans chaque environnement de conception ou de planification, des précédences existent et ont été formalisées dans les sections précédentes. La figure 3.13 ci-dessous illustre le couplage informationnel envisagé ainsi que les précédences nécessaires pour les changements d'état des attributs concernés. Les flèches orientées représentent les contraintes de précédence existant dans le changement d'état des attributs. Nous proposons ensuite de détailler indépendamment les couplages informationnels numérotés 1 et 2 sur la figure 3.13 respectivement dans les sections 3.1.4.1 et 3.1.4.2.

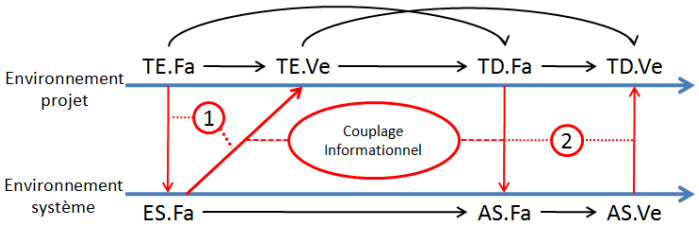


Figure 3.13 : Propositions de couplages informationnels

3.1.5.1 *Synchronisation des exigences système ES avec la tâche TE*

Nous abordons, dans un premier temps, le lien de couplage informationnel (noté ① dans la figure 3.13) concernant les changements d'état des attributs de faisabilité et de vérification de la tâche TE et l'attribut de faisabilité des exigences système ES.

D'après le processus intégré de conception / planification proposé dans la section 2.3, il apparait que le recueil des exigences ne peut débuter que dans la mesure où le responsable de conception a, au préalable, reçu un planning : la tâche TE doit donc être qualifiée de *faisable* avant que les exigences système puissent être recueillies et qualifiées à leur tour de *faisables*.

Nous avons défini précédemment que si les exigences système n'ont pas été qualifiées de *faisables* ou d'*infaisables*, cela signifie qu'elles n'ont pas été totalement recueillies (cf. section 3.1.2.1). Dans ce cas, la tâche TE n'est pas terminée. A contrario, si les exigences système ont été qualifiées de *faisables* ou d'*infaisables*, cela signifie en pratique, que la tâche TE est terminée.

La figure 3.14 illustre les différents états d'attributs possibles au cours de la conception de système et de la planification de projet ainsi que leurs évolutions. Nous détaillons la signification de chaque état et de chaque changement d'état.

Etat 1 : Le système et le projet de conception viennent d'être créés mais ne sont pas encore définis. De fait, la tâche TE ainsi que les exigences système ES créées automatiquement ne sont pas non plus définies. La faisabilité des exigences système ainsi que la faisabilité et la vérification de la tâche TE sont donc à l'état *non déterminé* (ES.Fa=ND, TE.Fa=ND et TE.Ve=ND). Il est nécessaire, afin de poursuivre le projet de conception, de planifier la tâche TE en respectant les contraintes projet avant de procéder effectivement au recueil des exigences système. Nous proposons donc la règle 25 représentant une connaissance méthodologique de couplage : tant que la tâche TE n'est pas qualifiée de *faisable*, les exigences système ES ne peuvent pas être qualifiées de *faisables* ou d'*infaisables*.

Règle 25 : Synchronisation de la faisabilité d'une tâche TE avec la faisabilité des exigences système ES associées

$$(TE.Fa \neq OK) \Rightarrow (ES.Fa = ND)$$

Il est donc possible, après planification de la tâche TE, de faire passer l'attribut de faisabilité de la tâche TE à *faisable* (transition t1 – état 2, fig. 3.14) si la planification respecte les contraintes projet ou à *infaisable* (transition t2 – état 3, fig. 3.14) dans le cas contraire.

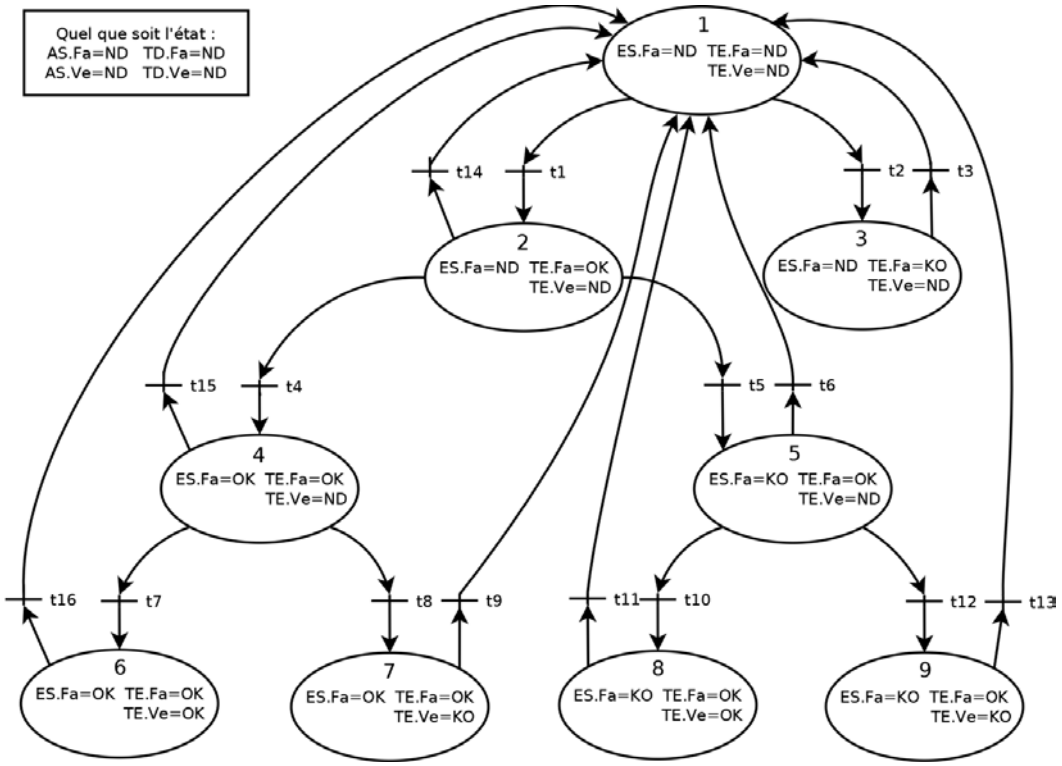


Figure 3.14 : Diagramme d'états-transitions pour les attributs de faisabilité des exigences système ES, de faisabilité et de vérification de la tâche TE associée

Etat 2 : La faisabilité de la tâche TE est qualifiée de *faisable* et les autres attributs sont *non déterminés* (ES.Fa=ND, TE.Fa=OK, TE.Ve=ND). Ici, une planification réaliste de la tâche TE a été produite, le travail du responsable de conception peut débuter dans ce cadre et un suivi peut être effectué. Il est nécessaire, afin de poursuivre le projet de conception, que le responsable de conception se soit prononcé sur la faisabilité des exigences système avant de procéder à la clôture de la tâche TE et donc à sa vérification. Nous proposons donc la règle 26 : tant que les exigences système ES ne sont pas qualifiées de *faisables* ou d'*infaisables*, la tâche TE ne peut pas être qualifiée de *vérifiée* ou *non vérifiée*.

Règle 26 : Synchronisation de la faisabilité des exigences système ES avec la vérification de la tâche TE associée

$$(ES.Fa = ND) \Rightarrow (TE.Ve = ND)$$

Quand les exigences système ont été effectivement recueillies et que le responsable de conception s'est prononcé sur la capacité de l'équipe de concepteurs à concevoir un système répondant aux exigences système, il peut faire passer l'attribut de faisabilité des exigences système à *faisable* (transition t4 – état 4, fig. 3.14) ou à *infaisable* (transition t5 – état 5, fig. 3.14).

Etat 3 : La faisabilité de la tâche TE est qualifiée d'*infaisable* et les autres attributs sont *non déterminés* (ES.Fa=ND, TE.Fa=KO, TE.Ve=ND). Ici, aucune planification réaliste de la tâche TE n'a été produite et, par conséquent, le projet de conception est « bloqué ». Pour sortir de cet état, il est possible de renégocier les contraintes projet (transition t3 – état 1, fig. 3.14) et d'essayer de replanifier.

Etat 4 : La faisabilité de la tâche TE est qualifiée de *faisable* et la vérification de la tâche TE est *non déterminée,* la faisabilité des exigences système est qualifiée de *faisable* (ES.Fa=OK, TE.Fa=OK, TE.Ve=ND, ES.Fa=OK). Dans ce cas, le travail du responsable de conception est terminé concernant les exigences système. Il est donc en attente de la clôture de la tâche TE. Il est donc

possible, après que le responsable de planification ait vérifié que les contraintes projet ont été respectées, de faire passer l'attribut de vérification de la tâche TE à *vérifié* (transition t7 – état 6, fig. 3.14) si c'est le cas ou à *non vérifié* (transition t8 – état 7, fig. 3.14) dans le cas contraire.

Etat 5 : La faisabilité de la tâche TE est qualifiée de *faisable*, la vérification de la tâche TE est *non déterminée* et la faisabilité des exigences système est qualifiée d'*infaisable* (ES.Fa=KO, TE.Fa=OK, TE.Ve=ND). Deux cas de figure peuvent se présenter :

- soit le responsable de conception négocie les exigences du client (transition t6 – état 1, fig. 3.14), l'ensemble des attributs revenant à l'état *non déterminé*,
- soit il ne peut pas renégocier les exigences du client, et il est donc en attente de la clôture de la tâche TE. Il est donc possible, après que le responsable de planification ait vérifié que les contraintes projet ont été respectées, de faire passer l'attribut de vérification de la tâche TE à *vérifié* (transition t10 – état 8, fig. 3.14) si c'est le cas ou à *non vérifié* (transition t12 – état 9, fig. 3.14) dans le cas contraire.

Etat 6 : La faisabilité de la tâche TE est qualifiée de *faisable,* la vérification de la tâche TE est qualifiée de *vérifiée* et la faisabilité des exigences système est qualifiée de *faisable* (ES.Fa=OK, TE.Fa=OK, TE.Ve=OK). Ici, le recueil des exigences s'est bien déroulé tant au niveau des alternatives qui vont être exploitées par la suite qu'au niveau de la planification. Le projet peut se poursuivre en actant le bon déroulement de la tâche TE.

Etat 7 : La faisabilité de la tâche TE est qualifiée de *faisable,* la vérification de la tâche TE est qualifiée de *non vérifiée* et la faisabilité des exigences système est qualifiée de *faisable* (ES.Fa=OK, TE.Fa=OK, TE.Ve=KO). Ici, le recueil des exigences s'est bien déroulé et des alternatives à exploiter sont envisagées mais les contraintes liées au projet n'ont pas été respectées. Pour sortir de cet état, la seule possibilité est de remettre en cause les exigences système et les contraintes projet (transition t9 – état 1, fig. 3.14). L'ensemble des attributs repasse à l'état *non déterminé.*

Si cette possibilité n'est pas envisageable, la non vérification de la tâche TE est actée. Néanmoins, le projet peut se poursuivre et tenant compte des contraintes de la tâche TE non respectées.

Etat 8 : La faisabilité de la tâche TE est qualifiée de *faisable*, la vérification de la tâche TE est qualifiée de *vérifiée* et la faisabilité des exigences système est qualifiée d'*infaisable* (ES.Fa=KO, TE.Fa=OK, TE.Ve=OK). Ici, le recueil des exigences n'aboutit pas à une solution, c'est-à-dire qu'aucune alternative n'est envisagée pour répondre aux exigences système ou des contradictions existent entre exigences. Cependant, les contraintes projet ont été respectées. Le projet ne peut donc pas se poursuivre si les responsables de conception et de planification ne remettent pas en cause les exigences système, ce qui implique de retourner à l'état initial (transition t11 – état 1, fig. 3.14). L'ensemble des attributs repassent à l'état *non déterminé.*

Etat 9 : La faisabilité de la tâche TE est qualifiée de *faisable*, la vérification de la tâche TE est qualifiée de *non vérifiée* et la faisabilité des exigences système est qualifiée d'*infaisable* (ES.Fa=KO, TE.Fa=OK, TE.Ve=KO). Ici, le recueil des exigences n'aboutit pas à une solution, c'est-à-dire qu'aucune alternative n'est envisagée pour répondre aux exigences système. De plus, les contraintes projet n'ont pas été respectées. Pour sortir de cet état, il est nécessaire de remettre en cause les exigences système et les contraintes projet et donc de retourner à l'état initial (transition t13 – état 1, fig. 3.14). Si la remise en cause des exigences système n'est pas possible, la non vérification de la tâche TE et des exigences système est actée et le projet est donc arrêté à ce stade de la conception.

À partir des états 2, 4 et 6, il est possible de remettre en cause les exigences système et/ou les contraintes projet. Dans ce cas, tous les attributs reviennent à l'état *non déterminé* (transitions t14, t15 et t16 – état 1, fig. 3.14) et le processus intégré peut recommencer.

3.1.5.2 *Synchronisation d'une alternative système AS avec une tâche TD*

Nous abordons dans cette section le lien de couplage informationnel (noté ② dans la figure 3.13) concernant les changements d'état des attributs de faisabilité et de vérification de la tâche TD et de l'alternative système AS associée.

Nous avons défini précédemment que si une tâche TD n'est pas qualifiée de *faisable*, le responsable de conception ne peut pas recevoir de planning et donc ne peut pas débuter la conception de l'alternative système AS associée : cette dernière ne peut donc pas être qualifiée de *faisable*. De plus, si une alternative système n'a pas été qualifiée de *vérifiée* ou *non vérifiée*, cela signifie qu'elle n'est pas totalement achevée (cf. section 3.1.2.2). Dans ce cas, la tâche TD n'est pas terminée. A contrario, si l'alternative système est qualifiée de *vérifiée* ou *non vérifiée*, cela signifie en pratique que la tâche TD est terminée.

Le diagramme d'états-transitions de la figure 3.15 ci-dessous correspond à la suite du diagramme d'états-transitions de la figure 3.14. À partir des états 6 ou 7, fig. 3.14 où les exigences système ES sont faisables (ES.Fa=OK) et la tâche TE est terminée (TE.Ve=OK ou TE.Ve=KO), nous décrivons les évolutions des attributs de faisabilité et de vérification d'une alternative système AS et d'une tâche TD associée. Nous représentons également l'état 1 de la figure 3.14 correspondant à la situation initiale du projet où l'ensemble des attributs est *non déterminé*.

Etat 6/7 : L'alternative système AS et la tâche TD viennent d'être créées mais ne sont pas encore définies. Les attributs de faisabilité et de vérification de la tâche TD et de l'alternative système AS sont donc dans l'état *non déterminé* (AS.Fa = ND, AS.Ve = ND, TD.Fa = ND et TD.Ve = ND). Il est nécessaire, afin de débuter le développement de l'alternative système AS, de planifier la tâche TD en respectant les contraintes projet avant de procéder effectivement au développement de l'alternative. Nous proposons la règle 27 comme connaissance de couplage méthodologique : tant que la tâche TD n'est pas qualifiée de *faisable*, l'alternative système ne peut pas être qualifiée de *faisable* ni d'*infaisable*.

Règle 27 : Synchronisation de la faisabilité d'une tâche TD avec la faisabilité de l'alternative système AS associée

$$(TD.Fa \neq OK) \Rightarrow (AS.Fa = ND)$$

Il est donc possible, après planification de la tâche TD, de faire passer son attribut de faisabilité à *faisable* (transition t17 – état 10, fig. 3.15) si la planification respecte les contraintes projet ou à *infaisable* (transition t18 – état 11, fig. 3.15) dans le cas contraire.

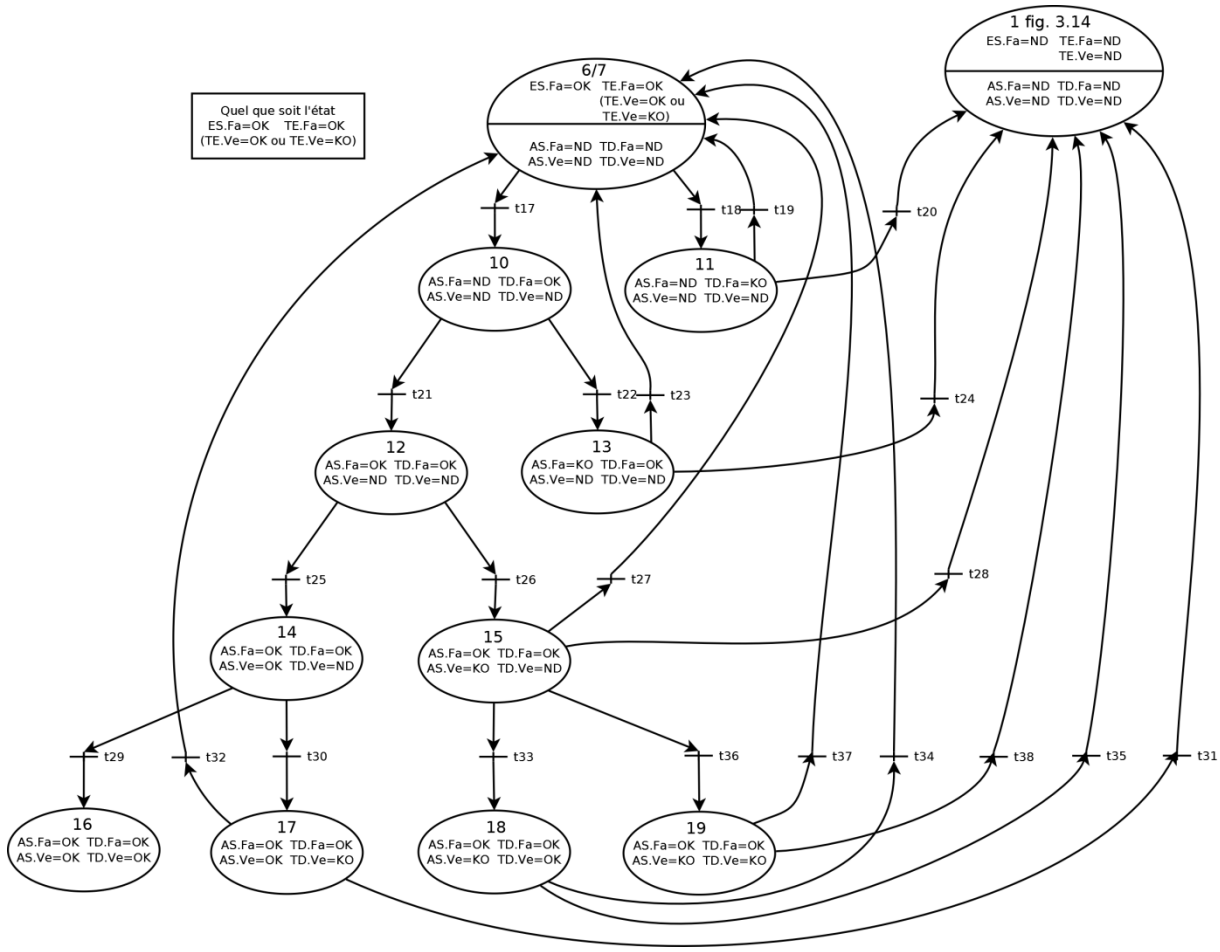


Figure 3.15 : Diagramme d'états-transitions pour les attributs de faisabilité et de vérification d'une alternative système AS et de faisabilité et de vérification de la tâche TD associée

Etat 10 : La tâche TD est qualifiée de *faisable,* les autres attributs sont *non déterminés* (AS.Fa = ND, AS.Ve = ND, TD.Fa = OK et TD.Ve = ND). Un planning de développement de l'alternative a été produit et ce dernier satisfait les contraintes projet. Toutefois, le responsable de conception ne sait pas, pour l'instant, si la solution logique étudiée peut mener à une solution physique correcte. Il est nécessaire, afin de poursuivre le développement de l'alternative, que le responsable de conception se soit prononcé sur la faisabilité de l'alternative système AS avant le développement de la solution physique qui pourra, par ailleurs, être vérifiée. Ceci a déjà fait l'objet de la règle 7 (voir section 3.1.2.2). Quand la solution logique de l'alternative système AS est produite et que le responsable de conception s'est prononcé sur la capacité de l'équipe de concepteurs à concevoir la solution physique correspondante, il peut faire passer l'attribut de faisabilité de l'alternative système à *faisable* (transition t21 – état 12, fig. 3.15) s'il pense être capable de concevoir ou à *infaisable* (transition t22 – état 13, fig. 3.15) dans le cas contraire.

Etat 11 : La tâche TD est qualifiée d'*infaisable,* les autres attributs sont *non déterminés* (AS.Fa = ND, AS.Ve = ND, TD.Fa = KO et TD.Ve = ND). Aucun planning de développement de l'alternative n'a pu être produit conformément aux contraintes projet. Pour sortir de cet état, il est possible :

- soit de renégocier les contraintes projet globales. Cette négociation implique que l'ensemble des attributs repasse à l'état *non déterminé* (transition t20 – état 1, fig. 3.14)

- soit de modifier uniquement les contraintes projet inhérentes à la tâche TD. Dans ce cas, seuls les attributs de la tâche TD et de l'alternative système AS reviennent à l'état *non déterminé* (transition t19 – état 6/7, fig. 3.15). Le responsable de planification doit ensuite replanifier la tâche TD en tenant compte des modifications.

Etat 12 : La tâche TD est qualifiée de *faisable,* l'alternative système est qualifiée de *faisable* et les autres attributs sont *non déterminés* (AS.Fa = OK, AS.Ve = ND, TD.Fa = OK et TD.Ve = ND). Ici, un planning de développement de l'alternative a été produit et ce dernier satisfait les contraintes projet et la solution logique envisagée pour l'alternative système semble pouvoir conduire à une solution physique satisfaisant les exigences système. Il est nécessaire, afin de poursuivre le développement de l'alternative, que le responsable de conception se soit prononcé sur la vérification de l'alternative système avant de procéder à la clôture de la tâche TD et donc à sa vérification. Nous proposons donc la règle 28 : tant que l'alternative système AS n'est pas qualifiée de *vérifiée* ou *non vérifiée*, la tâche TD ne peut pas être qualifiée de *vérifiée* ou *non vérifiée*.

Règle 28 : Synchronisation de la vérification de l'alternative système AS avec la vérification de la tâche TD associée

$$(AS.Ve = ND) \Rightarrow (TD.Ve = ND)$$

Quand la solution physique de l'alternative système AS est effectivement développée et que le responsable de conception a vérifié qu'elle répond bien aux exigences système, il peut faire passer l'attribut de vérification de l'alternative système AS à *vérifié* (transition t25 – état 14, fig. 3.15) si toutes les exigences système sont satisfaites ou à *non vérifié* (transition t26 – état 15, fig. 3.15) dans le cas contraire.

Etat 13 : La tâche TD est qualifiée de *faisable,* l'alternative système est qualifiée d'*infaisable* et les autres attributs sont *non déterminés* (AS.Fa = KO, AS.Ve = ND, TD.Fa = OK et TD.Ve = ND,). Ici, un planning de développement de l'alternative a été produit et ce dernier satisfait les contraintes projet, cependant, la solution logique proposée dans l'alternative système ne semble pas pouvoir déboucher vers une solution physique satisfaisant les exigences système. Pour sortir de cet état, il est possible :

- soit de renégocier les exigences système. Cette négociation implique que l'ensemble des attributs revient à l'état *non déterminé* (transition t24 – état 1, fig. 3.15) ;
- soit de modifier l'alternative système AS seule et donc la tâche TD associée, sans modifier les exigences système ES. Dans ce cas, seuls les attributs de la tâche TD et de l'alternative système repassent à l'état *non déterminé* (transition t23 – état 6/7, fig. 3.15)

Etat 14 : La tâche TD est *faisable* et sa vérification est *non déterminée,* l'alternative système est qualifiée de *faisable* et *vérifiée*, (AS.Fa = OK, AS.Ve = OK, TD.Fa = OK et TD.Ve = ND). Un planning de développement de l'alternative système a été produit et ce dernier satisfait les contraintes projet, la solution logique envisagée pour l'alternative système semble pouvoir conduire à une solution physique satisfaisant les exigences système et cette solution physique les satisfait effectivement. La conception étant techniquement terminée, le responsable de conception a fait passer l'attribut de vérification de l'alternative système AS à *vérifié.* Il est ensuite en attente de la clôture de la tâche TD par le responsable de planification. Le responsable de planification peut, après vérification que la tâche a bien respecté les contraintes projet, faire passer l'attribut de vérification de la tâche TD à *vérifié* (transition t29 – état 16, fig. 3.15) si toutes les contraintes projet sont satisfaites ou à *non vérifié* (transition t30 – état 17, fig. 3.15) dans le cas contraire.

Etat 15 : La tâche TD est *faisable* et sa vérification est *non déterminée* et l'alternative système est qualifiée de *faisable* et *non vérifiée* (AS.Fa = OK, AS.Ve = KO, TD.Fa = OK et TD.Ve = ND). Un planning de développement de l'alternative a été produit et ce dernier satisfait les contraintes projet, la solution logique envisagée pour l'alternative système semble pouvoir conduire à une solution physique satisfaisant les exigences système, cependant, la solution physique obtenue ne satisfait pas totalement les exigences système. Trois cas de figure peuvent se présenter :

- soit le responsable de conception négocie les exigences systèmes (transition t28 – état 1, fig. 3.15). L'ensemble des attributs revient à l'état *non déterminé*. Le processus complet doit être réitéré ;
- soit ce n'est pas possible et il souhaite modifier l'alternative système. Il envoie donc une demande au responsable de planification afin d'obtenir un délai et/ou des ressources supplémentaires pour réaliser cette modification. Ainsi, les attributs de la tâche TD et de l'alternative système repassent à l'état *non déterminé* (transition t27 – état 6/7, fig. 3.15) ;
- soit aucun des deux cas ci-dessus n'est envisagé. Le responsable de conception attend la clôture de la tâche TD. Le responsable de planification vérifie que les contraintes projet inhérentes à la tâche TD ont été respectées. L'attribut de vérification de la tâche TD passe à *vérifié* (transition t33 – état 18, fig. 3.15) si c'est le cas ou à *non vérifié* (transition t36 – état 19, fig. 3.15) dans le cas contraire.

Etat 16 : La tâche TD est *faisable* et *vérifiée*, l'alternative système est qualifiée de *faisable* et *vérifiée*, (AS.Fa = OK, AS.Ve = OK, TD.Fa = OK et TD.Ve = OK). Un planning de développement de l'alternative a été produit et ce dernier satisfait les contraintes projet. La solution logique envisagée pour l'alternative système semblait a priori pouvoir conduire à une solution physique satisfaisant les exigences système et cette solution physique, une fois obtenue, les satisfait effectivement. Enfin, la réalisation effective de la tâche TD s'est déroulée conformément aux contraintes projet.

Etat 17 : La tâche TD est *faisable* et *non vérifiée,* l'alternative système est qualifiée de *faisable* et *vérifiée*, (AS.Fa = OK, AS.Ve = OK, TD.Fa = OK et TD.Ve = KO). Un planning de développement de l'alternative a été produit et ce dernier satisfait totalement les contraintes projet cependant, la réalisation effective de la tâche TD ne s'est pas déroulée conformément aux attentes. D'autre part, la solution logique envisagée pour l'alternative système semblait a priori pouvoir conduire à une solution physique. Une fois obtenue, cette solution satisfait effectivement les exigences système. Pour sortir de cet état, il existe deux possibilités :

- soit renégocier les contraintes projet globales. Cette négociation implique que l'ensemble des attributs repasse à l'état *non déterminé* (transition t31– état 1, fig. 3.15),
- soit de modifier uniquement les contraintes projet inhérentes à la tâche TD. Dans ce cas, seuls les attributs de la tâche TD et de l'alternative système AS reviennent à l'état *non déterminé* (transition t32 – état 6/7, fig. 3.15). Le responsable de planification doit ensuite replanifier la tâche TD en tenant compte des modifications.

Si aucune des deux possibilités ci-dessus n'est envisageable, la non vérification de la tâche TD est actée. Néanmoins, le projet peut se poursuivre et tenant compte du non respect des contraintes projet lors de la tâche TD.

Etat 18 : La tâche TD est *faisable* et *vérifiée,* l'alternative système est qualifiée de *faisable* et *non vérifiée*, (AS.Fa = OK, AS.Ve = KO, TD.Fa = OK et TD.Ve = OK). Un planning de développement de l'alternative a été produit et ce dernier satisfait les contraintes projet. La réalisation

effective de la tâche TD s'est déroulée conformément aux contraintes projet. La solution logique envisagée pour l'alternative système semblait a priori pouvoir conduire à une solution physique satisfaisant les exigences système. Cependant, cette solution, une fois obtenue, ne satisfait pas totalement les exigences système. En pratique, il n'y a donc pas de système répondant aux besoins du client. Pour sortir de cet état, il existe deux possibilités :

- soit il est possible de remettre en cause les exigences système et donc de retourner à l'état initial (transition t35 – état 1, fig. 3.15) ;
- soit la remise en cause ne concerne que l'alternative système et la tâche TD. Leurs attributs repassent à l'état *non déterminé* (transition t34 – état 6/7, fig. 3.15). Un délai et/ou des ressources supplémentaires peuvent alors être planifiés par le responsable de planification. La poursuite, dans le nouveau cadre défini par la planification, doit permettre d'obtenir une alternative système AS répondant cette fois-ci aux exigences client.

Etat 19 : La tâche TD est *faisable* et *non vérifiée,* l'alternative système est qualifiée de *faisable* et *non vérifiée*, (AS.Fa = OK, AS.Ve = KO, TD.Fa = OK et TD.Ve = KO). Un planning de développement de l'alternative a été produit et ce dernier satisfait les contraintes projet. La solution logique envisagée pour l'alternative système semblait a priori pouvoir conduire à une solution physique satisfaisant les exigences système. Cette solution, une fois obtenue, ne satisfait pas totalement les exigences système. De plus, la réalisation effective de la tâche TD ne s'est pas déroulée conformément aux contraintes projet. Pour sortir de cet état, il existe deux possibilités :

- soit il est possible de remettre en cause les exigences système et les contraintes projet et donc de retourner à l'état initial (transition t38 – état 1, fig. 3.15) ;
- soit la remise en cause ne concerne que l'alternative système et la tâche TD. Leurs attributs repassent à l'état *non déterminé* (transition t37, fig. 3.15). La planification d'un délai et/ou de ressources supplémentaires pour le développement de l'alternative système AS doit permettre d'obtenir une tâche TD et d'une alternative système AS *vérifiées*.

Si aucune des deux possibilités ci-dessus n'est envisageable, la non vérification de la tâche TD et de l'alternative système est actée. S'il n'y a qu'une seule alternative, le projet est arrêté à ce stade de la conception.

À partir des états 10, 12, 14, 16, il est possible de remettre en cause les exigences système et/ou les contraintes projet. Dans ce cas, tous les attributs reviennent à l'état *non déterminé* (état 1, fig. 3.15) et le processus est réitéré entièrement. De même, il est possible de ne remettre en cause que l'alternative système et la tâche TD, seuls leurs attributs repassant à l'état *non déterminé* (état 6/7, fig. 3.15). Dans un souci de simplification de la lecture du la figure 3.15, ces transitions ne sont par représentées.

3.1.6 Définition du couplage informationnel

Dans la mesure où le couplage structurel garantit une cohérence entre structure système et structure projet, nous proposons dans cette section une définition du couplage informationnel. Il s'agit de garantir la cohérence des évolutions des entités de conception et de celles de planification. Le couplage informationnel permet de s'assurer que les changements d'état des attributs de faisabilité et de vérification des entités de planification et de conception sont déclenchés à des instants opportuns et cohérents. Le couplage informationnel consiste donc en une synchronisation du changement d'état de ces attributs pour chaque entité de conception et de planification couplée structurellement. Afin de le formaliser, des graphes représentant les différents états autorisés sont définis ainsi que 28 règles de

changements d'états autorisés. De plus, un processus basé sur le processus intégré de planification/conception du couplage structurel permet de définir les possibilités d'évolutions et de compléter cette connaissance méthodologique.

D'un point de vue plus opérationnel, le couplage informationnel permet :

- de garantir que les tâches de conception ne débutent que lorsque leur planning a été élaboré dans l'environnement de planification et que ce planning est juge réalisable,
- de garantir que les tâches de conception sont bien terminées avec des artefacts cohérents avant de pouvoir clore les tâches et les projets.

3.2 Evolution du diagramme de classe

Le diagramme de classe proposé dans la section 2.2.3 propose d'apparier :

- système S et projet de conception P,
- exigences système ES et tâche TE et,
- alternative système AS et tâche TD.

Dans cette section, nous proposons d'affiner ce diagramme de classe afin de prendre en compte le couplage informationnel présenté dans la section précédente tel qu'illustré sur la figure 3.16.

Ici, il s'agit d'ajouter un attribut de faisabilité aux classes Système et Projet de conception, Exigences système et Tâche TE et Alternative système et Tâche TD. De plus, nous proposons d'ajouter un attribut de vérification aux classes Système, Projet de conception, Tâche TE, Alternative système et Tâche TD.

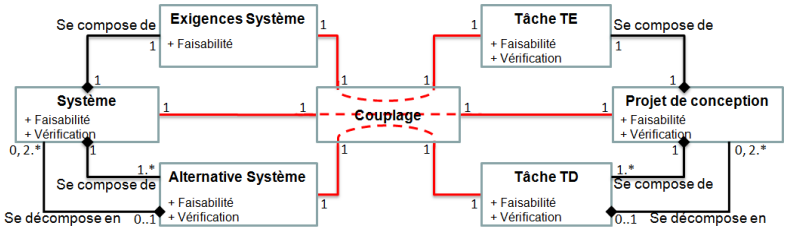


Figure 3.16 : Diagramme de classes supportant les couplages structurel et informationnel

3.3 Processus de mise en œuvre du couplage informationnel

Nous proposons dans cette section d'affiner le processus de conception et de planification proposé dans la section 2.3.2 qui concernait le couplage structurel afin d'y intégrer les principes du couplage informationnel. Dans un premier temps, la synchronisation entre les attributs des exigences système ES et de la tâche TE associée est détaillée dans la section 3.3.1. Puis, la synchronisation entre les attributs d'une alternative système AS et de la tâche TD associée est précisée dans la section 3.3.2.

3.3.1 Processus de synchronisation entre exigences système ES et tâche TE

Nous proposons dans cette section de reprendre le processus de conception et de planification proposé dans la section 2.3 en le complétant afin de prendre en compte la synchronisation des attributs des exigences système ES et de la tâche TE associée. Ce processus est illustré sur la figure 3.17.

Changement d'état de la faisabilité de la tâche TE : suite à la saisie des informations concernant la tâche TE puis à sa planification, le responsable de planification doit se prononcer sur sa faisabilité (activité 1, fig. 3.17). S'il qualifie cette dernière de *faisable* (TE.Fa = OK), il en informe le

directeur de programme et transmet le planning de la tâche TE au responsable de conception qui peut débuter le recueil des exigences système. S'il qualifie la tâche TE d'*infaisable* (TE.Fa = KO), cela signifie qu'il n'est pas en mesure de planifier convenablement la tâche TE au regard des informations saisies au préalable. Un choix est alors possible : soit modifier ces informations et réitérer la planification puis l'analyse de faisabilité de la tâche TE (TE.Fa = ND), soit abandonner la conception en informant le directeur de programme et le responsable de conception.

Changement d'état de la faisabilité des exigences système ES : suite à la réception du planning émis par le responsable de planification, le responsable de conception réalise le recueil des exigences système ainsi que la recherche d'alternatives permettant potentiellement de répondre à ces exigences. Quand ce travail est terminé, il doit se prononcer sur la faisabilité des exigences système (activité 2, fig. 3.17). S'il qualifie ces dernières de *faisables* (ES.Fa = OK), il en informe le directeur de programme et alerte le responsable de planification qui peut alors vérifier la tâche TE. S'il qualifie les exigences système d'*infaisables* (ES.Fa = KO), cela signifie qu'il n'est pas en mesure de développer de solution (alternative système) répondant aux exigences système. Un choix doit être réalisé par le responsable de conception : soit modifier les exigences système, c'est-à-dire poursuivre la tâche TE afin de lever l'infaisabilité (demande de dérogations, relaxation d'exigences, etc.), soit abandonner le développement en laissant l'attribut de faisabilité des exigences système à *infaisable* en alertant le directeur de programme. Dans le premier cas, une demande de replanification de la tâche TE est envoyée au responsable de planification et le responsable de conception attend son nouveau planning pour poursuivre la tâche TE. L'attribut de faisabilité des exigences système ES est mis à l'état *non déterminé* (ES.Fa = ND). Dans tous les cas, le responsable de planification est informé de la faisabilité des exigences système ES.

Changement d'état de la vérification de la tâche TE : à la fin de la tâche TE, le responsable de planification doit vérifier que les contraintes projet assignées à cette dernière ont bien été respectées (activité 3, fig. 3.17). Dans tous les cas, le directeur de programme est informé. Si les exigences système sont *infaisables* (ES.Fa = KO) et que le souhait de leur replanification est émis pour le responsable de planification suite à la demande du responsable de conception (TE.Fa = ND et TE.Ve = ND), le responsable de conception en est alerté, remet l'attribut de faisabilité des exigences système ES à non déterminé (ES.Fa = ND) et attend son nouveau planning pour la tâche TE. Le responsable de planification modifie les contraintes projet de la tâche TE et réitère la planification (activité 1, fig. 3.17).

Même si les exigences système sont jugées faisables pour le responsable de conception (ES.Fa = OK), une décision de replanifier la tâche TE peut être prise (TE.Fa = ND et TE.Ve = ND). Dans ce cas, le responsable de conception remet l'attribut de faisabilité des exigences système ES à *non déterminé* (ES.Fa = ND) et attend son nouveau planning pour la tâche TE.

3.3.2 Processus de synchronisation entre alternative système AS et tâche TD

Nous abordons dans cette section le processus de synchronisation entre les attributs de faisabilité et de vérification d'une tâche TD et de l'alternative système AS associée. Ce processus est illustré sur la figure 3.18.

Changement d'état de la faisabilité de la tâche TD : suite à la saisie des informations concernant la tâche TD puis à sa planification, le responsable de planification doit se prononcer sur sa faisabilité (activité 1, fig. 3.18). S'il qualifie cette dernière de *faisable* (TD.Fa = OK), il transmet le planning de la tâche TD au responsable de conception qui peut alors débuter la conception de

l'alternative système AS. S'il qualifie la tâche TD d'*infaisable* (TD.Fa = KO), cela signifie qu'il n'est pas en mesure de planifier convenablement la tâche TD au regard des contraintes projet saisies au préalable. Dans ce cas, 1) soit les informations concernant la tâche TD seule sont modifiées et la planification réitérée afin d'obtenir un nouveau planning pour cette tâche (TD.Fa = ND), 2) soit les contraintes projet globales sont remises en causes et le processus complet est remis en œuvre (évènement A, fig. 3.17 et 3.18), les attributs correspondants étant remis à l'état *non déterminé* (ES.Fa = ND, TE.Fa = ND, TE.Ve = ND et TD.Fa = ND), 3) soit les informations ne sont pas modifiées et l'alternative est abandonnée. Dans tous les cas, le directeur de programme et le responsable de conception sont alertés.

Changement d'état de la faisabilité de l'alternative système AS : suite à la réception du planning émis par le responsable de planification, le responsable de conception réalise la tâche de conception système (première sous-tâche de la tâche TD) (activité 2, fig. 3.18). À l'issue de cette dernière, il doit se prononcer sur la faisabilité de l'alternative système. S'il qualifie cette dernière de *faisable* (AS.Fa = OK), il alerte le responsable de planification et peut alors continuer la conception de l'alternative système. S'il qualifie l'alternative système d'*infaisable* (AS.Fa = KO), cela signifie qu'il n'est pas en mesure de développer la solution répondant aux exigences système au regard des exigences et/ou des contraintes projet. Soit le responsable de conception envoie une demande de replanification de la tâche TD afin de modifier sa solution logique (il attend alors le nouveau planning de la tâche TD), soit il envoie une demande de replanification du projet complet afin de modifier les exigences système (il attend alors le nouveau planning de la tâche TE). Dans les deux cas, la faisabilité de l'alternative système AS est remise à *non déterminée* (AS.Fa = ND) et le responsable de planification reçoit la demande de replanification. Suivant la demande, trois possibilités existent : 1) soit les informations concernant la tâche TD sont modifiées et sa planification réitérée afin d'obtenir un nouveau planning pour cette tâche (TD.Fa = ND), 2) soit les contraintes projet globales sont remises en causes et le processus complet est remis en œuvre (évènement A, fig. 3.17 et 3.18), les attributs correspondants étant remis à l'état *non déterminé* (ES.Fa = ND, TE.Fa = ND, TE.Ve = ND, AS.Fa = ND et TD.Fa = ND), 3) soit les informations ne sont pas modifiées et l'alternative est abandonnée, le responsable de conception est alors informé. Dans tous les cas, le directeur de programme est alerté.

Changement d'état de la vérification de l'alternative système AS : suite au travail de conception, le responsable de conception doit vérifier que l'alternative système AS satisfait toutes les exigences système (activité 3, fig. 3.18). S'il qualifie l'alternative système de *vérifiée* (AS.Ve = OK), le responsable de planification en est informé et peut alors vérifier la tâche TD. S'il qualifie l'alternative système d'*infaisable* (AS.Ve = KO), cela signifie que celle-ci ne satisfait pas toutes les exigences système. Dans ce cas, l'alternative système AS seule peut être abandonnée en l'état et la tâche TD est terminée (TD.Ve ≠ ND) ; le directeur de programme en est informé. L'alternative peut également être remise en cause, c'est-à-dire que la non-vérification peut être réévaluée suite à un travail de développement supplémentaire. Dans ce cas, le directeur de programme est alerté et les attributs de faisabilité et vérification de l'alternative système AS sont remis dans l'état *non déterminé* (AS.Fa = ND, AS.Ve = ND). Une demande de replanification, soit de la tâche TD seule, soit du projet complet, est envoyée au responsable de planification. En fonction de la demande reçue, le responsable de planification va replanifier soit la tâche TD (TD.Fa = ND, TD.Ve = ND), soit le projet complet (ES.Fa = ND, TE.Fa = ND, TE.Ve = ND et TD.Fa = ND), soit ne pas replanifier. Selon le cas, le responsable de conception attend soit le nouveau planning de la tâche TE, soit le nouveau planning de la tâche TD, soit l'information concernant la non replanification de la tâche TD.

Changement d'état de la vérification de la tâche TD : à la fin de la tâche TD, le responsable de planification doit vérifier que les contraintes projet assignées à cette dernière ont bien été respectées (activité 4, fig. 3.18). S'il qualifie la tâche TD de *vérifiée* (TD.Ve = OK), il informe le directeur de programme de son bon déroulement. S'il qualifie la tâche TD de *non vérifiée* (TD.Ve = KO), cela signifie que les contraintes projet assignées à cette dernière n'ont pas toutes été respectées (dépassement du délai ou du budget alloué par exemple). Le directeur de programme est alors alerté.

Quand la tâche TD est terminée et sa vérification réalisée (TD.Ve ≠ ND) et que l'alternative système AS est *non vérifiée* (AS.Ve = KO), il est encore possible de la remettre en cause. Pour cela, un délai et/ou des ressources supplémentaires sont nécessaires et il faut donc replanifier la tâche TD (TD.Fa = ND, TD.Ve = ND). Si tel est le cas, les attributs de faisabilité et de vérification de la tâche TD reviennent à l'état *non déterminé*. Similairement, les attributs de faisabilité et de vérification de l'alternative système reviennent à *non déterminé* (AS.Fa = ND, AS.Ve = ND) et le responsable de conception est informé. Il attend alors le nouveau planning lui permettant de poursuivre la tâche TD afin de rendre l'alternative système AS *vérifiée*. Il est également possible, selon le même principe, de remettre en cause l'ensemble du projet et de replanifier la tâche TE et la tâche TD. Dans ce cas, l'ensemble des attributs de toutes les entités sont remis à l'état *non déterminé*.

Même si l'alternative système AS est jugée *vérifiée*, une décision de replanifier la tâche TD (respectivement le projet complet) peut être prise. Dans ce cas, le responsable de conception remet les attributs de l'alternative système AS à *non déterminé* (AS.Fa = ND, AS.Ve = ND) (respectivement les attributs des exigences système et de l'alternative système à *non déterminé* (ES.Fa = ND, AS.Fa = ND, AS.Ve = ND) et attend son nouveau planning pour la tâche TD (respectivement pour la tâche TE).

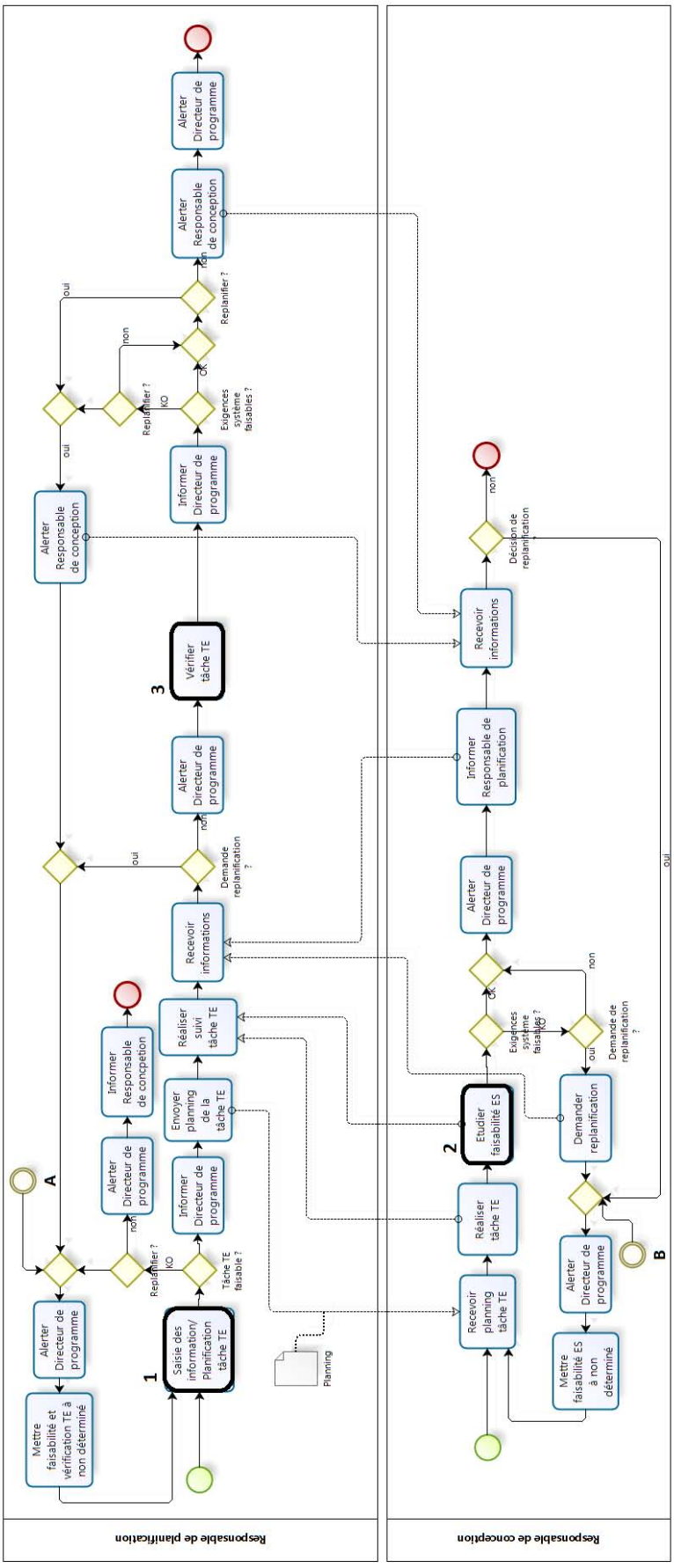


Figure 3.17 : Processus intégré de planification et de conception partiel permettant le changement des attributs des exigences système et de la tâche TE

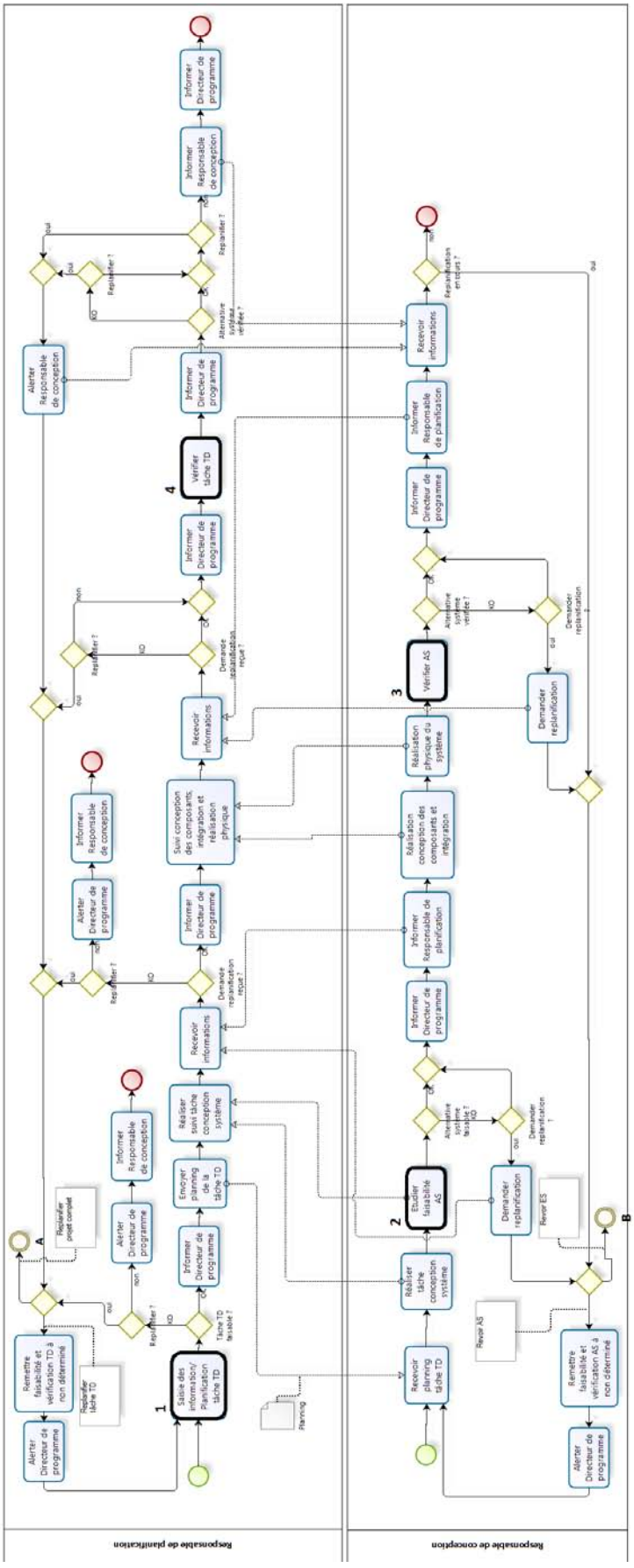


Figure 3.18 : Processus intégré de planification et de conception partiel permettant le changement des attributs de l'alternative système et de la tâche TD

3.4 Illustration du couplage informationnel

Cette section illustre l'application des processus proposés dans la section 3.3 sur la conception d'un avion d'affaire que nous avons débuté dans la section 2.4 en se focalisant sur les décisions prises par chaque acteur. Deux scénarios sont illustrés : la synchronisation entre la tâche TE et les exigences système ES associées et la synchronisation entre une tâche TD et une alternative système AS associée. Nous nous intéressons au projet de conception du sous-système « Structure » illustré dans la section 2.4.

3.4.1 Synchronisation entre exigences système ES et tâche TE

Nous considérons, dans ce premier scénario, la planification et le suivi d'une tâche TE dédiée au recueil des exigences système d'un système Structure, ainsi qu'au recueil des exigence système à proprement parler. Les étapes significatives de planification et de recueil des exigences système sont illustrées dans la figure 3.19.

Dans un premier temps, le responsable de planification reçoit les orientations du directeur de programme. Il peut alors saisir les informations puis planifier la tâche TE. Ici, la planification obtenue satisfait les orientations du projet : le responsable de planification (M. RespP1, voir section 2.4.3) qualifie donc la tâche TE de *faisable* (TE.Fa = OK). Le responsable de conception (M. RespC1) est alors averti de la faisabilité de la tâche TE et reçoit le planning correspondant. Il peut ainsi commencer le recueil des exigences système.

Quand le recueil des exigences système est terminé, M. RespC1 va se prononcer sur la faisabilité des exigences système. Ici, il pense pouvoir concevoir une solution correspondant aux exigences système : il qualifie donc les exigences système de *faisables* (ES.Fa = OK). M. RespP1 est alors averti que le travail de recueil des exigences système est terminé et peut donc vérifier la tâche TE.

M. RespP1 vérifie que le planning prévisionnel a bien été respecté. Ici, toutes les contraintes projet sont effectivement respectées : il qualifie donc la tâche TE de *vérifiée.*

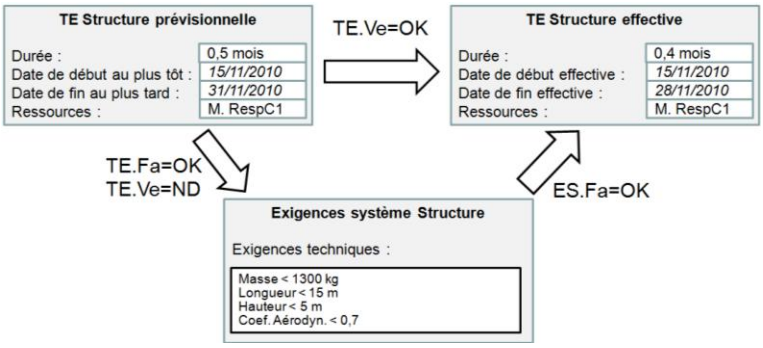


Figure 3.19 : Application du processus de conception/planification du recueil des exigences système pour un système « Structure »

3.4.2 Synchronisation entre alternative système AS et tâche TD

Nous considérons, dans ce second scénario, la planification et le suivi d'une tâche TD dédiée au développement d'une alternative système AS pour un système Structure, ainsi qu'au développement de cette alternative système à proprement parler. Les étapes significatives de planification et de conception sont illustrées sur la figure 3.20.

Ici, le responsable de planification a déjà reçu les contraintes du projet lors de la réalisation de la tâche TE. Il peut donc saisir les informations puis planifier la tâche TD. Ici, la planification obtenue satisfait les contraintes projet : le responsable de planification (M. RespP1) qualifie donc la tâche TD de *faisable* (TD.Fa = OK). Le responsable de conception (M. RespC1) est alors averti de la faisabilité de la tâche TD et reçoit le planning correspondant. Il peut ainsi commencer la conception système de l'alternative système.

Quand la conception de l'alternative système AS est terminée, M. RespC1, le responsable de conception, a produit une solution logique se décomposant en deux sous systèmes : un sous-système longeron qui fera l'objet de l'application présentée dans les chapitres suivants et un second sous-système comprenant les autres éléments de la structure. Le responsable de conception va alors se prononcer sur la faisabilité de l'alternative système. Ici, il pense pouvoir concevoir l'alternative en satisfaisant les exigences système : il qualifie donc l'alternative système de *faisable* (AS.Fa = OK).

Les sous-projets de conception des sous-systèmes sont alors planifiés et réalisés. À l'issue de leur conception, que nous ne détaillons pas dans cette section (voir section 2.3.4), aucun problème n'a été relevé, c'est-à-dire que les sous-systèmes et les sous-projets de conception associés sont *faisables* et *vérifiés*. Les sous-systèmes sont alors intégrés dans l'alternative système Structure et, le cas échéant, la réalisation physique de la structure est réalisée. Quand cela est fait, le responsable de conception va alors se prononcer sur la vérification de l'alternative système. Ici, la solution physique produite satisfait en tout point les exigences système : il qualifie donc l'alternative système de *vérifiée* (AS.Ve = OK). M. RespP1 est alors averti que la conception de l'alternative système AS est terminée et il peut alors effectuer la vérification de la tâche TD.

M. RespP1 vérifie que le planning prévisionnel a bien été respecté. Ici aussi, toutes les contraintes projet sont effectivement respectées : il qualifie donc la tâche TE de *vérifiée* (TE.Ve = OK).

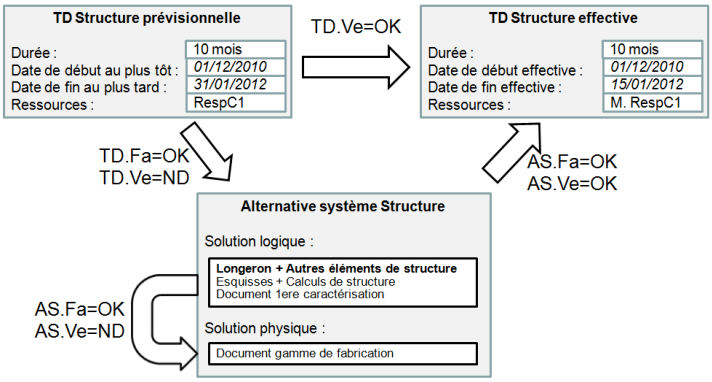


Figure 3.20 : Application du processus de conception/planification du développement d'une alternative système pour un système « Structure »

3.5 Conclusion sur le couplage informationnel

Nous avons proposé dans cette section de formaliser le couplage informationnel entre les entités de conception et de planification d'un projet de conception. Dans un premier temps, la sémantique du couplage informationnel a été définie afin d'envisager les possibilités de couplage et cibler celles retenues dans ce mémoire. Ensuite, nous avons proposé de compléter le diagramme de classe proposé dans la section 2.2 afin de supporter le couplage informationnel. Ensuite, nous avons décrit le processus de couplage informationnel à partir du processus intégré de planification et de

conception proposé dans la section 2.3. Enfin, nous avons illustré le déroulement de ce processus par un exemple de conception d'une structure d'avion d'affaire.

Le couplage informationnel s'inscrit en complément indispensable du couplage structurel proposé dans le chapitre 2. En effet, il est nécessaire, au-delà du couplage structurel :

- d'assurer une planification cohérente et un suivi tout au long du projet,
- d'assurer une conception cohérente au niveau technique,
- de détecter au plus tôt les infaisabilités de planification et de conception afin d'apporter les corrections nécessaires le plus rapidement possible,
- de détecter les non conformités (non vérification) dans la planification et la conception.

Cette section n'aborde cependant pas ou peu les processus permettant de pallier aux infaisabilités et aux non-vérifications. En effet, de nombreux cas de figure peuvent se produire. Nous proposons donc d'approfondir la question de la résolution des infaisabilités et des non-vérifications dans le chapitre 4 qui est dédié à la constitution de tableaux de bord et plus généralement à la prise de décision impactant à la fois la conception et la planification que nous nommons couplage décisionnel.

Chapitre 4 : Couplage décisionnel

4 Couplage décisionnel

Dans les chapitres précédents, nous avons proposé de mettre en place plusieurs types de couplage de la conception de systèmes et de la planification de projets associés. Chacun de ces couplages impliquent des prises de décision. Dans cette optique, nous proposons d'utiliser des tableaux de bord afin de faciliter la prise de décision par les acteurs concernés. Dans un premier temps, nous définissons la notion de tableau de bord, puis, nous nous intéressons au traitement des données nécessaires à la constitution d'un tableau de bord. Dans un troisième temps, nous proposons des processus pour la constitution d'un tableau de bord et la prise de décision et, enfin, nous illustrons ces processus pour la conception d'un avion d'affaire.

4.1 Définition d'un tableau de bord

4.1.1 Tableau de bord

Un tableau de bord est un instrument de mesure de la performance facilitant le pilotage "proactif" d'une ou plusieurs activités dans le cadre d'une démarche de progrès. Le tableau de bord contribue à réduire l'incertitude et facilite la prise de risque inhérente à toutes décisions. Le tableau de bord est un instrument d'aide à la décision [**Fernandez, 2005**].

Le tableau de bord présente donc un intérêt pour lever des incertitudes dans la conception de système et dans la planification de projet. En outre, il permet également de mesurer, prioriser et/ou évaluer plusieurs choix possibles : il s'agit de présenter différents scénarios (ou alternatives) et d'apporter des éléments de décision. Le tableau de bord doit permettre aux acteurs de prendre les meilleures décisions en mettant à leur disposition les éléments nécessaires et suffisants à la prise de décision.

Le tableau de bord n'est pas qu'un simple outil de suivi, il doit permettre réduire l'incertitude en permettant une meilleure appréhension du contexte de pilotage et, toute décision étant une prise de risque, de contribuer à une meilleure maîtrise du risque en disposant d'une vision stable et structurée de son environnement. De plus, le tableau de bord doit permettre de faciliter la communication entre les différents acteurs du projet et ainsi dynamiser la réflexion.

Un tableau de bord doit apporter des réponses aux questions suivantes :

- **Où en sommes-nous aujourd'hui ?** C'est-à-dire quel est l'état actuel de la conception et de la planification ;
- **Quel est le chemin parcouru ?** Autrement dit, il s'agit de mettre en lumière ce qui a été accompli au moment de la réalisation du tableau de bord et les décisions prises précédemment ;
- **Que reste-t-il à faire ?** Il s'agit de s'assurer qu'il n'y a pas de dérive par rapport au planning prévisionnel et que le contexte de la conception n'est pas défavorable.

4.1.2 Indicateurs

Un tableau de bord se compose d'indicateurs. Un indicateur est un instrument de mesure donnant des informations sur un paramètre [**Rousse, 2010**]. Il a pour fonction de suivre ou d'évaluer un paramètre de la conception ou de la planification, de diagnostiquer une situation ou un problème ou encore de contrôler l'environnement et ses changements.

La constitution de tableaux de bord ne se limite pas à regrouper sur un même espace quelques indicateurs traditionnellement utilisés dans le domaine concerné. Pour remplir son rôle d'assistance au

pilotage et ainsi faciliter la prise de décision, un indicateur pertinent présente les trois caractéristiques suivantes :

- il mesure la performance selon la direction de progrès qui a été choisie,
- il est adapté au contexte ainsi qu'aux moyens d'action disponibles et,
- il est en accord avec les besoins spécifiques de chaque décideur.

Un indicateur doit être simple et compréhensible, c'est-à-dire que l'information véhiculée par l'indicateur doit pouvoir être comprise immédiatement sans ambigüité possible et sans qu'il soit nécessaire de procéder à des transformations afin de la rendre exploitable. Dans cette optique, il doit être personnalisé, chaque projet étant unique, ce dernier a donc besoin d'indicateurs qui lui soient propres. Les indicateurs doivent également être validés dans le sens où l'information proposée doit avoir un sens pour le ou les utilisateurs. De plus, il doit être exploitable et exploité car un indicateur non exploité si, par exemple, l'information qu'il véhicule n'est pas fiable, ne sert à rien. En outre, il aussi être :

- utilisable en temps réel et donc proposer des indicateurs actualisés pour le système et le projet à contrôler,
- visualisable, il est en effet souhaitable que ceux qui ont à connaitre l'information donnée par l'indicateur voient la même chose,
- générateur d'actions, les décisions qui vont être prises le sont sur la base de informations proposées par les indicateurs,
- ergonomiquement présentable, de sorte que l'information et sa signification soit compréhensible le mieux possible.

Un indicateur peut être :

- un indicateur d'alerte signalant un état anormal nécessitant une intervention,
- un indicateur d'équilibration montrant l'avancement par rapport aux objectifs et pouvant induire des actions correctives voire de remettre en cause l'objectif en cas de dérive
- des indicateurs d'anticipation assurant une vision un peu plus large pouvant induire des changements de stratégie et d'objectifs et informant sur le contexte de conception.

4.1.3 Définition du couplage décisionnel

Compte tenu des éléments présentés dans les sections précédentes, nous définissons le couplage décisionnel comme la fédération, au sein d'un tableau de bord commun, d'indicateurs propres à la planification et d'indicateurs propres à la conception dans le but de permettre au directeur de programme de prendre des décisions optimales. Dans la mesure où il est en charge de définir les orientations de planification et de conception et d'assurer le suivi général du projet, l'ensemble des informations, présentées sous forme d'indicateurs, pertinentes et suffisantes peuvent lui permettre de prendre ses décisions en toute connaissance de cause. En cas de conflit entre planification et conception, la résolution en sera facilitée.

4.2 Processus de constitution de tableaux de bord

Nous avons vu précédemment qu'un tableau de bord n'est pas simplement un outil de reporting, il doit aider le décideur à prendre les meilleures décisions. Dans cette optique, Alain Fernandez [**Fernandez, 2005**] propose un processus de constitution de tableau de bord en cinq étapes. Ce processus est illustré sur la figure 4.1.

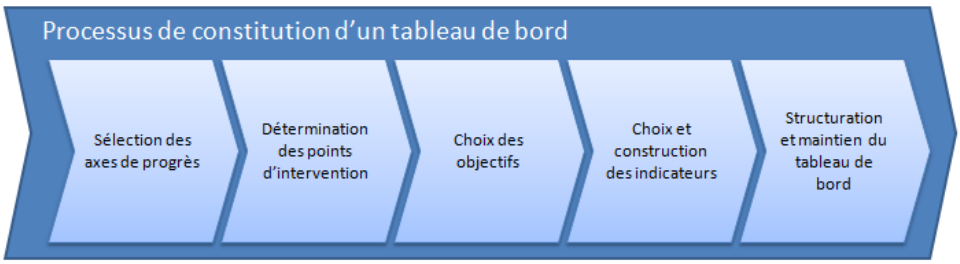


Figure 4.1 : Processus de constitution d'un tableau de bord

La **sélection des axes de progrès** consiste à déterminer la meilleure démarche de progrès. Cette étape est un préalable à la constitution d'un tableau de bord : il s'agit de définir les axes de progrès les plus profitables pour l'entreprise tout en tenant compte de ses spécificités, de son marché ainsi que des moyens disponibles. Cette étape se place plutôt en amont de la conception et donc du couplage de la conception et de la planification, nous ne l'aborderons donc pas dans ce mémoire.

La **détermination des points d'intervention** a pour but d'identifier exhaustivement les activités et processus critiques au sein des axes de progrès précédemment identifiés. Ici aussi, nous nous plaçons en amont de la conception et de sa planification. Nous ne traiterons donc pas cette étape.

Le **choix des objectifs** permet de définir et sélectionner, pour chaque acteur et groupe d'acteurs, les meilleurs objectifs « tactiques ». Il s'agit d'appliquer concrètement les axes de progrès définis dans la première étape. Ces objectifs vont devoir être suivis, ce qui fait l'objet des étapes suivantes. Cette étape est l'objet de la section 4.2.1.

Le **choix et la construction des indicateurs** consistent à sélectionner les indicateurs les plus pertinents en tenant compte des objectifs à suivre, du contexte et des habitudes de travail du décideur. Cette étape est détaillée dans les sections 4.2.2.1 et 4.2.2.2.

La **structuration et le maintien du tableau de bord** ont pour but d'agencer les indicateurs afin de permettre au tableau de bord d'être un véritable outil d'aide à la décision en permettant au décideur de visualiser l'état courant « d'un seul coup d'œil ». Cette étape fait l'objet de la section 4.2.2.3.

4.2.1 Choix des objectifs

Tous les acteurs du projet de conception (directeur de programme et responsables de conception et de planification) sont intéressés par les tableaux de bord, ces derniers pouvant être différents en fonction des objectifs qui leurs sont donnés. En effet, le directeur de programme doit avoir une vision globale du projet de conception et du système à concevoir tandis que les responsables de conception et de planification ont une vision plus resserrée et plus précise sur une partie du projet.

Chacun des acteurs du projet de conception doit donc définir son propre tableau de bord afin de contrôler ses propres objectifs à atteindre et, au-delà, de prendre les meilleures décisions en ayant connaissance des informations (c'est-à-dire des indicateurs) nécessaires.

4.2.2 Choix et construction des indicateurs

4.2.2.1 *Choix des indicateurs*

Le choix des indicateurs est crucial pour la constitution d'un tableau de bord. En effet, les indicateurs sont porteurs d'information et si le choix n'est pas judicieux, le décideur risque de disposer

d'informations non représentatives de l'objectif à suivre ou du contexte de conception. Dans cette optique, il y a six principes à respecter pour le choix et la construction d'un indicateur :

- *il doit mesurer le résultat par rapport à l'objectif à suivre*. Par exemple, si l'objectif vise au respect des délais pour une conception donnée, l'indicateur devra délivrer, d'une manière ou d'une autre, une information relative aux délais ;
- *il doit être fiable*. Seules les informations dignes de confiance sont susceptibles de contribuer à l'aide à la décision. En effet, un utilisateur ne prendra jamais de décision sur la base d'informations dont il n'est pas sur. Tant que la fiabilité de l'information n'est pas établie, il est inutile de l'utiliser pour construire un indicateur ;
- *il doit inciter à décider*. Un indicateur ne se contente pas de constater une situation. En effet, constater qu'un indicateur dérive n'a d'intérêt que si le décideur est en mesure de prendre les décisions opportunes pour corriger cette dérive ;
- *il doit pouvoir être construit sans trop de difficultés*. Les informations doivent être disponibles et les traitements pour l'obtenir doivent être relativement simples afin de limiter les risques d'erreurs et la maintenance sur les traitements ;
- *il doit être rafraîchi à temps.* L'information fournie par l'indicateur doit être suffisamment récente pour laisser le temps à la décision. Par exemple, un coût mensuel actualisé tous les ans ne sera pas utile pour le décideur.
- *il doit être délivré à coût acceptable.* La collecte d'information prend du temps et par conséquent présente un coût non négligeable. Il est important d'évaluer ce coût au préalable.

4.2.2.2 *Transformation des données en indicateurs*

En préambule du traitement des données utiles pour la prise de décision et donc la constitution d'un tableau de bord, nous nous intéressons où ces données sont stockées. Dans l'environnement de conception de système, les objectifs de conception sont stockés dans l'entité Exigences système et les solutions de conception sont regroupées dans la solution physique de chaque Alternative Système. Nous retrouvons également de données pertinentes dans les concepts qui sont associés aux systèmes et alternatives système. Dans l'environnement de planification, les objectifs de planification ainsi que les valeurs effectives sont comprises dans les projets ainsi que dans les tâches TE et TD.

Certaines données peuvent être utilisées sans modification et d'autres nécessitent un traitement afin de fournir un indicateur plus significatif et/ou agrégeant plusieurs données. Par exemple, la mesure de compatibilité entre une alternative système et des exigences système exprimées, proposée ultérieurement dans la section 6.2.4, est un indicateur qui est obtenu grâce à un traitement sur les données de conception.

Les indicateurs et plus généralement les tableaux de bord nécessaires à la prise de décision sont variables en fonction du système à concevoir et du projet de conception associé. Néanmoins, nous proposons quelques indicateurs utilisables en toutes circonstances :

- la faisabilité et la vérification des entités suivies,
- le coût système et le coût projet prévisionnel et effectif,
- l'avancement de la conception et de la planification,
- le risque système et projet.

Un indicateur mesure un objectif à suivre, il doit donc être fiable et inciter à décider. De plus il doit être facile à construire et actualité à temps. Enfin, il doit être réalisable à un coût acceptable. Les

formes d'indicateurs les plus courantes sont donc connues et peuvent aisément être mises en œuvre. Il peut s'agir :

- d'écarts, il s'agit de comparer une valeur réelle avec une valeur attendue (par exemple une valeur physique du système peut être comparée avec l'exigence système correspondante),
- de ratios, c'est-à-dire un rapport entre grandeurs significatives. Un ratio seul n'a pas de signification propre, c'est son évolution au cours du temps qui est significative,
- de graphiques offrant une bonne visibilité sur l'évolution des situations et des indicateurs (par exemple, le suivi du coût du projet),
- des clignotants permettant de générer des alertes en cas de dépassement d'un seuil fixé (par exemple, la faisabilité et la vérification peuvent être traités de cette manière).

Les indicateurs peuvent être représentés de différentes manières en fonction de ce qu'ils représentent. Nous proposons ci-dessous quelques exemples de représentation d'indicateurs :

- les indicateurs de type tout ou rien. Ce sont les indicateurs les plus élémentaires. Ils ne délivrent qu'une information binaire : oui/non, bon/mauvais, etc. Ils sont essentiellement utilisés pour les alarmes ;
- les indicateurs de type tout ou rien avec seuil d'alerte. Ils sont similaires aux précédents mais présentent un état supplémentaire à l'approche d'un seuil critique. C'est le principe du feu tricolore qui passe à l'orange avant de passer au rouge (indicateur a figure 4.2) ;
- les indicateurs de type mesure. Ils sont utilisés pour suivre l'évolution d'une entité et différentes présentations sont possibles comme :
    - le thermomètre ou la jauge. Comme tout indicateur, un seuil doit être fixé. Selon le type de mesure, le seuil d'alerte peut être défini comme « niveau bas » ou comme « niveau haut ». Le thermomètre change alors de couleur. Pour une meilleure anticipation, un second seuil intermédiaire peut être fixé (indicateur b figure 4.2) ;
    - le tachymètre qui présente l'évolution de la mesure à la manière d'un compteur de vitesse de voiture. Comme l'exemple précédent, un ou deux seuils peuvent être fixés (indicateur c figure 4.2) ;
- les indicateurs de type courbe qui permettent d'évaluer la mesure par rapport à une référence. Cette référence peut être un seuil ou une consigne. Cette consigne sera généralement une projection de tendance pour atteindre l'objectif. Par exemple, une évolution une évolution linéaire des coûts engagés jusqu'à la fin du projet sera une tendance à respecter et tout écart devra être corrigé,
- les indicateurs de type multi-seuils. Ils garantissent la transmission d'un message purement qualitatif comme « très faible-faible-modéré-fort-très fort »,
- les indicateurs complexes. Ces indicateurs, qui peuvent être entre autres des tableaux, des listes, etc., sont plutôt à écarter des tableaux de bord du fait de la quantité d'information importante qu'ils véhiculent.

Quelques exemples de représentation des indicateurs présentés ci-dessus sont proposés sur la figure 4.2.

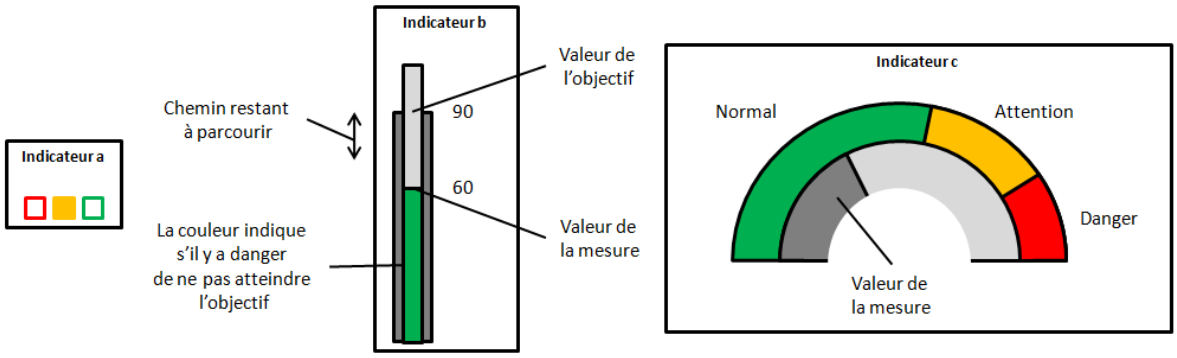


Figure 4.2 : Quelques exemples de représentations d'indicateurs

4.2.3 Structuration et maintien du tableau de bord

L'agencement des indicateurs au sein du tableau de bord est d'une importance significative. En effet, même si la préconisation d'un nombre d'indicateurs réduit (entre cinq et dix indicateurs par tableau de bord) cadre l'information transmise au destinataire du tableau de bord, toutes les informations ne sont pas équivalentes en importance. Il est donc nécessaire de mettre en exergue les indicateurs présentant pour l'utilisateur du tableau de bord, le plus d'intérêt. Dans ce sens, Alain Fernandez décompose le tableau de bord en quatre cadrans comme illustré dans la figure 4.3 [**Fernandez, 2005**].

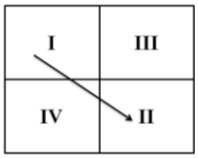


Figure 4.3 : Sens de lecture prioritaire d'un écran

Un écran se lit depuis l'angle en haut à gauche vers l'angle en bas à droite. Les cadrans I et II sont donc des zones de première importance. Les informations d'alertes essentielles ou les indicateurs les plus cruciaux seront placés dans ces deux cadrans en priorité. Les cadrans III et IV sont généralement lus au second coup d'œil. Les indicateurs moins critiques seront donc naturellement placés dans ces zones.

4.3 Illustration du couplage décisionnel

Dans cet exemple, nous présentons trois tableaux de bord proposés dans la conception d'un avion d'affaire. Le premier a été créé dans l'optique de choisir entre deux alternatives celle qui sera retenue. Le second a été réalisé suite à la non vérification d'une alternative système. Le dernier doit permettre la sélection d'une alternative système déjà conçue et réutilisé.

4.3.1 Choix entre deux alternatives

Dans cette section, nous allons constituer un tableau de bord permettant de comparer deux alternatives de conception et de planification. L'objectif de ce tableau de bord est de permettre au directeur de programme de choisir l'alternative répondant le mieux aux objectifs fixés. Ici, les objectifs principaux sont :

- de minimiser le risque de dépassement de délai,
- de respecter les contraintes de coût système et projet,
- de respecter les exigences système.

L'ensemble des informations nécessaires sont présentes dans la figure 4.4.

Indicateurs Produit				Indicateurs Projet		
COUTS						
	Avion1	Avion2			Avion1	Avion2
Coût Système (€)	*2,8M*	3M		Coût Projet (€)	1M	50k
Coût Syst. Engagé (€)	*10k*	10k		Coût Projet engagé (€)	10k	3k
RISQUES						
Risque Conception (./5)	*3*	2		Risque Planification (./5)	2	4
PERFORMANCES						
NbPass	*[24, 30]*	25		Délai (mois)	15	16
Poids (X 1000kg)	*≤10*	10,5		Marge	1.2	0.2
Distance (x 1000km)	*[2,5]*	4				
COMPLEMENTS						
Préférence	0	1		Préférence	1	0
Faisabilité	EC	OK		Faisabilité	EC	OK
Vérification	EC	OK		Vérification	EC	OK
Sélection	EC	EC		Sélection	EC	EC
Avancement Syst. (%)	2	100		Avancement Projet	2	15

Figure 4.4 : Ensemble des informations nécessaires à la constitution du tableau de bord pour le choix entre deux alternatives

Il s'agit alors pour le directeur de programme de définir et prioriser ses indicateurs puis de structurer son tableau de bord. Premièrement, le directeur de programme souhaite comparer les deux alternatives « Avion 1 » et « Avion 2 », il décide donc de décomposer son tableau de bord en deux parties identiques.

Ensuite, il souhaite en priorité connaitre l'état général de la conception et de la planification, il va donc placer en haut à gauche de chaque partie, des indicateurs de faisabilité et de vérification pour l'alternative système et pour la tâche TD associée. Ces indicateurs prendront la forme de feux tricolores (vert si l'alternative système ou la tâche TD est faisable ou vérifiée, rouge dans le cas contraire et orange si la faisabilité ou la vérification n'est pas déterminée).

Puis il souhaite connaître l'avis des responsables de conception et de planification sur les alternatives, il va donc disposer deux indicateurs tricolores exprimant ceci (vert si le responsable préfère cette solution, rouge s'il souhaite l'abandonner et orange s'il n'a pas de préférence ni d'opposition à l'alternative). Ces indicateurs seront placés en haut à droite de chaque partie. De plus, il placera à proximité deux indicateurs de risque pour la conception et la planification. Ces indicateurs sont de type « thermomètre ».

En bas à droite, il disposera deux indicateurs de type courbe présentant l'évolution de coûts système et projet.

Enfin, en bas à gauche, quelques indicateurs sur la conception système et la planification projet seront disposés afin de surveiller quelques aspects critiques de la conception et de la planification. Il s'agit ici d'indicateurs de type « thermomètre » ou « tachymètre ».

Le tableau de bord ainsi obtenu est présenté sur la figure 4.5.

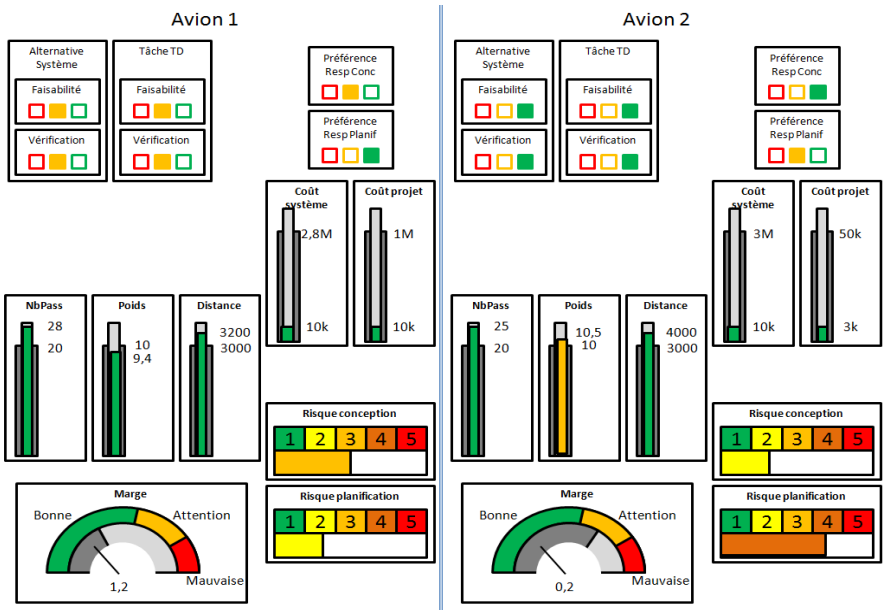


Figure 4.5 : Exemple de tableau de bord pour le choix entre deux alternatives

Ce tableau de bord doit servir de base de discussion entre le directeur de programme qui à demandé sa constitution, et les responsables de conception et de planification impliqués dans le choix de l'alternative. Suite à cette discussion, le choix est fait de conserver l'alternative « Avion 1 » et d'écarter l'alternative « Avion 2 » car le risque de dépassement de délai (risque de planification) est moindre pour l'alternative « Avion 1 ».

4.3.2 Suivi de développement d'une alternative

Cette section illustre la constitution d'un tableau de bord demandé par le responsable de conception suite à une non vérification de l'alternative système « Avion 1 ». Ce tableau de bord a pour objet de déterminer les leviers d'actions permettant de résoudre ce problème. L'objectif principal est, pour l'ensemble des acteurs impliqués dans la conception, de minimiser les non vérifications tant au niveau de la conception qu'au niveau de la planification.

Ce tableau de bord va ressembler à une partie du tableau de bord proposé dans la section précédente. En effet, les indicateurs permettant de déterminer les leviers d'actions sont assez proches des indicateurs ayant permis de choisir l'alternative. Cependant, les indicateurs de risque de conception et de planification ainsi que les préférences des responsables ont peu d'intérêt dans ce cas de figure puisqu'il n'y a qu'une seule alternative à examiner.

Dans le cas présent, les acteurs souhaitent en priorité connaitre l'état général de la conception et de la planification, il va donc placer en haut à gauche de chaque partie, des indicateurs de faisabilité et de vérification pour l'alternative système et pour la tâche TD associée. Ces indicateurs prendront la forme de feux tricolores (vert si l'alternative système ou la tâche TD est faisable ou vérifiée, rouge dans le cas contraire et orange si la faisabilité ou la vérification n'est pas déterminée).

Ensuite, ils veulent connaitre si les coûts engagés pour la conception et pour la planification sont corrects par rapport aux prévisions. Ceci est placé en haut à droite sous la forme de deux indicateurs de type « thermomètre ».

Enfin, le directeur de programme et les responsables de conception et de planification souhaitent disposer d'informations sur l'état de la conception et de la planification. Ici, la conception

est représentée par trois indicateurs de type « thermomètre » représentant le nombre de passagers (NbPass), le poids total de l'avion (Poids) et la distance pouvant être parcourue (Distance). De même, la planification est représentée par la marge disponible. Cet indicateur est de type « tachymètre ».

Le tableau de bord ainsi produit est illustré sur la figure 4.6.

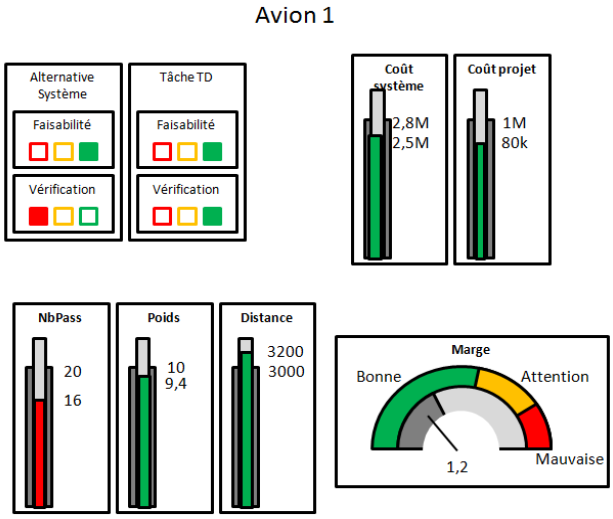


Figure 4.6 : Exemple de tableau de bord pour la résolution de non vérification

Ce tableau de bord sert de point de départ à la discussion entre le directeur de programme et les responsables de conception et de planification. Le constat établi par ce tableau de bord est que l'alternative système « Avion 1 » est non vérifiée car le nombre de passagers que l'avion peut contenir est inférieur aux exigences du client. À l'issue de cette discussion, le responsable de conception pense être en mesure d'augmenter le nombre de passager en modifiant légèrement se conception, c'est-à-dire en réduisant les réservoirs (et donc la distance pouvant être parcourue) et en augmentant légèrement le poids de l'avion. Cette solution nécessite un délai supplémentaire d'un mois qui peut être accordé car la marge est de 1,2 mois.

4.4 Conclusion sur le couplage décisionnel

L'information nécessaire pour décider à l'intérieur de son cadre de responsabilité n'est pas exclusivement détenue par soi. Il faut par conséquent définir le cadre que l'on veut, c'est-à-dire les indicateurs du tableau de bord, sous quelle forme et à quelle fréquence et l'alimenter. Ensuite, il reste à analyser la situation, mesurer les écarts, décider des actions futures puis passer à l'action.

C'est donc dans ce contexte que nous avons défini cette forme de couplage qualifiée de « décisionnelle ». En effet, l'agrégation, la fédération et l'affichage efficace d'informations provenant des deux domaines (planification et conception) permet aux décideurs (ici principalement le directeur de programme) de prendre des décisions, de régler des conflits et de définir les orientations du projet.

Cette forme de couplage est un peu particulière dans la mesure où la connaissance méthodologique ne précise que les informations de couplage qui peuvent être regroupées systématiquement dans le tableau de bord, les autres étant propres à chaque projet. Cependant, elle permet tout de même de guider dans l'élaboration du tableau de bord et à rendre obligatoire la constitution de cet espace d'échange où les conflits, les évolutions et les conséquences des décisions de couplage peuvent être observés

Conclusion sur le couplage méthodologique

Dans les trois chapitres précédents, nous avons proposé trois types de couplage basés sur des connaissances méthodologiques :

- le **couplage structurel** assurant une existence cohérente des entités de conception et de planification,
- le **couplage informationnel** assurant une évolution cohérente des entités de conception et de planification,
- le **couplage décisionnel** devant faciliter la prise de décision au regard des informations nécessaires et suffisantes.

Nous avons ainsi proposé des évolutions majeures dans la manière de coupler la planification d'un projet de conception avec les tâches de conception elles-mêmes par rapport aux pratiques classiques avec une vision très opérationnelles alors que les méthodes existantes sont bien souvent des modèles et processus de très haut niveau. En outre, dans la perspective de développement d'un outil logiciel d'aide à la décision, les modèles de données proposés (modèles de classes) ainsi que les processus sont pertinents et servent de guide méthodologique dans le projet ATLAS.

Ces trois types de couplage représentent un ensemble de « bonnes pratiques » utilisables en toutes circonstances de conception. Dans l'optique d'un projet de conception créative, les connaissances méthodologiques sont les seules connaissances formelles disponibles. Leur exploitation permet de cadrer l'activité de planification et de conception en garantissant de manière systématique que les informations sont saisies aux bons instants et qu'elles sont cohérentes. Elle permet également de s'assurer que les non-faisabilités, les conflits, les objectifs antagonistes sont détectés au plus tôt en offrant aux décideurs un environnement propice à leur résolution. Enfin, elles permettent de capitaliser les informations inhérentes à la planification et à la conception en s'appuyant sur une structure claire et non-ambigüe qui permettra ensuite une compréhension commune de ce qui a été conçu, en planification et en conception, à des fins d'assimilation par l'ensemble de l'entreprise.

Dans le cadre d'un projet de conception plus routinière, outre les activités de cadrage des activités par les processus et par la structure des informations qui doivent permettre d'accélérer le processus de conception en évitant les erreurs et en favorisant la standardisation, les connaissances méthodologiques offrent un socle indispensable à la formalisation d'un autre type de connaissance : la connaissance métier. Cette connaissance est spécifique à un domaine d'activité donné comme l'aéronautique.En effet, la connaissance métier est formalisée, capitalisée puis exploitée pour la conception de nouveaux systèmes et projets de conception associés. Ainsi, dans la section suivante (Partie II), nous proposons d'étendre le couplage à la formalisation, la capitalisation et l'exploitation des connaissances métier.

Partie II : Couplage basé sur les connaissances métier

Partie II : Couplage par les connaissances métier

Le couplage des activités de conception avec celles de définition et de planification de ces mêmes activités peut être intensifié par la capitalisation et l'exploitation des connaissances propres au domaine de conception. Ces connaissances sont appelées *connaissances métier.* Ces dernières sont fortement liées au contexte de conception et donc difficilement généralisables. Cependant, leur formalisation repose sur deux aspects : le premier est informationnel et concerne les données manipulables et le second est décisionnel et comprend des mécanismes de manipulation des informations et des règles. Cette partie s'intéresse à ces deux aspects. Nous proposons donc, d'une part, de *formaliser cette connaissance métier* et, d'autre part, de définir deux mécanismes de couplage utilisant les connaissances métier : un *couplage par réutilisation de cas* et un *couplage par contraintes*. Cette section est donc organisée en trois chapitres.

Le chapitre 5 traite de la formalisation de la connaissance métier. Elle consiste à définir une ontologie de concepts permettant la caractérisation des entités de conception et de planification sur une base sémantique commune à tous les acteurs du projet de conception. Ainsi, chaque concept est associé à un ensemble de modèles de variables, de modèles de domaines de définition pour ces modèles de variables et de modèles de contraintes qui pourront être appliqués aux entités de conception et de planification. Nous caractérisons donc ces entités à partir de ces modèles en permettant également aux différents acteurs du projet d'intégrer l'expression d'exigences et/ou contraintes supplémentaires.

Le chapitre 6 propose une forme de couplage par réutilisation de cas. Il s'agit d'exploiter les projets de conception et les systèmes conçus par le passé et de capitaliser les nouveaux projets et systèmes conçus. Ces informations étant stockées dans une base de cas, le couplage par réutilisation de cas consiste, à partir de la formalisation d'un problème de conception (exigences système), à chercher les solutions (alternatives système) précédemment conçues répondant le mieux à ce problème. Les solutions ainsi identifiées peuvent être adaptées ainsi que les tâches de développement associées.

Le couplage par contraintes, exposé dans le chapitre 7 exploite la connaissance métier telle que formalisée dans le chapitre 5 à l'aide de Problèmes de Satisfaction de Contraintes (CSP). Les CSP permettent de formaliser une connaissance générique pour chaque concept de l'ontologie proposée tant pour les concepts utilisés en conception que ceux utilisés en planification. Au-delà, nous définissons la notion de contrainte de couplage associant à la fois des variables d'un concept utilisé en conception et d'un concept utilisé en planification.

Chapitre 5 : Support de la formalisation des connaissances métier

5 Support de la formalisation des connaissances métier

La connaissance métier est selon China [**China, 1997**] « constituée de la somme des connaissances et de l'expérience nécessaire pour faire fonctionner le système, en vue d'atteindre des objectifs technico-économiques donnés et ce compte tenu des contraintes liées à l'environnement ». La formalisation de la connaissance métier permet, d'une part, une meilleure efficacité dans la réalisation des processus et, d'autre part, elle facilite la transmission du savoir et de l'expertise des personnels [**Beler, 2008**]. La formalisation de la connaissance métier au sein des entreprises est une activité généralement dévolue au management des connaissances (KM – *Knowledge Management*). Parmi les diverses méthodologies existantes, l'utilisation de la notion d'ontologie est très répandue (voir par exemple [**Richards et Simoff, 2001**] [**Brandt et al., 2008**]). Ainsi, nous proposons dans cette section de détailler la formalisation de connaissances métier à l'aide d'une ontologie de concepts afin de guider les décideurs dans leurs prises de décisions et de faciliter le couplage entre conception système et planification de projet. Nous nous plaçons dans cette partie dans le cadre d'une opérationnalisation des connaissances et donc nous nous intéressons particulièrement aux connaissances explicites.

Dans un premier temps, nous définissons les notions de concept et d'ontologie. Ensuite, nous expliquons comment les ontologies permettent de représenter la connaissance et, plus particulièrement, la connaissance métier. Dans un troisième temps, nous proposons une ontologie adaptée à nos besoins de couplage. Puis, nous faisons évoluer le modèle de classes proposé dans les chapitres précédents afin de prendre en compte cette connaissance métier. Enfin, nous illustrons le déroulement d'un processus intégré de planification de projet et de conception de système faisant intervenir la connaissance métier inhérente au projet et au système.

5.1 Concepts et ontologies

Un concept est une interprétation organisée et structurée d'une partie du monde réel qui est utilisée pour penser et communiquer sur cette partie du monde, c'est un modèle *mental interne* [**Darlington et Culley, 2008**]. Un concept est un constituant de la pensée sémantique évaluable et communicable. L'ensemble des attributs d'un concept, c'est-à-dire des descripteurs du concept, s'appelle sa compréhension ou son intension et l'ensemble des êtres qu'il englobe, son extension. L'intension est déterminée par l'identification de propriétés communes à tous les individus auxquels le concept s'applique, cet ensemble de caractères permet de définir le concept [**Gandon, 2002**].

Une ontologie est une spécification explicite d'une conceptualisation, la conceptualisation étant une interprétation organisée et structurée d'une partie du monde utilisée pour penser et communiquer à propos de ce monde [**Gruber, 1993**] [**Darlington et Culley, 2008**]. L'ensemble des concepts permet de construire une **ontologie** d'un domaine particulier, c'est-à-dire un ensemble structuré de termes partagés par une communauté d'experts pour exprimer des informations sémantiques consensuelles [**Studer et al., 1998**].

Souvent, le terme ontologie est utilisé comme synonyme de taxonomie de concepts, c'est-à-dire une méthode de classification des concepts dans une base de connaissances. En fait, ceci permet de s'affranchir d'une certaine complexité due principalement au caractère multidisciplinaire, dynamique et évolutif des domaines en se concentrant sur la sémantique des concepts manipulés [**Alberts, 1993**]. Cette classification peut être organisée selon tout type de principe dont le plus commun est la généralisation/spécialisation [**Sowa, 1984**]. Il s'agit, dans ce cas, de classer

hiérarchiquement les concepts dans une arborescence ou un graphe. Dans une ontologie, les concepts les plus généraux sont proches de la racine de l'arborescence (concept nommé « Universel »). Plus le concept est éloigné de la racine, plus il est spécialisé.

Les ontologies ont trouvé plusieurs utilisations en conception de systèmes. En effet, de nombreuses applications existent, notamment en ingénierie concourante et collaborative où il s'agit de permettre à plusieurs équipes de conception de communiquer grâce à un langage commun [**Roche, 2000**] [**Kim et al., 2006**]. Dans ce cas, les ontologies permettent de garantir une interopérabilité sémantique, c'est-à-dire la capacité de deux ou plusieurs concepteurs à se comprendre mutuellement. Un sens compréhensible par tous est ainsi donné aux informations échangées et ce sens doit être distribué dans tous les systèmes entre lesquels des échanges doivent être mis en œuvre [**Blanc, 2006**] [**Zhang et al., 2003**].

Une ontologie peut servir de guide pour les concepteurs qui vont, à un niveau abstrait, initier la conception en recherchant dans l'ontologie les concepts s'appliquant le mieux au système à concevoir. L'ontologie doit permettre de faciliter l'utilisation d'un langage commun et la compréhension commune de ce qui doit être conçu. Ceci doit permettre d'améliorer la communication entre les différentes parties prenantes de la conception par exemple dans un contexte d'ingénierie concourante où les travaux sont distribués sur plusieurs équipes, ou encore entre les acteurs du processus de conception, de l'identification des exigences à la définition des solutions. Dans la mesure où les ontologies sont définies et maintenues par des experts du domaine, une ontologie en conception peut donc être perçue comme la spécification d'une conceptualisation mais également comme un accord, une convention sur une conceptualisation [**Richards et Simoff, 2001**] dont les concepteurs peuvent s'inspirer pour réaliser leurs activités.

L'intérêt que présente l'utilisation des ontologies dans les processus d'ingénierie de conception est majeur. En effet, elles permettent de formaliser la connaissance d'un domaine dans le but de [**Darlington et Culley, 2008**] :

- partager une compréhension commune de la structure d'information entre les personnes,
- permettre la réutilisation du domaine de connaissance,
- expliciter des postulats sur un domaine donné [**Noy et McGuiness, 2000**].

Nous pouvons également citer l'utilisation des ontologies dans les processus de conception routinière de type configuration de produits [**Zhang et al., 2003**] [**Lau et al., 2009**]. En effet, des problèmes similaires mènent en général à des solutions similaires. Les ontologies permettent ici de retrouver les problèmes similaires dans un premier temps et les solutions qui ont été obtenues dans un second temps.

5.2 Modèle d'ontologie pour la formalisation des connaissances métier

D'un point de vue conceptuel, les systèmes conçus sont généralement caractérisés au moyen de paramètres de conception [**Suh, 1998**], ou de variables de conception [**Yannou et al., 2010**], c'est-à-dire un ensemble de descripteurs auxquels le concepteur donne des valeurs.

Les ontologies en conception sont souvent orientées « taxonomie », c'est-à-dire qu'un cadre est fourni par une hiérarchie de concepts avec, au mieux, la sémantique de chaque concept et, dans certains cas :

- l'ajout de règles comme dans [**Furtado et al., 2001**] ou [**Wriggers et al., 2007**] reflétant les dépendances entre les propriétés d'un concept, ou,
- l'intégration du concept de contrainte dans l'ontologie comme, par exemple dans [**Simoff et Maher, 1998**], [**Yang et al., 2008**] ou [**Mao et al., 2010**]. Ceci permet de comprendre les contraintes inhérentes au système à concevoir mais pas de guider dans leur utilisation.

Or, dans le cadre de nos travaux, nous avons besoin d'un référentiel permettant la capitalisation des connaissances en conception afin de :

- formaliser des connaissances de conception et de planification de projet grâce à des concepts, des variables, leurs domaines et des contraintes sur ces variables dans le but de :
  - fournir un cadre facilitant l'expression des exigences système et des solutions au sein des alternatives système,
  - fournir un cadre à la formalisation d'objectifs dans une structure générique de projets avec des concepts adaptés ainsi que leurs variables, domaines et contraintes,
  - réutiliser/adapter les solutions précédemment développées et ainsi réduire la durée et le coût du processus de planification projet/développement système,
- formaliser des connaissances inhérentes au couplage conception de système / planification dans le but de définir des méthodes d'aide à ce couplage.

De manière plus précise, le référentiel proposé est basé sur une ontologie permettant de :

- représenter une hiérarchie de concepts relatifs à la fois à la conception de systèmes et à la planification de projets,
- caractériser chaque concept à l'aide de variables,
- caractériser chaque variable grâce à son domaine admissible,
- capitaliser de la connaissance sur les concepts en rajoutant des contraintes sur les variables.

Le référentiel proposé permettra, par la suite :

- d'associer un concept à chaque système afin de le caractériser à un niveau conceptuel (concept), ainsi qu'à un niveau détaillé (variables),
- d'associer un concept à chaque alternative système afin de la caractériser et ainsi faciliter la formalisation des solutions en précisant les variables à renseigner,
- d'associer un concept à chaque projet afin de le caractériser et ainsi faciliter sa définition en précisant les variables à renseigner,
- d'associer un concept à chaque tâche afin de la caractériser et ainsi faciliter sa définition en précisant les variables à renseigner de même que sa planification au sein du projet.

Compte tenu des besoins exprimés ci-dessus, nous proposons une ontologie dont les concepts sont hiérarchisés selon une loi de type généralisation/spécialisation sans héritage multiple. L'ontologie est donc représentée à l'aide d'une arborescence dont la racine est le concept « Universel » et les feuilles sont les concepts les plus spécialisés. Dans le but de caractériser les systèmes, les alternatives système, les projets de conception et les tâches du projet, nous proposons que chaque concept soit décrit à l'aide :

- d'un ou plusieurs **modèles de variables**. Les modèles seront réutilisés, par recopie, afin de caractériser chacune de nos entités,
- d'un ou plusieurs **modèles de domaines**, propre à chaque variable et décrivant le domaine admissible (par exemple : $\mathbb{R}^+$, $\mathbb{N}$, {blanc, rouge, bleu}, etc.),
- d'un ou plusieurs **modèles de contraintes** afin de capitaliser des connaissances particulières à un concept donné. Chaque modèle de contrainte lie un ou plusieurs modèles de variables.

Une contrainte est une condition à satisfaire. Les modèles de contraintes varient en fonction de plusieurs critères :

- le type de variables sur lesquelles elles portent : symbolique, continue, temporelle…
- le type de contrainte : liste de valeurs autorisées (x = {1, 2, 3} ; y = {rouge, vert}), combinaisons de valeurs autorisées ((x = 1, y = rouge), (x = 2, y = vert)), de formules mathématiques ($y = x^2 + 2$) ou de règles logiques ($(x \vee y) \wedge z$).

Ainsi, conformément aux travaux présentés dans [**Benaissa et Lebbah, 2011**], un concept C_k est caractérisé, outre son nom et sa sémantique propre, par un **modèle de problème de satisfaction de contraintes**, c'est-à-dire par le triplet (V_{Ck}, D_{Ck}, Σ_{Ck}) avec :

- V_{Ck} : l'ensemble des modèles de variables du concept C_k,
- D_{Ck} : l'ensemble des modèles de domaines du concept C_k,
- Σ_{Ck} : l'ensemble des modèles de contraintes du concept C_k.

L'illustration d'un concept nommé « concept k » est proposée sur la figure 5.1.

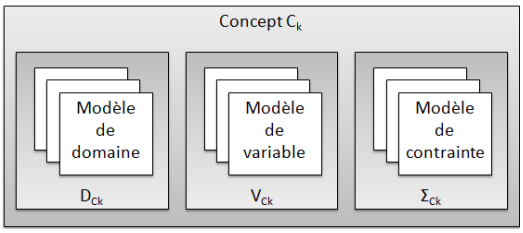


Figure 5.1 : Illustration d'un concept

Si un concept C_j a pour concept père C_i, alors le concept C_j hérite du concept C_i l'ensemble des modèles de variables, l'ensemble des modèles de domaines et l'ensemble des modèles de contraintes. En outre, le concept C_j possède d'autres modèles de variables ($V^0_{C_j}$), de domaines ($D^0_{C_j}$) et de contraintes ($\Sigma^0_{C_j}$) qui lui sont propres et le spécialisent. Ceci est formalisé ci-dessous.

- Modèle de variable : $V_{C_j} = V_{C_i} \cup V^0_{C_j}$

V_{C_i} : ensemble des modèles de variables du concept Ci

V_{C_j} : ensemble des modèles de variables du concept Cj tel que Cj est une spécialisation de Ci

$V^0_{C_j}$: ensemble des modèles de variables propres au concept Cj

- Modèle de domaine : $D_{C_j} = D_{C_i} \cup D^0_{C_j}$

D_{C_i} : ensemble des modèles de domaines du concept Ci

D_{C_j} : ensemble des modèles de domaines du concept Cj tel que Cj est une spécialisation de Ci

$D^0_{C_j}$: ensemble des modèles de domaines propres au concept Cj

- Modèle de contrainte : $\Sigma_{C_j} = \Sigma_{C_i} \cup \Sigma^0_{C_j}$

Σ_{C_i} : ensemble des modèles de contraintes du concept Ci

Σ_{C_j} : ensemble des modèles de contraintes du concept Cj tel que Cj est une spécialisation de Ci

$\Sigma^0_{C_j}$: ensemble des modèles de contraintes propres au concept Cj

Ces concepts pourront ensuite être associés à des systèmes S et à des alternatives système AS afin d'aider le concepteur à caractériser, d'une part, les exigences inhérentes au système à concevoir et, d'autre part, les solutions (alternatives système) qu'il conçoit pour répondre aux exigences. Similairement, des concepts pourront être associés à des projets et tâches de projets.

L'ensemble de cette connaissance métier est décrit et validé par un (ou des) expert(s) métier dont le rôle est de formaliser l'ontologie dans le but de la diffuser largement dans l'entreprise. Une ontologie n'est jamais figée. Des évolutions sont constamment possibles lorsque :

- de nouveaux concepts doivent être rajoutés (nouveaux concepts de systèmes par exemple),
- de nouveaux modèles de variables (avec leur modèle de domaines) sont requis afin de mieux caractériser des concepts existants,
- de nouveaux modèles de contraintes apparaissent, soit par exploitation de mécanismes de retour d'expérience, soit par génération de connaissances directement par les experts,
- des concepts, modèles de variables, de domaines ou de contraintes deviennent obsolètes et doivent être mis à jour.

5.3 Utilisation de notre proposition de modèle d'ontologie

Dans la suite de ce mémoire, nous proposons de noter en majuscule les modèles de variables (V), les modèles de domaines (D) et les modèles de contraintes (Σ). Les variables, domaines et contraintes utilisés dans la caractérisation des différentes entités de conception et de planification seront notés en minuscules (respectivement v, d et σ).

5.3.1 Caractérisation des exigences système ES

Dans le cas d'un système, le directeur de programme doit, en fonction des besoins exprimés par le client et de l'expertise du responsable de conception, choisir un concept CS (Concept Système) qu'il associe au système (activité « Demander création Projet et Système, fig. 2.13). Dès lors, l'ensemble des modèles de variables (noté V_{CS}), inhérent au concept est recopié dans les exigences système. Cette copie est appelée ensemble des **variables conceptuelles système,** notée v^S_{CS}. L'ensemble v^S_{CS} est défini par : $v^S_{CS} = \left\{ v^s_i \middle| v^s_i \overset{copie}{\Longleftarrow} V_i, \forall V_i \in V_{CS} \right\}$ *avec :*

CS : le concept associé par le concepteur au système S,

v^S_{CS} *: l'ensemble des variables conceptuelles système d'un système S,*

V_i *: le modèle de variable i,*

v^S_i *: la copie du modèle de variable* V_i*,*

V_{CS} *: l'ensemble des modèles de variables du concept CS,*

$\overset{copie}{\Longleftarrow}$ *: l'opérateur de copie du modèle de variable de droite dans la variable de gauche.*

Similairement, l'ensemble des modèles de domaines (noté D_{CS}), inhérent au concept CS est recopié dans les exigences système. Cette copie est appelée ensemble des **domaines des variables**

conceptuelles système, notée d_{CS}^{S}. L'ensemble d_{CS}^{S} est défini par : $d_{CS}^{S} = \left\{ d_i^S \middle| d_i^S \overset{copie}{\Longleftarrow} D_i, \forall D_i \in D_{CS} \right\}$ *avec :*

d_i *: la copie du modèle de domaine* D_i *pour un système S,*

D_i^S *: le modèle de domaine i,*

D_{CS} *: l'ensemble des modèles de domaines du concept CS.*

Similairement, l'ensemble des modèles de contraintes (noté Σ_{CS}) inhérent au concept est recopié dans les exigences système. Cette copie est appelée ensemble des **contraintes conceptuelles système**, noté σ_{CS}^{S}. L'ensemble σ_{CS}^{S} est défini par : $\sigma_{CS}^{S} = \left\{ \sigma_i^S \middle| \sigma_i^S \overset{copie}{\Longleftarrow} \Sigma_i, \forall \Sigma_i \in \Sigma_{CS} \right\}$ *avec :*

σ_i^S *: la copie du modèle de contrainte* Σ_i *pour un système S,*

Σ_i *: le modèle de contrainte i,*

Σ_{CS} *: l'ensemble des modèles de contraintes du concept CS*

Le concepteur peut alors exprimer les besoins du client sous forme de contraintes définies à partir des variables conceptuelles système (activité « Réaliser Tâche TE », fig. 2.13). Si nécessaire, l'ajout de nouvelles variables, non précisées par le concept CS et permettant une description plus précise des besoins est possible. Ces dernières sont nommées **variables supplémentaires système** et leur ensemble est noté v_{supp}^{S}. L'ajout des variables supplémentaires entraîne l'ajout de leur domaine de validité. Il ne s'agit donc pas de connaissances métiers mais de besoins exprimés par le concepteur lors de son activité. Ceci nous permet de définir d_{supp}^{S}, l'ensemble des domaines des variables supplémentaires ajoutées. L'ensemble des contraintes traduisant les exigences est noté σ_{Ex}^{S}. Ainsi, une entité Exigences Système ES est constituée de :

- l'ensemble des variables conceptuelles système (v_{CS}^{S}),
- l'ensemble des domaines des variables conceptuelles système (d_{CS}^{S}),
- l'ensemble des contraintes conceptuelles système (σ_{CS}^{S}),
- l'ensemble des variables supplémentaires système (v_{supp}^{S}),
- l'ensemble des domaines des variables supplémentaires système (d_{supp}^{S}),
- l'ensemble des contraintes exigences (σ_{Ex}^{S}) portant sur l'ensemble de variables $v_{CS}^{S} \cup v_{supp}^{S}$.

L'ensemble des variables conceptuelles système et l'ensemble des variables supplémentaires système constituent l'ensemble des **variables système** noté v^S, c'est-à-dire, des variables utilisées pour définir les exigences système. Cet ensemble est défini par : $v^S = v_{CS}^{S} \cup v_{supp}^{S}$.

Similairement, à partir de l'ensemble des domaines des variables conceptuelles système et de l'ensemble des domaines des variables supplémentaires système constituent l'ensemble des **domaines des variables système** noté d^S. Cet ensemble est défini par : $d^S = d_{CS}^{S} \cup d_{supp}^{S}$.

Enfin, l'ensemble des contraintes exigences et l'ensemble des contraintes système conceptuelles constituent l'ensemble des **contraintes système** (noté σ^S). Cet ensemble est défini par : $\sigma^S = \sigma_{CS}^{S} \cup \sigma_{Ex}^{S}$.

Les ensembles de variables, domaines et contraintes évoqués ci-dessus sont illustrés sur la figure 5.2.

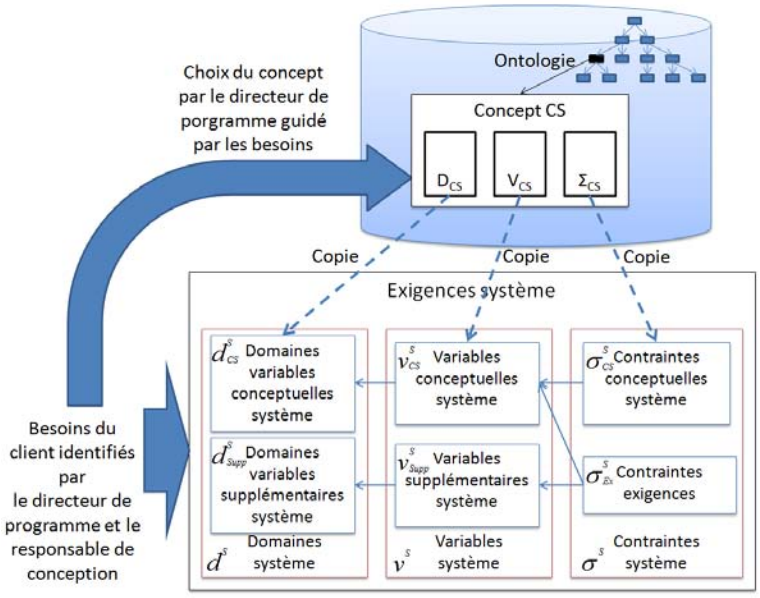


Figure 5.2 : Définition des variables, des domaines et des contraintes système

5.3.2 Caractérisation des alternatives système AS

Dans le cas d'une alternative système, il est nécessaire de disposer de l'ensemble des variables utilisées depuis le début de la conception système ainsi que de leur domaine afin de pouvoir leur affecter une valeur. En effet, une variable présente dans les exigences système doit être également présente dans chaque alternative système AS afin de dérouler la tâche de conception des alternatives et affecter une valeur à cette variable qui soit propre à l'alternative dans laquelle elle se trouve et répondant effectivement aux exigences système. De même, il est nécessaire de disposer, au sein de l'alternative système, des contraintes système afin de pouvoir vérifier la validité des valeurs des variables relativement à ces contraintes.

Pour une alternative système AS, le responsable de conception doit associer un concept noté CA de l'ontologie (activité « Demander création de m alternatives ASi », fig. 2.13). Ce concept doit être, soit identique au concept CS, le concept associé au système, soit une spécialisation de ce dernier.

Similairement au système, le concept CA fournit à l'alternative, par recopie :

- un ensemble de **variables conceptuelles alternative** noté v_{CA}^{AS},
- un ensemble de **domaines de variables conceptuelles alternative** noté d_{CA}^{AS},
- un ensemble de **contraintes conceptuelles alternative** noté σ_{CA}^{AS}.

L'ensemble des variables conceptuelles alternative v_{CA}^{AS} est défini par :

$$v_{CA}^{AS} = \left\{ v_i^{AS} \middle| v_i^{AS} \overset{copie}{\Longleftarrow} V_i, \forall V_i \in V_{CA} \right\}$$

avec :

V_i *: le modèle de variable i,*

v_i^{AS} *: la copie du modèle de variable* V_i *dans l'alternative système AS,*

V_{CA} *: l'ensemble des modèles de variables du concept CA.*

L'ensemble des domaines des variables conceptuelles alternative d_{CA}^{AS} est défini par :

$$d_{CA}^{AS} = \left\{ d_i^{AS} \middle| d_i^{AS} \overset{copie}{\Longleftarrow} D_i, \forall D_i \in D_{CA} \right\}$$

avec :

D_i *: le modèle de domaine i,*

d_i^{AS} *: la copie du modèle de domaine* D_i *dans l'alternative système AS,*

D_{CA} *: l'ensemble des modèles de domaines du concept CA.*

L'ensemble des contraintes conceptuelles alternatives σ_{CA}^{AS} est défini par :

$$\sigma_{CA}^{AS} = \left\{ \sigma_i^{AS} \middle| \sigma_i^{AS} \overset{copie}{\Longleftarrow} \Sigma_i, \forall \Sigma_i \in \Sigma_{CA} \right\}$$

avec :

Σ_i *: modèle de contrainte i,*

σ_i^{AS} *: copie du modèle de contrainte* Σ_i *dans l'alternative système AS,*

Σ_{CA} *: l'ensemble des modèles de contraintes du concept CA*

L'ensemble des **variables supplémentaires système** (ensemble v_{supp}^S), l'ensemble des domaines de ces variables (d_{supp}^S) ainsi que l'ensemble des **contraintes exigence système** (σ_{Ex}^S) sont recopiés dans l'alternative. Ceci permet au responsable de conception de :

- définir les valeurs de ces variables en restreignant leur domaine vers une valeur unique au cours de la conception,
- puis de réaliser l'activité de vérification de l'alternative en testant la satisfaction de chaque contrainte.

Ainsi, nous définissons v'^S_{supp}, la copie de l'ensemble v_{supp}^S par :

$$v'^S_{supp} = \left\{ v'_{supp\ j} \middle| v'_{supp\ j} \overset{copie}{\Longleftarrow} v_{supp\ j}, \forall v_{supp\ j} \in v_{supp}^S \right\}$$

avec :

v'^S_{supp} *: l'ensemble des copies des variables supplémentaires système d'un système S dans l'alternative système AS,*

$v_{supp\ j}$ *: la variable supplémentaire système j,*

$v'_{supp\ j}$ *: la copie de la variable supplémentaire système* v_j.

Nous définissons d'^S_{supp}, la copie de l'ensemble d_{supp}^S par :

$$d'^S_{supp} = \left\{ d'_{supp\ j} \middle| d'_{supp\ j} \overset{copie}{\Longleftarrow} d_{supp\ j}, \forall d_{supp\ j} \in d_{supp}^S \right\}$$

avec :

d'^S_{supp} *: l'ensemble des copies des domaines des variables supplémentaires système d'un système S dans l'alternative système AS,*

$d_{supp\ j}$ *: le domaine de la variable supplémentaire système j,*

$d'_{supp\ j}$ *: la copie du domaine* $d_{supp\ j}$.

Nous définissons également σ'^S_{Ex}, la copie de l'ensemble σ_{Ex}^S par :

$$\sigma'^S_{Ex} = \left\{ \sigma'_{Ex\ j} \middle| \sigma'_{Ex\ j} \overset{copie}{\Longleftarrow} \sigma_{Ex\ j}, \forall \sigma_{Ex\ j} \in \sigma_{Ex}^S \right\}$$

avec :

σ'^S_{Ex} *: l'ensemble des copies des contraintes exigence d'un système S dans l'alternative système AS,*

$\sigma_{Ex\ j}$ *: la contrainte exigence j associée au système S,*

$\sigma'_{Ex\ j}$ *: la copie de la contrainte exigence* $\sigma_{Ex\ j}$.

Les **variables conceptuelles système** (v_{CS}^{S}), les **domaines associés aux variables conceptuelles système** (d_{CS}^{S}) et les **contraintes conceptuelles système** (σ_{CS}^{S}) ne sont pas recopiées car le concept CA étant identique ou une spécialisation du concept CS, ces dernières sont déjà inclues dans l'ensemble des **variables conceptuelles alternative** (v_{CA}^{AS}), les **domaines associés aux variables conceptuelles alternative** (d_{CA}^{AS}) et dans l'ensemble des **contraintes conceptuelles alternative** (σ_{CA}^{AS}) grâce au mécanisme d'héritage.

Le concepteur de l'alternative système peut également rajouter des variables propres à l'alternative (activité « Processus de réalisation tâche TD », fig. 2.13). Ces variables sont inclues dans l'ensemble des **variables supplémentaires alternative** noté v_{supp}^{AS}. Parallèlement, les domaines de ces variables doivent être ajoutés. Ils constituent l'ensemble d_{supp}^{AS} des domaines des variables supplémentaires ajoutées. Similairement, le concepteur peut rajouter des contraintes propres à l'alternative. Ces contraintes sont inclues dans l'ensemble des **contraintes supplémentaires alternative** noté σ_{supp}^{AS}.

L'ensemble des variables conceptuelles alternative (v_{CA}^{AS}), l'ensemble des copies des variables supplémentaires système (v'^{S}_{supp}) et l'ensemble des variables supplémentaires alternative (v_{supp}^{AS}) constituent l'ensemble des **variables alternative** noté v^{AS}, c'est-à-dire, des variables utilisées pour définir l'alternative système. Cet ensemble est défini par :

$$v^{AS} = v_{CA}^{AS} \cup v'^{S}_{supp} \cup v_{supp}^{AS}$$

L'ensemble des domaines des variables conceptuelles alternative (d_{CA}^{AS}), l'ensemble des copies des domaines des variables supplémentaires système (d'^{S}_{supp}) et l'ensemble des domaines des variables supplémentaires alternative (d_{supp}^{AS}) constituent l'ensemble des **domaines des variables alternative** noté d^{AS}. Cet ensemble est défini par :

$$d^{AS} = d_{CA}^{AS} \cup d'^{S}_{supp} \cup d_{supp}^{AS}$$

Similairement, l'ensemble des contraintes conceptuelles alternative (σ_{CA}^{AS}), l'ensemble des copies des contraintes exigences (σ'^{S}_{Ex}) et l'ensemble des contraintes supplémentaires alternatives (σ_{supp}^{AS}) constituent l'ensemble des **contraintes alternative** (σ^{AS}). Cet ensemble est défini par :

$$\sigma^{AS} = \sigma_{CA}^{AS} \cup \sigma'^{S}_{Ex} \cup \sigma_{supp}^{AS}$$

À l'issue de la conception, toutes les variables alternative sont associées à des valeurs (leur domaine est un singleton). Ces valeurs pourront alors être comparées aux contraintes alternative. Si toutes les valeurs respectent toutes les contraintes, alors l'alternative répond bien aux besoins du client et peut être qualifiée de « vérifiée ».

Les ensembles de variables, domaines et contraintes évoqués ci-dessus sont illustrés sur la figure 5.3.

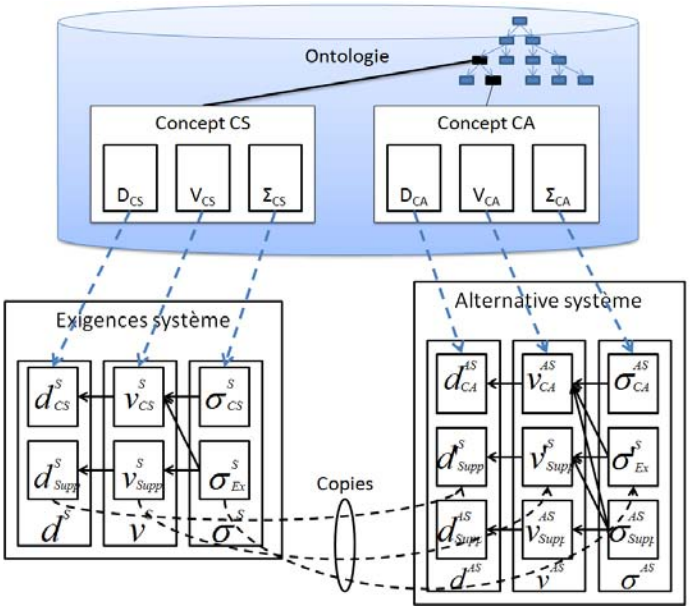


Figure 5.3 : Définition des variables, domaines et contraintes pour une alternative système

5.3.3 Caractérisation d'un projet et des tâches TE et TD

Similairement à la conception système, chaque entité de planification de projet peut être associée à un concept permettant de la caractériser au moyen de variables, domaines et contraintes. Ainsi, un projet de conception, une tâche TE ou une tâche TD est associé à un concept permettant de décrire leur type et de formaliser leurs caractéristiques. L'association de concepts à ces entités permet de les décrire, d'en faciliter le renseignement et la planification, de capitaliser les connaissances inhérentes et, in fine, de faciliter le couplage.

Nous proposons donc deux concepts différents pour caractériser ces entités :

- le concept Projet (associé à une entité Projet de conception),
- le concept Tâche (associé aux entités Tâche TE et Tâche TD).

Le concept Projet comporte :

- un ensemble de modèles de variables (V_{CP}),
- un ensemble de modèles de domaines (D_{CP}),
- un ensemble de modèles de contraintes (Σ_{CP}).

Le concept Tâche comporte :

- un ensemble de modèles de variables (V_{CT}),
- un ensemble de modèles de domaines (D_{CT}),
- un ensemble de modèles de contraintes (Σ_{CT}).

Les triplets (V_{CP}, D_{CP}, Σ_{CP}) et (V_{CT}, D_{CT}, Σ_{CT}) sont respectivement les modèles de problème de satisfaction de contraintes inhérents aux concepts Projet et Tâche.

Lors de la création d'un projet (activité « Demander création Projet et Système », fig.2.13), le concept Projet lui est associé. De même, chaque tâche TE ou TD du projet est associée au concept Tâche. Ainsi, le projet P se voit rajouter, par recopie :

- un ensemble de variables conceptuelles projet (v_{CP}^{P}),
- un ensemble de domaines de variables conceptuelles projet (d_{CP}^{P}),

- un ensemble de contraintes conceptuelles projet (σ_{CP}^{P}).

Pour un projet P, le responsable de planification peut souhaiter rajouter des variables supplémentaires projet, formant ainsi l'ensemble v_{supp}^{P} (non représenté sur la figure 2.13). Ceci implique l'ajout de leur domaine, formant ainsi l'ensemble des domaines de variables supplémentaires projet noté d_{supp}^{P}. De même, il est possible de rajouter des contraintes portant sur l'ensemble des variables et noté σ_{supp}^{P} (ces contraintes peuvent représenter des exigences sur le projet particulier). Un projet comporte in fine :

- un ensemble de variables projet noté v^P avec $v^P = v_{CP}^{P} \cup v_{supp}^{P}$,
- un ensemble de domaines de variables projet noté d^P avec $d^P = d_{CP}^{P} \cup d_{supp}^{P}$,
- un ensemble de contraintes projet noté σ^P avec $\sigma^P = \sigma_{CP}^{P} \cup \sigma_{supp}^{P}$.

Une tâche quelconque se voit également rajouter, par recopie :

- un ensemble de variables conceptuelles tâche (v_{CT}^{T}),
- un ensemble de domaines de variables conceptuelles tâche (d_{CT}^{T}),
- un ensemble de contraintes conceptuelles tâche (σ_{CT}^{T}).

Pour une tâche T, le responsable de planification peut rajouter des variables supplémentaires (ensemble v_{supp}^{T}) avec leur domaine (d_{supp}^{T}) ainsi que de nouvelles contraintes (σ_{supp}^{T}) (activités « Saisie des informations / Planification tâche TE » et « Processus de saisie des informations / Planification tâche TD », fig. 2.13).

Une tâche T comporte in fine :

- un ensemble de variables tâche noté v^T avec $v^T = v_{CT}^{T} \cup v_{supp}^{T}$,
- un ensemble de domaines de variables tâche noté d^T avec $d^T = d_{CT}^{T} \cup d_{supp}^{T}$,
- un ensemble de contraintes tâche noté σ^T avec $\sigma^T = \sigma_{CT}^{T} \cup \sigma_{supp}^{T}$.

Les ensembles de variables, domaines et contraintes pour un projet de conception P et pour une tâche quelconque T sont illustrés sur la figure 5.4.

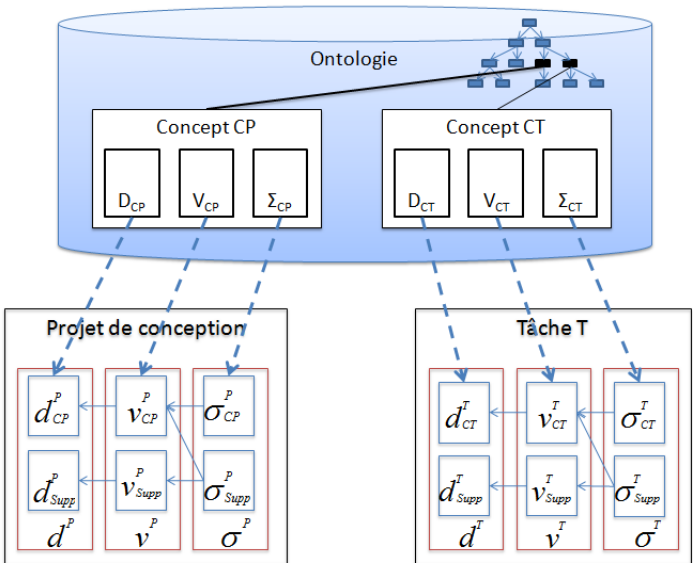


Figure 5.4 : Variables, domaines et contraintes pour un projet de conception et une tâche T

L'utilisation du modèle d'ontologie proposé dans cette section reste générique. Pour notre proposition, nous réduisons l'étendue du modèle dans la section suivante en justifiant nos choix.

5.4 Construction de l'ontologie proposée

Les concepts ayant pour but d'être associés aux systèmes et aux alternatives système sont en général liés au domaine métier auquel le système ou l'alternative se rapporte. Les modèles de variables peuvent donc être très différents en fonction du système à concevoir. Nous proposons cependant quelques variables (le coût, la masse et le risque), domaines associés et contraintes génériques pouvant se décliner quel que soit le système à concevoir. Ces modèles de variables génériques sont directement associés au concept « Système » dans l'ontologie, c'est-à-dire au concept le plus générique concernant les systèmes à concevoir. Ainsi, toutes les spécialisations du concept « Système » héritent de ces trois modèles de variable.

Concernant la planification de projet, les concepts Projet et Tâche peuvent être caractérisés avec un nombre réduit de variables, domaines et contraintes ne dépendant pas du domaine d'application. Ainsi, pour un projet, il s'agit de la durée, de la date de début, de la date de fin, des type et quantité de ressources, du coût et du risque. Pour une tâche, il s'agit de la durée, de la date de début, de la date de fin, des type et quantité de ressources, du coût et du risque. À partir de ces concepts, il est possible de définir de nouveaux concepts spécialisés dans l'ontologie. Par exemple, un concept de projet particulier peut être défini. Il en est de même pour une tâche.

La figure 5.5 ci-dessous illustre une ontologie générique, nécessairement partielle, faisant apparaître les concepts « Système », « Projet » et « Tâche » (les modèles de variables, de domaines et de contraintes ne sont pas représentés). Le concept « Universel » est un concept abstrait qui représente « tout ». Nous plaçons les concepts « Système », « Projet » et « Tâche » au même niveau immédiatement en dessous du concept universel. Dans une ontologie réelle, les concepts de systèmes seront une spécialisation du concept « Système » et seront donc placés en dessous. Nous illustrons cette ontologie sur un exemple dans la section 5.6.1.

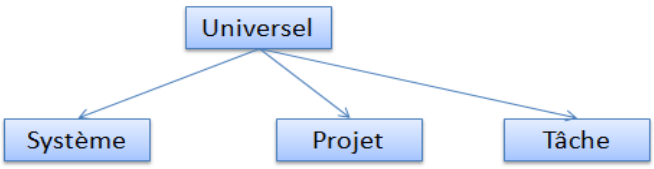


Figure 5.5 : Proposition d'ontologie partielle

Nous proposons de traiter les variables (et modèles de variable) symboliques, continues et temporelles. Par conséquent, les contraintes (et modèles de contrainte) s'appliquant sur ces variables sont :

- des listes de valeurs autorisées dans le cas d'une contrainte s'appliquant à une variable symbolique,
- des matrices de compatibilité pour les contraintes s'appliquant à plusieurs variables symboliques et/ou numériques,
- des fonctions mathématiques dans le cas d'une contrainte s'appliquant à une ou plusieurs variables numériques discrètes ou continues,
- des relations temporelles (par exemple précédence) pour les contraintes s'appliquant à des variables temporelles.

5.5 Evolution du diagramme de classes

À partir de ces définitions, nous complétons le modèle de classes proposé dans la section 2.2. D'une part, il s'agit de proposer une ontologie de concepts comprenant des modèles de variables, des modèles de domaine et des modèles de contraintes. Ces concepts sont associés aux systèmes S, aux alternatives systèmes AS, aux projets de conception P ou aux tâches TE et TD. L'entité exigences systèmes ES étant unique pour un système donné, elles ont le même concept que l'entité Système S à laquelle elles se rapportent. Un seul concept est associé à une instance des entités évoquées ci-dessus. D'autre part, le modèle ci-dessous permet d'associer des variables aux entités et des contraintes sur ces variables dans le but de représenter les besoins du client. Ces évolutions sont représentées dans les figures 5.6 et 5.7.

L'association réflexive sur la classe Concept représente la relation de généralisation/spécialisation entre concepts. La généralisation/spécialisation est un processus de modélisation permettant de lier un concept générique et un ou plusieurs concepts spécialisés. Le concept générique contient les propriétés (modèles de variable, de domaine et de contrainte) communes à tous les sous-concepts. Les concepts spécialisés héritent des propriétés du concept générique et de plus, définissent des propriétés qui leur sont spécifiques.

Les variables et les contraintes (tout comme les modèles de variables et modèles de contraintes) sont liées entre elles. En effet, une contrainte a pour objet de contraindre une ou plusieurs variables, une contrainte est donc composée d'un ensemble ordonné de variables.

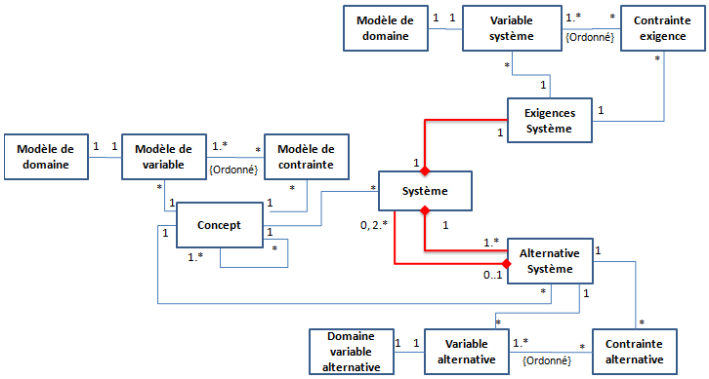


Figure 5.6 : Diagramme de classe modifié en conception de système

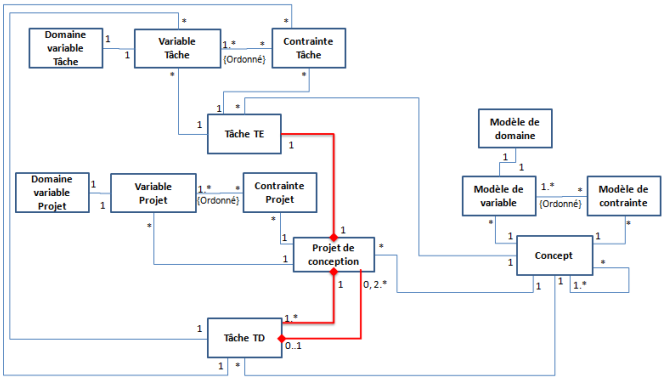


Figure 5.7 : Diagramme de classe modifié en planification de projet

5.6 Illustration de l'ontologie

L'exemple que nous proposons se décompose en deux parties distinctes. La première partie concerne la définition d'un système « Longeron » pour une aile d'avion. La seconde partie aborde la définition du projet de conception du longeron et de la tâche de recueil des exigences système et de recherche d'alternatives tâche TE.

5.6.1 Ontologie d'un avion d'affaire (longeron)

Nous proposons d'utiliser l'ontologie de concepts représentée sur la figure 5.8. Pour chaque concept C, le triplet (V, D, Σ) est représenté entre parenthèses.

- Universel (Ø, Ø, Ø)

- Système (
{Coût (C), Masse (M), Risque (R)} ;
{D_C : [0, +∞ [, D_M : [0, +∞[, D_R : [0, 1]} ;
Ø)

- Longeron (
{C, M, R, Longueur (Lo), Section (S)} ;
{D_C : [0, +∞ [, D_M : [0, +∞[, D_R : [0, 1], D_{Lo}: [0, +∞[, D_S: {U,I,T}};
Ø)

- Longeron_U (
{C, M, R, Lo, S} ;
{ D_C : [0, +∞ [, D_M : [0, +∞[, D_R : [0, 1], D_{Lo}: [0, +∞[, D_S: {U,I,T}} ;
{S = 'U'})

- Projet (
{Durée (Du), Date début (Dd), Date fin (Dt), Type ressources (Type_Ress), Quantité ressouces (Qté_Ress), Coût (C), Risque (Risk)} ;
{D_{Du}: [0, +∞[, D_{Dd}: [0, +∞[, D_{Dt}: [0, +∞[, D_{Type_Ress}: {Ingé_Calcul, Ingé_Structure, Ingé_Elec}, $D_{Qté\_Ress}$: [0, +∞[, D_C : [0, +∞[, D_{Risk} : [0, 1]} ;
{Dt ≤ Du + Dd})

- Tâche (
{Durée (Du), Date début (Dd), Date fin (Dt), Type ressources (Type_Ress), Quantité ressouces (Qté_Ress), Coût (C), Risque (Risk)} ;
{D_{Du}: [0, +∞[, D_{Dd}: [0, +∞[, D_{Dt}: [0, +∞[, D_{Type_Ress}: {Ingé_Calcul, Ingé_Structure, Ingé_Elec}, $D_{Qté\_Ress}$: [0, +∞[, D_C : [0, +∞[, D_{Risk} : [0, 1]} ;
{Dt ≤ Du + Dd})

Figure 5.8 : Ontologie utilisée

5.6.2 Définition d'un système S et d'une alternative système AS

Lors de la conception du système, le directeur de programme, assisté éventuellement par le responsable de conception, choisit un concept dans l'ontologie et l'associe au système à concevoir. Dans l'exemple, le concept choisi est celui du longeron. Celui-ci est associé au système ayant pour IdS = X9. Ce concept comporte :

- deux modèles de variables qui lui sont propres (S et Lo) et hérite également des variables de son concept père (C, M et R),
- les modèles de domaines associés à ces variables (D_C, D_M, D_R, D_S, D_{Lo})et,
- aucun modèle de contrainte.

Ces modèles vont être recopiés respectivement dans les variables conceptuelles système (c, m, r, s et lo) et les domaines des variables conceptuelles système (d_c, d_m, d_r, d_s, et d_{lo}) au sein des

exigences système ES. Ensuite, le responsable de conception définit les contraintes exigences représentant les besoins du client. Si nécessaire, il peut ajouter des variables et leurs domaines associés afin d'affiner l'expression des exigences du client : ici, il ajoute la variable épaisseur (e). Les contraintes exigences sont ici : lo > 1m et e < 2cm. Ceci est illustré sur la figure 5.9.

Quand les exigences sont définies, le responsable de conception définit l'alternative système qu'il souhaite développer, d'identifiant IdAS = X9.1. Similairement à la définition des exigences, il va commencer par choisir un concept (ici le concept Longeron_U) pour l'alternative système AS identique ou spécialisé par rapport au concept du système.

Ce concept comporte :

- cinq modèles de variables (C, M, R, S et Lo) hérités du concept Longeron,
- les modèles de domaines associés à ces variables (D_C, D_M, D_R, D_S et D_{Lo}) et,
- un modèle de contrainte (S = 'U').

Ces modèles vont être recopiés respectivement dans les variables conceptuelles alternative (c, m, r, s et lo), dans les domaines des variables conceptuelles alternatives (d_C, d_M, d_R, d_s et d_{lo}) et dans les contraintes conceptuelles alternative (s = 'U'). Ensuite, les contraintes exigences (lo > 1m et e < 2cm) ainsi que la variable supplémentaire (épaisseur (e)) définie dans les exigences système ES sont également recopiées dans l'alternative système AS. Enfin, le responsable de conception ajoute une variable supplémentaire permettant de caractériser l'alternative système : la variable matériau (m) dont le domaine est d_m = {'Alu'}. Il n'y a pas de contraintes supplémentaires alternative. Ceci est illustré sur la figure 5.10.

Lorsque la conception est terminée, les domaines des variables inclues dans l'alternative système AS doivent être réduits pour arriver à des singletons, qui pourront être comparés avec les contraintes afin de vérifier que l'alternative système AS satisfait bien les exigences système ES.

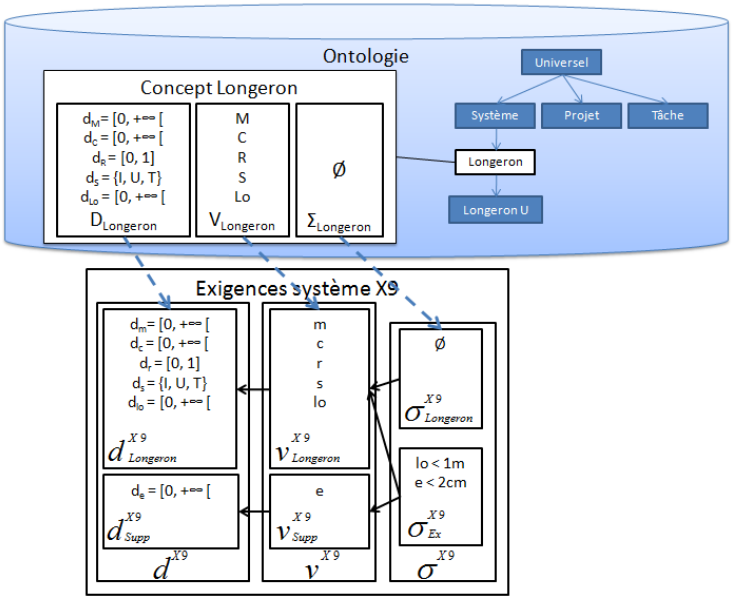


Figure 5.9 : Exemple de définition des exigences système pour la conception d'un longeron

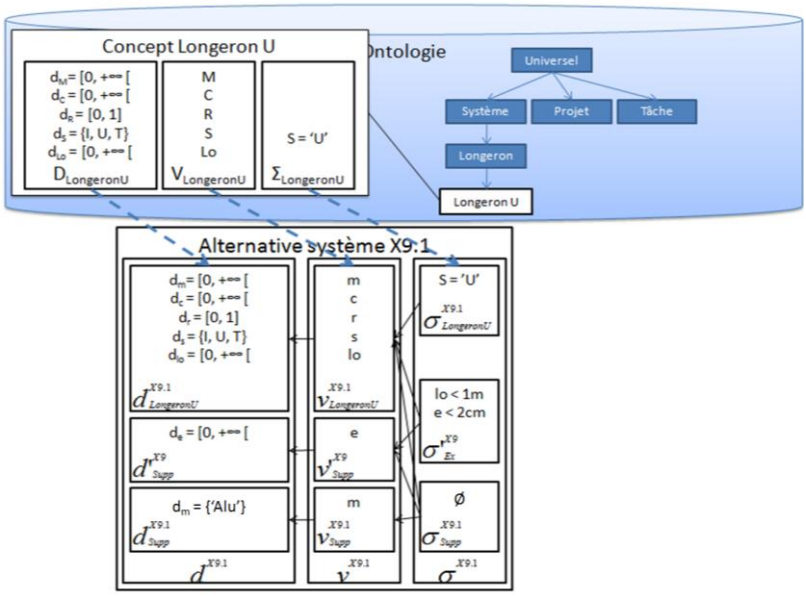


Figure 5.10 : Exemple de définition d'une alternative système pour la conception d'un longeron

5.6.3 Définition d'un projet de conception P et d'une tâche TE

Nous illustrons dans cet exemple la caractérisation d'un projet P et d'un tâche TE. Pour une tâche TD, la démarche est strictement identique.

Le directeur de programme va dans un premier temps associer le concept Projet au projet à planifier IdP = P9 (cf. section 5.6.1). Ce concept comporte :

- sept modèles de variables (Durée (Du), Date début (Dd), Date fin (Df), Type ressources (Type_Ress), Quantité ressouces (Qté_Ress), Coût (C), Risque (Risk)),
- sept modèles de domaines associés à ces variables (D_{Du}, D_{Dd}, D_{Dt}, D_{Ress}, $D_{Qté\_Ress,}$ D_C , D_{Risk}) et,
- un modèle de contrainte Dt ≤ Du + Dd.

Ces modèles vont être recopiés respectivement dans les variables conceptuelles projet (durée (du), date début (dd), date fin (df), type ressources (type_ress), quantité ressouces (qté_ress), coût (c), risque (risk)), dans les domaines de variables conceptuelles projet (d_{Du}, d_{Dd}, d_{Df}, d_{Ress}, $d_{Qté\_Ress,}$ d_C , d_{Risk}) et dans les contraintes conceptuelles projet (df ≤ du + dd). Ensuite, le responsable de planification définit les contraintes supplémentaires représentant les besoins du client. Si nécessaire, il peut ajouter des variables et les domaines associés permettant de définir les exigences du client (ici, ce n'est pas le cas). Les contraintes supplémentaires sont ici : dd > 0, df < 10, 5 < du et du < 8. Ceci est illustré sur la figure 5.11.

Similairement, le responsable de planification choisit le concept Tâche pour la tâche TE10 à planifier. Ce concept comporte :

- sept modèles de variables (Durée (Du), Date début (Dd), Date fin (Df), Type ressources (Type_Ress), Quantité ressouces (Qté_Ress), Coût (C), Risque (Risk)),
- les domaines associés à ces variables et,
- un modèle de contraintes (Df ≤ Du + Dd).

Ces modèles vont être recopiés respectivement dans les variables conceptuelles tâche, dans les domaines des variables conceptuelles tâche et dans les contraintes conceptuelles tâche. Ensuite, le responsable de planification définit les contraintes supplémentaires représentant les besoins du client.

Les contraintes exigences sont ici : dd = 0 ; df = 5 ; 2 < du ; du < 5 et type_ress = IC. Ceci est illustré sur la figure 5.12.

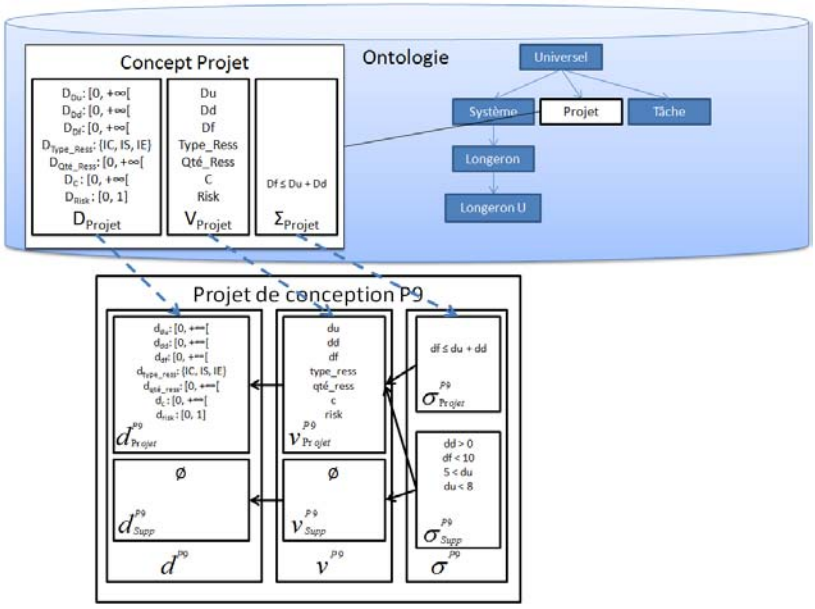


Figure 5.11 : Exemple de définition d'un projet de conception d'un longeron

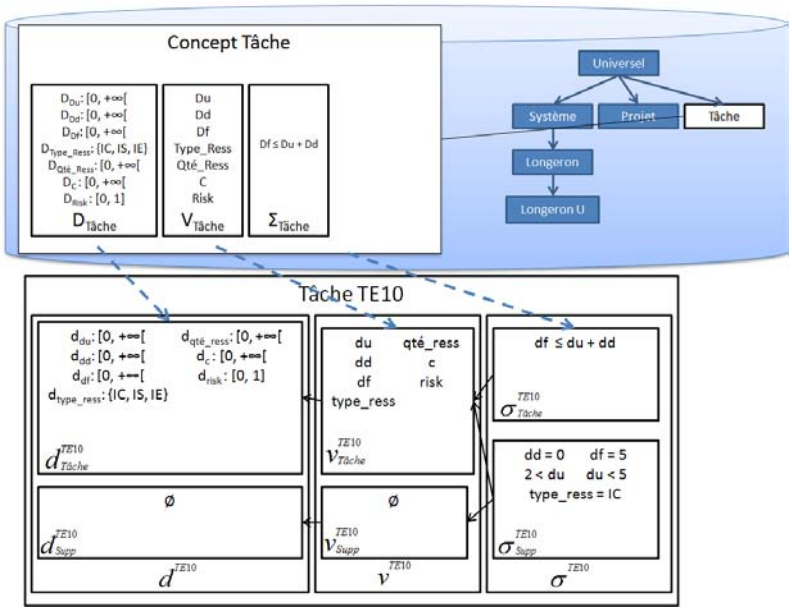


Figure 5.12 : Exemple de définition d'une tâche TE pour la conception d'un longeron

Après planification du projet complet, les domaines des variables sont réduits à des singletons correspondant à la solution de planification. Après réalisation du projet, l'activité de suivi pourra amener le responsable de planification et/ou le responsable de programme à modifier ces domaines en fonction des dates, durées et ressources effectivement mises en œuvre.

5.7 Conclusion sur la formalisation de la connaissance métier

Dans cette section, nous avons proposé une ontologie de concepts permettant d'aider à la conception des systèmes et à la planification des projets. Cette ontologie permet en outre de capitaliser les connaissances métier. Le formalisme choisi pour l'ontologie est celui d'une hiérarchie de concepts

liés entre eux par des liens de spécialisation/généralisation. Chaque concept comporte : des modèles de variables le caractérisant, des modèles de domaines limitant leur étendue et des modèles de contraintes portant sur des modèles de variables correspondant à une connaissance métier limitant les combinaisons de valeurs possibles pour chaque modèle de variable. L'ontologie proposée définit un cadre conceptuel pour des systèmes à concevoir ainsi que des projets et des tâches à planifier.

Ensuite, nous avons proposé de formaliser, à partir de l'ontologie, les variables, domaines et contraintes inhérents à des exigences systèmes, à des alternatives systèmes, à des projets ainsi qu'à des tâches. La formalisation permet de faire la part entre les exigences représentées par des contraintes portant sur des variables et les solutions représentées par des domaines de variables réduits à des singletons. Les propositions sont illustrées par un exemple.

Cette formalisation de connaissances métier propres à la conception de systèmes et propres à la planification de projets va permettre ensuite d'aider à formaliser une nouvelle forme de connaissances sur le couplage entre les deux domaines et de favoriser la réutilisation d'acquis de conception. Ainsi, dans la section suivante, nous formalisons un processus de réutilisation de solutions de planification/conception en mettant en perspective une méthodologie de Raisonnement à Partir de Cas (chapitre 6) et notre propre processus intégré de conception et de planification décrit dans la section 2.3. Ensuite, dans le chapitre 7, nous formalisons une nouvelle forme de connaissances sur le couplage projet/système en utilisant le formalisme des Problèmes de Satisfaction de Contraintes.

Chapitre 6 : Couplage par réutilisation de cas

6 Couplage par réutilisation de cas

La réutilisation de cas présente plusieurs intérêts pour la conception de systèmes. Le premier est le gain de temps dans le cycle de conception. En effet, le fait de réutiliser une solution déjà conçue par le passé pour une alternative système permet de réduire le temps de développement. La réutilisation de cas permet, lorsqu'elle est appliquée aux modèles couplés de conception et de planification que nous avons proposés dans le chapitre 2.1, à partir d'une alternative système correspondant aux exigences du client, de retrouver également la tâche de développement associée. Cette tâche peut alors être adaptée aux contraintes du nouveau client. De plus, le fait de réutiliser des cas antérieurs permet de réduire le cycle de conception et de favoriser la standardisation des systèmes (réutilisation de composants).

La réutilisation de cas consiste, à partir de la formalisation du problème de conception (exigences système), à chercher des solutions qui ont déjà été développées par le passé (alternatives système). Une fois ces solutions identifiées, elles sont adaptées pour satisfaire les exigences inhérentes au nouveau problème.

Dans un premier temps, nous posons certaines définitions du raisonnement à partir de cas, de la réutilisation de cas en conception et en planification et posons la définition du couplage par réutilisation de cas. Ensuite, nous formalisons le processus de réutilisation de cas et, dans un troisième temps, nous illustrons ce couplage pour la conception d'un longeron.

6.1 Définition de la réutilisation de cas

6.1.1 Définition du raisonnement à partir de cas

Le raisonnement à partir de cas (RàPC) s'inscrit bien souvent dans une boucle de retour d'expérience. « Le retour d'expérience est une démarche structurée de capitalisation et d'exploitation des connaissances issues de l'analyse d'évènements positifs et/ou négatifs. Elle met en œuvre un ensemble de ressources humaines et technologiques qui doivent être organisées pour contribuer à réduire les répétitions d'erreurs et à favoriser certaines pratiques performantes » [**Beler, 2008**] [**Clermont et al., 2007**] [**Rakoto et al., 2002**]. Le Raisonnement à Partir de Cas peut être considéré comme un sous-processus permettant de supporter cette démarche de retour d'expérience.

Nous rappelons que le RàPC est une méthodologie de résolution de problèmes basée sur l'exploitation des expériences passées. Les problèmes précédents et leurs solutions sont capitalisés dans une base d'expériences. Ces expériences constituent une base de cas (appelés cas sources) qui seront comparés à un nouveau problème (cas cible) afin de construire, par adaptation, une solution.

Le modèle classique de RàPC (ou Case-Based Reasoning – CBR) décrit dans [**Kolodner, 1993**] [**Aamodt et Plaza, 1994**] [**Kolodner et Leake, 1996**] est basé sur quatre activités : Recherche, Adaptation, Révision, Capitalisation. Dans ce modèle, le problème est considéré comme déjà défini. D'autres modèles plus récents considèrent une cinquième étape initiale de formalisation du problème (voir par exemple [**Finnie et Sun, 2003**] [**Cortes Robles et al., 2009**]). C'est ce dernier modèle que nous utilisons pour formaliser notre proposition.

Les cinq activités sont :

- la **définition du problème** : il s'agit de formaliser le problème avec une sémantique claire afin de construire le cas cible ;

- la **recherche** : il s'agit d'identifier, parmi les cas sources, ceux qui sont les plus similaires au cas cible afin d'être, le cas échéant, en mesure de les réutiliser pour bâtir la solution ;
- l'**adaptation** : suite au choix d'un cas source évalué suffisamment similaire, celui-ci doit être modifié, adapté afin de proposer une solution. Soit le cas source choisi répond parfaitement au problème et l'adaptation est immédiate, soit il ne correspond que partiellement et, dans ce cas, il doit être modifié ;
- la **révision** : la suggestion de solution issue de l'adaptation doit être révisée, réparée afin d'être cohérente, complète et pouvoir être ensuite capitalisée à son tour dans la base de cas. Souvent, la solution obtenue suite à l'adaptation est partielle et nécessite une analyse complémentaire afin d'être complétée puis vérifiée. Il ressort de cette étape un cas testé et/ou réparé ;
- enfin, la **capitalisation** : cette étape consiste à enregistrer le nouveau cas dans la base de cas. Ceci nécessite un certain niveau d'expertise afin de décider si le cas peut alimenter la base d'expériences et constituer ainsi un atome supplémentaire de connaissance contextualisée. Dans le cas présent, nous intégrons la capitalisation à l'intérieur du projet car cela permet de ne pas réaliser cette étape en la repoussant indéfiniment.

6.1.2 Réutilisation de cas en conception et en planification

Concernant la réutilisation en conception de systèmes, de nombreux travaux ont été menés. La plupart proposent des approches de Raisonnement à Partir de Cas dont la définition du problème à résoudre est basé sur un ensemble de descripteurs et des valeurs attendues pour ces descripteurs. Nous pouvons citer, entre autres, [**Maher et Gómez de Silva Garza, 1997**] qui présentent les applications du RàPC pour la conception et commentent les deux aspects de la conception basée sur les cas : l'aide à la conception et l'automatisation de la conception. Des applications récentes du RàPC existent comme [**Renaud et al., 2007**] qui proposent un panorama actuel de l'utilisation du RàPC en conception et en configuration de produits.

Concernant le raisonnement à partir de cas pour la planification de projets, nous pouvons citer les travaux menés sur la mémoire de projet tels que ceux exposés dans [**Matta et al., 2000**] ou dans [**Bekhti, 2004**]. La mémoire de projet est définie par « les leçons et expériences provenant d'un projet donné ». Il s'agit de considérer l'objectif du projet et son contexte, l'organisation du projet (équipes, participants, tâches, etc.), les références associées (règles, méthodes, directives, etc.), la réalisation du projet (problèmes rencontrés, solutions apportées, etc.) afin de capitaliser la connaissance propre à ce projet et donc de guider la définition et la mise en œuvre de projets ultérieurs.

Dans le cadre du couplage de la conception de système et de la planification de projet de conception, il n'existe pas, à notre connaissance, de travaux mettant en œuvre la réutilisation. Nous proposons donc que la réutilisation concerne la conception de systèmes mais également les projets de conception.

Aussi, nous pouvons constater que peu de travaux font appel à la notion de concept (et par extension à la notion d'ontologie) à l'exception de quelques travaux tels ceux présentés dans [**Wriggers et al., 2007**] qui proposent d'utiliser les ontologies associées au raisonnement à partir de cas pour supporter l'analyse de systèmes ou [**Lee et Liu, 2009**] qui proposent une approche similaire pour la conception des services d'un robot. De plus, peu de travaux offrent la possibilité d'exprimer le problème de conception sous forme de contraintes, variables et domaines comme nous le proposons

dans la section 5.1. Nous citons toutefois les travaux présentés dans [**Negny et Le Lann, 2008**] ou dans [**Yan-hong et Guang-xin, 2009**] qui proposent une démarche assez similaire.

Enfin, il est difficile d'exprimer des exigences système en introduisant de la flexibilité afin d'élargir le spectre de recherche de cas dans le but d'exprimer des préférences pour le concepteur ou encore de tenir compte de la notion de similarité entre valeurs d'une même variable. Toutefois, [**Negny et Le Lann, 2008**] proposent de prendre en compte l'imprécision grâce à la logique floue.

6.1.3 Définition du couplage par réutilisation de cas

Nous proposons ici une définition du couplage par réutilisation de cas. Nous avons vu dans la première partie du mémoire que le couplage structurel permettait de fédérer des artefacts de conception avec ceux de planification. Le couplage par réutilisation de cas est mis en œuvre chaque fois qu'une alternative système est conçue. Il consiste, une fois que les exigences système ont été formalisées :

- à rechercher des alternatives système antérieurement conçues et compatibles avec les nouvelles exigences système dans la base de cas,
- pour chacune, à identifier la tâche de développement d'alternative au sein du projet associé (par exploitation du couplage structurel),
- lorsque la décision de réutilisation d'une alternative système est prise, d'adapter la tâche de développement d'alternative TD associée et de la réviser afin qu'elle soit mise en œuvre puis,
- d'adapter et réviser l'alternative système pour élaborer la nouvelle solution,
- de capitaliser l'ensemble dans la base de cas.

Bien entendu, les principes du couplage informationnel sont également suivis afin de synchroniser les tâches et de réaliser les analyses de faisabilité et de vérification. Quant au couplage décisionnel, les tableaux de bord permettent de suivre l'évolution du processus et de prendre les décisions en consultant les indicateurs.

6.2 Formalisation du processus de réutilisation de cas

Le processus de Raisonnement à Partir de Cas peut être mis en perspective du processus de conception de systèmes décrit dans la section 2.3. Dans ce cas, la tâche de recueil des exigences système et de recherche d'alternatives correspond, dans le processus de Raisonnement à Partir de Cas, aux tâches de définition du problème et de recherche. De même, la tâche de développement d'alternative correspond aux tâches de réutilisation, d'adaptation et de capitalisation. Ces deux processus sont juxtaposés sur la figure 6.1.

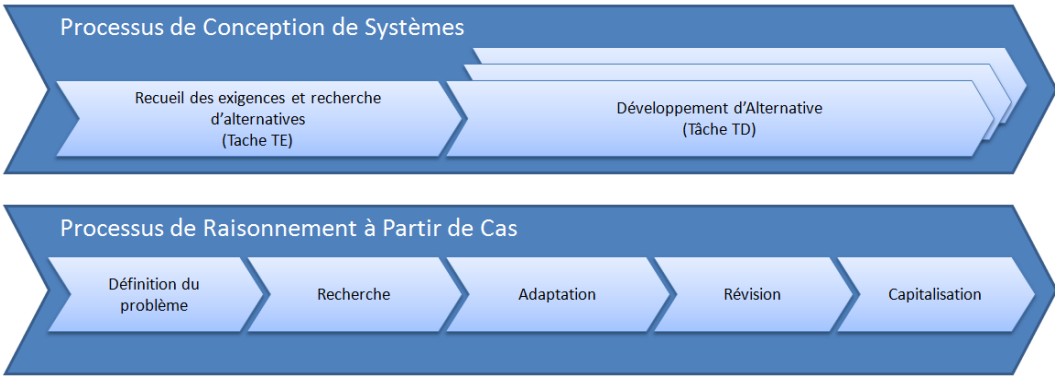


Figure 6.1 : Processus de conception de systèmes et de RàPC

Nous proposons d'adapter le processus de conception système proposé dans le chapitre 2.1 afin d'intégrer les principes du Raisonnement à Partir de Cas. En fait, nous avons vu dans les sections 2 et 3 sur les couplages structurel et informationnel que les activités de renseignement des entités de planification et celles de conception proprement dites étaient fortement couplées (cf. processus figure 2.13 section 2.3.2). Par conséquent, le modèle que nous proposons intègre ces deux dimensions : la réutilisation concerne aussi bien les informations sur les alternatives système AS que sur les tâches de développement de ces alternatives TD, notamment en exploitant le couplage structurel. Le modèle de réutilisation de cas est donc un modèle couplant la conception de systèmes et la planification de projet de conception.

Ainsi, l'identification dans la base de cas d'une alternative système AS réutilisable a priori mène, par l'exploitation du couplage structurel, à l'identification de la tâche de développement TD associée (ainsi que de son projet). Les informations de planification inhérentes à cette tâche TD peuvent alors être réutilisées par adaptation afin de renseigner la tâche TD de développement de la nouvelle alternative.

Le processus de réutilisation de cas en conception de systèmes et en planification de projet est illustré sur la figure 6.2. Ce processus reste totalement compatible avec les processus illustrant les couplages structurel et informationnel (cf. sections 2.3 et 3.3) car il s'agit d'une spécialisation prenant en compte la réutilisation.

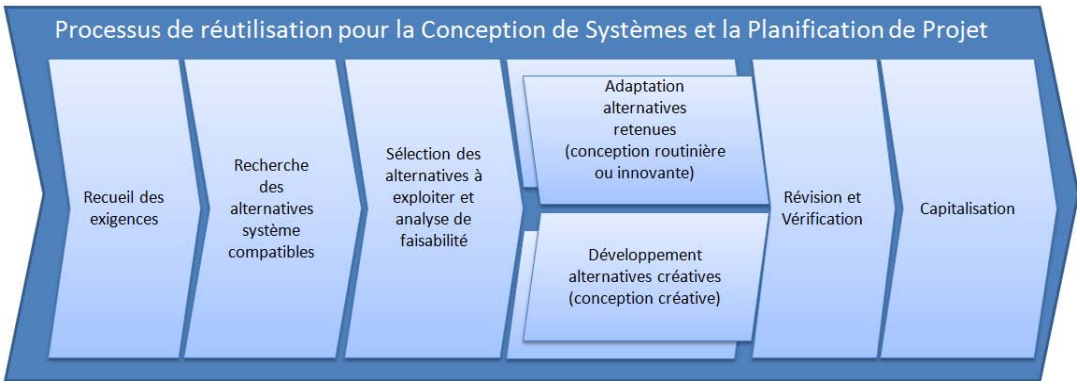


Figure 6.2 : Processus de Raisonnement à Partir de Cas pour la conception de systèmes

Les tâches de ce processus sont décrites ci-après.

La **tâche de recueil des exigences** consiste à définir les concepts, variables, domaines et contraintes exprimant les besoins du client en introduisant une part de flexibilité pour la future recherche d'alternatives compatibles. Il s'agit ici, soit de permettre au concepteur d'exprimer des préférences de recherche sur les contraintes précédemment exprimées tel que décrit dans le chapitre précédent, soit, à partir de fonctions de similarité exprimées a priori et hors ligne par un expert, d'élargir le champ de recherche par rapport aux besoins réels. L'introduction de cette flexibilité dans les critères de recherche permettra d'éviter d'écarter des cas sources proches du problème cible mais ne satisfaisant pas totalement les exigences exprimées. Cette tâche est décrite dans la section 6.3.

La **tâche de recherche des alternatives système compatibles** permet de rechercher, parmi les alternatives système précédemment réalisées et capitalisées dans la base de cas, celles qui correspondent le mieux aux exigences système définies lors de la tâche précédente. Ces alternatives sont alors dites « compatibles ». Cette tâche est décrite dans la section 6.4.

La **tâche de sélection des alternatives à exploiter et d'analyse de faisabilité** permet de choisir la ou les alternatives système et, grâce au couple structurel, la ou les tâches TD associées qui devront être développées ainsi que leur nombre. Ces alternatives système peuvent être, soit adaptées de celles retrouvées dans la base de cas et recopiées dans le nouveau cas, soit totalement nouvelles. À l'issue de cette opération, l'ensemble des attributs proposés dans la section 3 sont à l'état *non déterminé* TD.Fa = ND ∧ AS.Fa = ND. Il est donc nécessaire dans un premier temps de s'assurer de la faisabilité des tâches de développement TD associées à ces alternatives. La faisabilité d'une tâche TD est d'autant plus facile à déterminer que la tâche est réutilisée. Ensuite, l'analyse de faisabilité de chaque alternative système AS que l'on a choisi de développer est réalisée. Ici également, l'analyse de faisabilité réalisée a priori est facilitée par la consultation de l'existant capitalisé.

Selon les choix réalisés lors de la tâche précédente, la **tâche d'adaptation** est différente :

- conception routinière : une alternative système évaluée comme totalement compatible est réutilisée en l'état. L'effort d'adaptation est minimal puisqu'il s'agit de réutiliser totalement ;
- conception innovante : dans la mesure où, pour ce genre de conception, on réutilise une bonne partie de l'existant afin de concentrer l'innovation sur des parties ou composants précis, la tâche d'adaptation consiste à modifier une (ou plusieurs) solutions existantes ;
- conception créative : ici, aucune adaptation n'est possible a priori, cependant le concepteur peut être « guidé » dans ses choix par l'existant (technologies, composants, etc.).

Ainsi, les deux premiers types de conception correspondent à la **tâche d'adaptation des alternatives retenues** (décrite dans la section 6.5) et le troisième correspond à la tâche de **développement d'alternatives créatives** pour laquelle aucune adaptation n'est nécessaire.

La **tâche de révision et vérification** permet de compléter et/ou modifier la solution adaptée afin qu'elle soit cohérente par rapport aux exigences (Révision). Elle permet également de contrôler que toutes les exigences système sont satisfaites (Vérification) et ainsi de mettre à jour les attributs de vérification correspondants.

La **tâche de capitalisation** a pour objet de déterminer si l'alternative système doit être stockée ou non dans la base de cas.

Les tâches de révision et vérification et de capitalisation sont présentées dans la section 6.6.

Dans la suite de cette section, nous détaillons ces tâches sous l'angle des outils et formalismes nécessaires. Ainsi, dans la section 6.3, nous détaillons la tâche de recueil des exigences en proposant un formalisme de représentation des exigences système qui introduit une certaine flexibilité exploitable en phase de recherche. Dans la section 6.4, l'algorithme proposé pour la recherche d'alternatives compatibles est détaillé. La section 6.5 aborde l'adaptation des alternatives et tâche de développement du point de vue des informations réutilisables et de la manière de les adapter. La section 6.6 traite des tâches de révision et de vérification et de capitalisation. Enfin, la section 6.7 illustre la réutilisation de cas pour la conception d'un longeron utilisé en aéronautique.

6.2.1 Recueil et expression des exigences

Nous avons vu dans la section 5 comment formaliser les exigences système à partir de la notion de variables rattachées à des concepts. Il s'agit donc, pour cette tâche, de proposer, à partir des besoins du client et de connaissances conceptuelles issues de l'ontologie, un ensemble de variables

(v^S), l'ensemble des domaines associés à ces variables (d^S) et l'ensemble des contraintes portant sur ces variables (σ^S).

Nous rappelons qu'une contrainte est un ensemble de combinaisons de valeurs autorisées pour un ensemble de variables. Nous considérons dans notre proposition qu'il s'agit :

- de contraintes de compatibilités définissant les combinaisons de valeurs possibles dans le cas de variables discrètes numériques ou symboliques ou,
- de fonctions mathématiques dans le cas de variables continues.

Parallèlement à la formalisation de ces exigences, nous proposons de formaliser une méthode d'évaluation de leur degré de satisfaction. Ceci va permettre, lors de la phase de recherche d'alternatives compatibles, d'évaluer la compatibilité de solutions de la base de cas par rapport aux nouvelles exigences à satisfaire.

Nous décrivons cette méthode pour des contraintes unaires dans un premier temps et ensuite sur des contrainte n-aires.

6.2.1.1 *Compatibilité dans le cas de contraintes unaires*

Une contrainte unaire ne porte que sur une seule variable. Ce type de contraintes précise les valeurs autorisées pour les valeurs de cette variable.

Dans le cas d'une variable discrète, la contrainte prend la forme d'un ensemble de valeurs autorisées. Les deux exemples ci-dessous montrent deux contraintes σ_1 et σ_2 portant respectivement sur les variables x et y où x est une variable discrète numérique et y une variable discrète symbolique. Leurs domaines sont respectivement d_x et d_y.

$$\sigma_1 \ : \ x \in [1,3] \ \ avec \ \ d_x = \{[1,5]\}$$

$$\sigma_2 : \ y \in \{bleu, rouge, blanc\} \ \ avec \ \ d_y = \{bleu, rouge, blanc, vert, jaune\}$$

Dans le cas des variables continues, nous proposons de représenter la contrainte à l'aide d'une fonction mathématique continue bijective dont la valeur pour une valeur de la variable est comprise entre deux bornes. L'exemple ci-dessous montre une contrainte σ_3 sur la variable continue réelle z dont le domaine est d_z.

$$\sigma_3 \ : \ z^2 + 2z + 1 \leq 50 \ \ avec \ \ d_z = [0, +\infty[$$

À partir d'une contrainte unaire, nous proposons de définir une fonction de compatibilité de valeurs possibles pour la variable par rapport à la contrainte. Une fonction de compatibilité précise donc dans quelle mesure chaque valeur possible de la variable, c'est-à-dire chaque valeur appartenant au domaine associé à la variable, est compatible avec la contrainte, la compatibilité pouvant être partielle. Ceci s'apparente à une fonction d'appartenance à un sous ensemble flou des valeurs de variable acceptables tel que proposé par la logique floue [**Zadeh, 1965**]. Nous n'utilisons cependant pas ce formalisme dans notre proposition.

Nous proposons de définir ces fonctions de compatibilité en introduisant une forme de flexibilité. Cela signifie qu'à partir de l'expression d'une contrainte, nécessairement « stricte » au sens où elle est respectée ou ne l'est pas par rapport à une valeur de la variable, la fonction de compatibilité proposée indique avec quel degré la contrainte est satisfaite ou non.

Afin de construire une fonction de compatibilité à partir d'une contrainte stricte, nous proposons d'utiliser deux critères :

- les préférences du concepteur ou,
- la connaissance experte de la similarité entre deux valeurs de :
  - la variable impliquée dans la contrainte dans le cas de variables discrètes,
  - la fonction représentant la contrainte dans la mesure où cette similarité est évaluable (cas de variables numériques).

Le domaine d'une fonction de compatibilité est compris entre 0 et 1. Plus le résultat retourné par la fonction de compatibilité est proche de 1, plus la valeur de la variable est satisfaisante par rapport à la contrainte. Plus elle est proche de 0, moins la contrainte est satisfaite.

Domaine discret pour une variable numérique

Une contrainte stricte σ_i sur la variable discrète numérique x est définie par :

$$\sigma_i \ : \ x \in \{x_m, x_{m+1}, \dots, x_{p-1}, x_p\} \quad avec \quad d_x = \{x_O, \dots, x_M\} \quad et \quad \{x_m, x_{m+1}, \dots, x_{p-1}, x_p\} \subset d_x$$

La contrainte σ_i définie par $\{x_m, x_{m+1}, \dots, x_{p-1}, x_p\}$ correspond à un ensemble de valeurs autorisées ordonnées par ordre croissant.

Si le concepteur ne souhaite pas intégrer ses préférences et que nous ne disposons pas de mesures de similarité entre valeurs possibles de x, la fonction de compatibilité, notée C_{σ_i} , est définie par :

$$C_{\sigma_i}(x) = \begin{cases} 1 & si \quad x \in \{x_m, x_{m+1}, \dots, x_{p-1}, x_p\} \\ 0 & sinon \end{cases}$$

La figure 6.3 illustre la fonction de compatibilité C_{σ_i} ne faisant intervenir ni les préférences du concepteur, ni les fonctions de similarité. Il s'agit donc de l'évaluation de la satisfaction stricte de la contrainte σ_i .

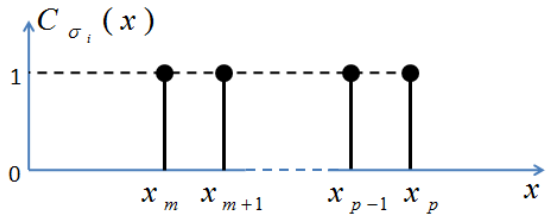


Figure 6.3 : Exemple de fonction de compatibilité représentant une contrainte stricte

Si le concepteur souhaite intégrer ses préférences pour un ensemble de valeurs de variables V_P tel que $V_P \not\subset \{x_m, x_{m+1}, \dots, x_{p-1}, x_p\}$, alors la fonction de compatibilité est définie par :

$$C_{\sigma_i}(x) = \begin{cases} 1 & si \quad x \in \{x_m, x_{m+1}, \dots, x_{p-1}, x_p\} \\ \alpha(x_k) & si \quad x = x_k \ , \forall x_k \in V_P \ / \ V_P \not\subset \{x_m, x_{m+1}, \dots, x_{p-1}, x_p\} \\ 0 & sinon \end{cases}$$

avec :
$\alpha(x_k)$ *: le degré de préférence (* $0 \leq \alpha(x_k) \leq 1$ *) donné par le concepteur pour la valeur* x_k*,*
x_k *: la* $k^{ème}$ *valeur préférée par le concepteur pour la variable x n'appartenant pas à l'ensemble des valeurs autorisées par la contrainte.*

Si le concepteur ne souhaite pas exprimer de préférence mais plutôt utiliser la connaissance a priori sur les similarités entre valeurs possibles pour la variable x, la fonction de compatibilité C_{σ_i} est définie, dans le cas d'une variable numérique, par :

$$C_{\sigma_i}(x) = \begin{cases} 1 & si \quad x \in \{x_m, x_{m+1}, \dots, x_{p-1}, x_p\} \\ sim(x, x_m) & si \quad x < x_m \\ sim(x, x_p) & si \quad x > x_p \end{cases}$$

avec :
$sim(x, x_i)$ *: la valeur retournée par la fonction de similarité entre x et* x_i *telle que* $0 \leq sim(x, x_i) \leq 1$.

Comme la variable x est à valeurs discrètes, l'ensemble des valeurs retournées par la fonction de similarité peut être calculé a priori et regroupé dans une matrice de similarités comme illustré sur la figure 6.4. La matrice de similarités contient les similarités entre les valeurs du domaine de la variable x (ici le domaine $d_x = \{x_O, \dots, x_M\}$). La similarité d'une valeur de la variable avec elle-même est totale $(sim(x_i, x_i) = 1)$.

$$\begin{array}{c} \\ x_O \\ \\ x_M \end{array} \begin{array}{c} \begin{array}{ccc} x_O & & x_M \end{array} \\ \begin{bmatrix} 1 & \cdots & sim(x_O, x_M) \\ \vdots & \ddots & \vdots \\ sim(x_M, x_O) & \cdots & 1 \end{bmatrix} \end{array}$$

Figure 6.4 : Exemple de matrice de similarités

La figure 6.5 illustre une fonction de compatibilité pour une variable numérique discrète x. Les valeurs respectant totalement la contrainte stricte σ_i ont une compatibilité totale (voir la partie centrale de la figure) ; les valeurs inférieures et supérieures aux bornes x_m et x_p ont une compatibilité qui dépend de leur similarité avec ces bornes.

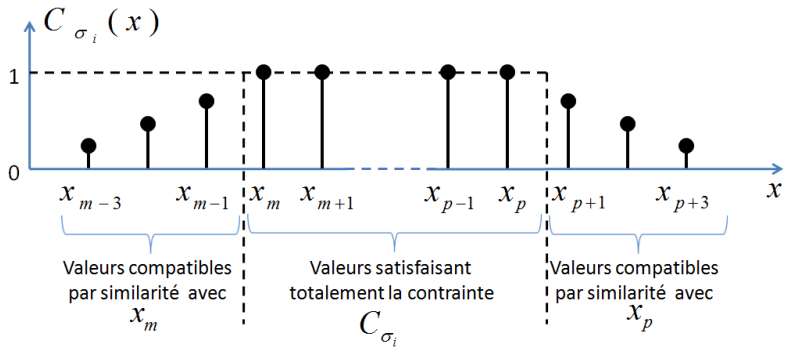


Figure 6.5 : Exemple de fonction de compatibilité avec mesure de similarité aux bornes

Domaine discret pour une variable symbolique

Dans le cas d'une variable symbolique x, une contrainte unaire σ_j est définie par :

$$\sigma_j \ : \ x \in \{S_m, \dots, S_p\} \ \ avec \ \ d_x = \{S_0, \dots, S_n\} \ \ et \ \ \{S_m, \dots, S_p\} \subset d_x$$

avec :
S_i : le symbole i,
d_x : le domaine de x.

La contrainte σ_j définie par $\{S_m, \dots, S_p\}$ constitue l'ensemble des valeurs symboliques de la variable x autorisées.

Similairement aux contraintes discrètes numériques, si le concepteur ne souhaite pas intégrer ses préférences et que nous ne disposons pas de mesures de similarités entre valeurs possibles de x, la fonction de compatibilité C_{σ_j} est définie par :

$$C_{\sigma_j}(x) = \begin{cases} 1 & si \quad x \in \{S_m, \dots, S_p\} \\ 0 & sinon \end{cases}$$

Si le concepteur souhaite exprimer ses préférences quant à un ensemble V_P de symboles, la fonction de compatibilité est définie par :

$$C_{\sigma_i}(x) = \begin{cases} 1 & si \quad x \in \{S_m, \dots, S_p\} \\ \alpha(x_k) & si \quad x = x_k \, , \forall x_k \in V_P \, / \, V_P \not\subset \{S_m, \dots, S_p\} \\ 0 & sinon \end{cases}$$

avec :
$\alpha(x_k)$: le degré de préférence ($0 \leq \alpha(x_k) \leq 1$) exprimé par le concepteur pour la valeur x_k,
x_k : la $k^{ème}$ valeur préférée par le concepteur pour la variable x et n'appartenant pas à l'ensemble des valeurs autorisées par la contrainte.
V_P : l'ensemble des valeurs possibles de x sur lesquelles le concepteur a émis une préférence et tel que $V_P \not\subset \{S_m, \dots, S_p\}$.

Si le concepteur ne souhaite pas exprimer de préférence mais utiliser la connaissance a priori sur les similarités entre valeurs de la variable x, la fonction de compatibilité C_{σ_j} est définie, dans le cas d'une variable symbolique, par :

$$C_{\sigma_j}(x) = \begin{cases} 1 & si \quad x \in \{S_m, \dots, S_p\} \\ \max_i \left(sim(x, S_i)\right) & \forall S_i \in \{S_m, \dots, S_p\} \quad si \quad x \notin \{S_m, \dots, S_p\} \end{cases}$$

avec :
$sim(x, S_i)$: la similarité entre les valeurs symboliques x et S_i.

La fonction de compatibilité retourne la valeur maximale de 1 si la contrainte est respectée pour la valeur de x. Elle correspond à la plus grande similarité entre la valeur de x et les valeurs autorisées par la contrainte dans le cas contraire.

Similairement aux variables discrètes numériques, l'ensemble des valeurs retournées par la fonction de similarité pour les symboles possibles du domaine d_x peut être regroupé dans une matrice de similarités selon la même démarche.

Domaine continu pour une variable numérique

Nous considérons dans ce cas que la contrainte σ_j portant sur une variable y de domaine d_y est représentée par une fonction mathématique $f(y)$ encadrée par ses bornes inférieures et supérieures. La contrainte s'exprime alors ainsi :

$$\sigma_k \ : \ f_{min} \leq f(y) \leq f_{max} \quad avec \quad d_y = \{[y_{min}, y_{max}]\}$$

Si le concepteur ne souhaite pas intégrer ses préférences et que nous ne disposons pas de mesures de similarité entre valeurs possibles de f(y), la fonction de compatibilité C_{σ_k} est définie par :

$$C_{\sigma_k}(y) = \begin{cases} 1 & si \quad f_{min} \leq f(y) \leq f_{max} \\ 0 & sinon \end{cases}$$

La figure 6.6 illustre la fonction de compatibilité de la valeur de $f(y)$ par rapport à la contrainte C_{σ_k} en ne faisant intervenir ni les préférences du concepteur, ni les fonctions de similarité.

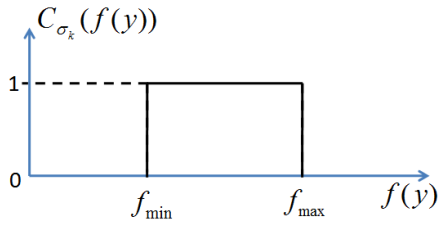


Figure 6.6 : Exemple de représentation d'une fonction de compatibilité d'une contrainte stricte dans le domaine continu

Si le concepteur souhaite intégrer ses préférences quant aux valeurs de $f(y)$ qu'il juge acceptables par rapport aux bornes f_{min} et f_{max} mais ne respectant pas la contrainte σ_k , la fonction de compatibilité est définie par :

$$C_{\sigma_k}(y) = \begin{cases} 1 & si \quad f_{min} \leq f(y) \leq f_{max} \\ pref\big(f(y)\big) & si \quad f(y) \in [f_{min}^{-}, f_{min}[\cup]f_{max}, f_{max}^{+}] \\ 0 & sinon \end{cases}$$

avec :
$pref\big(f(y)\big)$ *: une fonction de préférence appliquée à la fonction f de la contrainte* σ_k *,*
f_{min}^{-} *: la borne inférieure en dessous de laquelle, le concepteur exprime dans ses préférences que les valeurs de* $f(y)$ *ne sont plus compatibles,*
f_{max}^{+} *: la borne supérieure au-delà de laquelle, le concepteur exprime dans ses préférences que les valeurs de* $f(y)$ *ne sont plus compatibles.*

On peut trouver différentes manières pour formaliser des similarités ou des préférences. Par exemple, il est possible d'utiliser des fonctions seuil, des fonctions linéaires, des fonctions sigmoïdes, des fonctions exponentielles [**Bergmann, 2002**].

Nous proposons d'utiliser la fonction de préférence linéaire suivante, illustrée sur la figure 6.7 :

$$pref(f(y)) = \begin{cases} \dfrac{f(y) - f_{min}^{-}}{f_{min} - f_{min}^{-}} & si \quad f_{min}^{-} \leq f(y) \leq f_{min} \\ \dfrac{f_{max}^{+} - f(y)}{f_{max}^{+} - f_{max}} & si \quad f_{max} \leq f(y) \leq f_{max}^{+} \\ 0 \quad sinon \end{cases}$$

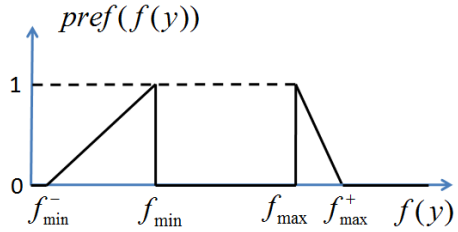


Figure 6.7 : Exemple de fonction de préférences aux bornes d'une contrainte σ_k .

Dans le cas où le concepteur souhaite utiliser la connaissance a priori sur les similarités entre $f(y)$ et les bornes f_{min} et f_{max}, nous proposons d'utiliser une fonction de similarité entre une valeur de la fonction $f(y)$ et une borne de la contrainte σ_k (borne inférieure f_{min} ou borne supérieure f_{max}). Ceci est possible dans deux situations :

- f(y) = y : dans ce cas, la fonction de similarité entre valeurs de la variable y est utilisée,
- autres cas : soit un expert est capable d'exprimer la similarité entre valeurs de la fonction, soit il est impossible d'utiliser les similarités.

Un exemple de fonction de similarité dans le cas de bornes inférieure et supérieure $(sim(f(y), f_{min}))$ et $(sim(f(y), f_{max}))$ est illustrée sur la figure 6.8.

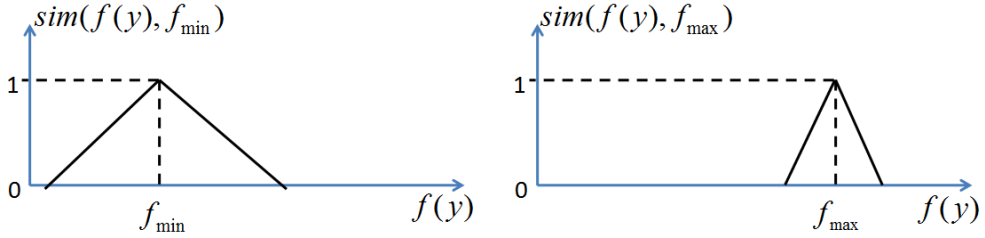


Figure 6.8 : Exemple de fonction de similarité pour une valeur de la fonction $f(y)$ et les bornes f_{min} et f_{max}

Dans le cas où le concepteur souhaite utiliser la connaissance sur les similarités, la fonction de compatibilité $C_{\sigma_k}(y)$ est alors définie par :

$$C_{\sigma_k}(y) = \begin{cases} 1 \quad si \quad f_{min} \leq f(y) \leq f_{max} \\ sim(f(y), f_{min}) \quad si \quad f(y) < f_{min} \\ sim(f(y), f_{max}) \quad si \quad f(y) > f_{max} \end{cases}$$

Nous abordons dans la section suivante le cas des contraintes n-aires.

6.2.1.2 *Compatibilité dans le cas de contraintes n-aires*

Une contrainte n-aire porte sur plusieurs variables. Une telle contrainte est formalisée par un ensemble de valeurs autorisées pour un ensemble de variables.

Domaine discret pour des variables symboliques ou numériques

Dans le cas des variables discrètes, nous considérons que la contrainte précise les combinaisons de valeurs de variables autorisées. Une contrainte σ_l faisant intervenir un ensemble de variables $X = \{x_1, x_2, \dots, x_n\}$ est donc définie en extension par :

$$\sigma_l : X \in X_{autorisé} \; tel \; que \; X_{autorisé} = \{(x_{11}, x_{21}, \dots, x_{n1}), (x_{12}, x_{22}, \dots, x_{n2}), \dots, (x_{1p}, x_{2p}, \dots, x_{np})\}$$

Nous notons $X_{autorisé}$ l'ensemble des n-uplets autorisés. La fonction de compatibilité de la contrainte stricte σ_l est donc définie par :

$$C_{\sigma_l}(X) = \begin{cases} 1 & si \quad X \in X_{autorisé} \\ 0 & sinon \end{cases}$$

Afin d'exprimer ses préférences, le concepteur peut proposer une liste de n-uplets X_P tel que $X_P \not\subset X_{autorisé}$ avec sa préférence pour chacun. La fonction de compatibilité $C_{\sigma_l}(X)$ de l'ensemble de variables X, issue de la contrainte σ_l et des préférences du concepteur, est donc définie par :

$$C_{\sigma_l}(X) = \begin{cases} 1 & si \quad X \in X_{autorisé} \\ \alpha(X_k) & si \quad X = X_k, \forall X_k \in X_P \,/\, X_P \not\subset X_{autorisé} \\ 0 & sinon \end{cases}$$

Si le concepteur ne souhaite pas exprimer ses préférences mais qu'il souhaite utiliser la connaissance a priori sur les similarités entre les valeurs des variables X, nous proposons, pour tout n-uplet autorisé $X_j \in X_{autorisé}$, de mesurer la similarité locale entre chacune des valeurs des variables de X avec la valeur de la variable correspondante du n-uplet autorisé X_j. Ensuite, nous proposons d'agréger ces similarités afin de calculer une similarité globale entre X et X_j. Enfin, nous ne retenons que la similarité maximale obtenue pour l'ensemble des n-uplets autorisés. La fonction de compatibilité $C_{\sigma_l}(X)$ est donc définie par :

$$C_{\sigma_l}(X) = \begin{cases} 1 & si \quad X \in X_{autorisé} \\ \max\limits_{j} \left(sim(X, X_j)\right) & \forall X_j \in X_{autorisé} \end{cases}$$

avec :
$sim(X, X_j)$ *: la fonction de similarité globale entre le n-uplet X et le n-uplet* X_j *autorisé par la contrainte* σ_l *définie par :*

$$sim(X, X_j) = \left[\sum_{i=1}^{n} w_i (sim(x_i, x_{ij}))^{\beta}\right]^{\frac{1}{\beta}}$$

avec :
w_i *: le poids donné à la variable* x_i *tel que* $\sum_{i=1}^{n} w_i = 1$ *et* $0 \leq w_i \leq 1$,

n : le nombre de n-uplets autorisés,
x_{ij} *: la* $i^{ème}$ *valeur du n-uplet autorisé* X_j*,*
x_i *: la* $i^{ème}$ *variable du n-uplet X,*
$sim(x_i, x_{ij})$ *: la fonction de similarité locale entre la valeur de* x_i *et la valeur* x_{ij}*.*

La fonction d'agrégation basée sur la fonction de Minkowski (voir par exemple [**Bergmann, 2002**] [**Núñez et al., 2004**] [**Avramenko et Kraslawski, 2006**]) est utilisée pour calculer la similarité globale entre X et X_j à partir des mesures de similarités locales entre valeurs de variables (connaissance a priori). Il existe plusieurs fonctions d'agrégation usuelles comme, par exemple, [**Bergmann, 2002**] : les fonctions produit, minimum, maximum, moyenne ou encore la fonction de Minkovski.

Les fonctions minimum et produit proposent une vision pessimiste de l'agrégation. En effet, si un des attributs à agréger est nul, le résultat de l'agrégation est nul. A contrario, la fonction maximum propose une vision optimiste de l'agrégation puisque si un seul attribut est égal à 1, le résultat de l'agrégation est égal à 1. La fonction moyenne est la fonction la plus courante où chacun des attributs contribue au résultat de manière pondérée. La fonction de Minkowski est une généralisation de la moyenne pondérée dans laquelle, plus la valeur de β est grande, plus l'influence des attributs possédant des similarités locales élevées est importante.

Domaine continu pour des variables numériques

Nous considérons dans ce cas qu'une contrainte σ_m fait intervenir un ensemble de variables continues $Y = \{y_1, y_2, \dots, y_n\}$. Chaque variable $y_i \in Y$ a pour domaine $d_{y_i} \in d_y$. Nous considérons que la contrainte σ_m est représentée par une fonction mathématique continue bijective $f(Y)$ encadrée par ses bornes inférieure (f_{min}) et supérieure (f_{max}). La contrainte s'exprime alors ainsi :

$$\sigma_m \ : \ f_{min} \leq f(Y) \ et \ f(Y) \leq f_{max} \quad avec \quad d_Y = \{d_{y_1}, d_{y_2}, \dots, d_{y_n}\}$$

Si le concepteur ne souhaite pas intégrer ses préférences et que nous ne disposons pas de mesures de similarité entre les valeurs possibles de y, la fonction de compatibilité C_{σ_m} est définie par :

$$C_{\sigma_m}(Y) = \begin{cases} 1 & si \quad f_{min} \leq f(Y) \leq f_{max} \\ 0 & sinon \end{cases}$$

La figure 6.9 illustre la fonction de compatibilité des valeurs de $f(Y)$ pour la contrainte C_{σ_m} ne faisant intervenir ni les préférences du concepteur, ni les fonctions de similarité.

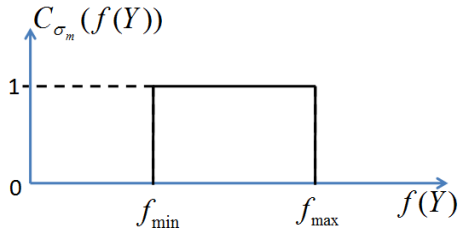


Figure 6.9 : Exemple de représentation d'une fonction de compatibilité d'une contrainte n-aire stricte dans le domaine continu

Si le concepteur souhaite intégrer ses préférences quant aux valeurs de $f(Y)$ qu'il juge acceptables par rapport aux bornes f_{min} et f_{max} mais ne respectent pas la contrainte σ_m, la fonction de compatibilité est définie par :

$$C_{\sigma_m}(Y) = \begin{cases} 1 & si \quad f_{min} \leq f(Y) \leq f_{max} \\ pref\big(f(Y)\big) & si \quad f(Y) \in [f_{min}^{-}, f_{min}[\cup]f_{max}, f_{max}^{+}] \\ 0 & sinon \end{cases}$$

avec :
$pref\big(f(Y)\big)$ *: fonction de préférence appliquée à la fonction f de la contrainte* σ_m *,*
f_{min}^{-} *: la borne inférieure en dessous de laquelle, le concepteur exprime dans ses préférences que les valeurs de* $f(Y)$ *ne sont plus compatibles,*
f_{max}^{+} *: la borne supérieure au-delà de laquelle, le concepteur exprime dans ses préférences que les valeurs de* $f(Y)$ *ne sont plus compatibles.*

Nous proposons la fonction de préférence suivante (illustrée sur la figure 6.10) :

$$pref(f(Y)) = \begin{cases} \dfrac{f(Y) - f_{min}^{-}}{f_{min} - f_{min}^{-}} & si \quad f_{min}^{-} \leq f(Y) \leq f_{min} \\ \dfrac{f_{max}^{+} - f(Y)}{f_{max}^{+} - f_{max}} & si \quad f_{max} \leq f(Y) \leq f_{max}^{+} \\ 0 & sinon \end{cases}$$

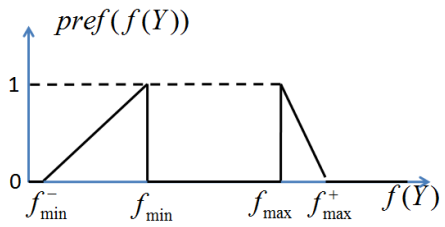


Figure 6.10 : Exemple de fonction de préférences aux bornes d'une contrainte σ_m .

Dans le cas où le concepteur souhaite utiliser la connaissance a priori sur les similarités entre $f(Y)$ et les bornes f_{min} et f_{max}, nous proposons d'utiliser une fonction de similarité entre une valeur de la fonction $f(Y)$ et une borne de la contrainte σ_m (borne inférieure f_{min} ou borne supérieure f_{max}). Ceci n'est possible que dans le cas où un expert est en mesure d'évaluer la similarité entre les valeurs de la fonction $f(Y)$.

Un exemple de fonction de similarité dans le cas des bornes inférieure et supérieure $\big(sim(f(Y), f_{min})\big)$ et $\big(sim(f(Y), f_{max})\big)$ est proposée sur la figure 6.11.

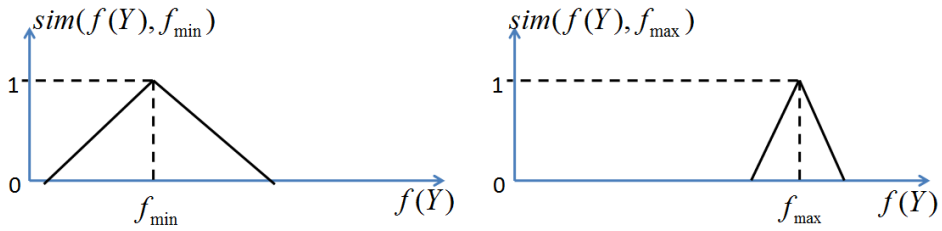


Figure 6.11 : Exemple de fonction de similarité pour une valeur de la fonction $f(Y)$ et les bornes f_{min} et f_{max}

La fonction de compatibilité $C_{\sigma_m}(Y)$ est alors définie par :

$$C_{\sigma_m}(Y) = \begin{cases} 1 \quad si \quad f_{min} \leq f(Y) \leq f_{max} \\ sim(f(Y), f_{min}) \quad si \quad f(Y) < f_{min} \\ sim(f(Y), f_{max}) \quad si \quad f(Y) > f_{max} \end{cases}$$

Nous venons d'aborder comment exprimer des fonctions de compatibilité à partir de contraintes pouvant être unaires ou n-aires, symboliques ou numériques et discrètes ou continues. Les fonctions de compatibilité que nous décrivons permettent de modéliser des contraintes de manière stricte et d'y intégrer une forme de flexibilité à partir de l'expression de préférences du concepteur ou bien de connaissances expertes sur les similarités entre valeurs. Cette flexibilité permet d'élargir le spectre de la recherche d'alternatives système compatibles dans l'étape suivante.

6.2.2 Recherche des alternatives système compatibles

La recherche de cas compatibles consiste à trouver une ou plusieurs alternatives système dans la base de cas satisfaisant totalement ou partiellement l'ensemble des contraintes définies dans la section précédente. Nous proposons, afin d'identifier ces alternatives compatibles, la démarche suivante :

- la **présélection des alternatives système dans la base de cas** : le principe est d'identifier un ensemble d'alternatives dont le concept associé est le plus similaire au concept identifié pour le nouveau système à concevoir,
- l'**évaluation de la compatibilité de chaque alternative présélectionnée** par rapport aux exigences système exprimées sous forme de contraintes,
- la **sélection des alternatives à exploiter** parmi celles évaluées comme étant compatibles.

6.2.2.1 *Présélection des alternatives système*

La présélection des alternatives consiste à rechercher les alternatives dont le concept est « proche » du concept choisi pour définir les exigences système. Dans un premier temps, nous proposons de rechercher dans la base de cas les alternatives système dont le concept est, soit identique au concept choisi pour la définition des exigences système, soit une spécialisation de ce dernier (cf. section 5.4).

Dans un second temps, et si aucune alternative n'a été trouvée, nous proposons de rechercher les alternatives système dont le concept est, soit le concept père du concept choisi pour la définition des exigences système, soit une spécialisation de ce dernier (en éliminant les concepts ayant déjà été utilisés dans la première phase).

Pour chacun des concepts associés aux alternatives présélectionnées, nous proposons de calculer une mesure de similarité sémantique entre ce concept et le concept choisi pour définir les exigences systèmes (Concept système). La mesure proposée est celle définie par [**Wu et Palmer, 1994**] dans laquelle, la similarité est définie par rapport à la distance (en nombre d'arcs) qui sépare deux concepts dans l'ontologie. Wu et Palmer définissent la similarité entre deux concepts C_1 et C_2 par :

$$Sim\ (C_1, C_2) = \frac{2 * profondeur(C)}{profondeur(C_1) + profondeur(C_2)}$$

avec :
C_1 *et* C_2 *: deux concepts dont on souhaite mesurer la similarité,*
C *: le plus petit concept généralisant* C_1 *et* C_2*,*
$profondeur(C)$ *: le nombre d'arc entre le concepts C et la racine de l'ontologie (c'est-à-dire le concept « Universel »).*

La mesure de Wu et Palmer est légèrement modifiée dans notre approche car nous calculons les nombres d'arcs à partir du concept « Système » de l'ontologie et non du concept « Universel »

Cette mesure présente l'intérêt d'être simple à mettre en œuvre même si d'autres mesures peuvent être utilisées (voir par exemple [**Cordì et al., 2005**] [**Pirró, 2009**] [**Al-Mubaid et Nguyen, 2009**] [**Batet et al., 2010**]). La mesure de Wu et Palmer prend en compte le fait que :

- plus deux concepts sont éloignés (par le nombre d'arcs qui les séparent) dans l'ontologie, moins ils sont similaires,
- considérant deux paires de concepts séparés par le même nombre d'arcs dans l'ontologie, la paire dont le plus petit concept généralisant est le plus éloigné de la racine aura la similarité la plus importante.

Exemples de calcul de similarité

Soit l'ontologie simplifiée illustrée sur la figure 6.12.

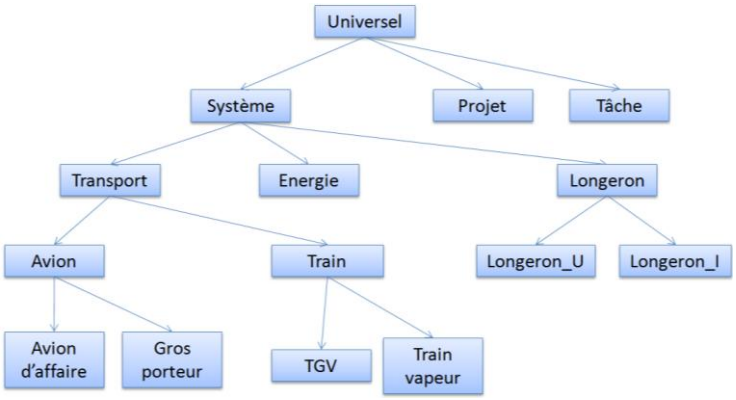


Figure 6.12 : Exemple d'ontologie simplifiée

La similarité entre les concepts « Avion d'affaire » et « Gros porteur », les concepts « Avion d'affaire » et « TGV » et les concepts « Avion » et « Train » sont données par :

$$Sim\ (Avion\ d'affaire, Gros\ porteur) = \frac{2*2}{3+3} = 0{,}67$$

$$Sim\ (Avion\ d'affaire, TGV) = \frac{2 * 1}{3 + 3} = 0{,}33$$

$$Sim\ (Avion, Train) = \frac{2 * 1}{2 + 2} = 0{,}5$$

La mesure prend bien en compte le fait que les paires de concepts (Avion d'affaire, Gros porteur) et (Avion d'affaire, TGV), bien que distants chacun de deux arcs, ont une mesure de similarité différente (0,67 et 0,5) :

- pour la paire (Avion d'affaire, Gros porteur), le plus petit concept généralisant est « Avion », distant de deux arcs du concept « Système »,
- pour la paire (Avion d'affaire, TGV), le plus petit concept généralisant est « Transport », distant d'un seul arc du concept « Système »

Nous présentons dans l'Algorithme 1 la présélection des alternatives système et au calcul de la distance sémantique pour les concepts des alternatives système présélectionnées. Nous posons comme hypothèse qu'il existe au moins une alternative système dans la base de cas.

Quand tous les concepts des alternatives système ont été comparés au concept C et à ses descendants, si aucune alternative système n'a été sélectionnée, le concept C est remplacé par son concept père afin d'élargir la recherche, jusqu'au concept « Système » si nécessaire.

6.2.2.1 *Mesure de compatibilité d'une alternative système présélectionnée par rapport aux exigences système*

Nous avons précédemment défini les fonctions de compatibilité pour des contraintes (voir section 6.1). Il s'agit, dans cette section, de définir une mesure de compatibilité globale d'une alternative présélectionnée AS_i vis-à-vis de l'ensemble des contraintes système rassemblées dans σ^S. Le principe est de construire cette mesure de compatibilité globale à partir des mesures de compatibilité locales évaluées pour chaque contrainte $\sigma_i \in \sigma^S$.

Mesure de compatibilité locale entre une alternative AS_j et une contrainte σ_i

Deux cas de figure peuvent se présenter pour une contrainte σ_i donnée :

- toutes les variables concernées par la contrainte σ_i sont présentes dans l'alternative système AS_j. Dans ce cas, on évalue la compatibilité de l'ensemble des valeurs de variables de l'alternative système AS_j par rapport à la contrainte ;
- il existe au moins une variable concernée par la contrainte σ_i qui n'est pas présente dans l'alternative système AS_j. Dans ce cas, il y a deux moyens d'évaluer la compatibilité :
  - on calcule la compatibilité en considérant uniquement les variables concernées dans la contrainte σ_i. La compatibilité de la contrainte n'est évaluée que sur un sous-ensemble des variables normalement concernées par celle-ci ;
  - on considère que la compatibilité des valeurs présentes dans l'alternative par rapport à la contrainte est nulle.

Algorithme de présélection d'alternatives système et calcul de la similarité sémantique entre le concept cible et le concept de l'alternative présélectionnée

Entrée :

S : Système à concevoir.

Sortie :

PRE : l'ensemble des alternatives système présélectionnées (ensemble de cas sources) ainsi que, pour chacune, la similarité entre le concept du système à concevoir et le concept de l'alternative présélectionnée.

Notations :

Soit C un concept de l'ontologie,
CF(C) : l'ensemble des concepts spécialisant C,
CP(C) : le concept père de C,
CS : le concept associé au système,
AS_i : l'alternative système i présente dans la base de cas,
AS : l'ensemble des alternatives système contenues dans la base de cas tel que AS non vide,
CA_i : le concept de l'alternative système AS_i,
C_cible : l'ensemble des concepts utilisés pour la présélection,
$Sim\_CS\_CA_i$: la mesure de similarité entre le concept CS et le concept CA_i.

début

```
C = CS                                   /* le concept C est le concept cible */
C_cible = {C} ∪ CF(C)                    /* sélection du concept C et de ses descendants */
PRE = Ø                                  /* aucune alternative système n'est présélectionnée*/
Répéter
    Pour tout ASi ∈ AS faire             /* une seule alternative système testée à la fois*/
        Si (CAi ∈ C_cible) Alors         /* le concept de l'alternative système i est compatible
                                            avec celui des exigences système*/
            Sim_CS_CAi = Sim(CS, CAi)    /* mesure de Wu et Palmer */
            PRE = PRE ∪ (ASi, Sim_CS_CAi)   /* l'alternative système est */
            Finsi                        /* présélectionnée avec mesure de */
    Finpour                              /* similarité entre concepts*/
    Si (PRE = Ø et C ≠ Système) Alors
        C = CP(C)                        /* le concept C est remplacé par son père dans
                                            l'ontologie */
        C_cible = ({C} ∪ CF(C))\C_cible  /* pour sélectionner les concepts spécialisant C
                                            sauf ceux déjà testés */
    Finsi
Jusqu'à (PRE ≠ Ø)
```

fin

Algorithme 1 : Algorithme de présélection d'alternatives système et calcul de la similarité sémantique entre le concept cible et le concept de l'alternative présélectionnée

Dans la suite de cette section, nous proposons d'utiliser la deuxième alternative en considérant que la compatibilité locale est nulle si l'ensemble des variables de la contrainte n'est pas totalement renseigné. En procédant ainsi, s'il existe au moins une variable de la contrainte qui n'est pas renseignée dans l'alternative, la compatibilité locale est très fortement pénalisée, traduisant le fait qu'une solution partiellement définie par rapport à une nouvelle exigence n'est pas compatible par rapport à celle-ci. Cette pénalité sera ensuite intégrée dans le calcul de la compatibilité globale.

La compatibilité des valeurs de variables de l'alternative AS_j par rapport à la contrainte σ_i est définie par :

$$C_{\sigma_i}^{AS_j} = \begin{cases} C_{\sigma_i}(Val\left(d_{\sigma_i}^{AS_j}\right) \; si \; \forall v_n \in v_{\sigma_i}^{S}, \exists v_n \in v_{\sigma_i}^{AS_j} \\ 0 \; sinon \end{cases}$$

avec :
AS_j *: l'alternative système j,*
σ_i *: la contrainte i portant sur un ensemble de variables,*
v^{AS_j} *: l'ensemble des variables de l'alternative système* AS_j *tel que* $v^{AS_j} = \left\{v_n / v_n \in v^{AS_j}\right\}$,
d^{AS_j} *: l'ensemble des domaines des variables de l'ensemble* v^{AS_j} *tel que* $d^{AS_j} = \left\{d_n \; tel \; que \; d_n est \; le \; domaine \; de \; v_n, \forall v_n \in v^{AS_j}\right\}$,
$v_{\sigma_i}^{AS_j}$ *: les variables liées à la contrainte* σ_i *et présentes dans l'alternative système* AS_j ($v_{\sigma_i}^{AS_j} \subset v^{AS_j}$),
$d_{\sigma_i}^{AS_j}$ *: l'ensemble des domaines des variables de l'ensemble* $v_{\sigma_i}^{AS_j}$,
$d_n.val$ *: valeur de la variable* v_n *obtenue par réduction à un singleton de son domaine* d_n ($v_n \in v^{AS_j}$ *et* $d_n \in d^{AS_j}$),
$Val\left(d_{\sigma_i}^{AS_j}\right)$ *: ensemble des valeurs des variables de l'alternative* AS_j *pour la contrainte* σ_i *tel que* $Val\left(d_{\sigma_i}^{AS_j}\right) = \left\{d_n.val \; tel \; que \; d_n \; est \; le \; domaine \; de \; v_n, \forall v_n \in v_{\sigma_i}^{AS_j}\right\}$,
$v_{\sigma_i}^{S}$ *: l'ensemble des variables concernées par la contrainte* σ_i *et issues des exigences du système S,*
$C_{\sigma_i}(X)$ *: fonction de compatibilité de l'ensemble de valeurs de X par rapport à la contrainte* σ_i *(cf. section 6.3).*

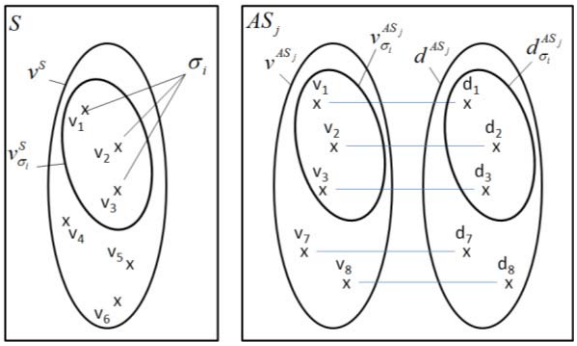


Figure 6.13 : Exemple d'éléments permettant le calcul de la compatibilité entre une contrainte σ_i et une alternative système AS_j présélectionnée

Lorsque toutes les contraintes de σ^S ont été traitées de cette manière, une opération d'agrégation doit être effectuée afin d'évaluer la compatibilité globale de l'alternative AS_j par rapport à l'ensemble des contraintes système σ^S, noté $C_{\sigma^S}^{AS_j}$. De nouveau, la fonction de Minkowski est utilisée car elle permet de prendre en compte toutes les mesures de compatibilités locales (contrairement aux fonctions minimum et maximum) tout en ne pénalisant pas complètement les alternatives système ne pouvant pas répondre à toutes les contraintes (contrairement à la fonction produit).

La compatibilité globale d'une alternative système AS_j par rapport aux exigences du système S est définie par :

$$C_{\sigma^S}^{AS_j} = \left(\sum_{i=1}^{N} w_i * (C_{\sigma_i}^{AS_j})^{\beta} \right)^{1/\beta}$$

avec :
N : le nombre de contraintes système ($N = |\sigma^S|$).

Chaque alternative présélectionnée doit être évaluée de cette façon au regard de l'ensemble des contraintes σ^S. In fine, le concepteur dispose, pour chaque alternative AS_j présélectionnée :

- d'une mesure de compatibilité de cette alternative par rapport aux contraintes liées au système à concevoir (après l'introduction éventuelle de flexibilité sur les contraintes),
- d'une mesure de similarité du concept de l'alternative système AS_j par rapport au concept choisi pour le système S.

Ces deux indicateurs doivent permettre de guider le concepteur dans le choix des alternatives à réutiliser. Celles qui seront retenues doivent ensuite être adaptées.

6.2.3 Adaptation des alternatives retenues

Certaines approches de RàPC sont basées sur une adaptation réalisée « automatiquement » par l'application de règles de modification [**Leake et Powell, 2007**]. Dans la mesure où ces règles sont rarement complètes et disponibles [**Lee et al., 1998**], nous ne nous plaçons pas dans ce mode de fonctionnement. Ainsi, la tâche d'adaptation est réalisée manuellement dans notre approche par les responsables de planification et de conception.

L'adaptation concerne en premier lieu la réutilisation des informations propres aux tâches TD_j associées aux alternatives système AS_j compatibles. Chaque tâche TD_j envisagée doit au préalable être renseignée puis planifiée. Donc, suite au choix des alternatives système AS_j compatibles à réutiliser, chaque tâche TD_j peut être adaptée des cas précédents puis planifiée. Quel que soit le type de conception, la connaissance des durées et ressources des tâches précédemment réalisées pour des alternatives système compatibles doit faciliter la construction des plannings des nouvelles tâches par adaptation.

Dans cette section, nous abordons la question de l'adaptation des alternatives système et des tâches de développement. Nous détaillons ci-dessous les informations réutilisables en planification de projet et en conception de système.

6.2.3.1 *Informations de planification réutilisables*

Autant il est envisageable de réutiliser entièrement sans adaptation une solution de conception système, autant pour un projet et ses tâches, la réutilisation est nécessairement partielle car dépendant du contexte et du responsable de planification [**Lee et Lee, 2006**]. Nous envisageons la réutilisation des informations temporelles, de ressources, de risque et de coût projet liées aux tâches :

- *informations temporelles (durées)* : si le niveau de réutilisation d'une alternative système est important (peu ou pas d'adaptation), les durées de développement vont être réduites. Par conséquent, ces informations peuvent être réutilisées par le responsable de planification uniquement comme guide pour définir les dates et les durées. Par exemple, considérons une alternative système AS et sa tâche de développement TD stockées dans la base de cas. La durée de la tâche TD était de 3

semaines lors de sa première mise en œuvre. Le choix de réutiliser l'alternative système AS compatible comme solution à de nouvelles exigences inhérentes à un nouveau système à concevoir va mener le responsable de planification à fixer la durée de développement de l'alternative à 3 jours correspondant à la durée de l'adaptation de l'alternative système compatible. Dans cet exemple, l'adaptation pourra consister à s'assurer que les composants sont toujours disponibles et/ou trouver de nouveaux fournisseurs. L'information concernant la durée de 3 semaines aura été adaptée au cas courant (par réduction). En revanche, les informations temporelles concernant les tâches de « Production » (l'approvisionnement des composants et/ou la fabrication des composants et/ou l'intégration des sous-systèmes qui ont été physiquement réalisés au niveau inférieur) peuvent être réutilisées totalement. Par exemple, une durée d'assemblage de 10 jours peut être conservée dans la mesure où le processus de fabrication reste valide ;

- *informations sur les ressources* : pour ce type d'information concernant la ou les ressources affectées à la réalisation d'une tâche TD, la réutilisation peut être totale, partielle ou nulle. Il peut s'agir, par exemple, d'un concepteur spécialisé dans un métier particulier que le responsable de planification peut souhaiter réaffecter à une nouvelle tâche TD ;
- *informations sur le risque et le coût des tâches* : concernant les informations concernant le risque, le fait de réutiliser une tâche TD réalisée auparavant peut réduire le risque. Concernant le coût projet, il est en partie fonction des informations temporelles et des ressources affectées. Le coût peut donc être réduit s'il y a réutilisation de la tâche TD.

6.2.3.2 *Informations de conception système réutilisables*

L'ensemble des informations caractérisant une solution de conception est réutilisable. Ainsi, les solutions logiques (avec leurs sous-systèmes s'ils existent dans la solution) et les solutions physiques sont réutilisables soit totalement, soit partiellement.

Dans le cas où l'on choisit de réutiliser une alternative système en la modifiant, plusieurs cas de figure sont envisageables :

- sur la solution logique :
  - modifier certains principes de fonctionnement décrits dans la solution logique (équations, modèles, schémas, etc.) ;
  - et/ou modifier la structure du système en remplaçant certains sous-systèmes par d'autres ou en rajoutant/supprimant des sous-systèmes ;
  - et/ou modifier les sous-systèmes existants en redéfinissant leurs exigences ;
- sur la solution physique :
  - substituer certains composants de l'alternative par d'autres composants existants « sur étagère » ;
  - et/ou modifier la structure du système (ajout/suppression de composants).

Il est nécessaire, après toute modification, de s'assurer que l'alternative système conçue satisfait bien les exigences système.

6.2.3.3 *Récursivité de la tâche d'adaptation*

Nous abordons ici l'aspect récursif de la tâche d'adaptation. En effet, lorsqu'une alternative système compatible est adaptée, et que sa solution logique comporte des sous-systèmes, son adaptation

peut être soit réalisée automatiquement en une seule fois pour toute la hiérarchie de sous-systèmes et sous-projets associés, soit de manière récursive et réalisée niveaux après niveaux. Nous analysons ces deux possibilités.

L'adaptation est réalisée automatiquement pour l'ensemble de la structure de systèmes, sous-systèmes, projets et sous-projets

Cela signifie que la réutilisation d'une alternative système compatible comportant des sous-systèmes entraine la réutilisation automatique de l'ensemble de ses sous-systèmes et ce jusqu'aux feuilles de l'arborescence sans analyser ni vérifier ces solutions. L'avantage est que si le choix de réutiliser une alternative système est fait à un certain niveau, l'ensemble de la hiérarchie peut être réutilisé tel quel. Dans le cas d'un système très complexe avec de multiples niveaux, cette démarche peut s'avérer relativement simple. En revanche, rien ne garantit que toutes les solutions de conception des sous-systèmes sont toujours viables et répondent toujours aux exigences pour lesquelles elles ont été créées initialement. De plus, coté projet, nous perdons l'aspect bijectif : le développement de la hiérarchie de sous-systèmes réutilisés entièrement sans détailler les niveaux se retrouve piloté par une unique tâche de développement d'alternative. Il sera ainsi impossible de piloter une activité d'étude de faisabilité ou de vérification d'une alternative d'un sous-système de bas niveau. Nous obtenons ici un cas où plusieurs niveaux de décomposition en conception sont pilotés par un seul projet (voir section 2.1.3).

L'adaptation est réalisée récursivement niveau par niveau

Dans ce cas, le choix d'adapter une alternative système compatible en conservant ses sous-systèmes entraine la mise en œuvre du processus de réutilisation pour chacun des sous-systèmes et donc, chacun des sous-projets associés. Chaque sous-système réutilisé étant associé à son propre sous-projet de développement, ce sous-projet et ses tâches doivent être adaptés, permettant ainsi d'offrir un cadre de pilotage de l'activité d'identification des exigences inhérentes à ce sous-système, de recherche d'alternatives système compatibles, d'adaptation de ces sous-systèmes, etc. L'avantage est qu'à chaque niveau, les exigences ou les solutions d'un sous-système, même si elles sont réutilisées sans modification, peuvent être remises en question lors de la phase d'analyse de faisabilité en prenant en compte d'inévitables évolutions de normes, de contextes, de technologies, de fournisseurs, etc. Les limites de cette démarche viennent du fait que, pour des systèmes comportant de très nombreux niveaux de sous-systèmes, le processus récursif doit être réitéré de très nombreuses fois, en s'imposant d'analyser faisabilité et vérification pour chaque entité de sous-projet et de sous-système associée. La figure 6.14 illustre le processus d'adaptation récursive.

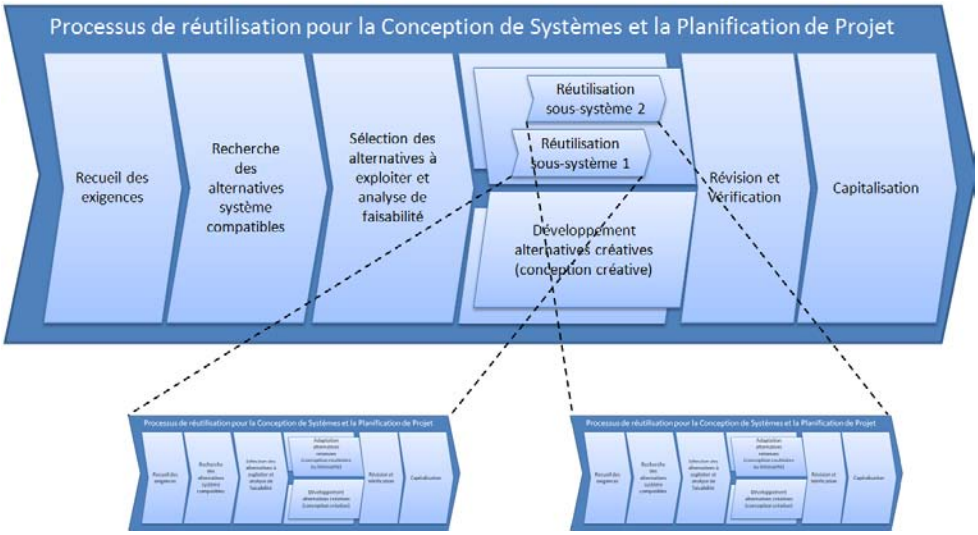


Figure 6.14 : Exemple d'adaptation récursive

6.2.4 Révision, Vérification et Capitalisation

6.2.4.1 *Révision et Vérification*

La **tâche de révision et vérification** permet de :

- compléter, modifier la solution adaptée afin qu'elle soit cohérente par rapport aux exigences (Révision) et,
- contrôler que toutes les exigences système sont satisfaites (Vérification) et mettre à jour les attributs correspondants.

Ici également, les deux domaines conception et planification sont considérés :

- l'adaptation d'informations sur une tâche TD peut être révisée (sa faisabilité est remise en cause) ;
- l'adaptation d'une alternative système compatible peut être révisée (sa faisabilité est remise en cause) ;
- une tâche TD doit être vérifiée après réalisation ;
- une alternative système doit être vérifiée après développement.

Concernant la vérification, cette dernière est réalisée conformément aux propositions faites dans la section 3.

6.2.4.2 *Capitalisation*

La tâche de capitalisation a pour objet de déterminer si l'alternative système et la tâche de développement d'alternative associée doivent être stockées ou non dans la base de cas. Dans les approches de RàPC, une analyse experte de la solution peut éventuellement empêcher sa mémorisation. Pour notre approche, un cas est systématiquement capitalisé, même si les attributs de faisabilité et de vérification ne sont pas respectivement dans l'état « faisable » et « vérifié ». De tels systèmes et projets ne peuvent être réutilisés en l'état. De plus, leur analyse ultérieure pourra permettre de dégager des règles à respecter pour éviter de réitérer les mêmes erreurs et constituer ainsi de nouvelles connaissances. D'autres travaux, non présentés ici, abordent par exemple cet aspect en utilisant un processus de résolution de problème de type 8D ou 6-sigma sur les projets qui se sont bien déroulés et ceux qui se sont mal déroulés [**Kamsu Foguem et al., 2008**].

6.3 Illustration du couplage par réutilisation de cas

Dans l'exemple, nous proposons, à partir des exigences systèmes de l'exemple de la section 5.6, de procéder à une recherche d'alternatives système compatibles puis, après sélection d'une alternative système, d'adapter cette dernière ainsi que la tâche de développement qui lui est associée. Enfin, nous vérifierons et capitaliserons cette nouvelle conception dans la base de cas.

6.3.1 Recueil des exigences

Les exigences système ES du système d'identifiant IdS = X9 utilisées dans cet exemple sont celles proposées dans l'exemple du chapitre 5. Nous rappelons que, pour cet exemple, les exigences système comportent deux contraintes : $l_0 < 1m$ et $e < 2cm$.

Le responsable de conception souhaite exprimer ses préférences sur la contrainte $l_0 < 1m$. La valeur limite au-delà de laquelle le responsable de conception n'est plus intéressé est 1,2m. La fonction de compatibilité pour cette contrainte est représentée dans la figure 6.15.

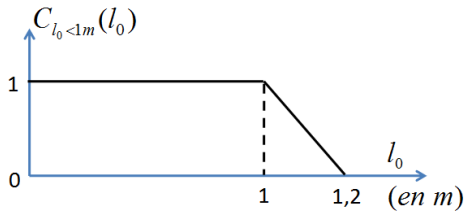


Figure 6.15 : Fonction de compatibilité pour la contrainte $l_0 < 1m$ intégrant les préférences du concepteur

Par ailleurs, il souhaite utiliser la connaissance a priori sur les similarités entre valeurs de la variable e. Par ailleurs, une fonction de similarité sur cette valeur de la variable e existe. La fonction de compatibilité pour cette contrainte est également représentée dans la figure 6.16.

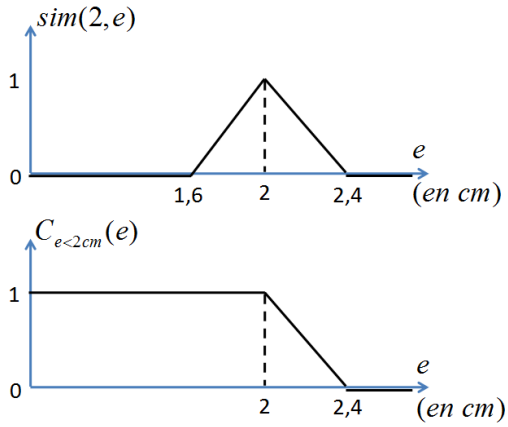


Figure 6.16 : Fonction de similarité entre la valeur 2 de la variable e et les valeurs du domaine de la variable e et fonction de compatibilité pour la contrainte e < 2cm intégrant les similarités

6.3.2 Recherche des alternatives compatibles

Une fois les exigences définies, le responsable de conception peut lancer la recherche d'alternatives système compatibles. Ainsi, la présélection d'alternatives système met en avant les deux alternatives système d'identifiants IdAS = X5.1 et IdAS = X7.1 comme présenté en figure 6.17.

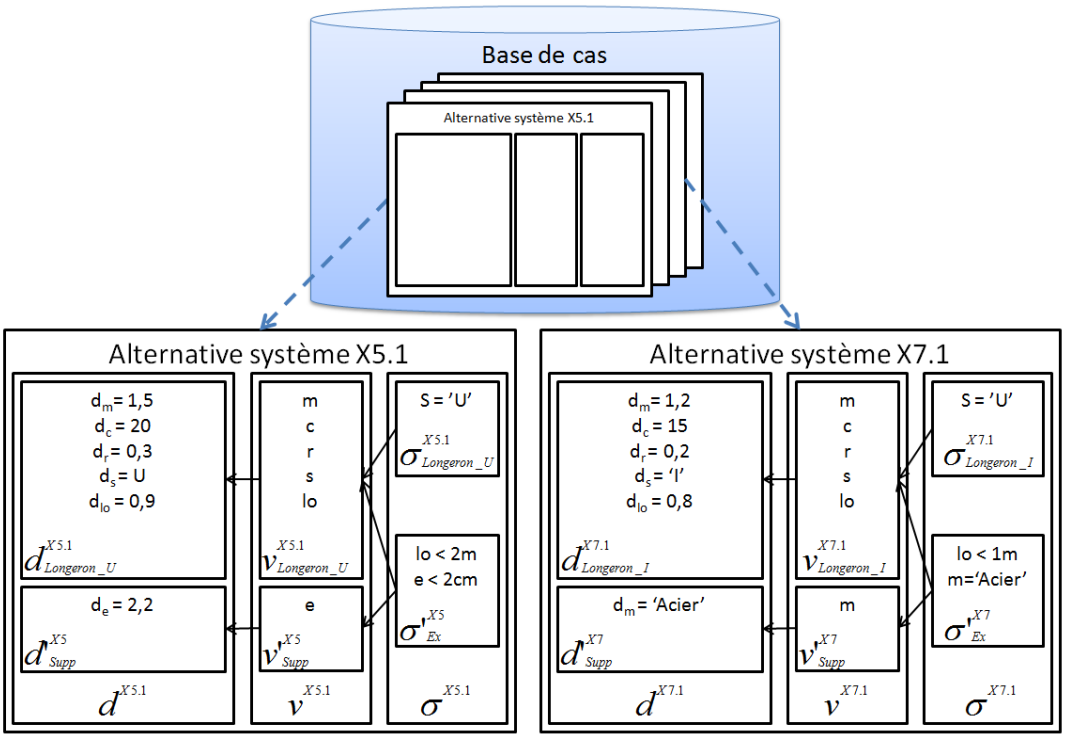


Figure 6.17 : Alternatives présélectionnées

Ces deux alternatives système sont respectivement associées aux concepts « Longeron_U » et « Longeron_I ». Conformément à l'ontologie partielle proposée sur la figure 5.8 (section 5.6.1), ces deux concepts sont des concepts fils du concept associé aux exigences système (« Longeron »). Ils possèdent donc la même distance sémantique :

$$Sim(Longeron, Longeron\_U) = Sim(Longeron, Longeron\_I) = \frac{2 * 2}{3 + 2} = 0{,}8$$

La mesure de compatibilité globale de chacune de ces alternatives système par rapport aux exigences systèmes est alors calculée. Dans un premier temps, la compatibilité de chaque alternative système avec chaque contrainte des exigences système piouioyoiuiouiouiouiouiouioest calculée ci-dessous :

$$C^{X5.1}_{(l_0<1m)} = 1 \qquad C^{X5.1}_{(e<2cm)} = 0{,}5 \qquad C^{X7.1}_{(l_0<1m)} = 1 \qquad C^{X7.1}_{(e<2cm)} = 0$$

Dans un second temps, la mesure de compatibilité globale peut alors être calculée à partir des compatibilités locales. Dans le cas présent, les paramètres de la fonction de Minkowski permettant d'agréger les compatibilités locales sont $p = 2$ et $w_{(l_0<1m)} = w_{(e<2cm)} = 0{,}5$. Les compatibilités globales des alternatives système IdAS = X5.1 et IdAS = X7.1 sont donc :

$$C^{X5.1}_{\sigma^{X9}} = [0{,}5 * (1)^2 + 0{,}5 * (0{,}5)^2]^{1/2} = 0{,}79$$

$$C^{X7.1}_{\sigma^{X9}} = [0{,}5 * (1)^2 + 0{,}5 * (0)^2]^{1/2} = 0{,}71$$

L'alternative système d'identifiant IdAS = X5.1 étant la plus compatible avec les exigences système, le responsable de conception décide de la réutiliser en l'adaptant puisqu'elle n'est pas totalement compatible. À ce stade de la conception, l'alternative système d'identifiant IdAS = X5.1 et

la tâche de développement d'alternative d'identifiant IdTD = TD5.1 sont dupliquées avec un nouvel identifiant et les attributs de faisabilité et de vérification de ces copies sont à « non déterminé ».

6.3.3 Adaptation de la tâche de développement et de l'alternative système

Nous avons vu dans la section 6.5 que la tâche de développement d'alternative devait être d'abord dupliquée et la nouvelle tâche TD adaptée avant d'entreprendre la duplication et l'adaptation de l'alternative système retenue. La figure 6.18 illustre la tâche de développement d'identifiant IdTD = TD5.1 dupliquée en IdTD = TD9.2 associée à l'alternative d'identifiant IdAS = X5.1 d'une part, et la duplication et l'adaptation qui en est faite en X9.2 par le responsable de planification. Les adaptations sont les suivantes :

- la durée (du) de la tâche TD9.2 passe de 3 à la valeur 1,5,
- la date de fin passe de 8 à 6,
- le coût projet passe de 200 à 80 et,
- le risque est diminué, passant de 0,3 à 0,2.

La date de début au plus tôt et les type et quantité de ressources restent inchangées par rapport à la tâche de développement d'alternative d'identifiant IdTD = TD5.1.

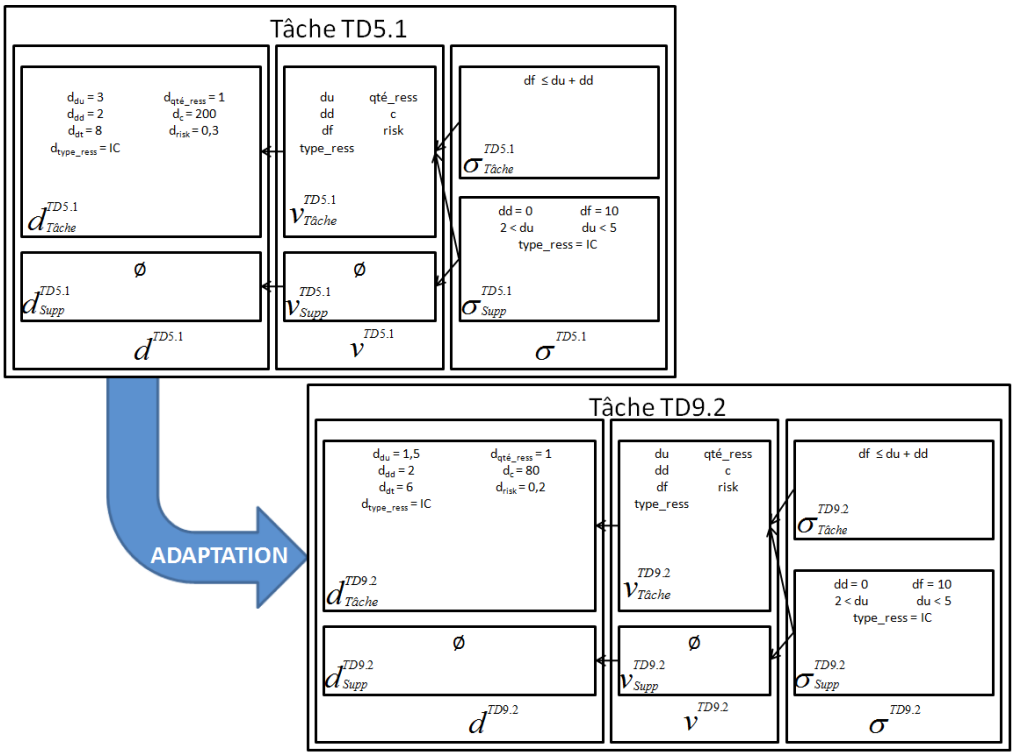


Figure 6.18 : Adaptation d'une tâche de développement d'alternative

Après que la tâche de développement d'identifiant IdTD = TD9.2 ait été planifiée, l'alternative peut à son tour être adaptée. Ici, seule l'épaisseur (e) n'est pas compatible avec les exigences système initiales. Le responsable de conception propose donc d'utiliser un nouveau profilé d'épaisseur inférieure. Suite à une recherche sur catalogue, il a identifié un profilé utilisable dont l'épaisseur est de 1,8cm. L'alternative X5.1 à adapter et, d'autre part, l'alternative X9.2 qui est le résultat de l'adaptation sont illustrés sur la figure 6.19.

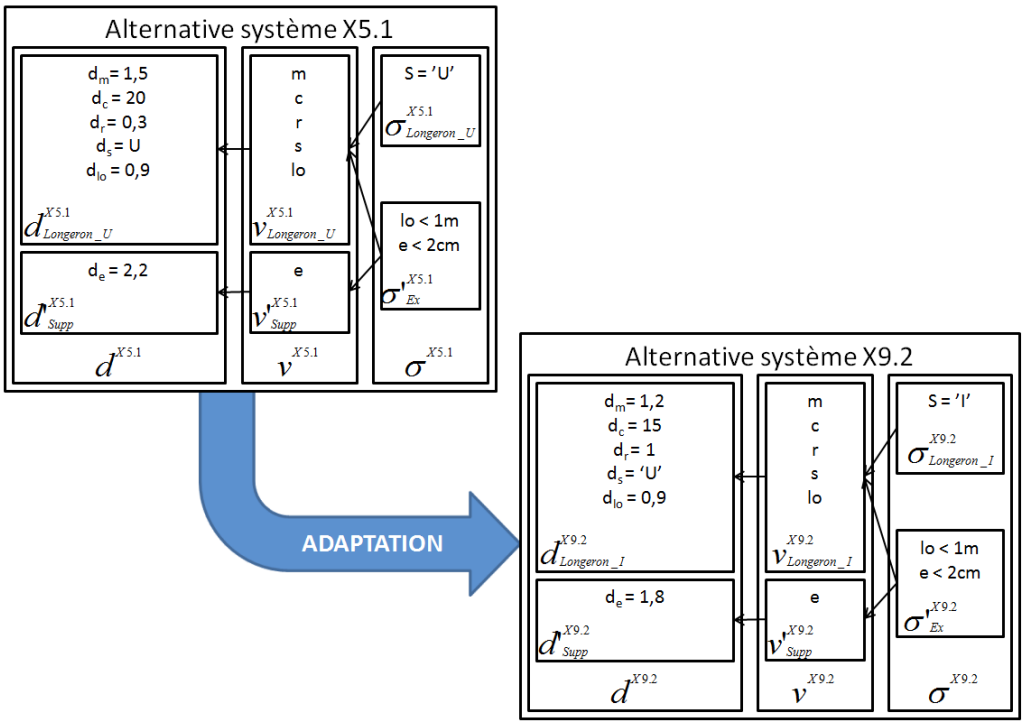


Figure 6.19 : Adaptation d'une alternative système

Quand le développement de l'alternative système d'identifiant IdAS = X9.2 est terminé, cette dernière doit être vérifiée. Ainsi, on s'assure que toutes les exigences système ont bien été satisfaites. Ici, on a $l_0 = 0{,}9 < 1m$ et $e = 1{,}8 < 2cm$. Les exigences sont bien respectées, l'alternative système d'identifiant IdAS = X9.2 est qualifiée de *vérifiée*. La même démarche est lancée par le responsable de planification afin de s'assurer que la tâche de développement d'identifiant IdTD = TD9.2 satisfait bien les contraintes qui lui étaient assignées.

À ce stade, la conception et la planification sont terminées, l'alternative système X9.2 et la tâche de développement d'alternative TD9.2 sont enregistrées dans la base de cas.

6.4 Conclusion sur le couplage par réutilisation de cas

Nous avons proposé dans cette section un processus de réutilisation de cas pour la conception de systèmes et la planification des projets associés. Ce processus consiste à :

- définir les exigences système et, éventuellement, y intégrer les préférences du concepteur ou de la connaissance experte sur les similarités entre valeurs de variables,
- rechercher les alternatives système compatibles avec les exigences système,
- adapter la ou les tâches de développement d'alternative puis les alternatives système retenues,
- vérifier, après réalisation, que ces dernières satisfont bien les contraintes qui leur étaient assignées et,
- capitaliser les alternatives système et les tâche de développement associées dans une base de cas.

Quand plusieurs cas proches sont stockés dans la base de cas, cela signifie que les concepteurs et planificateurs ont acquis une connaissance sur ce type d'alternatives. Dans certaines situations, il est possible de dégager des règles générales de conception, et/ou de planification. La définition ainsi que l'exploitation de ces règles sont l'objet du chapitre suivant.

Chapitre 7 : Couplage par contraintes

7 Couplage par contraintes

Dans le chapitre précédent, nous avons proposé des mécanismes de réutilisation de cas utilisant la connaissance capitalisée au fur et à mesure des différents projets de conception. Nous abordons maintenant une autre forme de connaissance métier basée sur la proposition de relations génériques. Ces relations, nommées contraintes, sont définies par des experts du domaine en amont de la conception et de la planification. Elles peuvent provenir soit de connaissances académiques (lois physiques par exemple), des bonnes pratiques de l'entreprise, de connaissances d'experts et/ou être extraites des cas stockés. Ces contraintes sont associées aux concepts de l'ontologie proposée dans le chapitre 5. Dans le but de coupler les entités de conception et de planification, nos propositions sont limitées à un niveau de décomposition, c'est-à-dire que nous ne traitons pas les contraintes pouvant exister entre un système et ses sous-systèmes, un projet et ses sous-projets et, au-delà, les contraintes « diagonales » entre système et sous-projets ou projet et sous-systèmes.

Dans ce chapitre, nous définissons, dans un premier temps, les problèmes de satisfaction de contraintes (CSP) et leur lien existant avec l'ontologie que nous avons proposé dans le chapitre 5. Puis nous abordons l'utilisation des CSP pour l'aide à la décision dans les environnements de conception et de planification pour aboutir à une définition du couplage par contraintes. Dans un deuxième temps, nous formalisons ce couplage en complétant le modèle de classe proposé dans la section 5 et en positionnant un processus de propagation de contraintes dans le processus intégré de conception et de planification. Enfin, nous illustrons notre proposition de processus de propagation de contraintes dans la conception/planification d'un longeron.

7.1 Définition des problèmes de satisfaction de contraintes et lien avec l'ontologie

Dans cette section, nous définissons, dans un premier temps, les problèmes de satisfaction de contraintes (CSP) ainsi que ses liens avec l'ontologie proposée dans la section 5. Puis, nous présentons les méthodes de traitement des CSP permettant l'aide à la décision : la résolution et le filtrage. Enfin, nous synthétisons les approches retenues dans le cadre de nos travaux de thèse.

7.1.1 Définition d'un CSP et lien avec l'ontologie

Un problème de satisfaction de contraintes est défini par un triplet (X, D, C) [**Montanari, 1974**] où :

- X est un ensemble fini de variables,
- D est un ensemble fini de domaines de définition des variables et,
- C est un ensemble fini de contraintes portant sur les variables où une contrainte décrit les combinaisons autorisées ou exclues des valeurs des variables.

Les variables peuvent prendre des valeurs soit numériques continues, soit numériques ou symboliques discrètes, soit des valeurs temporelles conformément à la classification proposée dans [**Vareilles, 2005**]. Nous parlons de CSP discret dès lors que l'ensemble des variables est discret, continu quand l'ensemble des variables est continu, temporel si toutes les variables ont des valeurs temporelles et mixte si les variables sont de différents types.

Les contraintes peuvent être :

- des contraintes de compatibilité qui permettent de définir les combinaisons de valeurs autorisées pour un ensemble de variables. Ces contraintes peuvent prendre la forme de

tables de compatibilité continues, mixtes ou discrètes, de formules mathématiques ou de formules logiques [**Gelle, 1998**]. Par exemple, C_1 : $m > 2 * nb\_pass$ est une contrainte de compatibilité entre les deux variables continues m (masse totale de l'avion) et nb_pass (nombre de passagers que peut accueillir l'avion) ;

- des contraintes d'activations qui permettent de gérer de manière explicite la pertinence [**Van Oudenhove de Saint Géry, 2006**] des variables en autorisant ou en interdisant leur activation [**Mittal et Falkenhainer, 1990**]. Par exemple, prenons une variable continue ra (rayon d'action) initialement active et une variable discrète res_supp (réservoir supplémentaire) initialement inactive, et la contrainte suivante C_2 : $ra > 2000 \xrightarrow{ACT} res\_supp$. Cette contrainte d'activation spécifie que si la valeur de ra est supérieure à 2000 (km) il faut un réservoir supplémentaire : la variable res_supp s'active et vient s'ajouter au problème courant. Ce type de contrainte ne sera pas abordé dans ce mémoire.

D'après la définition des concepts proposée dans la section 5.1, ces derniers disposent d'un triplet (X, D, C) que l'on peut apparenter à un CSP « mixte » c'est à dire possédant des variables continues et discrètes. En effet, dans les concepts spécialisés de « Système », les variables décrivant les concepts peuvent être continues (par exemple, le rayon d'action) ou discrètes (par exemple, le nombre de passagers). De même, dans les concepts « Projet » et « Tâche », les variables décrivant les concepts peuvent être continues (par exemple, la durée d'une tâche) ou discrète (par exemple, le type de ressource nécessaire à la réalisation d'une tâche).

Suite au choix d'un concept par le concepteur, ou le directeur de programme, le CSP mixte présent dans le concept est associé à l'entité à décrire. Son utilisation va permettre de réduire le domaine des variables de concept au fur et à mesure de la description des besoins, solutions, projets et tâches, et ainsi guider la conception et la planification.

7.1.2 Résolution et filtrage d'un CSP

Il existe deux types de traitement pour un CSP en fonction des attentes de l'utilisateur. Ce dernier peut vouloir :

- trouver une solution ou toutes les solutions (résolution) ou,
- réduire les domaines des variables en supprimant les valeurs ne menant pas à une solution (filtrage) ou,
- valider ses exigences par rapport aux connaissances du domaine (résolution ou filtrage).

7.1.2.1 *Résolution d'un CSP*

La résolution d'un CSP consiste à trouver une ou plusieurs solutions à un problème donné, c'est-à-dire une ou plusieurs instanciations de toutes les variables impliquées dans le CSP en respectant toutes les contraintes. Il s'agit, à partir de l'instanciation d'un ensemble de variables correspondant aux exigences, de fournir une ou plusieurs solutions respectant, d'une part, les contraintes du CSP, et d'autre part, ses exigences.

Il existe deux types de méthodes de résolution d'un CSP :

- les méthodes complètes qui explorent de manière systématique l'espace de recherche et sont capables de fournir toutes les solutions d'un problème comme la méthode de *retour arrière* [**Golumb et Baumbert, 1965**] et,

- les méthodes incomplètes qui n'explorent pas de façon systématique l'espace de recherche et ne fournissent donc qu'un sous-ensemble des solutions d'un problème comme la méthode de *recherche tabou* [**Glover et Laguna, 1993**] ou le *recuit simulé* [**Kirkpatrick et al., 1984**].

À l'issue de la résolution complète ou incomplète d'un CSP, nous obtenons un ensemble de solutions cohérentes avec les contraintes et les exigences. L'avantage de la résolution d'un CSP réside principalement dans l'exactitude des solutions proposées si elles existent. En effet, toutes les solutions proposées répondent parfaitement aux contraintes inhérentes au CSP et aux exigences émises par l'utilisateur. En revanche, le temps de calcul de ces solutions peut être important si le CSP est de taille importante (complexité exponentielle) ou possède des variables continues.

Par exemple, prenons un CSP défini par :

- trois variables : Nb_Pass (le nombre de passagers), RA (le rayon d'action) et Mot le type de moteur,
- les domaines de définition : d_{Nb_Pass} = {20, 50, 100}, d_{RA} = {1000, 2000, 5000} et d_{Mot} = {'M1', 'M2'}, il existe donc a priori 3x3x2= 18 solutions possibles,
- trois contraintes de compatibilité réduisant le nombre de solutions autorisées à 9 :

C1 :

Nb_Pass	RA
20	1000
50	≤ 2000
100	≤ 5000

C2 :

Mot	RA
M1	≤ 2000
M2	≤ 5000

C3 :

Mot	Nb_Pass
M1	≤ 50
M2	≤ 100

L'utilisateur pose comme exigences : Nb_Pass = 50 et RA = 2000. L'algorithme de résolution va alors évaluer chaque combinaison et établir si chacune respecte les contraintes inhérentes au CSP et aux exigences émises par l'utilisateur. Ici, il existe deux solutions : S1 (50, 2000, M1) et S2 (50, 2000, M2).

7.1.2.2 *Filtrage d'un CSP*

Dans les méthodes de résolution présentées ci-dessus, les contraintes sont utilisées de manière passives, c'est-à-dire qu'elles sont uniquement exploitées pour tester la cohérence des affectations partielles et complètes. Au contraire, les méthodes de filtrage utilisent les contraintes de manière active pour effectuer des déductions sur le problème. L'objectif des méthodes de filtrage est la détection d'affectations localement ou globalement incohérentes. La technique la plus fréquemment utilisée est la cohérence d'arc qui vérifie que toutes les valeurs d'une variable sont compatibles avec chaque contrainte prise séparément. La cohérence d'arc se décline suivant le type de CSP à traiter, nous pouvons citer, entre autres, [**Montanari, 1974**] pour les CSP discrets ou [**Lhomme, 1993**] pour les CSP continus.

Les méthodes de filtrage permettent de répercuter les choix de l'utilisateur sur le problème courant. Les contraintes étant non orientées, le filtrage permet, en modifiant le domaine de définition d'une variable donnée, de réduire les domaines de définition des autres variables impliquées dans la contrainte et ce quelle que soit la variable d'entrée. La recherche de solution est alors interactive : c'est la séquence de choix cohérents qui conduit à une ou plusieurs solutions.

À l'issue du filtrage d'un CSP, les solutions non cohérentes avec les contraintes et les exigences proposées par l'utilisateur sont écartées. Nous obtenons donc un espace réduit de solutions

possibles. L'avantage du filtrage d'un CSP est que toutes les solutions sont conservées. Cependant, des non-solutions peuvent également être conservées selon le degré de filtrage utilisé (arc, chemin, k,.. consistance).

Reprenons l'exemple de la section précédente. Dans le cas du filtrage, l'utilisateur va renseigner ses exigences une par une et filtrer à chaque fois. La première exigence est Nb_Pass = 50. Le filtrage va donc permettre d'éliminer toutes les valeurs de chaque variable incompatibles. L'état des domaines des variables sera le suivant : d_{Nb_Pass} = {50}, d_{RA} = {1000, 2000} et d_{Mot} = {'M1', 'M2'}. La seconde exigence est RA = 2000. Similairement, le filtrage va éliminer les valeurs de chaque variable qui sont incompatibles : d_{Nb_Pass} = {50}, d_{RA} = {2000} et d_{Mot} = {'M1', 'M2'}.

Il est également possible d'utiliser les méthodes de filtrage afin d'améliorer les algorithmes de résolution. Il s'agit ici de réduire l'espace de solutions avant d'appliquer l'algorithme de résolution. Par exemple, le *Forward checking* [**Haralick et Elliot, 1980**] combine le filtrage par arc-cohérence avec un algorithme de retour arrière.

Dans la mesure où nous souhaitons fournir aux utilisateurs une aide à la décision interactive au niveau de la conception, de la planification et de leur couplage, nous utiliserons donc les méthodes de filtrage de CSP.

7.2 CSP, conception système et planification de projet

Dans cette section, nous nous intéressons dans un premier temps à l'utilisation des CSP pour l'aide à la conception et à la planification. Puis nous détaillons, pour ces deux aspects, les points communs avec l'ontologie définie dans la section 5.

7.2.1 Utilisation des CSP pour l'aide à la conception et la planification

Dans la section 5 et de manière identique aux travaux présentées dans [**Benaissa et Lebbah, 2011**], nous avons proposé d'associer à chaque entité de conception et de planification un concept choisi au préalable. Les modèles de variables, modèles de domaines et modèles de contraintes sont alors copiés dans l'entité concernée. Soit X un système, une alternative système, un projet ou une tâche. Le problème de conception ou de planification peut alors être décrit avec :

- des modèles de variables inhérents au concept v_{CX}^{X} soit en réduisant les domaines de définition des modèles de variables (un sous ensemble des valeurs du domaine de définition initial),
- et/ou de nouvelles contraintes et le cas échéant, de nouvelles variables v_{supp}^{X} avec leurs domaines de définition (comme proposé dans la section 5.3.1).

La figure 7.1 illustre un problème de conception ou de planification. Les variables conceptuelles v_{CX}^{X} peuvent soit être liées entre elles par des contraintes conceptuelles (pointillés noirs), soit ne pas être liées. De même, les variables supplémentaires v_{supp}^{X} peuvent soit être liées entre elles, soit être liées avec des variables conceptuelles (pointillés gris), ou encore ne pas être liées.

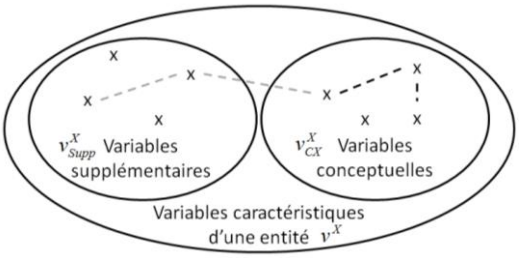


Figure 7.1 : Illustration d'un problème de conception ou de planification

Deux possibilités d'aide à la décision sont données à l'utilisateur :

- soit il renseigne l'intégralité de ses exigences et demande ensuite un filtrage. Ceci permet de confronter les exigences exprimées au CSP et ainsi s'assurer que les exigences sont en conformité avec la connaissance du concept. Ainsi, nous obtenons soit un ensemble de domaines des variables réduits si les exigences sont conformes, soit aucune solution dans le cas contraire ;
- soit il renseigne une exigence, demande un filtrage et réitère cette opération autant de fois que nécessaire. Dans ce cas, les domaines des variables sont réduits progressivement. Ceci permet donc une définition progressive des exigences en conformité avec la connaissance associée au concept.

Dans les deux cas, le filtrage du CSP sur des domaines réduits de variables de v^X permet de qualifier en partie la faisabilité de l'entité concernée.

Nous proposons de ne pas automatiser le déclenchement du filtrage concernant les contraintes propres à un environnement donné. Le déclenchement de la propagation est initié par un des utilisateurs, c'est-à-dire par le responsable de conception ou le responsable de planification car la propagation des contraintes doit être perçue comme une aide à la décision qui peut être utilisée à la demande.

7.2.2 CSP et conception système

Nous avons proposé dans la section 5.2 que tout concept descendant du concept « Système » soit associé à un CSP. Ceci permet, via le concept, d'associer de la connaissance de conception, modélisée sous forme d'un CSP, à une entité de conception. Ces connaissances peuvent être très différentes en fonction du type de système à concevoir. En effet, les connaissances impliquées pour la conception d'un composant mécanique (par exemple, le volume) ne seront vraisemblablement pas les mêmes que pour un composant électronique (par exemple, la puissance consommée) et par extension, il n'est pas possible d'établir des connaissances applicables à tout type de système : chaque concept disposera donc de son propre modèle de connaissances formalisé sous forme d'un CSP. Toutefois, nous avons proposé dans la section 5.4 de dégager trois variables (et leurs domaines de définition) indissociables d'un système quel qu'il soit : le coût système, le risque associé à la conception du système et la masse du système.

La connaissance d'un concept, modélisée sous la forme d'un CSP, peut se décliner sur l'ensemble des variables inclues dans le concept. L'utilisation des méthodes de filtrage permet de réduire les domaines des variables grâce aux contraintes et ainsi d'aider à la conception.

7.2.3 CSP et planification de projet

La connaissance propre aux concepts « Projet » et « Tâche » est définie similairement à celle des concepts « Système », c'est-à-dire à l'aide d'un CSP (voir section 5.2). Mais contrairement aux

concepts descendants du concept « Système », la diversité est moindre pour les concepts « Projet » et « Tâche ». Il est donc défini, dans la section 5.4, des modèles de variables valables quel que soit le projet. Pour rappel, ces modèles de variables sont : le début, la fin, la durée, le type et la quantité de ressources, le risque et le coût du projet.

Conformément aux travaux de [**Barták et al., 2010**], nous faisons la distinction entre deux types de contraintes en planification :

- les contraintes temporelles liant le début, la fin et/ou la durée d'une même tâche ou les débuts et/ou fins de plusieurs tâches,
- les contraintes liant la durée au type et à la quantité de ressources.

Similairement aux travaux présentés dans [**Djefel, 2010**], nous limitons les contraintes aux variables de durée, type de ressources, quantité de ressources, risque et coût. Par exemple, si la ressource de type « ingénieur senior » est employée pour une tâche, alors sa durée estimée sera réduite de 15% par rapport à la durée normale. Cette contrainte peut être modélisée de la manière suivante :

C1 :

Type_Ress	Coef
ingénieur junior	1
ingénieur sénior	0,85

C2 :

Durée_estim = Coef x Durée

Les connaissances liant les dates de début et de fin, comme par exemple les contraintes liant une ressource à la date de début de la tâche (M. RespC, début = janvier) correspondent plus à des exigences projet qu'à des connaissances génériques : nous prendrons donc en compte ces contraintes comme exigences de planification.

7.2.4 **Définition du couplage par contraintes**

Le couplage par contraintes permet de propager les décisions de conception vers la planification et inversement, les décisions de planification vers la conception sur un même niveau de décomposition. Les contraintes étant non orientées, elles permettent de supporter ces échanges d'informations dès lors qu'elles relient des variables de conception et des variables de planification. La figure 7.2 illustre le couplage d'un CSP lié à un concept descendant du concept « Système » et d'un CSP lié à un concept « Tâche ». Chaque croix représente une variable et chaque ligne en pointillé symbolise une contrainte, soit propre à un environnement en gris, soit de couplage en noir.

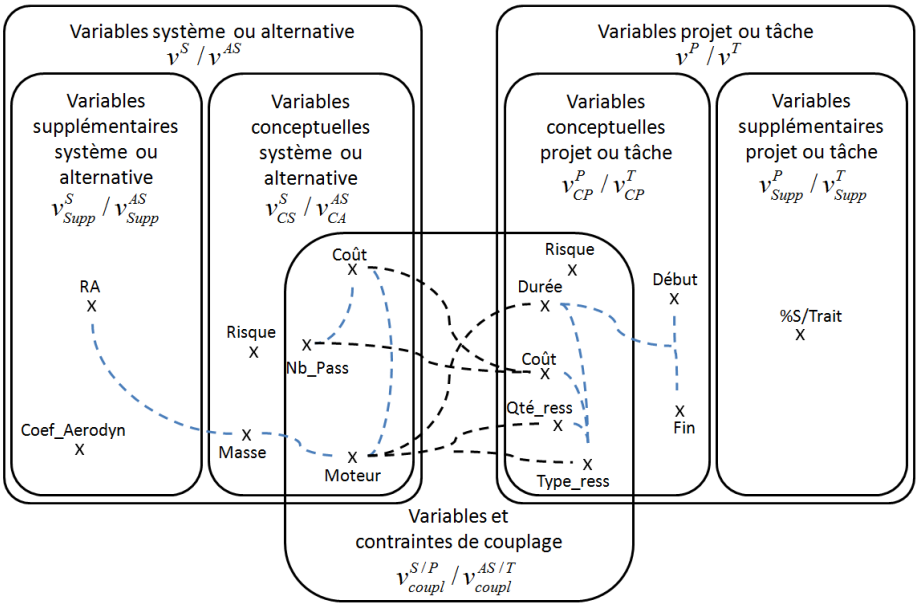


Figure 7.2 : Couplage de deux CSP par des contraintes de couplage

Similairement aux travaux de [**Djefel, 2010**] et en nous basant sur les ensembles de variables proposés dans la section sur la formalisation de la connaissance métier (section 5), nous distinguons trois ensembles de variables :

- l'ensemble des variables système $v^S = v^S_{CS} \cup v^S_{supp}$ ou alternative $v^{AS} = v^{AS}_{CA} \cup v'^S_{supp} \cup v^{AS}_{supp}$,
- l'ensemble des variables tâche $v^{TD} = v^{TD}_{CT} \cup v^{TD}_{supp}$ ou $v^{TE} = v^{TE}_{CT} \cup v^{TE}_{supp}$,
- l'ensemble des variables de couplage qui peuvent être des variables conceptuelles système v^S_{CS} et projet v^{TE}_{CT} ou des variables conceptuelles alternative v^{AS}_{CA} et tâche v^{TD}_{CT}. Si les contraintes de couplage lient un système à un projet, nous notons l'ensemble des variables de couplage $v^{S/TE}_{coupl}$ et si elles lient une alternative système à une tâche, l'ensemble est noté $v^{AS/TD}_{coupl}$.

Toutes les variables conceptuelles système v^S_{CS} ou alternative v^{AS}_{CA} sont susceptibles d'être des variables de couplage. En revanche, au niveau des variables conceptuelles tâche v^{TD}_{CT} ou v^{TE}_{CT}, nous limitons les variables de couplage aux : type de ressource, quantité de ressource et, par extension, à la durée, au risque et au coût. En effet, un choix technologique (par exemple la décision de concevoir un composant électronique) en conception peut avoir un impact sur les ressources en termes de quantité et de compétence en planification (l'affectation d'une ressource de type électronicien) et inversement. De plus, il existe des contraintes en planification liant type et quantité de ressources à la durée, au coût ainsi qu'au risque : il est envisageable, pour simplifier, de lier directement durée, coût et risque à des variables système ou alternative.

Les contraintes de couplage représentent de la connaissance de couplage qui doit être générique, c'est-à-dire applicable quelle que soit la conception ou la planification : ni le responsable de conception, ni le responsable de planification ne peuvent modifier ces contraintes ou en ajouter en cours de conception et de planification. Leur création et leur mise à jour, sont réalisées par un expert hors du contexte d'une conception ou d'une planification spécifique.

7.3 Formalisation du couplage par contraintes

Dans cette section, nous formalisons le couplage par contraintes. Dans un premier temps, nous identifions deux types de connaissances de couplage, et formalisons le couplage par propagation de contraintes puis nous présentons le processus associé. Enfin, nous positionnons le processus de filtrage dans le processus de planification et de conception intégré global et nous évoquons le processus de mise à jour des modèles de connaissances.

7.3.1 Supports des connaissances de couplage

Nous avons identifié deux types de connaissances de couplage :

(1) des contraintes liant concept(s) et variable(s), c'est-à-dire que le choix d'un concept a un impact sur les valeurs des variables d'un autre concept,

(2) des contraintes liant des variables de concepts différents, c'est-à-dire que le choix d'une valeur pour une variable d'un concept impacte les valeurs d'autres variables appartenant à un autre concept.

(1) Le choix d'un concept peut impacter les valeurs d'une ou plusieurs variables, c'est-à-dire qu'une connaissance sur le concept d'un système à concevoir va induire d'autres connaissances sur les variables de tâche et inversement. Par exemple, si le concept « Avion bimoteur » est choisi pour une entité de conception donnée, cela implique que la durée de la tâche associée ne pourra pas être inférieure à 6 mois. De même, si le concept « Avion monomoteur » est choisi pour une entité de conception donnée, cela implique que la durée de la tâche associée ne pourra pas être inférieure à 3 mois (contrainte CC1). Inversement, si la durée de la tâche est égale à quatre mois, alors le concept « Avion bimoteur » ne pourra être choisi. Cette contrainte est modélisée ci-dessous.

CC1 :

Concept	Durée
Avion bimoteur	> 6 mois
Avion monomoteur	> 3 mois

Nous avons également considéré de cas où un concept « Tâche » impacte les connaissances sur les variables d'un concept descendant de « Système ». En se basant sur l'ontologie proposée dans la section 5, ce type de contrainte a peu d'utilité. Cependant, si l'ontologie se trouvait plus détaillée (présence de sous-concepts au niveau des branches projet et tâche), nous imaginons que ce type de contraintes pourrait prendre du sens. Dans l'état actuel de nos travaux, nous nous limiterons aux contraintes liant concept descendant de « Système » et variables tâche. Dans ce cas, nous considérons le nom du concept comme une variable conceptuelle à part entière inclue dans l'ensemble v_{CS}^{S} ou v_{CA}^{AS}.

(2) Les modèles de contraintes de couplage peuvent associer les variables d'un concept descendant du concept « Système » et les variables d'un concept « tâche », c'est-à-dire que des exigences exprimées au travers de certaines variables d'un système vont avoir des conséquences sur la tâche associée et inversement. Par exemple, le choix d'un composant X en conception va impliquer une durée de tâche comprise entre 2 et 3 mois en planification (contrainte CC2). Inversement, une durée de tâche inférieure à 3 mois va impliquer un nombre de passagers (Nb_Pass) inférieur à 20 (contrainte CC3). Ces deux contraintes sont modélisées ci-après :

CC2 :

Composant	Durée
X	[2, 3]
T	[3, 5]

CC3 :

Nb_Pass	Durée
> 20	> 3
≤ 20	≤ 3

De manière similaire à l'aide à la décision propre à un environnement, nous proposons de ne pas automatiser le déclenchement du filtrage concernant les modèles de contraintes de couplage. Le déclenchement de la propagation vers l'autre environnement est initié par un des utilisateurs, c'est-à-dire par le responsable de conception ou le responsable de planification.

7.3.2 Formalisation du couplage par propagation de contraintes

Il s'agit, dans cette section, de formaliser les contraintes liant des modèles de variables des concepts descendants du concept « Système » et des modèles de variables des concepts « Tâche » afin de propager les décisions prises dans un environnement sur l'autre. Nous proposons donc d'insérer une classe à part entière nommée « Modèle de contrainte de couplage » liant les modèles de variables associées aux concepts descendants du concept « Système » et les modèles de variables associés au concept « Tâche ».

La figure 7.3 reprend le diagramme de classe proposé dans la section 5.2 augmenté de la classe « Modèle de contraintes de couplage » proposée ci-dessus.

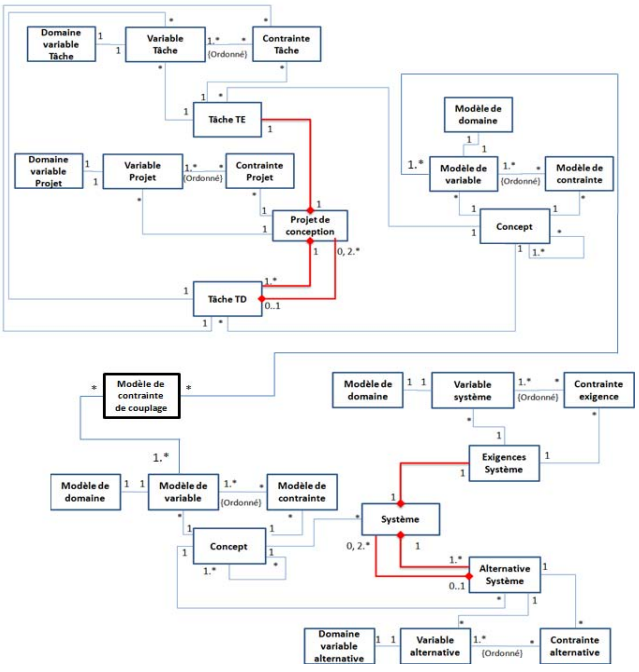


Figure 7.3 : Diagramme de classe supportant le couplage par propagation de contraintes

7.3.3 Processus de filtrage sur l'environnement intégré de conception et de planification

Nous rappelons que le déclenchement du processus de filtrage n'est pas automatique, c'est-à-dire que l'initiation du filtrage est demandée par un des utilisateurs de l'environnement de conception ou de planification. Ceci est valable pour un filtrage propre à un environnement donné ou pour la propagation des résultats d'un filtrage d'un environnement vers l'autre via les contraintes de couplage.

La figure 7.4 illustre le processus de propagation de contraintes entre les CSP de conception et de planification.

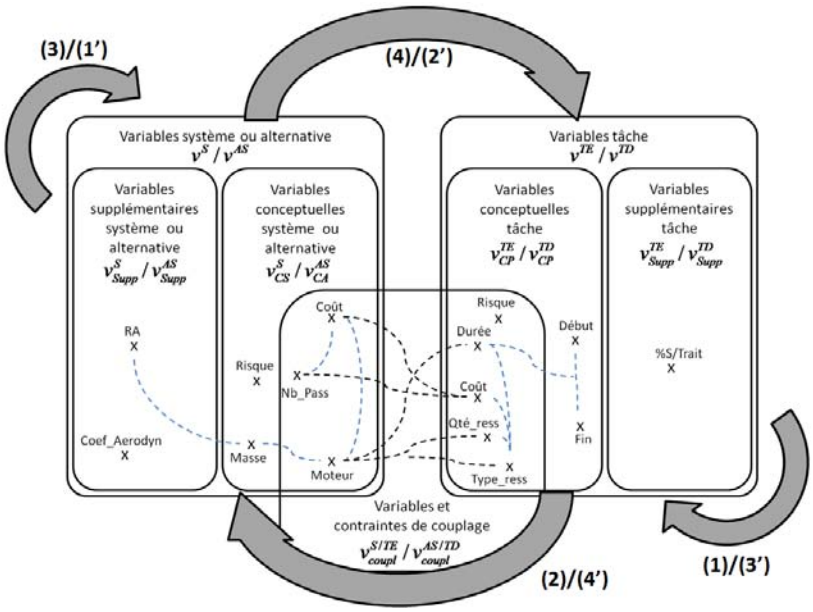


Figure 7.4 : Processus de propagation de contraintes entre les CSP de conception et de planification

(1/1') Le responsable de planification (resp. de conception) fait des choix de planification, c'est-à-dire qu'il réduit les domaines de définition de certaines variables tâche (resp. système), puis décide ou non de propager ses décisions dans son propre environnement. Grâce à ce filtrage, il obtient une réduction des domaines de définition des variables de l'ensemble v^{TE} ou v^{TD} (resp. v^S ou v^{AS}).

(2/2') Si des variables de couplage $v^{S/TE}_{coupl}$ ou $v^{AS/TD}_{coupl}$ ont été impactées par cette propagation, le responsable de planification (resp. de conception) peut décider ou non de propager ses décisions vers l'autre environnement. Si tel est le cas, il y a une réduction des domaines des variables de l'ensemble v^S ou v^{AS} (resp. v^{TE} ou v^{TD}) via les contraintes de couplage.

(3/3') Le responsable de conception (resp.de planification) est informé de la propagation initiée par le responsable de planification (resp. de conception) et peut à son tour décider de propager à son propre environnement. Si tel est le cas, les domaines réduits des variables appartenant à l'ensemble$v^{S/TE}_{coupl}$ ou $v^{AS/TD}_{coupl}$ sont propagés dans l'autre environnement v^S ou v^{AS} (resp. v^{TE} ou v^{TD}).

(4/4') Similairement à l'environnement de planification (resp. de conception), si des variables de couplage $v^{S/TE}_{coupl}$ ou $v^{AS/TD}_{coupl}$ ont été impactées par cette propagation, le responsable de conception (resp. de planification) peut décider ou non de propager ses décisions vers l'autre environnement.

Ce processus (1, 2, 3, 4 ou 1', 2', 3', 4') peut se répéter tant que l'ensemble des domaines réduits des variables n'est pas stabilisé.

7.3.4 Positionnement du processus de filtrage dans le processus global

Dans la section précédente, nous avons proposé un processus de filtrage applicable dans l'environnement de planification et de conception intégré présenté dans la section 2.3. Nous nous intéressons maintenant à son positionnement dans ce dernier.

Nous distinguons deux instants dans la planification/conception où il est possible d'utiliser le processus de filtrage. Ceci est illustré dans la figure 7.5. Il s'agit :

- du suivi de la réalisation de la tâche TE dans le processus de planification (a) et de la réalisation de cette dernière dans le processus de conception (b), c'est-à-dire lors de la définition des exigences système. Ceci est réalisable après le choix du concept système réalisé dans l'activité « Demander création Projet et Système » fig. 2.13 et la définition du planning prévisionnel de la tâche TE ;
- du suivi de la réalisation de la tâche TD dans le processus de planification (c) et de la réalisation de cette dernière dans le processus de conception (d), c'est-à-dire lors de la définition de l'alternative système associée. Ceci est réalisable après le choix du concept alternative réalisé dans l'activité « Demander création m alternatives système AS » fig. 2.13 et la définition du planning prévisionnel de la tâche TD ou en cas de réutilisation.

Les ensembles de variables impliquées dans le premier cas sont les variables système v^S et les variables projet v^{TE} et les variables de couplage $v_{coupl}^{S/TE}$. Dans le second cas, les ensembles de variables impliqués sont les variables alternative v^{AS} et les variables projet v^{TD} et les variables de couplage $v_{coupl}^{AS/TD}$.

Pour chaque activité (a, b, c ou d), lors de la saisie des informations inhérentes à l'activité, le responsable de planification ou le responsable de conception peut demander un filtrage dans son propre environnement sur les variables renseignées. Ce filtrage propre à un environnement est modélisé par la boucle (fl) sur chacune des activités.

Le responsable de planification ou le responsable de conception peut alors demander à répercuter ses modifications dans l'autre environnement : les réductions de domaines sont propagées dans l'autre environnement (modélisé par les flèches (fc) sur le diagramme de la figure 7.5).

Ensuite, comme indiqué dans le processus de filtrage proposé dans la section précédente, la propagation d'un environnement vers l'autre peut se poursuivre jusqu'à stabilisation des domaines de définition de l'ensemble des variables des deux environnements.

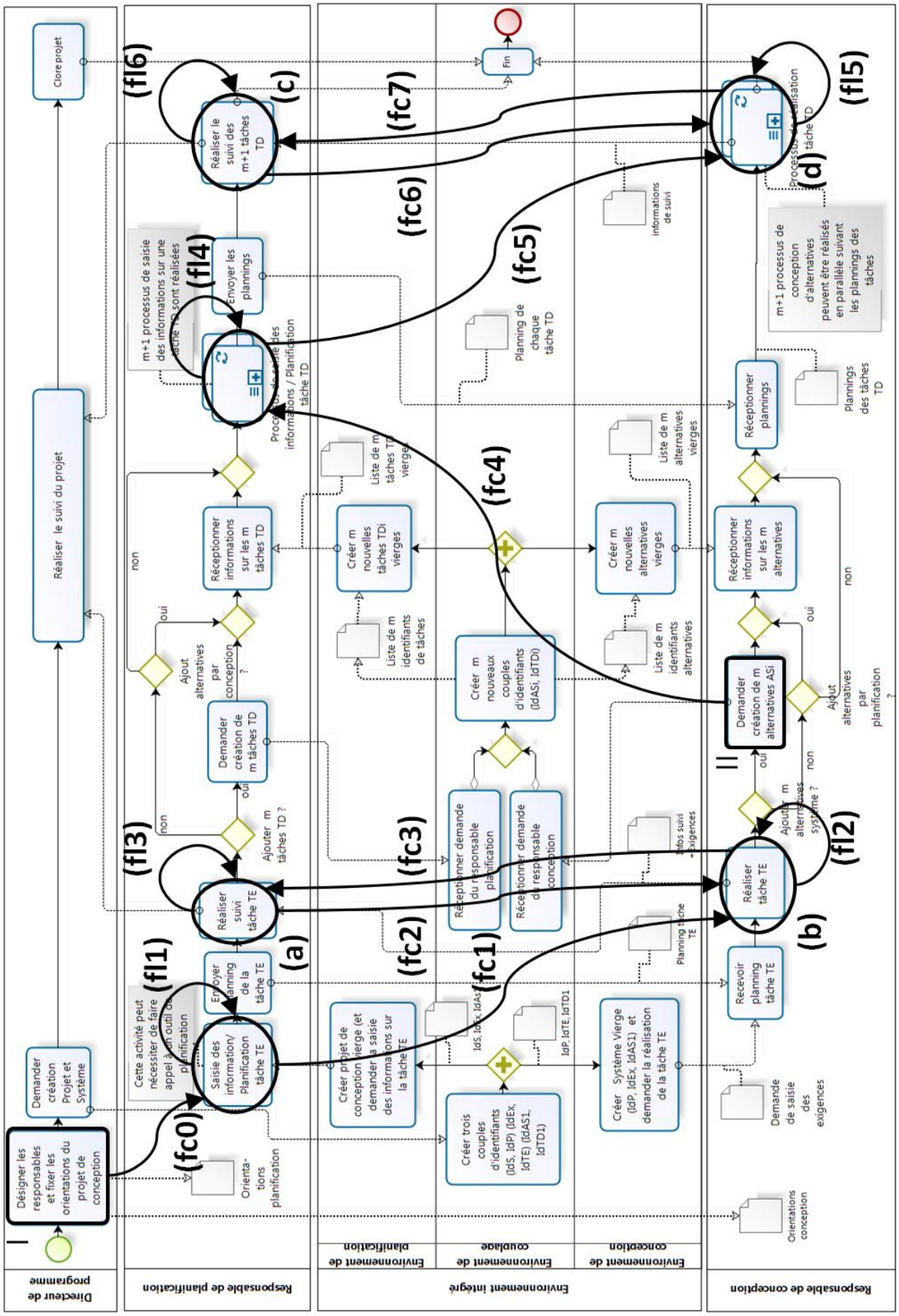


Figure 7.5 : Positionnement du processus de filtrage dans le processus de planification et de conception intégré

7.3.5 Processus de création et de mise à jour des modèles de CSP

Nous avons proposé, en introduction de la section 7, que les connaissances modélisées sous la forme d'un CSP ne puissent pas être modifiées lors de la conception d'un système particulier ou de la planification d'un projet donné. En revanche, les modèles de contraintes de couplage peuvent être créés et mis à jour par un expert hors du contexte d'un projet et/ou d'un système donné.

Nous distinguons deux possibilités de créer ou de mettre à jour les modèles de CSP, c'est-à-dire les modèles de variables, les modèles de domaines associés aux modèles de variables et les modèles de contraintes.

La première possibilité provient de la connaissance de l'expert lui-même (connaissance métier générique et implicite qu'il décide de formaliser) ou de la connaissance générale du domaine (par exemple, lois physiques, électroniques, règles de planification, etc.).

La deuxième possibilité repose sur l'analyse des cas passés stockés dans la base de données (voir section 6). Il s'agit ici de dégager à partir de plusieurs cas, des contraintes génériques applicables en toutes circonstances. Il peut s'agir de modèles de contraintes pour la conception, pour la planification pour un concept particulier ou encore de modèles de contraintes de couplage pour des couples de concepts.

7.4 Illustration du couplage par contraintes

Dans cette section, nous illustrons l'utilisation du couplage par contraintes pour la définition d'un projet et des exigences système associées puis d'une tâche de développement d'alternative et de l'alternative système associée. Nous n'aborderons pas la création des entités qui est déjà illustrée dans la section 2.4.1 et supposerons que les entités ont déjà été créées au préalable.

7.4.1 Structure du problème de conception et de planification

Nous nous situons dans le cadre de la conception d'un longeron. Nous n'étudierons ici qu'un seul niveau de décomposition c'est-à-dire le système de concept « Longeron » et son projet de conception associé et deux alternatives systèmes de concepts respectifs « Longeron_I » et « Longeron_Caisson » ainsi que les tâches de développement d'alternative qui leur sont associées. La figure 7.6 illustre la structure du problème de conception et de planification.

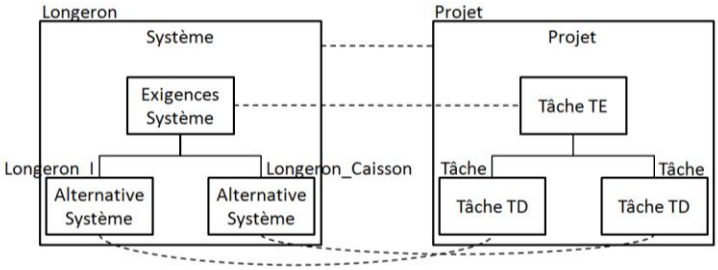


Figure 7.6 : Problème de conception et de planification étudié

Concernant l'ontologie, nous avons fait évoluer cette dernière par rapport à l'exemple proposé dans la section 5.6.1. En effet, suite à la rencontre d'un expert du domaine aéronautique, nous avons pu affiner les modèles de variables, de domaines et de contraintes proposées lors de la précédente version de l'ontologie. Comme indiqué dans le chapitre 5, une ontologie ne doit pas être un référentiel figé : il est nécessaire de la faire évoluer en fonction de l'apparition de nouvelles connaissances générales. L'ontologie ainsi modifiée est présentée ci-dessous. Nous limitons cette illustration aux concepts utilisés dans le déroulement de l'exemple.

- Universel (Ø, Ø, Ø)

- Système (
{Nom_Concept (Nom), Type_Concept (Type), Coût (C), Masse (M), Risque (R)} ;
{D_{Nom} : {Longeron, Longeron_I, Longeron_Caisson}, D_{Type} : {ES, AS}, D_C : [0, +∞ [, D_M : [0, +∞[, D_R : [0, 1]} ;
Ø)

- Longeron (
{Nom, Type, C, M, R, Réservoir (Res), Charge (Ch), Accroche_Longeron (AC_Long), Hauteur_Longeron (H_Long), Longueur_Longeron (L_Long), Matériau (Mat), Section_principale (Sect), Epaisseur_bout (EB), Epaisseur_accroche (EA), Nb_Longeron (Nb_Long)} ;
{D_{Nom}, D_{Type}, D_C, D_M : [60, 300], D_R, D_{Res}: {O, N}, D_{Ch}: [8, 15], $D_{Ac\text{-}Long}$: {Carré, Rect}, $D_{H\text{-}Long}$: [10, 50], $D_{L\text{-}Long}$: [8, 10] , D_{Mat} : {Acier, Carbone}, D_{Sect} : {I, Tube, Caisson}, D_{EB} : [5, 20] , D_{EA} : [5, 20] , D_{Nb_Long} : [1..5]};
{C1, C2, C3 : Nom = Longeron})

- Longeron_I (
{Nom, Type, C, M, R, Res, Ch, Ac_Long, H_Long, L_Long, Matériau (Mat), Section_principale (Sect), Epaisseur_bout (EB), Epaisseur_accroche (EA), Nb_Longeron (Nb_Long)} ;
{D_{Nom}, D_{Type}, D_C, D_M : [60, 300], D_R, D_{Res}: {O, N}, D_{Ch}: [8, 15], $D_{Ac\text{-}Long}$: {Carré, Rect}, $D_{H\text{-}Long}$: [10, 50], $D_{L\text{-}Long}$: [8, 10], D_{Mat} : {Acier, Carbone}, D_{Sect}, D_{EB} : [5, 20] , D_{EA} : [5, 20] , D_{Nb_Long} : [1..3]};
{C1, C2, C3 : Nom = Longeron_I, C4 : Sect = I})

- Longeron_Caisson (
{Nom, Type, C, M, R, Res, Ch, Ac_Long, H_Long, L_Long, Matériau (Mat), Section_principale (Sect), Epaisseur_bout (EB), Epaisseur_accroche (EA), Nb_Longeron (Nb_Long), Taille_Reservoir (T_Res)} ;
{ D_{Nom}, D_{Type}, D_C, D_M : [60, 300], D_R, D_{Res}: {O, N}, D_{Ch}: [8, 15], $D_{Ac\text{-}Long}$: {Carré, Rect}, $D_{H\text{-}Long}$: [10, 50], $D_{L\text{-}Long}$: [8, 10], D_{Mat} : {Acier, Carbone}, D_{Sect}, D_{EB} : [5, 20] , D_{EA} : [5, 20] , D_{Nb_Long} : [1..5], D_{T_Res} : [0, 1000]};
{C1, C2, C3 : Nom = Longeron_Caisson, C4 : Sect = Caisson })

- Tâche (
{Durée (Du), Date début (Dd), Date fin (Df), Ingé_Calcul (IC), Ingé_Structure (IS), Ingé_Electronicien (IE), Coût (C), Risque (Risk)} ;
{D_{Du} : [5, 10[, D_{Dd} : [0, +∞[, D_{Dd_tt} : [0, +∞[, D_{Dd_td} : [0, +∞[, D_{IC} : {Expert, Junior, Aucun}, D_{IS} : {Expert, Junior, Aucun}, D_{IE} : {Expert, Junior, Aucun}, D_C : [0, +∞[, D_{Risk} : [0, 1]} ;
{C4 : Df ≥ Dd + Du})

Nous signalons que, par souci de lisibilité et de leur non implication dans les contraintes, nous ne mentionneront plus les variables Coût et Risque pour les concepts descendants de « Système », les systèmes ou les alternatives système.

Les contraintes C1, C2 et C3 sont :

C1 :

Res	Sect
O, N	Caisson
N	I, Tube, Caisson

C2 :

Sect	Nb_Long
I	[1, 3]
Tube	[2, 4]
Caisson	[1, 5]

C3 :

Res	T_Res
N	0
O	> 0

Deux contraintes de couplage eCC1 et eCC2 associent des variables de différents concepts :

eCC1 :

Type	Nom	IC	IS	IE
AS / ES	Longeron	Junior/ Expert	Junior/Expert	Aucun
AS / ES	Longeron_I	Junior/ Expert	Junior/Expert	Aucun
AS / ES	Longeron_Tube	Junior/ Expert	Expert	Aucun
AS / ES	Longeron_Caisson	Expert	Expert	Aucun

eCC2 :

Type	Mat	Ac_Long	Nb_Long	Res	Du
AS	Acier	Carré	[1, 5]	N	[5, 6]
AS	Acier	Carré	[1, 5]	O	[6, 7]
AS	Acier	Rectangulaire	[1, 5]	N	[6, 7]
AS	Acier	Rectangulaire	[1, 5]	O	[7, 8]
AS	Carbone	Carré	[1, 2]	N	[7, 8]
AS	Carbone	Carré	[1, 2]	O	[6, 7]
AS	Carbone	Carré	[3, 5]	N	[6, 7]
AS	Carbone	Carré	[3, 5]	O	[7, 8]
AS	Carbone	Rectangulaire	[1, 2]	N	[7, 8]
AS	Carbone	Rectangulaire	[1, 2]	O	[8, 9]
AS	Carbone	Rectangulaire	[3, 5]	N	[8, 9]
AS	Carbone	Rectangulaire	[3, 5]	O	[9, 10]
ES	∀	∀	∀	N	[1, 2]
ES	∀	∀	∀	O	[1, 3]

7.4.2 Définition d'un projet de conception et des exigences système associées

En premier lieu, un projet de conception est créé et le concept « Projet » lui est associé. Un système est alors automatiquement créé et le directeur de programme choisit de lui associer le concept « Longeron » (activité « Demander création Projet et Système » fig. 2.13). Conformément aux propositions de la section 5.3, les modèles de variables, de domaines et de contraintes sont copiés dans le système et le projet. De plus, similairement aux propositions de la section 3, les attributs de faisabilité et de vérification de ces deux entités sont, à leur création, non déterminés.

Le responsable de planification décide de propager le choix du concept « Longeron » dans son environnement de planification (fc0) : la contrainte eCC1 permet de réduire le domaine $D_{IE} = \{0\}$ de la tâche TE. Le responsable de planification décide alors d'ajouter quatre contraintes supplémentaires projet (C5, C6, C7 et C8) correspondant au coût de main d'œuvre pour chaque ressource en fonction de sa compétence, ce qui entraîne également l'ajout de trois variables supplémentaires projet (CMo_Calcul, CMo_Struct et CMo_Elec).

C5 : C = (CMo_Calcul + CMo_Struct + CMo_Elec) x Du

C6 :

CMo_Calcul	IC
0	Aucun
50	Junior
100	Expert

C7 :

CMo_Struct	IS
0	Aucun
60	Junior
120	Expert

C8 :

CMo_Elec	IE
0	Aucun
50	Junior
90	Expert

Il renseigne alors les variables projet :

- Dd : 01/05/2011 ;
- Df : 31/04/2012 ;
- Du : [8, 9] ;
- C ≤ 1600
- Risk ≤ 0,5 ;
- IC : Expert ;
- IS : Expert.

Puis il décide de propager ses choix à nouveau dans son propre environnement (fl1) afin de s'assurer de la cohérence de ces derniers : la contrainte C5 ne peut pas être respectée car, au mieux, le coût est supérieur à 1600 (Coût = (100 + 120) x 8 = 1760 > 1600). Il décide donc d'employer un ingénieur calcul junior (IC : Junior) et propage à nouveau sa décision dans son environnement (fl1) : le domaine de la variable Cout est réduit à l'intervalle [1360, 1530] et les choix du responsable de planification sont conformes avec l'ensemble des contraintes de son environnement. Il qualifie donc la tâche TE de *faisable* (TE.Fa = OK) et transmet le planning de la tâche TE au responsable de conception (message m40, fig. 2.13). Il décide aussi de propager ses choix vers l'environnement de conception (fc1).

Le responsable de conception, alerté de cette demande de propagation, propage les contraintes de la tâche TE dans son environnement (fl2) et constate qu'il n'y a pas d'incidence. Il décide également d'ajouter deux variables supplémentaires système (Prof_Aile et Long_Aile) ainsi que leurs domaines respectifs afin de recueillir les exigences système. Ensuite, il renseigne les variables système :

- Res : N
- Ch : 10 T
- M : [60, 120]kg
- Ac_Long : Carré
- H_Long : [10, 20] cm
- L_Long : [8, 8,5] m
- Prof_Aile : Profil A
- L_Aile : [8, 9]m

Puis il décide de propager ses choix dans son propre environnement afin de s'assurer de la cohérence de ces derniers : ses choix sont conformes avec l'ensemble des contraintes de son environnement, il peut donc qualifier les exigences système de *faisables* (ES.Fa = OK) (message m50, fig. 2.13). Il décide alors de propager ses choix vers l'environnement de planification (fc3) : les décisions du responsable de conception n'ont pas d'incidence sur les domaines des variables tâche, le responsable de planification peut donc qualifier la tâche TE de *vérifiée* (TE.Ve = OK). La figure 7.7 illustre l'ensemble des variables et domaines réduits à la fin du recueil des exigences système.

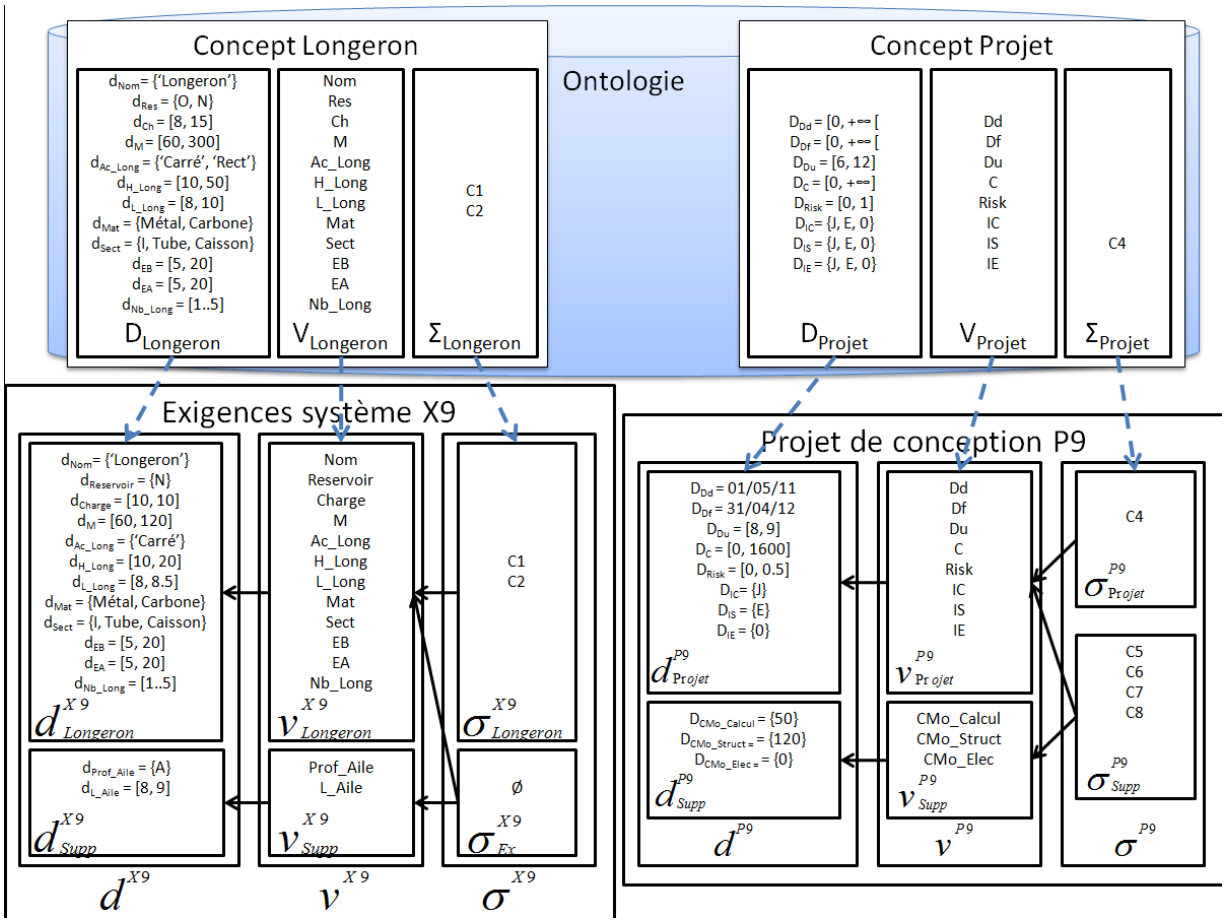


Figure 7.7 : Exigences système et projet de conception à la fin du recueil des exigences système

7.4.3 Définition des tâches de développement d'alternatives et des alternatives système associées

À l'issue de la définition des exigences système, il est décidé conjointement par les responsables de conception et de planification d'étudier deux alternatives. Deux tâches de développement d'alternative (TD) sont alors créées et le concept « Tâche » leur est associé. Deux alternatives système sont également créées et le responsable de conception choisit de leur assigner respectivement les concepts « Longeron_I » et « Longeron_Caisson ». Conformément aux propositions de la section 5, les modèles de variables, de modèles et de contraintes associés à chaque concept sont copiés dans l'entité correspondante. De plus, les variables supplémentaires ainsi que les domaines réduits au cours du recueil des exigences système sont également copiés. Nous rappelons également que l'ensemble des attributs de faisabilité et de vérification sont, à la création des entités, à l'état non déterminé.

7.4.3.1 *Illustration sur le développement d'un longeron en I*

Le responsable de planification décide de propager le choix du concept « Longeron_I » dans son environnement (fc4) : les domaines des variables tâche TD ne sont pas affectés.

Le responsable de planification renseigne alors les variables tâche TD :

- Dd : 01/07/2011 ;
- Df : 31/04/2012 ;
- Du : [5, 8] ;
- C ≤ 1050

Puis il décide de propager à nouveau ses choix dans son propre environnement (fl4) afin de s'assurer de la cohérence de ces derniers : la contrainte C5 ne peut être respectée que si la durée est

inférieure ou égale à 7, le domaine de la variable Du devient donc [5, 6], le domaine de la variable Cout est également réduit à l'intervalle [850, 1020] et les choix du responsable de planification sont conformes avec l'ensemble des contraintes de son environnement. Il qualifie donc la tâche TD de *faisable* (TD.Fa = OK) et transmet le planning de la tâche TD au responsable de conception (message m100, fig. 2.13).

Il décide alors de propager ses choix vers l'environnement de conception (fc5) : après acceptation du responsable de conception, en respectant la contrainte de couplage eCC2, le matériau du longeron ne peut être que l'acier (D_{Mat} = {Métal}), l'accroche doit être carrée (D_{Sect} = {Carré}) et il ne doit pas y avoir de réservoir (D_{Res} :{N}).

Le responsable de conception décide alors d'ajouter une variable supplémentaire alternative (Nb_Renforts) ainsi que son domaine ($D_{Nb\_Renforts}$ = [1..4]). Il peut alors renseigner les variables alternatives :

- Res : N
- Ch : 10 T
- M : 80 kg
- Ac_Long : Carré
- H_Long : 13 cm
- L_Long : 8,2 m
- Prof_Aile : Profil A
- L_Aile : 8,3 m
- Nb_Renforts : 2

Puis il décide de propager ses choix dans son propre environnement (fl5) afin de s'assurer de la cohérence de ces derniers : ses choix sont conformes avec l'ensemble des contraintes de son environnement, il peut donc qualifier l'alternative système de faisable (AS.Fa = OK). Il décide alors de propager ses choix vers l'environnement de planification (fc7) : les décisions du responsable de conception n'ont pas d'incidence sur les domaines des variables tâche. La figure 7.7 illustre l'ensemble des variables et domaines réduits à la fin du recueil des exigences système.

7.4.3.2 *Illustration sur le développement d'un longeron en caisson*

Le responsable de planification décide de propager le choix du concept « Longeron_Caisson » vers l'environnement de planification : la contrainte eCC1 impose pour ce concept l'emploi d'un ingénieur structure expert, or cette possibilité a été exclue lors de la définition du projet. Trois options sont alors possibles pour le responsable de planification :

- reconsidérer les exigences afin de trouver une configuration convenable, la faisabilité de la tâche TD reste non déterminée,
- abandonner la tâche de développement et donc l'alternative système associée, la faisabilité de la tâche TD passe à KO ou,
- passer outre cette alerte et poursuivre la planification et la conception sans faire appel à la propagation de contraintes, la faisabilité reste non déterminée.

7.5 Conclusion sur le couplage par contraintes

Nous avons proposé dans ce chapitre l'utilisation de contraintes de couplage entre les concepts associés à la conception et ceux associés à la planification. Ces contraintes sont utilisées grâce à un processus de propagation de contraintes permettant d'une part la propagation des contraintes propres à un environnement, et d'autre part, la propagation d'un environnement vers l'autre.

Ces contraintes de couplage, formalisant des connaissances métier entre la conception et la planification, ont pour but de répercuter les décisions dans un environnement sur l'autre. Ceci est une forme d'aide à la définition intégrée de la conception et de la planification.

Conclusion sur le couplage par connaissances métier

Dans les trois chapitres précédents, nous avons proposé un formalisme de représentation des **connaissances métier** et deux formes de couplage supplémentaires basées sur ces dernières :

- un **couplage par réutilisation de cas** qui permet de rechercher des solutions de planification et de conception pouvant satisfaire totalement ou partiellement les exigences exprimées puis, le cas échéant, les adapter, les réviser et les capitaliser dans la base de cas,
- un **couplage par contraintes** permettant de propager les décisions de la conception vers la planification et inversement durant la réalisation du processus intégré de planification / conception.

Ainsi, dans le chapitre 5, nous avons formalisé une ontologie permettant de fédérer dans une même hiérarchie une taxonomie de concepts dont les liens sont des relations de spécialisation/généralisation. Les concepts concernent aussi bien les systèmes que les projets et leurs tâches. Chaque concept contient, outre les classiques définitions/sémantiques propres au concept, différents modèles (variables, domaines, contraintes). Ceci permet de formaliser de la connaissance propre à chaque concept. Il s'agit donc d'une connaissance métier que chaque entreprise doit pouvoir formaliser. L'intérêt majeur de la proposition est double : formaliser ces connaissances dans le but de les exploiter ensuite dans deux formes de couplage (par réutilisation de cas et par contraintes) et de permettre la diffusion et l'acceptation de cette connaissance dans l'entreprise.

Dans le chapitre 6, nous avons proposé le couplage par réutilisation de cas. Celui-ci consiste à rechercher, à partir de l'expression des exigences, des cas préalablement conçus contenant des entités de planification/conception couplées structurellement afin de les adapter, les réviser et les capitaliser. Il s'agit donc d'une forme de processus de raisonnement à partir de cas, mais adapté au couplage planification/conception. En effet, lorsqu'une solution est identifiée comme étant compatible par rapport aux exigences, la tâche de développement de l'alternative système correspondante est d'abord adaptée puis révisée et ensuite, l'alternative système elle-même est adaptée et révisée. L'ensemble couplé structurellement est ensuite capitalisé. Afin de ne pas écarter des cas proches de ce qui est recherché mais ne répondant que partiellement aux nouvelles exigences, nous avons intégré une certaine flexibilité dans la recherche avec la possibilité, pour le concepteur, soit d'utiliser des fonctions de similarités, soit des préférences. Dans ce processus, la connaissance capitalisée dans l'ontologie est exploitée. La contribution majeure de ce chapitre concerne l'intégration du processus intégré de planification/conception dans un processus de raisonnement à partir de cas, ce que nous appelons couplage par réutilisation de cas ainsi que la possibilité de formaliser les préférences de conception dans la recherche d'alternatives.

Enfin, dans le chapitre 7, nous avons proposé le couplage par contraintes. Il s'agit de formaliser des connaissances de couplage à l'aide de contraintes liant la conception et la planification. La réalisation de choix dans un environnement est propagée dans l'autre environnement grâce à un mécanisme de filtrage. Ainsi, le couplage par contraintes permet, en choisissant un concept ou en fixant des valeurs à des variables (ou en réduisant leur domaine) dans un environnement et en appliquant une méthode de filtrage, de réduire l'espace des possibles dans l'autre environnement. Les choix de planification réduisent ainsi les choix possibles en conception et inversement, des choix faits en conception limitent les possibilités en planification. Une fois cette connaissance métier capitalisée sous forme de contraintes, elle peut être systématiquement utilisée à chaque fois que des concepts identiques seront utilisés. La contribution majeure de ce chapitre concerne l'intégration d'un processus de filtrage de contraintes dans le processus intégré de planification/conception.

Ce couplage par les connaissances métier permet de bénéficier de la connaissance capitalisée à chaque projet de conception. Les responsables de conception et de planification disposent ainsi, dès le début du projet, de connaissances spécifiques sur le projet à planifier et sur le système à concevoir. Cette connaissance n'est pas figée, c'est-à-dire qu'au fur et à mesure des projets de conception réalisés ou des apports des experts, de nouvelles connaissances viennent compléter, rectifier ou supprimer des connaissances plus anciennes.

Conclusion générale

Conclusion générale

Les travaux de recherche présentés dans ce mémoire ont porté sur le couplage de la planification de projet de conception avec la conception de systèmes. Apres avoir présenté un état de l'art sur les deux domaines et leur couplage, puis formalisé notre problématique, nos propositions ont été élaborées selon deux axes complémentaires. Le premier axe, développé dans la première partie (chapitres 2, 3 et 4), est basé sur la formalisation et l'exploitation de connaissances méthodologiques, c'est-à-dire des connaissances basées sur les informations et les processus d'obtention de ces informations pour le couplage. Le second axe, développé dans la seconde partie (chapitres 5, 6 et 7), est basé sur des connaissances métier, c'est-à-dire les connaissances qui, une fois formalisées et capitalisées, peuvent aider au pilotage de projets de planification et développement pour des métiers particuliers.

Dans la partie relative au couplage basé sur des connaissances méthodologiques, nous avons proposé en premier lieu une forme de couplage structurel. Ce couplage permet de garantir que chaque système à concevoir est cadré par un projet de conception propre mettant en œuvre des tâches pour le recueil des exigences et d'autres pour le développement de solutions. Le modèle proposé permet de fédérer les informations inhérentes à chaque domaine et de maintenir en permanence leur association. Ainsi, nous avons formalisé des associations entre : un projet de développement d'un système et ce système, une tâche de recueil des exigences systèmes et ces exigences et enfin, entre une tâche de développement d'une solution (appelée alternative système) et cette solution elle-même. Ce modèle est récursif dans la mesure où un système peut être décomposé en sous-systèmes et le projet associé, en sous projets, en maintenant toujours la bijection entre entités des deux domaines (un projet / un système). D'autre part, nous avons proposé un processus intégré de planification et de conception orchestrant la création des entités de conception et de planification puis leur renseignement et enfin, le suivi de leur élaboration. Le processus proposé permet de garantir que les informations sur les entités couplées sont renseignées au moment opportun et de manière cohérente en permettant aux décideurs de réaliser des choix synchronisés. Les synchronisations ont été possibles grâce à la définition d'indicateurs de faisabilité et de vérification propres aux entités et dont le changement d'état doit être contrôlé. La synchronisation dans ces changements d'état a été nommée « couplage informationnel ». Après avoir établi des graphes représentants les états possibles des entités au regard des indicateurs de faisabilité et de vérification, nous avons formalisé une forme de connaissance méthodologique à l'aide de règles. Enfin, nous avons proposé une dernière forme de couplage dans laquelle des tableaux de bord permettent de fédérer des informations pouvant aider les décideurs à faire des choix lorsque plusieurs alternatives sont possibles.

Ensuite, nous avons développé la partie concernant le couplage exploitant les connaissances métiers. Dans un premier temps, nous avons formalisé une partie de cette connaissance grâce à une ontologie de concepts. L'ontologie proposée permet de regrouper des connaissances sur les artefacts à concevoir, que ce soit en planification ou en conception. Le principe est de regrouper des concepts propres aux systèmes et à la planification au sein d'une taxonomie hiérarchisée dont les liens entre concepts sont définis par des relations de généralisation/spécialisation. Chaque concept fédère de la connaissance sur la sémantique du concept lui-même (définition, variables caractérisant le concept, domaines de définition, etc.) et sur les contraintes propres au concept (contraintes sur les variables). Cette connaissance permet de guider les planificateurs et concepteurs en leur imposant de renseigner des informations sur des variables et en imposant des limites sur les valeurs acceptables pour ces variables. À partir de cette formalisation, nous avons développé un mécanisme de recherche de cas compatibles avec les exigences permettant la réutilisation d'acquis de planification et de conception. Un processus de planification/conception, intégré au processus décrit dans la partie sur les

connaissances méthodologiques et basé sur des principes de Raisonnement à Partir de Cas a été proposé. Le principe consiste, à partir des exigences systèmes, à rechercher, dans une base de cas capitalisant toutes les expériences passées de planification/conception, celles qui sont les plus compatibles afin de les adapter manuellement puis de les capitaliser et accélérer ainsi le processus. Une forme de flexibilité dans la définition des exigences ainsi que dans la recherche a été proposée afin de ne pas écarter trop rapidement des cas proches mais ne répondant pas totalement aux nouvelles exigences. Une mesure de compatibilité caractérisant la proximité entre ce qui est recherché et ce qui existe dans la base de cas a été proposée. Finalement, nous avons proposé de formaliser et d'exploiter des connaissances de couplage grâce au formalisme des problèmes de satisfaction de contraintes (CSP). Le principe est de formaliser des contraintes entre concepts et/ou variables des deux domaines permettant de réduire les valeurs autorisées. Un choix réalisé dans l'un des deux domaines permet, après la mise en œuvre d'un mécanisme de filtrage, de réduire l'espace des possibles dans l'autre domaine. Le mécanisme a été également intégré au processus intégré de la première partie.

D'un point de vue industriel, l'enquête que nous avons menée auprès d'acteurs industriels du pôle de compétitivité Aerospace Valley a permis d'aider à la formalisation de la problématique de nos travaux. En effet, les entretiens ont permis de mettre en lumière un besoin de suivi simultané des entités de conception et de planification de même que le besoin d'outils adaptés pour garantir la cohérence des décisions, faire des choix concertés au plus tôt dans le processus de développement et éviter des erreurs et des remises en causes trop fréquentes. Ceci nous a guidés afin de formaliser le couplage informationnel. Nous avons également identifié un besoin concernant la constitution d'un environnement décisionnel de type tableau de bord. Enfin, les résultats présentés dans ce mémoire ont contribué fortement aux spécifications pour le développement du démonstrateur ATLAS. Par ailleurs, les exemples présentés tout au long des chapitres ont servi à la mise au point du prototype ATLAS dont le fonctionnement est illustré dans l'annexe 4.

Les perspectives de ce travail peuvent être déclinées selon les axes académique et industriel. D'un point de vue académique, le choix argumenté d'une bijection entre les entités de conception et de planification nous a permis de proposer des modèles et processus relativement simples et fonctionnels. Ce choix est parfaitement réaliste aux niveaux les plus hauts de la structure projet/système, notamment pour des systèmes complexes et dont la conception et/ou la réalisation est distribuée. Lorsqu'on atteint des niveaux plus bas, ce choix peut s'avérer contraignant en imposant systématiquement d'associer un projet à un système, même si ce dernier est très simple. Il est donc tout à fait envisageable, pour de tels systèmes, que les concepteurs et planificateurs souhaitent s'affranchir de cette contrainte pour les niveaux de conception inférieurs en associant, par exemple, un seul projet au développement d'une hiérarchie de systèmes/sous-systèmes simples. Dans ce cas, les modèles et les processus doivent être modifiés et adaptés. En outre, les modèles proposés pour les systèmes sont bien adaptés à des données statiques, en revanche, leur aspect dynamique n'est pas réellement pris en compte. Une autre perspective est donc d'intégrer dans ces modèles des données sur le comportement dynamique des systèmes tant au niveau de la définition des exigences que de la représentation des solutions. Il peut s'agir par exemple de données concernant des contraintes de temps réel, de durées de traitement d'informations ou encore de comportements mécaniques dynamiques.

Concernant l'exploitation des connaissances métiers par réutilisation des acquis, une perspective concerne l'intégration de mécanismes de recherche de cas plus élaborés. Pour la modélisation des exigences, nous avons limité notre approche à des contraintes portant sur des variables soit discrètes, soit continues. Une perspective est d'intégrer des contraintes mixtes et portant sur d'autres types de variables (temporelles par exemple). Les mécanismes de recherche de cas que nous avons proposé ne peuvent fonctionner que niveau après niveau et il n'est possible d'évaluer la

compatibilité entre exigences et solutions capitalisées que sur un seul niveau de la structure système. Un mécanisme de recherche plus élaboré tenant compte des sous-systèmes présents dans une solution et des informations les concernant est donc envisagé. Le but est de définir un mécanisme permettant de rechercher les informations susceptibles de satisfaire les exigences mais sans connaître a priori à quel niveau se situent ces informations ni même si elles s'y trouvent. Pour conclure sur les perspectives concernant la réutilisation d'acquis, l'intégration du Retour d'Expérience pourrait permettre de guider les choix des décideurs, par exemple, en fonction des risques identifiés par l'expérience sur un type donné de conception.

Concernant l'exploitation des connaissances métier par propagation de contraintes, nous avons limité notre proposition à des contraintes de compatibilité entre valeurs de variables et des mécanismes de filtrage qui n'interviennent que sur un seul niveau. Une première perspective concerne l'utilisation de contraintes d'activation, c'est-à-dire qu'une valeur précise d'une variable va activer de nouvelles variables. Ceci permettrait de disposer de connaissances plus complètes et optionnelles pour la définition des exigences et des solutions. Une seconde perspective est d'étendre la propagation des contraintes sur plusieurs niveaux de décomposition. En effet, dans le cadre d'une conception de type configuration de produits, une décision sur un système donné peut avoir un impact sur ses sous-systèmes et, par couplage, sur les sous-projets associés.

D'un point de vue industriel, une perspective est de développer, sur la base du démonstrateur développé dans le cadre du projet ATLAS, un outil industriel et commercialisable afin de, à terme, l'implanter dans un environnement industriel. Dans cette optique, il est nécessaire de réaliser des interfaces entre l'environnement intégré de conception/planification avec d'autres progiciels communément utilisés dans les entreprises. Par exemple, des interfaces avec des progiciels comme CATIA V5® utilisé en conception et MS Project® utilisé en planification sont nécessaires. Pour le couplage décisionnel, nous avons limité notre proposition à la constitution de tableaux de bord relativement simples impliquant peu d'entités. Dans le cas de la conception de systèmes complexes où les variantes et possibilités sont importantes, la combinatoire devient rapidement très importante, la constitution de tableaux de bord ne pourra pas être gérée telle qu'elle a été proposée. Il conviendra donc de s'intéresser à la manière de présélectionner les meilleures solutions à présenter au(x) décideur(s). Dans cette optique, une solution pourrait être l'utilisation de stratégies mixtes d'exploration / regroupement exploitant, d'une part, l'exploration des arborescences système et projet et, d'autre part, le regroupement de plusieurs solutions de caractéristiques proches.

Bibliographie

Bibliographie

Aamodt, A. et Plaza, E. (1994) Case-based reasoning: foundational issues, methodological variations, and system approaches, *Artificial intelligence Communications, IOS Press, vol.7.1*, pp. 39-59.

Abeille, J., Coudert, T., Vareilles, E., Geneste, L., Aldanondo, M. et Roux, T. (2010) Formalization of an integrated system/project design framework: first models and processes, Complex System Design and Management, Paris, 207-217.

Abeille, J., Goris, G., Vareilles, E., Roux, T., Aldanondo, M. et Coudert, T. (2009) Couplage de la conception de produit et de la planification de projet : Première analyse des pratiques industrielles, Congrés International de Génie Industriel CIGI09, Bagnères de Bigorre.

AFIS (2004) *Glossaire de base de l'Ingénierie Système*, AFIS.

AFITEP-AFNOR (2000) *Dictionnaire de management de projet*.

Alberts, L.K. (1993) *YMIR : an ontology for engineering design*, University of Twente.

Aldanondo, M., Vareilles, E., Baron, C., Lahmar, Y. et Geneste, L. (2007) Product Development and Product Management : toward a Constraint based Approach for Cooperation, International Conference on Industrial Engineering and Systems Management, (ESM 2007), Beijing, China.

Aldanondo, M., Vareilles, E., Djefel, M., Gaborit, P. et Abeille, J. (2009) Coupling product configuration and process planning with constraints, 13th IFAC Symposium on Information Control Problems on Manufacturing (INCOM'09), Moscow, Russia.

Al-Mubaid, H. et Nguyen, H.A. (2009) Measuring Semantic Similarity Between Biomedical Concepts Within Multiple Ontologies, *IEEE Trans. SMC-C*, vol. 39, no. 4, pp. 389-398, Available: Digital Object Identifier 10.1109/TSMCC.2009.2.

Antón, A.I. et Potts, C. (2003) Functional Paleontology: The Evolution of User-Visible System Services, *IEEE Trans. Software Eng.*, vol. 29, no. 2, pp. 151-166.

Avramenko, Y. et Kraslawski, A. (2006) Similarity concept for case-based design in process engineering, *Computers and Chemical Engineering*, vol. 30, pp. 548-557.

Barták, R., Salido, M. et Rossi, F. (2010) Constraint Satisfaction Techniques in Planning and Scheduling, *Journal of Intelligent Manufacturing*, vol. 21, no. 1, pp. 5-15.

Batet, M., Sanchez, D., Valls, A. et Gibert, K. (2010) Exploiting Taxonomical Knowledge to Compute Semantic Similarity: An Evaluation in the Biomedical Domain, *Lecture Notes in Computer Science*, vol. 6096/2010, pp. 274-283.

Bekhti, S. (2004) Mémoire de projet : Une approche de modélisation et de réutilisation du contexte et de la logique de conception, *Coopération et organisation numériques*, pp. 137-150, Available: DOI 10.3166/dn.8.1.137-150.

Beler, C. (2008) *Modélisation générique d'un retour d'expérience cognitive Application à la prévention des risques*. Thèse de doctorat. Institut National Polytechnique de Toulouse.

Benaissa, M. et Lebbah, Y. (2011) A Constraint Programming Based Approach to Detect Ontology Inconsistencies, *The International Arab Journal of Information Technology*, vol. 8, no. 1.

Bergmann, R. (2002) Experience Management: Foundations, Development methodology and, Internet-based Applications, *Lecture Notes in Artificial Intelligence*.

Blanc, S. (2006) *Contribution à la caractérisation et à l'évaluation de l'interopérabilité pour les entreprises collaboratives*. Thèse de doctorat. Ecole Doctorale des Sciences Physiques et de l'Ingénieur.

Bonjour, E. (2008) *Contributions à l'instrumentation du métier d'architecte système : de l'architecture modulaire du produit à l'organisation du système de conception*. Habilitation à Diriger des Recherches. Université de Franche-Comté, UFR Sciences et Techniques.

Brandt, S.C., Morbach, J., Miatidis, M., Theiben, M., Jarke, M. et Marquardt, W. (2008) An ontology based approach to knowledge management in design processes, *Computers and Chemical Engineering*, vol. 32, pp. 320-342.

Brown, D.C. et Chandrasekaran, B. (1985) Expert Systems for a Class of Mechanical Design Activity, in J.S.Gero (ed.) *Knowledge Engineering in Computer-Aided Design*, North Holland.

Chen, C.H., Ling, S.H. et Chen, W. (2003) Project scheduling for collaborative product development using DSM, *International Journal of Project Management*, vol. 21, no. 4, pp. 291-299.

China, M. (1997) Incidence de la réduction d'effectifs sur la perte de savoir-faire de l'entreprise, *Connaissance et savoir-faire en entreprise : intégration et capitalisation, coordination Fouet J. M*, pp. page 119-126.

Clermont, P., Beler, C., Rakoto, H., Desforges, X. et Geneste, L. (2007) Capitalisation et exploitation du retour d'expérience : un raisonnement à partir de cas étendu aux systèmes socio-économiques, in *Raisonnement à partir de cas, conception et configuration de produits*, Série Informatique et Systèmes d'Information edition, Hermès.

Cordì, V., Lombardi, P., Martelli, M. et Mascardi, V. (2005) An Ontology-Based Similarity between Sets of Concepts, *WOA 2005: Dagli Oggetti agli Agenti. 6th AI*IA/TABOO Joint Workshop "From Objects to Agents": Simulation and Formal Analysis of Complex Systems, 14-16 November 2005*, Available: ISBN 88-371-1590-3.

Cortes Robles, G., Negny, S. et Le Lann, J.M. (2009) Case Based Reasoning and TRIZ : a coupling for Innovative conception in Chemical Engineering, *Chemical Engineering and Processing*, vol. 48, no. 1, pp. 239-249, Available: ISSN 0255-2701.

Danilovic, T. et Browning, R. (2007) Managing complex product development projects with design structure matrices and domain mapping matrices, *International Journal of Project Management*, vol. 25, pp. 300-314.

Darlington, M.J. et Culley, S.J. (2008) Investigating ontology development for engineering design support, *Advanced Engineering Informatics*, vol. 22, pp. 112-134.

Dechter, R. (2003) *Constraint processing*, San Mateo.

Defense Acquisition University (2001) *Systems Engineering Fundamentals: January 2001*, Defense Acquisition University Press.

Dieter, G.E. (2000) *Engineering design : a materials and processing approach*, Boston: McGraw-Hill.

Djefel, M. (2010) *Couplage de la configuration de produit et de projet de réalisation : exploitation des approches par contraintes et des algorithmes évolutionnaires*. Thèse de doctorat. Institut National Polytechnique de Toulouse.

Electronic Industries Alliance (1999) *EIA-632 – Processes for Engineering a System*.

Eynard, B. (2005) *Gestion du cycle de vie des produits et dynamique des connaissances industrielles en conception intégrée*. Habilitation à Diriger des Recherches. Université Technologique de Troyes.

Fernandez, A. (2005) *L'essentiel du tableau de bord*, Editions d'organisation.

Finkelstein, A.C. et Finkelstein, L. (1995) Review of Design Methodology Design And Systems, in Publishers, T. (ed.) *in Design and Systems Praxiology: The International Annual of Practical Philosophy and Methodology*, New Brunswick (U.S.A) and Lono (U.K).

Finnie, G.R. et Sun, Z. (2003) R5 model for case-based reasoning, *Knowl.-Based Syst.*, vol. 16, no. 1, pp. 59-65.

Fixson, S.K. (2005) Product architecture assessment: a tool to link product, process, and supply chain design decisions, *Journal of Operations Management*, vol. 23, p. 345–369.

Froman, B. et Gourdon, C. (2003) *Dictionnaire de la qualité*, AFNOR.

Furtado, E., Furtado, V., Silva, W., Rodrigues, D., Taddeo, L., Limbourg, Q. et Vanderdonckt, J. (2001) An Ontology-Based Method for Universal Design of User Interfaces, *Proceedings of Workshop on Multiple User Interfaces over the Internet: Engineering and applications Trends*, p. Accessible in http://www.cs.concordia.ca/%7Efaculty/seffah/ihm2001/pr ogram.html.

Gandon, F. (2002) *Distributed Artificial Intelligence and Knowledge Management: ontologies and multi-agent systems for a corporate semantic web*. Scientific Philosopher Doctorate Thesis. Doctoral School of Sciences and Technologies of Information and Communication (S.T.I.C.).

Gelle, E. (1998) *On the generation of locally consistent solution spaces in mixed dynamic contraint problems*. Thèse de doctorat. Ecole Polytechnique Fédérale de Lausanne, Suisse.

Gero, J.S. (1990) Design prototypes : a knowledge representation schema for design, *AI Magazine*, vol. 11, no. 4, pp. 26-36.

Glover, F. et Laguna, M. (1993) *Tabu search*, Kluwer Academic Publisher.

Golumb, S. et Baumbert, I. (1965) Backtrack programming, *Journal of ACM*, vol. 12, no. 4, pp. 516-524.

Gómez de Silva Garza, A. et Maher, M.L. (1996) Design by interactive exploration using memory-based techniques, *Knowl.-Based Syst.*, vol. 9, no. 3, pp. 151-161.

Gonçalves-Coelho, A., Mourão, A. et Pereira, Z. (2005) Improving the use of QFD with Axiomatic Design, *Concurent Engineering: Research and Applications*, vol. 13, no. 3.

Gorschek, T. et Davis, A.M. (2008) Requirements engineering: In search of the dependant variables, *Information and Software Technology*, vol. 50, pp. 67-75.

Gourc, D. (2005) *Vers un modèle général du risque pour le pilotage et la consuite des activités de biens et de services : Propositions pour une conduite de projets et une gestion des risques intégrées*. Habilitation à Diriger des recherches. Institut National Polytechnique de Toulouse.

Gruber, T.R. (1993) A Translation Approach to Portable Ontology Specification, *Knowledge acquisition*, vol. 5, no. 2, pp. 199-220.

Haralick, R. et Elliot, G. (1980) Increasing tree search efficiency for constraint satisfaction problem, *Artificial Intelligence*, vol. 14, pp. 263-313.

Hatchuel, A. et Weil, B. (2002) La théorie C-K : Fondements et usages d'une théorie unifiée de la conception, Colloque "Science de la conception", Lyon.

Jiao, J., Zhang, L., Zhang, Y. et Pokharel, S. (2008) Association rule mining for product and process variety mapping, *International Journal of Computer Integrated Manufacturing*, vol. 21, no. 1, pp. 111-124.

Kamsu Foguem, B., Coudert, T., Béler, C. et Geneste, L. (2008) Knowledge formalization in experience feedback processes: An ontology-based approach, *Computers in Industry*, vol. 59, no. 7, pp. 694-710.

Kim, K.Y., Manley, D.G. et Yang, H. (2006) Ontology-based assembly design and information sharing for collaborative product development, *Computer-Aided Design*, vol. 38, p. 1233–1250.

Kirkpatrick, S., Gelatt, C. et Vecchi, M. (1984) Optimization by simulated annealing, *Science*, vol. 200, pp. 671-680.

Kolodner, J. (1993) *Case-Based Reasoning*, San Mateo, CA: Morgan Kaufman.

Kolodner, J. et Leake, D. (1996) A Tutorial Introduction to Case-Based Reasoning, *To appear in Case-Based Reasoning: Experiences, Lessons, and Future Directions*.

Lau, A., Tsui, E. et Lee, W.B. (2009) An ontology-based similarity measurement for problem-based case reasoning, *Expert Systems with Applications 36*, pp. 6574-6579.

Leake, D. et Powell, J. (2007) Mining Large-Scale Knowledge Sources for Case Adaptation Knowledge, *Lecture Notes in Computer Science, Volume 4626, Case-Based Reasoning Research and Development*, pp. 209-223.

Lee, K.J., Kim, H.W., Lee, J.K. et Kim, T.H. (1998) Case- and Constraint-Based Project Planning for Apartment Construction, *AI Magazine Volume 19 Number 1*.

Lee, J. et Lee, N. (2006) Least modification principle for case-based reasoning : a software project planning experience, *Expert Systems with Applications 30*, pp. 190-202.

Lee, C.H.L. et Liu, A. (2009) Designing Robot Services with Ontology and Learning, *Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics, San Antonio, TX, USA*.

Lhomme, O. (1993) Consistency techniques for numeric CSP, *International Joint Conference on Artificial Intelligence , Chambéry, France*, pp. 232-238.

Lindemann, U. (2007) A vision to overcome « chaotic » design for X processes in early phases, International Conference on Engineering Design (ICED'07), Paris.

Lu, R., Peng, W. et Wang, C. (2008) Integration of Product Design Process and Task Management for Product Data Management Systems, *Integration of Product Design Process and Task Management for Product Data Management Systems, IFIP International Federation for Information Processing, Volume 254, Research and Practical Issues of Enterprise Information Systems*, vol. 1, no. 2, pp. 207-218.

Maher, M.L. et Gómez de Silva Garza, A. (1997) Case-Based Reasoning in Design, *IEEE Expert*, vol. 12, no. 2, pp. 34-41.

Maheswari, J.A. et Varghese, K. (2005) Project Scheduling using Dependency Structure Matrix, *International Journal of Project Management*, vol. 23, pp. 223-230.

Mao, M., Peng, Y. et Spring, M. (2010) An adaptive ontology mapping approach with neural network based constraint satisfaction, *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 8, no. 1, pp. 14-25.

Matta, N., Ribière, M., Corby, O., Lewkowicz, M. et Zacklad, M. (2000) *Project Memory in Design, Industrial Knowledge Management - A Micro Level Approach*, SPRINGER-VERLAG : RAJKUMAR ROY.

Mittal, S. et Falkenhainer, B. (1990) Dynamic contraint satisfaction problems, *AAAI*, pp. 25-32.

Montanari, U. (1974) Networks of contraints: fundamental properties and application to picture processing, *Information sciences*, vol. 7, pp. 95-132.

Negny, S. et Le Lann, J.L. (2008) Case-based reasoning for chemical engineering design, *Chemical Engineering research and Design*.

Noy, N. et McGuiness, D.L. (2000) *Ontology Development 101: A guide to Creating your First Ontology*. URL : www.smi.stanford.edu/projects/protege:publications/ontology_development/ontology101.pdf: Stanford Medical Informatics Technical Report No. SMI-2001-0880.

Núñez, H., Sànchez-Marrè, M., Cortés, U., Comas, J., Martínez, M., Rodríguez-Roda, I. et Poch, M. (2004) A comparative study on the use of similarity measures in case-based reasoning to improve the classification of environmental system situations, *Environmental Modelling & Software*, vol. 19, p. 809–819.

Oosterman, B. (2001) *Improving Product Development Projects by Matching Product Architecture and Organization*. Thèse de doctorat. Université Groningen.

Pahl, G. et Beitz, W. (1996) *Engineering Design : a Systematic Approach*.

Pirró, G. (2009) A semantic similarity metric combining features and intrinsic information content, *Data & Knowledge Engineering*, vol. 68, p. 1289–1308.

Pitiot, P. (2009) *Amélioration des techniques d'optimisation combinatoire par utilisation d'un mécanisme de retour d'expérience : Application à la sélection de scénarios en conception préliminaire de produit / projet*. Thèse de doctorat. Institut National Polytechnique de Toulouse.

Pitiot, P., Coudert, T., Geneste, L. et Baron, C. (2010) Hybridation of Bayesian networks and Evolutionary Algorithms for multi-objective optimization in an integrated product design and project management context, *Engineering Applications of Artificial Intelligence (EAAI)*, vol. 23, no. 5, pp. 830-843.

Project Management Institute (2004) *A guide to Project management body of knowledge: PMBOK Guide*, 3rd edition.

Rakoto, H., Hermosillo Worley, J. et Ruet, M. (2002) Integration of experience based decision support in industrial processes, *International Conference on Systems, Man and Cybernetics, IEEE SMC 02, Bridging the Digital Divide, Cyber-development, Human Progress, Peace and Prosperity*, Available: ISBN : 2-9512309-4-X.

Ramesh, B., Cao, L. et Baskerville, R. (2010) Agile requirements engineering practices and challenges: an empirical study, *Info Systems Journal*, vol. 20, pp. 449-480.

Renaud, J., Chebel-Morello, B., Fuchs, B. et Lieber, J. (2007) *Raisonnement à partir de cas : conception et configuration de produits, Traité IC2, série Informatique et Systèmes d'information*, Lavoisier.

Richards, D. et Simoff, S.J. (2001) Design ontology in context – a situated cognition approach to conceptual modelling, *Artificial Intelligence in Engineering*, vol. 15, pp. 121-136.

Roche, C. (2000) Corporate ontologies and concurrent engineering, *Journal of Materials Processing Technology*, vol. 107.

Rochet, S. (2007) *Formalisation des processus de l'Ingénierie Système : Proposition d'une méthode d'adaptation des processus génériques à différents contextes d'application*. Thèse de doctorat. Institut National des Sciences Appliquées de Toulouse.

Rochet, S. et Baron, C. (2008) Towards a controlled specialization of system engineerning processes, Extended Product and Process Analysis and Design, Bordeaux.

Rousse, C. (2010) *Les tableaux de bord dans le suivi des projets d'innovation*. Mémoire de mastère spécialisé Audit interne & contrôle de gestion. ESC Toulouse.

Sharon, A., de Weck, O. et Dori, D. (2009) Is there a Complete Project Plan? A Model-Based Project Planning Approach, Nineteenth Annual International Symposium of the International Council on Systems Engineering, Singapore.

Sharon, A., Perelman, V. et Dori, D. (2008) A Project-Product Lifecycle Management approach for improved systems engineering practices, Eighteenth Annual International Symposium of the International Council on Systems Engineering (INCOSE), Utrecht, the Netherlands.

Simoff, S. et Maher, M.L. (1998) Designing with the activity/Space ontology, *Artificial Intelligence in Design*, pp. 23-44.

Sowa, J.F. (1984) *Conceptual Structures. Information Processing in Mind and Machine*, Reading, MA: Addison.

Sriram, D., Stephanopoulos, G., Logcher, R., Gossard, D., Groleau, N., Serrano, D. et Navinchandra, D. (1989) Knowledge-Based System Applications in Engineering Design: Research at MIT, *AI Magazine*, vol. 10, no. 3, pp. 79-96.

Steward, D. et Tate, D. (2000) Integration of axiomatic design and project planning, First International Conference on Axiomatic Desing (ICAD2000), Cambridge, USA.

Studer, R., Benjamins, R. et Fensel, D. (1998) Knowledge Engineering: Principles and methods, *Data & Knowledge Engineering*, vol. 25, pp. 191-197.

Suh, N.P. (1990) *The principles of design*, Oxford Press, Oxford University.

Suh, N.P. (1998) Axiomatic Design Theory for Systems, *Research in Engineering Design*, vol. 10, p. 189–209.

Tsang, E. (1993) *Fundation of constraints*, London: Accademic press.

Van Oudenhove de Saint Géry, T. (2006) *Contribution à l'élaboration d'un formalisme gérant la pertinence pour les problèmes d'aide à la conception à base de contraintes*. Thèse de doctorat. Institut National Polytechnique de Toulouse.

Vareilles, E. (2005) *Conception et approches par propagation de contraintes : contribution à la mise en oeuvre d'un outil d'aide interactif*. Thèse de doctorat. Institut National Polytechnique de Toulouse.

Vareilles, E., Aldanondo, M., Gaborit, P., Auriol, G., Rochet, S. et Baron, B. (2008) A Prospective Proposal for Coupling Product and Project Configuration with Constraints, International Conference on Software, Knowledge, Information Management and Applications (SKIMA 2008), Kathmandu, Nepal.

Vermaas, P.E. (2007) On the conceptual framework of John Gero's FBS-model and the perspective aims of design methodology, *Design Studies*, vol. 28, pp. 133-157.

Wriggers, P., Siplivaya, M., Joukova, I. et Slivin, R. (2007) Intelligent support of engineering analysis using ontology and case-based reasoning, *Engineering Applications of Artificial Intelligence*, vol. 20, p. 709–720.

Wu, Z. et Palmer, M. (1994) Verb Semantics and Lexical Selection, Proceedings of the 32nd Annual Meetings of the Associations for Computational Linguistics, 133-138.

Yang, D., Dong, M. et Miao, R. (2008) Development of a product configuration system with an ontology-based approach, *Computer-Aided Design*, vol. 40, pp. 863-878.

Yan-hong, Q. et Guang-xin, W. (2009) Product Configuration Based on CBR and CSP, 2009 International Conference on Measuring Technology and Mechatronics Automation, Zhangjiajie, Hunan.

Yannou, B., Mazur, C. et Yvars, P.-A. (2010) *Parameterization and dimensioning of the multi-disc clutch int the CO4 environment, Technical report, Laboratoire Génie Industriel, Ecole Centrale Paris, September 2010. (Cahiers de recherche 2010-21)*. Technical report. Laboratoire Génie Industriel, Ecole Centrale Paris (Cahiers de recherche 2010-21).

Zadeh, L.A. (1965) Fuzzy Sets, *Information and Control*, no. 8, pp. 338-353.

Zhang, J., Zhang, S., Chen, C. et Wang, B. (2003) Semantic Interoperability Based on Ontology Mapping in Distributed Collaborative Design Environment, *ADVANCES IN WEB INTELLIGENCE, Lecture Notes in Computer Science*, vol. 2663, pp. 208-217.

Annexes

Annexe 1 : Questionnaire, fiche d'entretien et tableau synthétique de l'enquête auprès d'acteurs industriels

Cette annexe présente le questionnaire type ainsi que la fiche d'entretien ayant servi de trame à la direction des entretiens puis un tableau synthétique de l'analyse des entretiens.

Questionnaire type

Ce document est juste une aide sur les thèmes qu'il est nécessaire d'aborder.

L'entretien est enregistré (ou non selon le souhait de l'industriel) afin de pouvoir travailler dessus sans l'interviewé ; il est donc important qu'il soit le plus exhaustif possible tout en restant dans le cadre du couplage conception et planification. Il faut récupérer toutes les informations nécessaires à la construction des modèles. Les points suivants peuvent être abordés dans n'importe quel ordre et sans séparation distincte.

Date : **Poste de l'interviewé :**
Nom de l'interviewé :

Société de l'interviewé :
Son métier :

- Qu'est ce qu'il fait ?
  - En quoi consiste son travail pour coupler conception et planification ? (description complète)
  - Quelles sont ses activités en rapport avec le couplage ?
  - Quels sont les inputs et les outputs ?
  - Quelles connaissances pense-t-il utiliser ? (savoir, savoir-faire, savoir-être)
- Cadre de son activité :
  - Qui fait quoi ?
  - Qui sait quoi ?
  - Qui décide, informe, et/ou agit ? avec quoi ?
- Comment fait-il ?
  - Qu'est ce qu'il utilise ? (ressources techniques, humaines, informationnelles et connaissances)
  - Quelles méthodes utilise-t-il ? (pas les normes ou les processus mais ceux que lui fait vraiment)
  - Liste complète des actions à effectuer ? Dans quel ordre (séquentiellement, en parallèle) ? Par qui ? Avec quoi ? Pendant combien de temps ?
- Pourquoi ?
  - Dans quels buts ?
  - Qu'est ce qu'il cherche à obtenir ? à éviter ? à maîtriser ? à inhiber ? …
  - Dans quels contextes ?
  - Quels sont les causes de telle ou telle décision ?
  - Quels paramètres sont à prendre en compte pour choisir ?

Fiche d'entretien

Présentation du projet :

- Projet ATLAS : Aides et assisTances pour la conception, la conduite et leur coupLage par les connAissanceS.
- But : créer un démonstrateur d'**aide** à la conception et à la planification, on ne veut ni remplacer ni supprimer les opérateurs.
- Etude de l'interaction entre l'objet de la conception (le produit) et le processus de réalisation (le projet) ainsi que son optimisation.
- Démonstrateur basé sur la connaissance des opérateurs récupérée par les entretiens.

Présentation du déroulement des entretiens :

- Premier entretien enregistré, l'opérateur parle sur ce qu'il fait, on écoute et on ne pose que quelques questions.
- Ensuite on va retranscrire les propos de l'opérateur et « mettre » ses connaissances dans différents modèles.
- Puis nous allons revenir pour que l'opérateur puisse modifier les modèles et nous aider à combler les manques.
- Enfin on poursuit les entretiens de validation jusqu'à ce qu'on obtienne un résultat satisfaisant.

Présentation du but des entretiens :

- Récupérer la connaissance des opérateurs sur les phénomènes de couplage entre la conception d'un projet et son management.
- Nous avons déjà des idées sur la façon de coupler la conception et la planification, mais nous voulons savoir comment font les opérateurs afin de répondre réellement à leurs besoins.
- Nous voulons récupérer **ce que font les opérateurs** pour accorder la conception et la planification.
- Les normes ou processus défini par l'entreprise ou autres ne nous intéressent pas.

Présentation du rôle de l'interviewer :

- Nous ne sommes là ni pour juger ni pour auditer.
- Ce que nous dira l'opérateur restera confidentiel.
- L'opérateur possède un droit de regard et de censure.
- Précisions sur l'entretien : Qu'est ce qu'il fait ? Comment ? **Pourquoi ?** (contexte)

Nous voulons qu'il nous explique comment il fait pour **coupler la conception et la planification**.

Il s'agit d'un entretien non guidé, on doit laisser l'opérateur parler et l'interrompre le moins possible.

Il faut l'inciter à parler, le recadrer si nécessaire mais surtout pas l'orienter pour faire correspondre ses connaissances à celles attendues.

Tableau synthétique des entretiens

Entretien	Activité	Conception produit	Planification projet	Méthodes externes	Méthodes internes	Cycle de développement
1	Equipementier-Maintenance Aero	CAO CATIA v5 - Simul AMESIM - Calculs COSMOS	Plan de MP	EIA 632, TPS Lean manufacturing	Rex	Cycle en V
2	Equipementier-Maintenance Aero	SMARTEAM, CATIA v5	GESTPROJ, PQP	EIA 632		Cycle en V
3	Industrie ferroviaire	CATIA v5	Plan de MP	AMDEC, INCOSE	Livres de savoir-faire, Rex, Normes INCOSE	Cycle en V
4	Industrie électronique			ISO, SE, CMM, CMMI, IEEE, JEDEC, 6 sigma	IP Pipeline, MRD/PRD, Rex feuilles Excel, Quality for design, Quality driven-technology, Best practices methodologies	Cycle en V
5	Industrie aéronautique	CAO CATIA v5	Plan de MP, MS Project, PS Next, DMU Milestones	EIA 632, Lean manufacturing, IS INCOSE	Ingénierie éxigences CARE	Cycle en V=IVVQ
6	Industrie aéronautique	SAP BWS, PDP, Démarche MRPII, ANSYS, ZEBULON	Plani budgétaire Hypérion, ARTEMIS, Dev Projet générique	EIA 632, Lean manufacturing, Qualité PDCA, 5S, AMDEC	Action V, Amélioration continue	Cycle en V
7	Industrie spatiale	CATIA v5, Mentor	Portail collaboratif Intranet, SPS	CMMI, ECSS, INCOSE	CMMI3, Méthodes propriétaires	Cycle en V
8	Industrie systèmes embarqués	SIMULINK, State Meth, SAP	MS Project		CMMI3	Cycle en V
9	Industrie et services informatiques	WP Codes sources, ERP SAP, SGBD DB2	Référentiel PRINCE2, PMP PMI, IBM Cognos8, Outils Rational	PMI(OBI/WBS), UMML, CMMI, ITIL, COBIT	RUP Méthodes agiles	Cycle en Y
10	SSII	WP Codes sources		UML, ITIL, COBIT		Cycle en V
11	SSII	Java, J2EE, SAP, WP Codes sources	NQI Orchestra	TMM, PRAXIS, 6 sigma, PRAXEME, eSCM, BPM	CMMI 3, Process UP	Cycle en Y
12	SSII	Java, J2EE, SAP, WP Codes sources	Plan de MP	UML, CMMI, COBIT, ITIL		Cycle en V ou en cascade
13	Ingénierie et services SIG	UML Java, OpenGL	Plan de MP, MS Project	ECSS, Echelle Maturité TRL		Cycle en V

Annexe 2 : Formalisme BPMN

La notation BPMN (Business Process Modelling Notation) est une notation graphique permettant d'exprimer des processus d'entreprise. L'objectif de BPMN est de fournir un cadre permettant de décrire un processus d'une manière commune à tous les utilisateurs. La norme BPMN définit deux concepts pour organiser les processus :

- les orchestrations,
- les chorégraphies.

Les orchestrations sont internes, elles définissent ce qui se passe à l'intérieur d'un groupement. Les chorégraphies sont interprocessus, elles définissent les communications entre groupements (entre processus). Les échanges internes à un processus BPMN sont effectués au travers d'enchaînements d'activités. Ils signifient qu'un enchainement entre les différentes tâches du processus est effectué. Les échanges entre les processus sont eux représentés par des messages et répondent au nom de chorégraphie.

Dans la suite de cette annexe, nous décrivons une à une les différents items utilisés pour la formalisation des processus de ce mémoire.

Item **Evènement** : un évènement est quelque chose d'extérieur au processus qui se produit. Ces évènements peuvent affecter le déroulement du processus et ont généralement une cause ou un impact. Nous avons utilisé deux types d'évènements : les évènements Début et les évènements Fin représentés ci-dessous.

Figure A2.1 : Représentation des évènements début et fin

Item **Activité** : une activité est un terme générique pour un travail à réaliser. Une activité peut être atomique (non décomposée) ou non atomique (l'activité peut être affinée en un sous-processus).

Figure A2.2 : Représentation des activités atomiques et non atomiques

Item **Porte** : les portes sont utilisées pour contrôler la divergence et la convergence dans le déroulement du processus. Elles représentent les choix pouvant être faits durant le processus ou les différents chemins à prendre simultanément. Nous avons utilisé trois représentations de portes : les portes ET (représentée à gauche) dans le cas de chemins à réaliser en parallèle et des portes OU EXCLUSIF (les deux représentations de droite sont équivalentes) permettant de représenter des choix.

Figure A2.3 : Représentation des portes ET et OU EXCLUSIF

Item **Flux séquentiel** : Cet item est utilisé pour représenter l'ordre dans lequel les activités s'enchainent.

Figure A2.4 : Représentation d'un flux séquentiel

Item **Flux informationnel** : Cet item est utilisé pour représenter le flux de messages entre deux processus. Dans la situation illustrée dans la figure A2.5, l'activité attend l'arrivée du message pour se poursuivre.

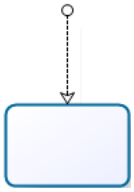


Figure A2.5 : Représentation d'un flux informationnel

Item **Bassin** : Cet item représente un participant du processus. L'ensemble des activités que le participant doit accomplir est représentée à l'intérieur du bassin.

Figure A2.6 : Représentation d'un bassin

Item **Ligne** : Une ligne est une sous partition d'un bassin. Elle a principalement pour but d'organiser et de classer les activités.

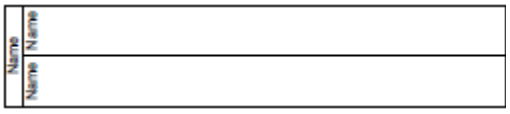


Figure A2.7 : Représentation d'une ligne

Item **Données Objet** : Les données objet sont considérées comme des artefacts dans le mesure où elles n'ont pas d'impact direct sur le processus. Cependant, elles contiennent des informations sur ce que les activités ont besoin pour être réalisées et/ou ce qu'elles produisent.

Figure A2.8 : Représentation d'une donnée objet

Annexe 3 : Processus

Cette annexe reprend les différents processus précédemment proposés dans ce mémoire.

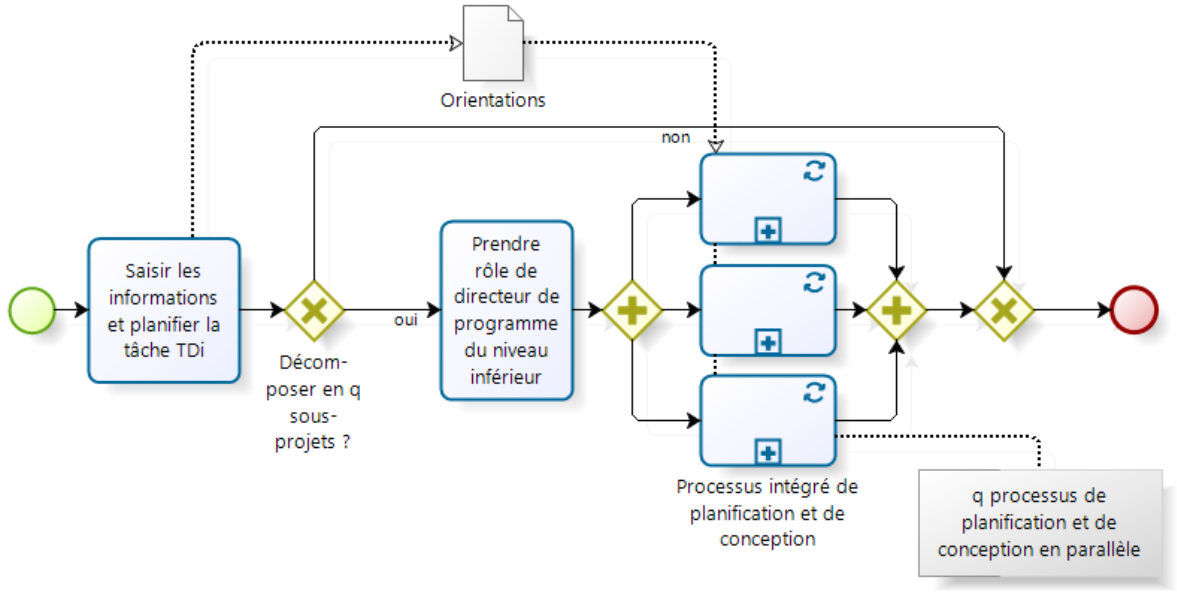


Figure A3.1 : Processus de saisie des informations / Planification tâche TD

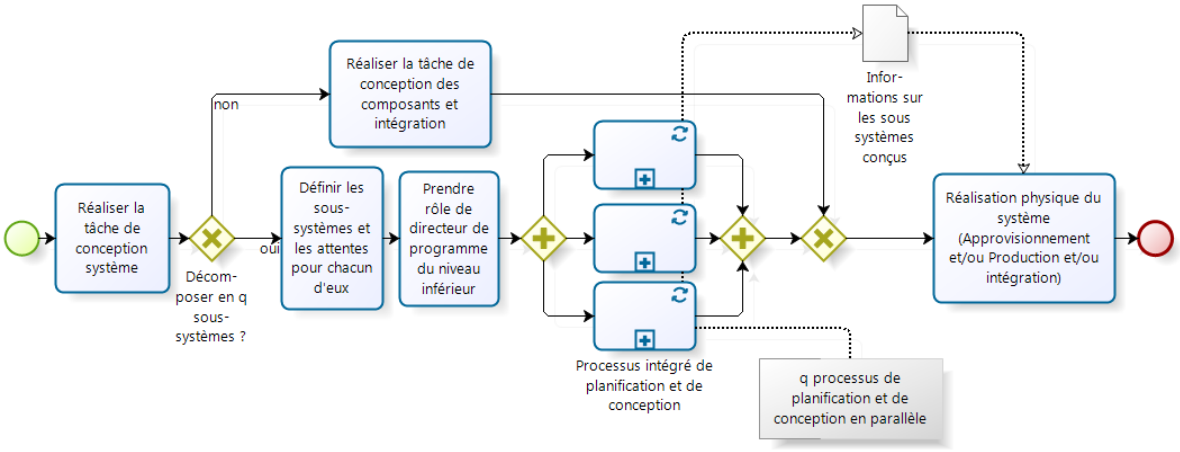


Figure A3.2 : Processus de réalisation d'une tâche TD

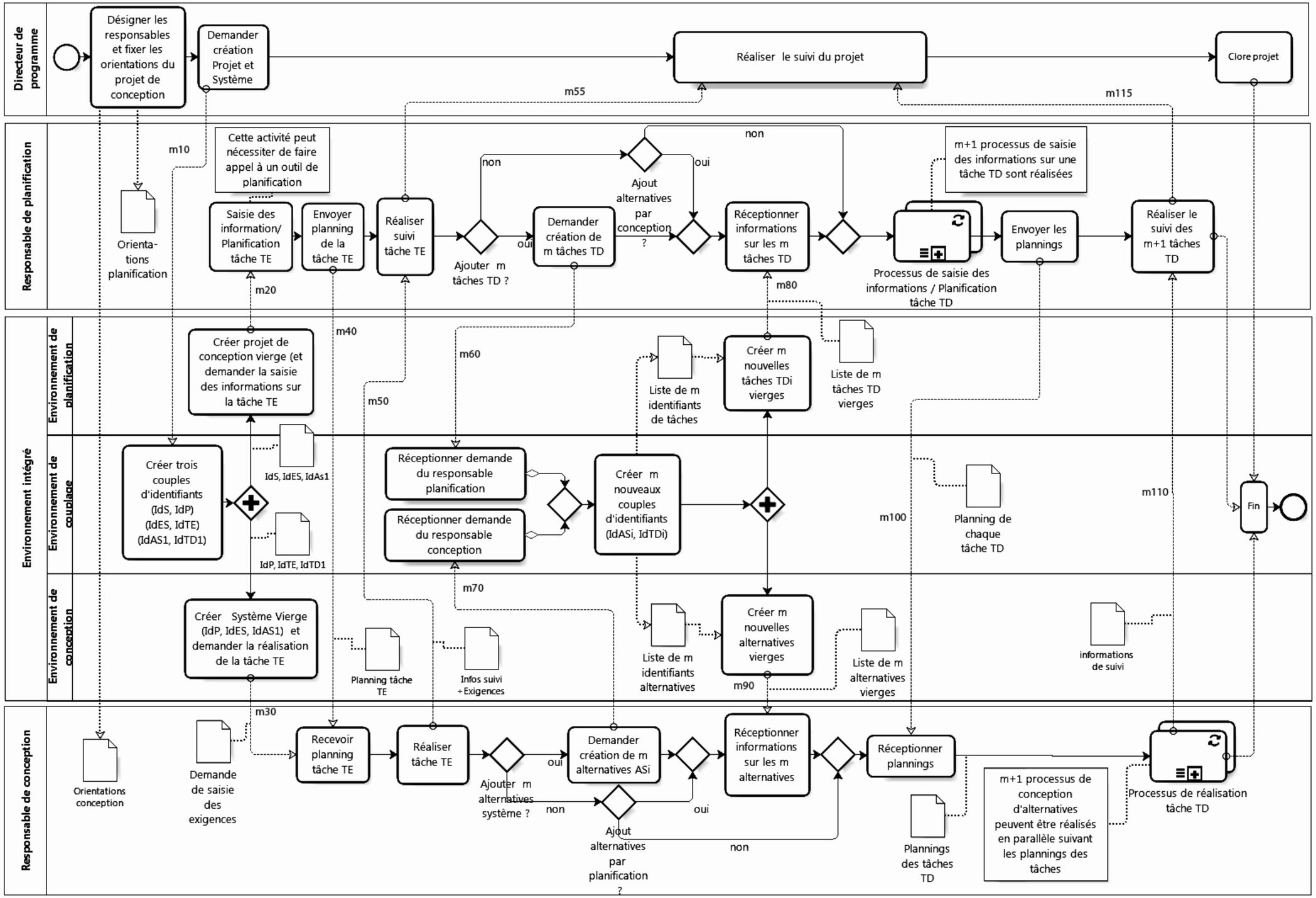


Figure A3.3 : Processus intégré de planification et de conception

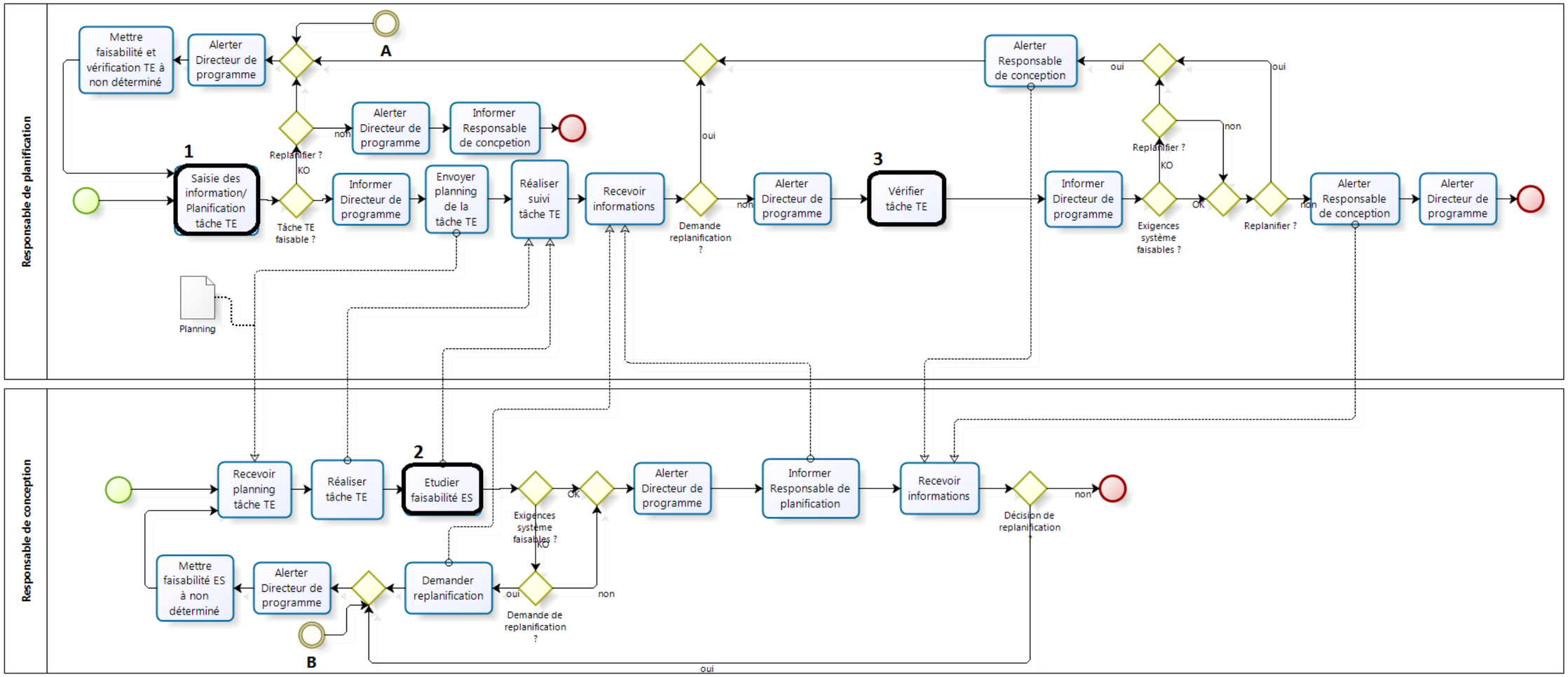


Figure A3.4 : Processus intégré de planification et de conception partiel permettant le changement des attributs des exigences système ES et de la tâche TE

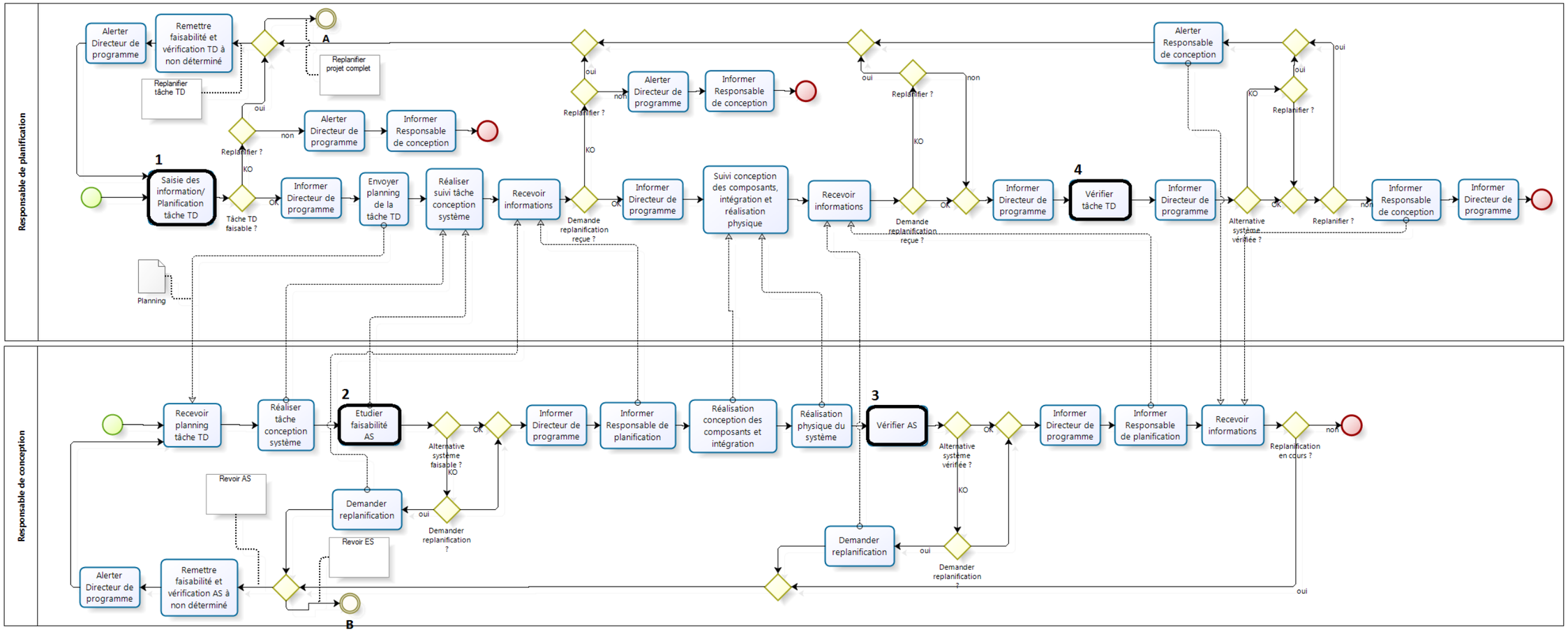


Figure A3.5 : Processus intégré de planification et de conception partiel permettant le changement des attributs d'une alternative système AS et de la tâche TD

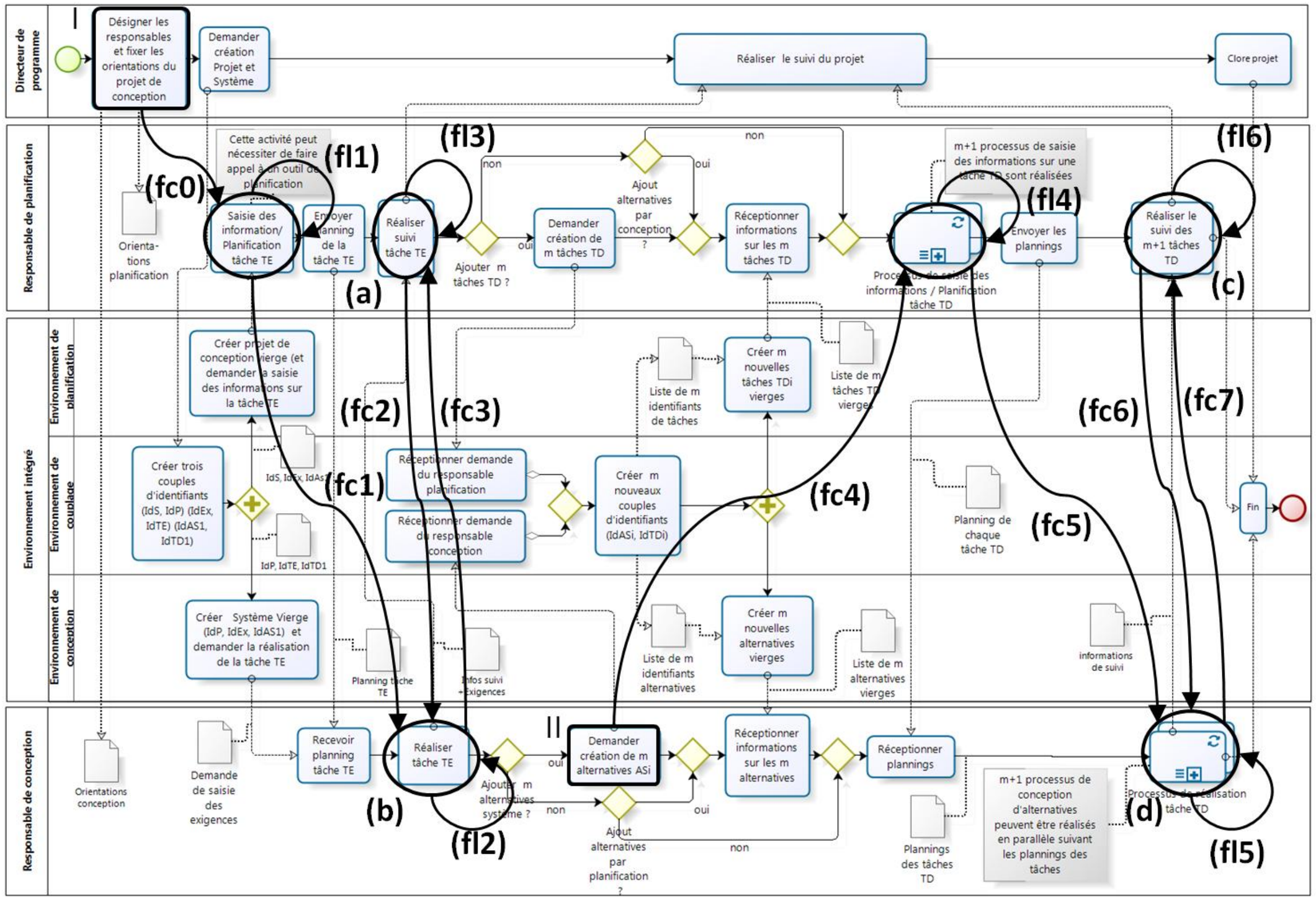


Figure A3.6 : Positionnement du processus de filtrage dans le processus intégré de planification et de conception

Annexe 4 : Illustration du fonctionnement du prototype ATLAS

Cette annexe présente quelques fonctionnalités réalisables par le prototype logiciel développé dans le cadre du projet ATLAS par le CRC-IDCE de l'ENI de Tarbes.

La figure A4.1 illustre l'ontologie de concepts proposée dans le chapitre 5. Il s'agit d'une taxonomie liant les concepts entre eux par des liens de spécialisation/généralisation. En effet, les concepts « Avion », « Structure » et « Longeron » sont au même niveau dans l'ontologie. En revanche, les concepts « Longeron_I » et « Longeron_Caisson » sont des spécialisations du concept « Longeron ».

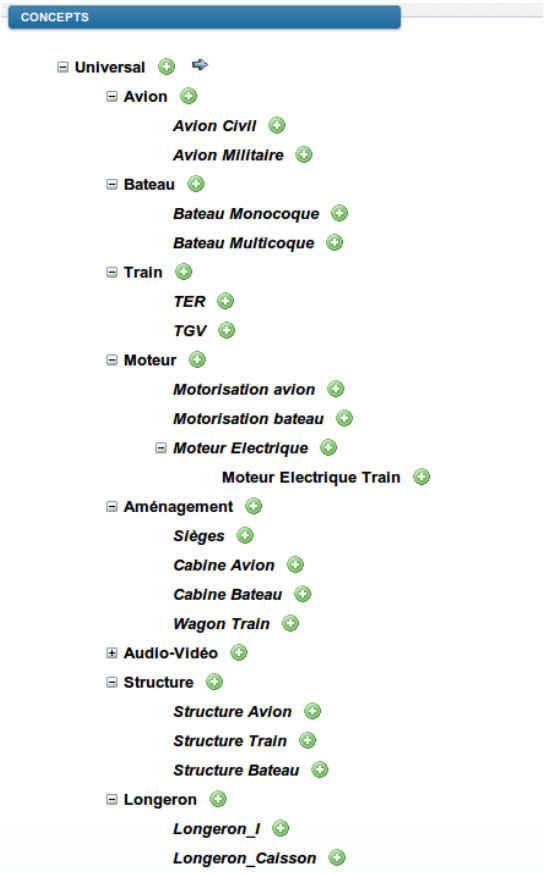


Figure A4.1 : Structure générale de l'ontologie

En cliquant sur l'icône ⊕ à droite du nom du concept, il est possible d'éditer ses caractéristiques. Les icônes ⊞ et ⊟ à gauche d'un concept permettent de déployer ou replier la hiérarchie. La figure A4.2 illustre un concept particulier de l'ontologie : le concept « Longeron ». Ce concept, outre son nom, comprend des modèles de variables. Ces dernières sont elles-mêmes composées d'un nom, d'un identifiant CoFiADe[1] (utilisé pour la propagation des contraintes), d'une unité de mesure et d'un domaine de définition pouvant être symbolique ou numérique. Ici, le concept « Longeron » comprend 15 modèles de variables.

[1] CoFiADe est un outil de filtrage de contraintes développé par le laboratoire CGI de l'Ecole de Mines d'Albi-Carmaux

Cofiade	Nom	Unité	Domaine
accroche_longeron	Accroche Longeron	type	"carré" "rectangulaire"
charge	Charge	tonnes	[8, 15]
cout	Coût	euros	[0, 1000000]
epaisseur_accroche	Epaisseur Accroche	cm	[5, 20]
epaisseur_bout	Epaisseur en bout	cm	[5, 20]
hauteur_longeron	Hauteur Longeron	cm	[10, 50]
longueur_longeron	Longueur Longeron	metre	[8, 10]
masse_longeron	Masse longeron	kg	[60, 300]
materiau	Matériau	type	"acier" "carbone"
nom_concept	Nom concept	texte	"Longeron" "Longeron_I" "Longeron_Caisson"
nombre_longerons	Nombre de Longerons	qté	[1, 5]
presence_reservoir	Présence réservoir	oui non	"oui" "non"
risque	Risque	pourcentage	[0, 100]
section_principale	Section Prncipale	type	"I" "tube" "caisson"
type_concept	Type Concept	texte	"ES" "AS"

Figure A4.2 : Modèles de variables et modèles de domaines de définition du concept « Longeron » de l'ontologie

De plus, un concept peut comprendre des modèles de contraintes entre les modèles de variables. Pour le concept Longeron, ces modèles de contraintes sont illustrés dans la figure A4.3. Ces modèles de contraintes correspondent aux contraintes C1, C2 et eCC2 présentés dans la section 7.4.1. Elles répertorient les combinaisons possibles entre les variables impliquées.

Contraintes du concept :

```
constraint C1 using (presence_reservoir, section_principale)
table {
{"oui", "caisson"},
{"non", "caisson"},
{"non", "tube"},
{"non", "I"},
};

constraint C2 using (section_principale, nombre_longerons)
table {
{"I", [1,3]},
{"tube", [2,4]},
{"caisson", [3,5]},
};


constraint eCC2 using (materiau, accroche_longeron, nombre_longerons,
presence_reservoir, T31)
table {
{"acier", "carre", [1,5], "non", [5,6]},
{"acier", "carre", [1,5], "oui", [6,7]},
{"acier", "rectangulaire", [1,5], "non", [6,7]},
{"acier", "rectangulaire", [1,5], "oui", [7,8]},
{"carbone", "carre", [1,2], "non", [7,8]},
{"carbone", "carre", [1,2], "oui", [6,7]},
{"carbone", "carre", [3,5], "non", [6,7]},
{"carbone", "carre", [3,5], "oui", [7,8]},
{"carbone", "rectangulaire", [1,2], "non", [7,8]},
{"carbone", "rectangulaire", [1,2], "oui", [8,9]},
{"carbone", "rectangulaire", [3,5], "non", [8,9]},
{"carbone", "rectangulaire", [3,5], "oui", [9,10]},
};
```

Figure A4.3 : Contraintes inhérentes au concept Longeron de l'ontologie

La figure A4.4 présente la décomposition partielle d'un système « Avion essai2 » en sous-systèmes. Ce dernier ne dispose que d'une alternative (Alternative 1) elle-même composée de trois sous-systèmes (« Moteur avion essai2 », « Structure avion essai2 » et « Cabine avion essai2 »). Le sous-système « Structure avion essai2 » dispose d'une alternative dont l'un des sous-systèmes est le système « Longeron ».

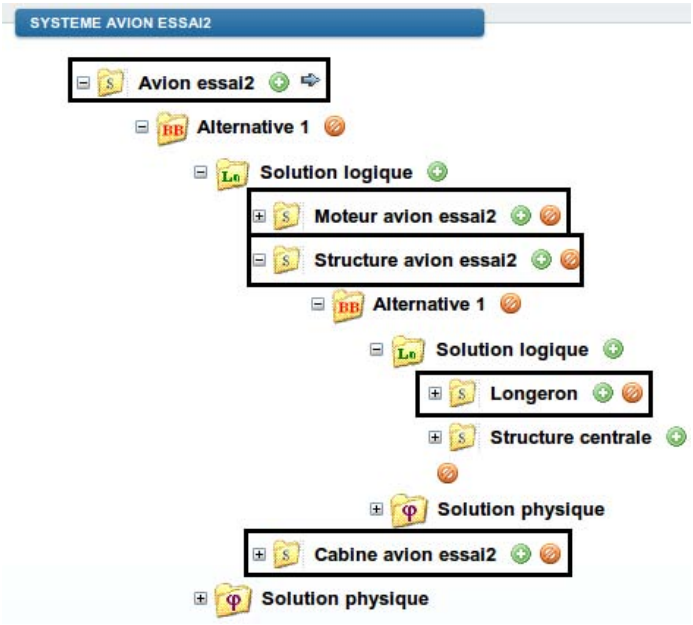


Figure A4.4 : Décomposition d'un système en sous-systèmes

Pour la planification du projet de conception, la structure projet est similaire à la structure du système : le projet de plus haut niveau est le projet relatif à la conception du système « Avion essai2 ». Ce dernier se compose de deux tâches : une tâche de recueil des besoins et exigences techniques et une tâche de recherche d'alternatives (l'ensemble étant noté tâche TE dans ce mémoire) et une tâche de développement d'alternative (tâche TD). Cette dernière comporte, en outre, une tâche de « Conception N+1 » à laquelle les projets de conception des sous-systèmes (« Moteur avion essai2 », « Structure avion essai2 » et « Cabine avion essai2 ») sont rattachés. Ceci est illustré sur la figure A4.5. Similairement, un sous-projet de conception du système « Longeron » est rattaché à la tâche « Conception N+1 » du projet de conception du sous-système « Structure avion essai2 » (non représenté sur la figure A4.5). En dessous de chaque tâche, les dates de début et de fin au plus tôt sont représentées ainsi que les dates de début et de fin au plus tard.

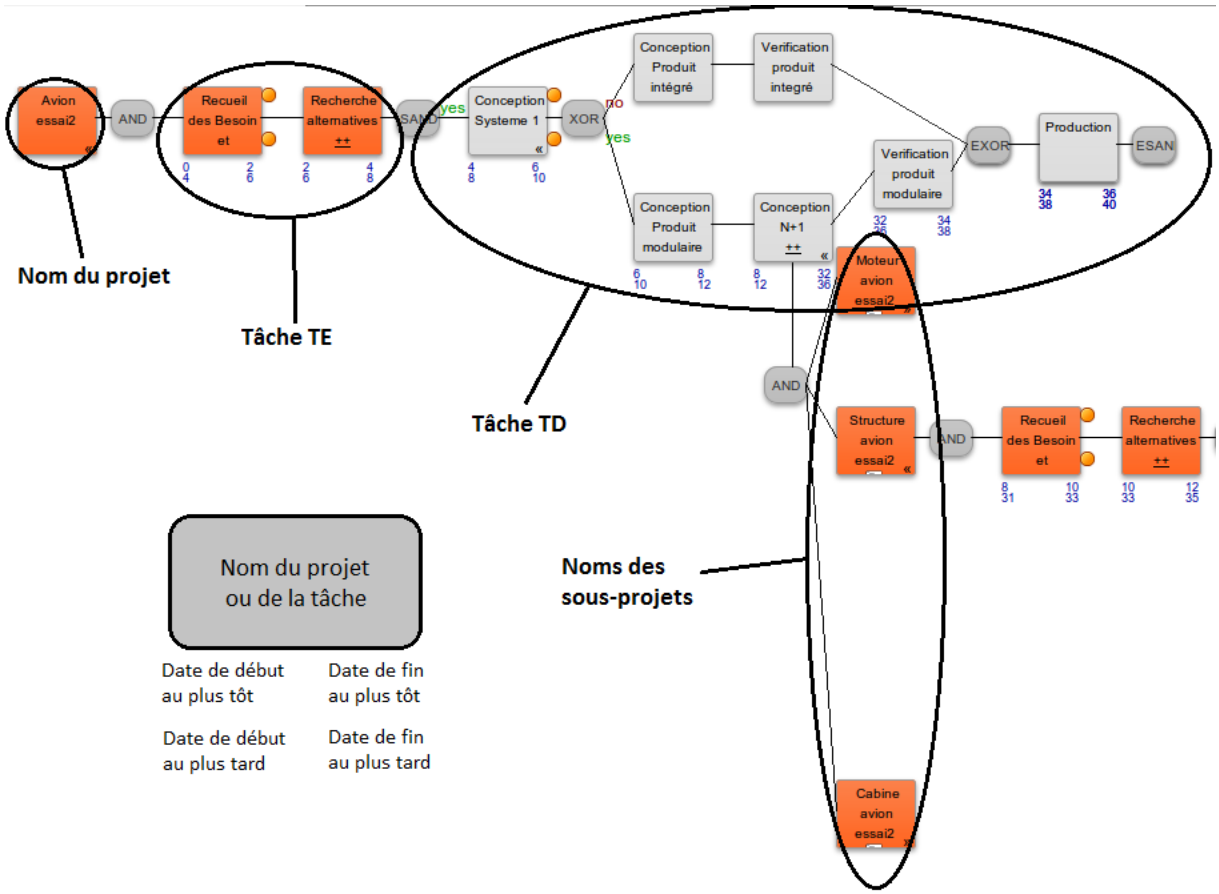
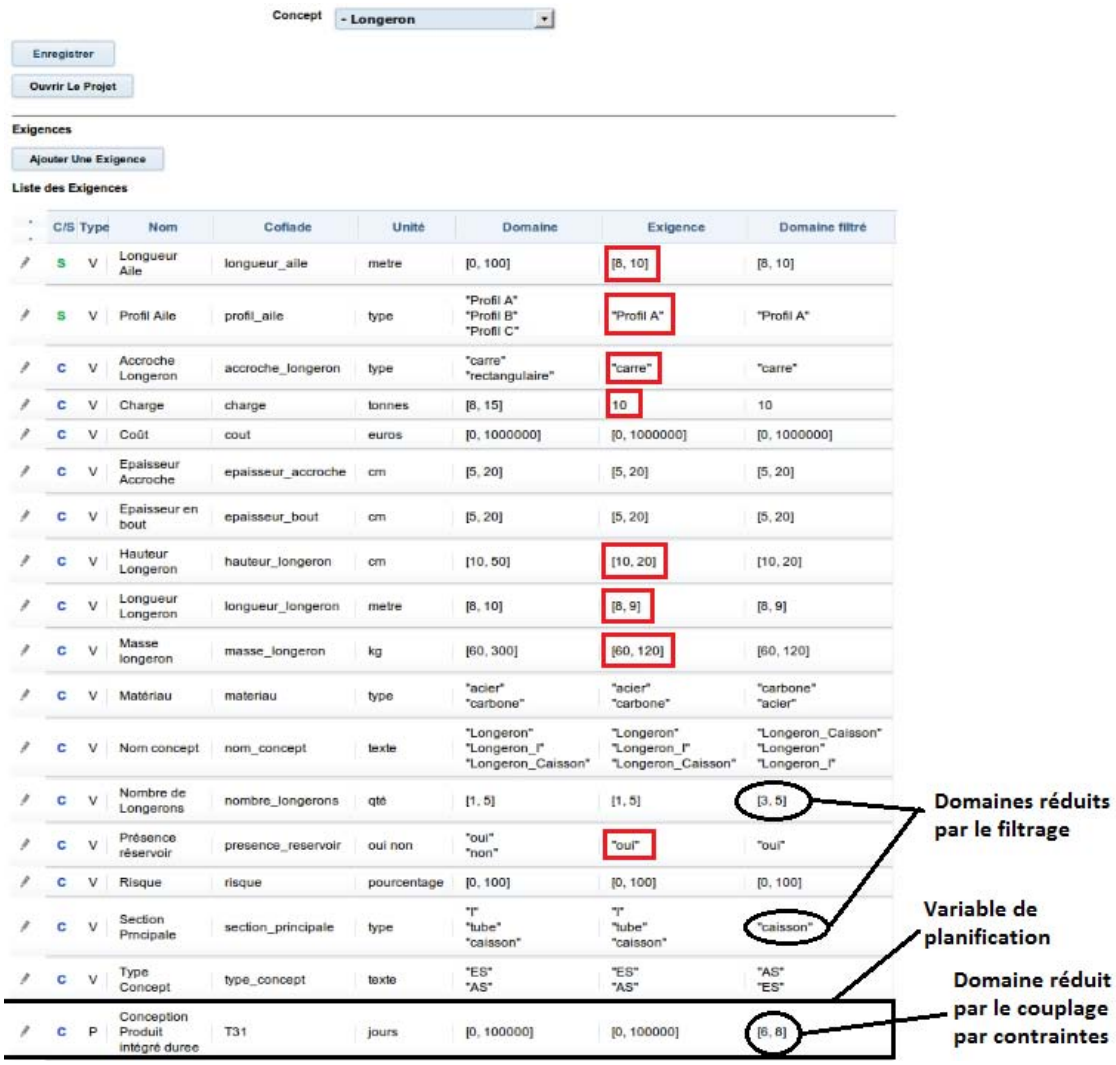


Figure A4.5 : Décomposition d'un projet de conception en sous-projets

La figure A4.6 illustre la définition du système Longeron. Le concept Longeron lui est assigné : les modèles de variables et leurs modèles de domaines de définition inhérents au concept sont dupliqués dans le système (marquées d'un C dans la colonne C/S). Les variables supplémentaires « Longueur Aile » et « Profil Aile » ainsi que leurs domaines de définition ont été ajoutées (marquées d'un S dans la colonne C/S). Des exigences sont renseignées (encadrées dans la colonne « Exigences » dans la figure A4.6). Ces dernières restreignent les domaines de définition des variables. S'il n'y a pas d'exigence sur une variable donnée, le domaine de définition reste inchangé.

Suite au renseignement des exigences, le responsable de conception a choisi d'utiliser la propagation de contraintes (ces dernières, émanant du concept, sont présentées sur la figure A4.3) : les domaines de définition des variables « Nombre Longeron » et « Section Principale » sont réduits par filtrage via les contraintes C1 et C2. Concernant la propagation vers l'environnement de planification, la durée de la tâche de « Conception produit Intégrée » est également réduite via la contrainte eCC2. Le résultat de la propagation est proposé dans la colonne « Domaine filtré » de la figure A4.6.



	C/S	Type	Nom	Cofiade	Unité	Domaine	Exigence	Domaine filtré
	S	V	Longueur Aile	longueur_aile	metre	[0, 100]	[8, 10]	[8, 10]
	S	V	Profil Aile	profil_aile	type	"Profil A" "Profil B" "Profil C"	"Profil A"	"Profil A"
	C	V	Accroche Longeron	accroche_longeron	type	"carre" "rectangulaire"	"carre"	"carre"
	C	V	Charge	charge	tonnes	[8, 15]	10	10
	C	V	Coût	cout	euros	[0, 1000000]	[0, 1000000]	[0, 1000000]
	C	V	Epaisseur Accroche	epaisseur_accroche	cm	[5, 20]	[5, 20]	[5, 20]
	C	V	Epaisseur en bout	epaisseur_bout	cm	[5, 20]	[5, 20]	[5, 20]
	C	V	Hauteur Longeron	hauteur_longeron	cm	[10, 50]	[10, 20]	[10, 20]
	C	V	Longueur Longeron	longueur_longeron	metre	[8, 10]	[8, 9]	[8, 9]
	C	V	Masse longeron	masse_longeron	kg	[60, 300]	[60, 120]	[60, 120]
	C	V	Matériau	materiau	type	"acier" "carbone"	"acier" "carbone"	"carbone" "acier"
	C	V	Nom concept	nom_concept	texte	"Longeron" "Longeron_I" "Longeron_Caisson"	"Longeron" "Longeron_I" "Longeron_Caisson"	"Longeron_Caisson" "Longeron" "Longeron_I"
	C	V	Nombre de Longerons	nombre_longerons	qté	[1, 5]	[1, 5]	[3, 5]
	C	V	Présence réservoir	presence_reservoir	oui non	"oui" "non"	"oui"	"oui"
	C	V	Risque	risque	pourcentage	[0, 100]	[0, 100]	[0, 100]
	C	V	Section Principale	section_principale	type	"I" "tube" "caisson"	"I" "tube" "caisson"	"caisson"
	C	V	Type Concept	type_concept	texte	"ES" "AS"	"ES" "AS"	"AS" "ES"
	C	P	Conception Produit integré duree	T31	jours	[0, 100000]	[0, 100000]	[6, 8]

Figure A4.6 : Définition d'un système « Longeron » et propagation de contraintes

Concernant la planification du projet de conception du système « Longeron », il s'agit d'attribuer des durées minimales et maximales à chaque tâche. Ici, toutes les tâches ont une durée comprise entre 2 et 3 unités de temps à l'exception de la tâche « Conception Produit Intégré » dont la propagation de contraintes réalisée précédemment (voir dernière ligne de la figure A4.6) a permis de déterminer que sa durée était comprise entre 6 et 8 unités de temps. La planification consiste, pour chaque tâche du projet, à déterminer les dates de début et de fin au plus tôt et au plus tard si cela est possible. La planification du projet de conception du système « Longeron » est présentée en figure A4.7.

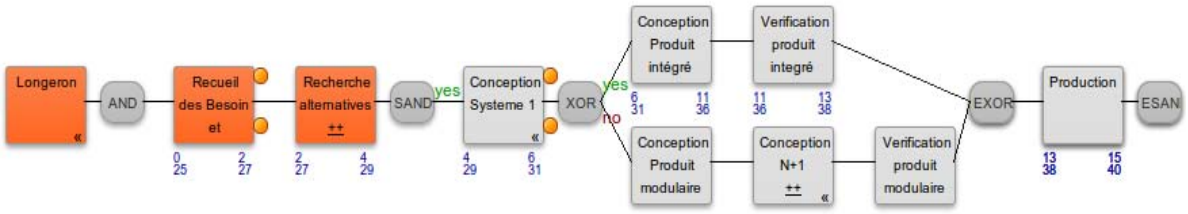


Figure A4.7 : Planification du projet de conception d'un système « Longeron »

Concernant la réutilisation de cas, la figure A4.8 illustre le formulaire de saisie d'exigences en vue d'une recherche d'alternatives existantes. En premier lieu, il est nécessaire de choisir un concept sur lequel la définition des exigences va s'appuyer. Ces exigences systèmes sont définies comme indiqué sur la figure A4.6. Une fois, ces exigences renseignées, il est possible, pour chaque variable contrainte par une exigence, d'étendre le domaine de validité (colonne « Domaine étendu » de la figure A4.8) et de pondérer les variables (colonne « Poids » de la figure A4.8). Il s'agit, d'une part, d'élargir la recherche à des cas proches et, d'autre part, de prendre en compte, pour le calcul de compatibilité, les exigences en fonction de leur importance pour le concepteur. Dans l'exemple, proposé dans la figure A4.8, les exigences sont Longueur_longeron ≤ 9m et Epaisseur_bout ≤ 10cm. Cette dernière exigence est étendue aux épaisseurs inférieures à 14cm, la compatibilité est donc de 1 jusqu'à 10cm et décroit linéairement de 1 à 0 entre 10 et 14cm. Le poids de ces deux exigences est équivalent (0,5).

Figure A4.8 : Formulaire de saisie pour la réutilisation de cas

À partir de ces exigences et du concept préalablement choisi, une recherche de cas compatibles est effectuée. Il en résulte le tableau présenté dans la figure A4.9. Ce tableau propose le nom des alternatives système compatibles (« LongeronX7.1 » et « LongeronX8.1 »), les concepts qui leur sont associés (respectivement « Longeron_Caisson » et « Longeron_I »), la similarité de ces derniers avec le concept du système à concevoir (0,8 pour les deux alternatives) et les mesures de compatibilité entre les alternatives et les exigences étendues (respectivement 0,646 et 0.29).

Alternative	Concept	Sim Wu...	Compatibilté
LongeronX7.1	Longeron_Caisson	0,8	0,646
LongeronX8.1	Longeron_I	0,8	0,29

Figure A4.9 : Résultat de la recherche de cas compatibles avec les exigences système

On choisit l'alternative « LongeronX7.1 ». Cette dernière n'étant pas totalement compatible avec les exigences systèmes, un travail d'adaptation et de révision est nécessaire. Par exploitation du couplage structurel, on identifie la tâche TD associée à ce longeron. Cette tâche TD est dupliquée dans l'environnement et doit être adaptée en conséquence. Une fois, celle-ci adaptée, planifiée et jugée faisable, le travail d'adaptation de l'alternative peut débuter.

Résumé :

Les travaux présentés dans cette thèse s'inscrivent dans une problématique d'aide à la conception de systèmes, à la planification de leur projet de développement et à leur couplage. L'aide à la conception et à la planification repose sur la formalisation de deux grands types de connaissances : les connaissances méthodologiques utilisables quel que soit le projet de conception et, les connaissances métier spécifiques à un type de conception et/ou de planification donné. Le premier chapitre de la thèse propose un état de l'art concernant les travaux sur le couplage des processus de conception de systèmes et de planification des projets associés et expose la problématique de nos travaux. Deux partie traitent ensuite, d'une part, des connaissances méthodologiques et, d'autre part, des connaissances métier. La première partie expose trois types de couplages méthodologiques. Le couplage structurel propose de formaliser les entités de conception et de planification puis permet leur création et leur association. Le couplage informationnel définit les attributs de faisabilité et de vérification pour ces entités et synchronise les états de ces dernières vis-à-vis de ces attributs. Enfin, le couplage décisionnel consiste à proposer, dans un même espace et sous forme de tableau de bord, les informations nécessaires et suffisantes à la prise de décision par les acteurs du projet de conception. La seconde partie propose de formaliser, d'exploiter et de capitaliser la connaissance métier. Après avoir formalisé ces connaissances sous forme d'une ontologie de concepts, deux mécanismes sont exploités : un mécanisme de réutilisation de cas permettant de réutiliser, en les adaptant, les projets de conception passés et un mécanisme de propagation de contraintes permettant de propager des décisions de la conception vers la planification et réciproquement.

Mots-clés : Ingénierie Système, Conception, Planification, Couplage conception/planification, Raisonnement à Partir de Cas, Problèmes de Satisfaction de Contraintes, Ingénierie des Connaissances

---

Abstract :

The work presented in this thesis deals with aiding system design, development project planning and its coupling. Aiding design and planning is based on the formalization of two kind of knowledge: methodological knowledge that can be used in all kind of design projects and *business* knowledge that are dedicated to a particular kind of design and/or planning. The first chapter presents a state of the art about coupling system design process and project planning process and gives the problem of our work. Then, two parts deal with design and planning coupling thanks to, on one hand, methodological knowledge, and on the other hand, business knowledge. The first part presents three types of methodological coupling. The structural coupling defines design and planning entities and permits its simultaneous creation of and its association. The informational coupling defines feasibility and verification attributes for these entities and synchronizes its attribute states. Finally, the decisional coupling consists in proposing, in a single dashboard, the necessary and sufficient information to make a decision by the design project actors. The second part proposes to formalize, to exploit and to capitalize business knowledge. This knowledge is formalized with ontology of concepts. Then, two mechanisms are exploited: a case reuse mechanism that permits to reuse and adapt former design projects and a constraint propagation mechanism that allows propagating decisions from design to planning and reciprocally.

Keywords : System Engineering, Design, Planning, Design/Planning Coupling, Case-Based Reasoning, Constraint Satisfaction Problems, Knowledge Engineering