UNIVERSITÉ CHEIKH ANTA DIOP DE DAKAR

ECOLE DOCTORALE DE MATHEMATIQUES ET INFORMATIQUE
FACULTE DES SCIENCES ET TECHNIQUES

Année : 2015 **N° d'ordre :**

THESE DE DOCTORAT

Mention : INFORMATIQUE ET TELECOMMUNICATIONS
Spécialité : INFORMATIQUE

Présentée par :
Sidi Mohamed Ould Moulaye Abdellahi MAKHTOUR

Titre : Modèle et Processus de Lignes de Produits pour les Systèmes d'Information Web

---

Soutenue le 23/07/2015 devant le jury composé de :

Président :	M. Hamidou DATHE	Professeur	UCAD, Dakar, Sénégal
Rapporteurs :	M. Joel RODRIGUES	Professeur	UBI, Covilhã, Portugal
	M. Mohamed Tayeb LASKRI	Professeur	UBM, Annaba, Algérie
Examinateurs:	M. Abdourahmane RAIMY	Maître de Conférences	UCAD, Dakar, Sénégal
	M. Oumar DIANKHA	Professeur	UCAD, Dakar, Sénégal
Directeurs de thèse :	M. Mbaye SENE	Maître de Conférences	UCAD, Dakar, Sénégal
	M. Mohamed Taher KIMOUR	Professeur	UBM, Annaba, Algérie

Résumé : Les systèmes dǐnformation web sont des éléments cruciaux dans la gestion des entreprises. Les facteurs de productivité, coût et délais de mise sur le marché sont des paramètres clés lors de leur développement. Lǐngénierie des lignes de produit logiciel est une nouvelle approche de génie logiciel ayant principalement pour buts de répondre au mieux à ces exigences.

Cette thèse présente un modèle et un processus de développement de systèmes dǐnformation web, basé sur le concept des lignes de produit logiciel. Ce concept offre une grande souplesse et un haut degré de réutilisablité de nature à concevoir et réaliser efficacement ces systèmes. Basé sur UML et défini selon les trois vues de contenu, de navigation et de présentation, le modèle proposé est doté de règles de passage entre les features comme étant un moyen de spécification des besoins et lǐarchitecture qui représente la conception du système.

Mots clés : Systèmes dǐnformation Web, Lignes de Produits Logiciels, UWE, Features.

Abstract: Web information systems are crucial elements for enterprise management. Productivity factors, cost and time-to-market are key parameters for the development of such systems. Product-line engineering is a new approch in software engineering, especially aiming to efficiently achieve such requirements. This thesis presents a model and a development process of web information systems, based on the concept of software product lines.

This concept offers great flexibility and a high degree of reusability in order to effectively design such systems. Based on UML and defined according to three views of content, navigation and presentation, the proposed model offers transition rules between features as a requirements specification tool and the architecture that represents the system design.

Keywords : Web information system, Software product line, UWE, Features.

REMERCIEMENTS

Je tiens à remercier:

Monsieur M. Hamidou DATHE, Professeur à lťUniversité de Cheikh Anta Diop pour lťhonneur quťil me fait de présider mon jury de thèse.
Monsieur Joel Rodrigues, Professeur à lťUniversité de Beira Interior pour sťêtre montré particulièrement intéressé par mon travail et pour avoir accepté de le rapporter. Je le remercie pour les remarques et les conseils quťil a formulés à propos de mon manuscrit.
Monsieur Mohamed Tayeb LASKRI, Professeur à lťUniversité Badji Mokhtar de Annaba pour lťintérêt quťil a témoigné pour mon travail et pour avoir accepté dťêtre rapporteur de ma thèse. Chacune de ses remarques est pour moi un enrichissement et un plaisir.

Messieurs Abdourahmane RAIMY, Maitre de Conférences à lťUniversité de Cheikh Anta Diop et Omar DIANKA, Professeur à lťUniversité de Cheikh Anta Diop qui mťont honoré en acceptant dťêtre Examinateurs dans ce jury. Je suis très reconnaissant envers eux.

Monsieur Mbaye Babacar SENE, Maitre de Conférences à lťUniversité de Cheikh Anta Diop pour m'avoir fait confiance malgré les connaissances plutôt légères, puis pour m'avoir guidé, encouragé, conseillé et pour son soutien sans faille dans la direction de cette thèse tout en laissant libre cours à ma créativité. Je sais avoir beaucoup appris grâce à lui.

Monsieur Mohamed Taher KIMOUR, Professeur à lťUniversité Badji Mokhtar de Annaba pour la gentillesse et la patience qu'il a manifestées à mon égard durant cette thèse, pour tous les conseils et les programmes qu'il a bien voulu m'envoyer, pour l'hospitalité dont il a fait preuve envers moi lors des séjours que j'ai effectués dans son groupe, et aussi pour m'avoir fait l'honneur de participer au Jury de soutenance. Je ne pourrais malheureusement jamais exposer en quelques lignes lťaide infinie quťil mťa apportée et pour laquelle je lui suis profondément reconnaissant. Je tiens toutefois à le remercier sincèrement, dans le cadre précis de la thèse, pour ses nombreuses relectures et ses conseils avisés sur mon texte.

Je ne sais pas comment exprimer ma gratitude à ces deux personnes autrement qu'en leur promettant d'agir comme eux avec des étudiants dans ma situation, si un jour l'occasion m'en est donnée.
Je tiens aussi à remercier Mr. Deye.. Cťest en partie grâce à vous si jťai toujours gardé le moral pendant ces années de thèse.

Tous ceux et toutes celles sans qui cette thèse ne serait pas ce qu'elle est : aussi bien par les discussions que j'ai eu la chance d'avoir avec eux, leurs suggestions ou contributions. Je pense ici en particulier aux membres de la famille.

TABLE DES MATIERES

FIGURES

TABLEAUX

1. INTRODUCTION

Les Systèmes dĩnformation Web (SIW) sont des systèmes informatiques dĩentreprise, fondés sur les technologies web [32, 53, 58, 59]. Cette dernière décade, ils ont pris de lĩampleur et sont devenus une composante fondamentale de toute organisation. Un SIW est utilisé dans un but informatif ou applicatif et lĩadhésion de lĩinformation dans ce concept a fortement évolué depuis lĩusage plus courant des logiciels et matériels informatiques dans les entreprises [27, 28, 31, 33, 34, 48, 49, 55]. Il est à noter que la traditionnelle transmission dĩnformation via des méthodes traditionnelles est dépassée par rapport au SIW qui se base sur lĩinternet et sa vertu internationale pour promouvoir un produit, un service ou simplement pour faire connaître une entreprise [37, 39, 50, 51, 82, 83, 85, 86].

Les SIW sont des systèmes complexes. Ils sont fondamentaux et indispensables pour la compétitivité des entreprises et leur performance. Ils présentent des exigences de plus en plus pressantes en matière de coût, délais de mise sur le marché et de sécurité [17, 20, 26, 29, 31, 32, 35]. La réutilisation sĩimpose de plus en plus comme une solution technologique et méthodologique pour le développement de SIW, qui sont des systèmes complexes à prépondérance logicielle [41, 44, 46, 48, 52].

Par ailleurs, lĩingénierie des lignes de produits logiciels est un paradigme qui sĩappuie principalement sur la réutilisation et la configuration en masse [54, 56, 57, 65]. Cette ingénierie prône une vision de développement et de modélisation dans laquelle lĩobjectif attendu nĩest plus lĩobtention dĩun produit logiciel, mais dĩun ensemble de produits logiciels présentant des caractéristiques communes [1, 2, 3, 4, 6, 9, 10, 12, 74, 75, 76].

Dans la littérature, différentes approches de développement de systèmes dĩnformation web ont été proposées [8, 17, 28, 40, 45, 56, 60, 64, 68, 69, 72, 76, 78], parfois avec des outils supports. Les plus notables de ces méthodes sont: WebML [28, 33,34, 44, 45, 68]; OOHDM [17, 20, 38, 40, 42], UWE [60, 61, 62, 63, 64, 69]. Toutefois, Lĩimplication des lignes de produits dans le domaine des SIW est à son étape embryonnaire [56] et des efforts sont nécessaires afin dĩobtenir des résultats plus satisfaisants.

En effet, le concept de lignes de produits logiciels (LPL) permet la création de plusieurs produits présentant de grandes similarités mais qui, au final, restent différentes. Elles sont utilisées à des fins de «réutilisation», cĩest-à-dire quĩon se sert dĩun produit spécifique pour créer dĩautres dérivés de ce dernier à partir de ses coordonnées [76, 81,90, 91, 92].

Les avantages dĩune ligne de produits sont considérables. Etant une «famille de produits», elle permet, par exemple, à un constructeur automobile de créer plusieurs modèles à partir dĩun prototype donné. Ces modèles détiennent les similarités de base présentes dans le prototype comme des pneus achetés chez le même fournisseur; cependant, ils ont également leur spécificité, telle que la présence dĩun toit ouvrant pour une voiture ou lĩouverture télécommandée des portières pour une autre, alors que ces deux voitures sont issues dĩune même ligne de produits [74].

En même temps, les lignes de produits permettent de produire de nombreux produits sans trop de contraintes du point de vue financier, temps ou main dĩœuvre. Elles permettent donc une facilitation de la production, surtout grâce à son principe de réutilisation [8, 9, 81, 87, 88, 89,].

Les lignes de produits en rapport avec les logiciels sont appelées «lignes de produits logiciels» [16, 65, 71, 74, 75, 76]. Il sřagit dř «un ensemble de systèmes partageant un ensemble de fonctionnalités communes, satisfaisant des besoins spécifiques pour un domaine particulier et développés de manière contrôlée à partir dřun ensemble commun dřéléments réutilisables» [74, 75].
En intégrant le concept des lignes de produits dans le domaine de lřinformatique, les entreprises de développement des logiciels qui sont les plus à même dřutiliser ce concept anticipent lřinflation des coûts liés à la construction de plusieurs logiciels et se permettent de produire des divers logiciels à la fois, grâce à lřutilisation du paradigme des Lignes de produits. Leur but est dřaméliorer la productivité de logiciels tout en maximisant leur qualité et en diminuant le coût et le délai de production.

Lřévolution du génie logiciel vers le paradigme des lignes de produit a donc permis dřenvisager, puis de réaliser la création de lignes de produits logiciels pour les SIW. Dans cette optique, les SIW sont conçus en se basant sur le principe des lignes de produits [8, 56, 72, 76, 78]. Toutefois, cette évolution nřa pas encore donné pleine satisfaction et beaucoup de challenges sont identifiés, tels que sur le plan notation, processus, configuration, et traçabilité [1, 2, 8, 12, 13, 14, 15]. Les méthodes existantes les plus notables de développement dřapplication Web et qui intègrent le concept LPL se focalisent surtout sur lřarchitecture fonctionnelle et ne couvrent pas tout le cycle de vie du système web. Par exemple, le modèle de navigation, ainsi que celui de la présentation, qui sont des éléments fondamentaux de la conception dřapplication web, ne sont pas du tout pris en compte.

Cette thèse porte principalement sur lřapplication du concept du LPL dans le développement des SIW. Il sřagit donc dřélaborer un modèle et un processus LPL pour le développement des systèmes d'information web en utilisant les éléments et mécanismes de variabilité et commonalité définis par le paradigme des lignes de produits.

Dans le but de lřintégration du concept de ligne de produit dans le processus de développement des SIW, nous avons analysé différentes méthodes existantes de développement web, les plus notables. Notre choix a porté sur la méthode UWE pour les principaux avantages quřelle présente. Principalement, UWE couvre tout le cycle de vie dřune application Web. Elle utilise UML [11,73] comme notation standard international. UWE est une méthode qui est supportée par un environnement de développement offrant des outils prenant en charge les différents modèles à chaque phase de développement [64,73].

Dans cette thèse, nous avons adapté la méthode UWE au paradigme LPL au moyen de lřextension de sa notation mais aussi de la redéfinition de son processus de façon à intégrer la phase de lřingénierie du domaine et la phase de lřingénierie de lřapplication.

La structure de la thèse est composée de cinq chapitres. Le premier chapitre est consacré à lřétat de l'art des lignes de produits. Nous y verrons la définition des lignes de produits, leurs principes de gestion, leurs objectifs de la mise en place, leurs avantages et défis à relever.

Le second chapitre décrit lřarchitecture et les missions des systèmes dřinformation aux services des applications web. Il présente lřanalyse des méthodes les plus notables de développement des applications Web. Nous y détaillerons les techniques, méthodes et outils Web. Puis, nous nous livrerons à une étude comparative entre les Systèmes d'Information Traditionnels et les systèmes d'Informations Web dřun côté, et à une comparaison entre les différentes méthodes de développement des applications Web dřun autre côté.

Le troisième chapitre est consacré à lřapplication du concept de lignes de produits aux systèmes dřinformation Web. A ce niveau nous avons analysé les approches les plus notables, tout en mettant en relief leurs limites.

Le quatrième chapitre développe la première partie de notre contribution, consistant en un ensemble de notation et un processus pour le développement de SIW selon lřapproche LPL.

Le cinquième chapitre développe la deuxième partie de notre contribution, consistant en un framework pour lřingénierie dřapplication web pour la sante, intégrant la dimension LPL.

En dernier lieu, nous concluons la thèse par une présentation des résultats, des points forts, des limites, et des perspectives de ce travail de recherche.

2. ETAT DE L'ART DES LIGNES DE PRODUITS

2.1. Définition des lignes de produit

En terme général, les lignes de produits représentent un ensemble de produits de la même spécificité avec les mêmes caractéristiques dans un seul et même ensemble. En dřautres termes, elles sont comparables à une structure mère comprenant plusieurs éléments relatifs à cette structure. Ces éléments qui sont des produits sont identiques sur quelques points mais différents sur dřautres.

Lorsque les lignes de produits entrent en interaction avec les Systèmes dřInformation Web, elles sont communément appelées « lignes de produits logiciels ». Une ligne de produits logiciels est « un ensemble de systèmes partageant un ensemble de fonctionnalités communes, satisfaisant des besoins spécifiques pour un domaine particulier et développés de manière contrôlée à partir dřun ensemble commun dřéléments réutilisables » [3]. Elle vise une réduction des coûts de production et de maintenance tout en favorisant la création de plusieurs logiciels à la fois, regroupés en son sein [1, 2, 5, 18, 19].

Le champ dřaction des lignes de produits est actuellement axé autour des entreprises. Ces dernières se servent de ce concept pour standardiser leur processus de développement dřentreprises centré sur la production de systèmes, de logiciels ou de matériels et pour accroitre leur taux de productivité.
Ces entreprises prennent en compte les éléments communs et variables constituant les logiciels ainsi que les éléments réutilisables qui ont servi à leur production.

Si tels sont les avantages de ce concept, il contient toutefois quelques limites et lacunes notamment au niveau de lřinsuffisance en nombre des logiciels détenant un grand nombre de points communs ou des différences. Lřélaboration des lignes de produit permet dřarrêter de concevoir des logiciels un à un mais plutôt de les produire en masse grâce aux « assets » dits « éléments réutilisables » [22 , 23, 24, 25].

Les « assets » sont des documents, des matériaux constitutifs, des plans ou des modèles à suivre. Les différences entre les logiciels sont appelés « variabilités». La qualité dřune ligne de produits repose entièrement sur la bonne gestion de ces différences et similarités.

Lřexemple le plus concret concernant les lignes de produits est la production automobile, plus précisément celle dřune chaîne automobile. Les véhicules sont conçus avec des points communs tels que les roues et le volant et des points de différence tels que lřajout de système de climatisation. Enfin, les logiciels conçus dans ces lignes de produits peuvent être à vocation matérielle ou commerciale.

La conception dřune ligne de produits demande lřapplication de deux types dřingénierie : lř« Ingénierie du domaine » et lř« Ingénierie de lřapplication ». Lřingénierie du domaine est dédiée à

la construction des assets utiles pour la réutilisation. Lřingénierie de lřapplication est celle qui consiste à utiliser les assets développés pour la production des produits [1, 87].

Les lignes de produits logiciels font partie du génie logiciel inventé en 1960. Plusieurs projets européens, à savoir ESAPS, CAFE et FAMILIES ont permis aux lignes de produits dřintégrer le génie logiciel.
La ligne de produit a gagné du terrain auprès des entreprises grâce à une action menée par le Software Engineering Institute (SEI) qui, ayant testé et appliqué cette approche, a partagé au grand public ses vertus et avantages. Ainsi, des grands génies de la téléphonie tels que Nokia ont adopté les lignes de produits en accord avec les différents logiciels quřils intègrent dans leurs téléphones mobiles. On note, par exemple, la diversité des langues (plus de 60) disponibles lors de la manipulation du téléphone qui est caractérisé dřune variabilité [1].

La ligne de produits a également été abordée par cette industrie du fait quřelle doit développer et concevoir un grand nombre de téléphones en une année, les LPL lui permettent donc dřaccroître sa production et de suivre le flux de production exigé, ou même de lřexcéder afin de générer plus de bénéfices tout en garantissant la qualité et lřinnovation dans les produits.

Pour bien cerner le concept de ligne de produits, il est essentiel de connaitre la signification des différents termes techniques qui lui sont propres [1,74] :

- Les similarités ou « commonalités » sont les points communs que chaque logiciel détient,

- Les variabilités représentent les différences que les logiciels présentent,

- « Un domaine est un secteur de métier ou de technologies ou des connaissances caractérisées par un ensemble de concepts et de terminologies compréhensibles par les utilisateurs de ce secteur ».

Les lignes de produits sont donc particulièrement utiles, voire même indispensables aux industries, surtout aux industries automobiles et à celles œuvrant dans la téléphonie mobile. Leurs avantages sont indéniables, pour ne citer que la réduction des coûts de production au profit dřune croissance au niveau de la production elle-même.

Les objectifs des lignes de produits logiciels sont dřéviter la conception dřun logiciel ou dřun produit « à partir de rien » [1], mais plutôt de produire un bon nombre de produits à partir de la réutilisation dřun logiciel déjà existant. Cela implique quřun concepteur réalise et fabrique un produit à partir dřéléments réutilisables de sorte que plusieurs autres produits similaires puissent être conçus au même moment [81].

Les lignes de produits logiciels reposent donc essentiellement sur le procédé de réutilisation. Cette « solution méthodologique et technologique » [2] est surtout utilisée pour le développement de systèmes logiciels complexes et fastidieux. Les concepteurs se concentrent donc sur lřoptimisation de la réutilisation et donc sur la conception de meilleurs éléments réutilisables.

Dans cette optique, on note lřusage dřune ingénierie des domaines dont lřobjectif principal est de créer une gamme de systèmes logiciels avec des traits communs plutôt que dřun seul système logiciel uniquement.

Les lignes de produit logiciels passent par plusieurs phases telles que la phase de modélisation lors de leur conception. Ces phases sřappuient sur la réutilisation des assets et sur une configuration commune à tous les logiciels produits. La phase de modélisation est lřétape primordiale constituant la conception des lignes de produits. Elle permet de définir les éléments similaires et les éléments différents dans un produit et dans lřensemble de logiciels de produits conçus.

La modélisation consiste en une approche par un modèle comme celui du modèle par les caractéristiques ou celui des modèles UML (ou Unified Modeling Language). Elle peut constituer une contrainte ou une limite lorsque le concepteur de la lige de produits en confronté à une ligne de produit compliquée à réaliser ou complexe dans son développement ou dont les membres doivent être plus nombreux que dřhabitude.
En effet, lorsquřil est amené à produire des logiciels de type complexe, il nřest pas évident pour lui de trouver des caractéristiques en masse, surtout du point de vue variabilité. De plus, il peut aussi faire face aux demandes ou aux exigences tendancielles des clients, des utilisateurs de tels logiciels et de la technologie qui devra assouvir, dřoù une pression énorme et un effort en plus à déployer.

La modélisation est une phase cruciale dans la conception dřun système de logiciels dans la mesure où elle permet de visualiser la composition de ce dernier et de déterminer les fonctions et rôles de chaque intervenant dans la conception de la ligne de produits logiciels [72, 80].

En pratique, la tâche de modélisation est distribuée sur différents intervenants ou équipes. Elle est effectuée selon les exigences et les types de clients. Les intervenants réalisent chacun des modèles, combinés, qui constitueront le modèle global du système de logiciel. La ligne de produits est confrontée à un problème majeur au niveau de la composition de chaque produit. On dit que les logiciels à mettre en place sont souvent de grande taille, rendant compliqué la composition de chaque modèle. Cette difficulté se fait sentir au niveau des procédures et des techniques à mettre en place afin dřassembler et de travailler les composants des différents modèles. Ce cas concerne surtout les logiciels compliqués qui nécessitent des caractéristiques et une structure hors du commun.

Dans la conception de lignes de produits, le plus important est de se focaliser sur les variabilités présentes dans chaque modèle. En effet, elle doit être visible et chacun devrait pouvoir différencier chaque modèle réalisé dřun autre malgré leurs caractéristiques communes. La conception de ces variabilités est compliquée dans la mesure où chaque variabilité doit être représentée par une « information de variabilité » lui étant spécifique. Cřest dans cette optique quřon parle de difficultés ou contraintes relatives aux variabilités, un des éléments cruciaux lors de la composition et de la modélisation dřun système de logiciels.

Ce qui rend complexe la conception des variabilités, cřest le fait de définir lřinformation de variabilité, mais aussi les contraintes de variabilité pour chaque élément composant chaque modèle. Ces dernières doivent être identifiées et conçues de sorte que tous les modèles composant une ligne de produits soient dépourvus des contraintes qui risquent de les rendre invalides et donc impossibles à mettre sur le marché.

Le dernier élément important constituant un modèle dans une ligne de produits, et non le moindre, est une « sémantique bien définie qui reflètera la portée et la visée de la ligne de produits. Tous ces éléments sont à définir pendant la phase de composition dřune ligne de

produits. Une fois quřils sont bien inclus dans chaque modèle, ils doivent être vérifiés à la fin de la modélisation et de la composition avant de déclarer que tel ou tel modèle est bien valide.

Le processus de modélisation étant un processus de base dans la conception dřune ligne de produits, il doit être choisi suivant les objectifs et les modèles dřune ligne de produits. Il convient donc de choisir la bonne architecture pour les variabilités, telle quřUML ou le formalisme orthogonal.

La modélisation et la composition des lignes de produits font partie de leur phase de conception. Cette phase est essentiellement compose de deux ingénieries : ingénierie de domaine et ingénierie dřapplication. Ces ingénieries ont pour but de concevoir et de mettre en place différents modèles de produits issus dřune seule et même ligne en peu de temps, à faible coût et de manière qualitative. Cřest cette ingénierie qui permet la conception de plusieurs modèles à la fois, au lieu dřun modèle à la fois comme cřétait le cas auparavant. Lřingénierie des LPL est donc la source de modélisation de la LPL elle-même.

La ligne de produits présente divers avantages indéniables. Basée sur la réutilisation qui, selon [3], est *"Le processus de création des systèmes logiciels à partir des logiciels existants au lieu de les créer à partir de zéro",* elle évite alors de faire des efforts pour la production dřun seul produit mais plutôt dřutiliser des éléments appelés assets pour créer une famille de produits simultanément. La réutilisation constitue un des atouts majeurs de la conception des lignes de produits, elle sert, depuis quelques décennies, à réduire le coût de production tout en assurant une amélioration de la qualité.

La réutilisation, dans le domaine des lignes de produits logiciels, cache plusieurs défis tels que lřindentification des assets ou éléments à réutiliser et lřintégration de ces derniers dans chaque produit avec moins de difficulté, de temps et de coût que lorsquřon part de zéro pour concevoir un produit. Ces contraintes ont été abolies avec lřapparition de la ligne de produits puisque ces efforts ont été réduits par ce concept dès lors quřun modèle doit présenter des similarités.

Pour mener à bien lřacte de réutilisation, une bonne planification est requise, sinon les budgets alloués à cette dernière risqueraient dřêtre nettement supérieurs au budget de la création à partir de zéro. Néanmoins, ce point est aussi résolu en termes de ligne de produits logiciels. En effet, la qualité tant prônée chez les LPL résulte de la planification étudiée, testée et prouvée en matière de réutilisation.

La réutilisation puise ses forces en réutilisant les éléments les plus onéreux et contraignants comme les structures, et les composantes dřun logiciels. Ces éléments seront réutilisés chez les autres méthodes avec plus dřamélioration et de perfectionnement, rendant la LdP plus qualitative.

Mais la réutilisation à elle-seule nřest pas suffisante pour la conception de plusieurs modèles identiques et différents à la fois. Elle est alors accompagnée de la configuration de masse que [4] définit de la sorte : *"La production à grande échelle de biens adaptés aux besoins individuels des clients ".* La configuration de masse est une étape cruciale dans la conception de la ligne de produits puisquřelle constitue une étape primordiale dans lřingénierie de la LPL.

Elle permet dřintégrer les éléments communs et les éléments différents dans chaque modèle en accord avec les exigences dřun public cible dřutilisateurs prédéfinis. Ainsi, lřingénierie permet de faire en sorte à ce que plusieurs modèles de produits soient pensés, créés et intégrés dans une

même famille de produits en vue de satisfaire un large public de consommateurs et dřutilisateurs.

Outre la configuration de masse, on note aussi la modélisation et la composition des lignes de produits logiciels. La modélisation est une abstraction définie par [5] comme étant *"la traçabilité des correspondances à partir d'une représentation d'un problème vers une autre représentation qui préserve certaines propriétés souhaitées et qui réduit la complexité".*

Approches de modélisation orientée aspects

Les approches de modélisation orientée aspects séparent les éléments de modèles qui appartiennent à différentes préoccupations. Un modèle primaire ou aussi de base représente le cœur fonctionnel du système. Plusieurs modèles dřaspects sont aussi modélisés pour représenter les préoccupations transverses du système, comme par exemple la sécurité et la persistance. Les modèles dřaspects doivent alors être composés avec le modèle primaire afin dřobtenir un modèle de conception intègre. La communauté de développement de logiciels par aspects porte un intérêt particulier à lřexploitation des mécanismes utilisés pour le développement des lignes de produits logiciels.

Plusieurs travaux se sont focalisés sur la capture des points communs et des points variables dans une ligne de produits logiciels en utilisant les aspects à une étape précoce [4, 5, 7, 8, 9] ainsi quřà lřutilisation de la technologie de lřaspect au niveau implémentation pour représenter les caractéristiques des lignes de produits. Par exemple, [10, 11, 12] représentent les caractéristiques dřune ligne de produits logiciels au niveau modèle par des aspects (qui sont aussi des modèles). Au niveau implémentation, les aspects encapsulent les implémentations des caractéristiques. Pour obtenir un produit de la ligne de produits, les aspects sont composés avec la base selon une sélection des caractéristiques pour une configuration donnée. Cřest ainsi quřils expriment la variabilité par les aspects.

Parmi les approches existantes pour la composition des modèles de lignes de produits logiciels par utilisation des technologies de lřaspect, deux approches sont sélectionnées et analysées dans cette partie.

Dans [8, 9] les auteurs proposent de fusionner les modèles en deux phases. La première phase consiste à comparer les modèles à fusionner pour identifier les éléments qui représentent un même concept. Ces éléments sont alors dits des éléments correspondants. Ils sont ensuite fusionnés durant la phase de fusion. Pour identifier les éléments correspondants, lřapproche propose de se baser sur la comparaison des signatures des éléments. Tout type dřélément possède un type de signature défini par ses propriétés syntaxiques pour assurer lřunicité des éléments. Deux éléments sont alors fusionnés sřils ont des signatures équivalentes.

Par exemple, le type de signature dřune classe UML est défini par la propriété name qui indique le nom de la classe ainsi que la propriété isAbstract qui indique si la classe est abstraite ou non. La signature de la classe Sensing de la figure 2.1 est donc définie par {name= Sensing, isAbstract=false}. Si deux vues de la classe Sensing appartiennent aux modèles à fusionner et quřelles ont le même nom et la même valeur de la propriété isAbstract, elles sont alors correspondantes et elles sont fusionnées durant la phase de fusion pour former une seule classe dans le modèle de fusion résultant.

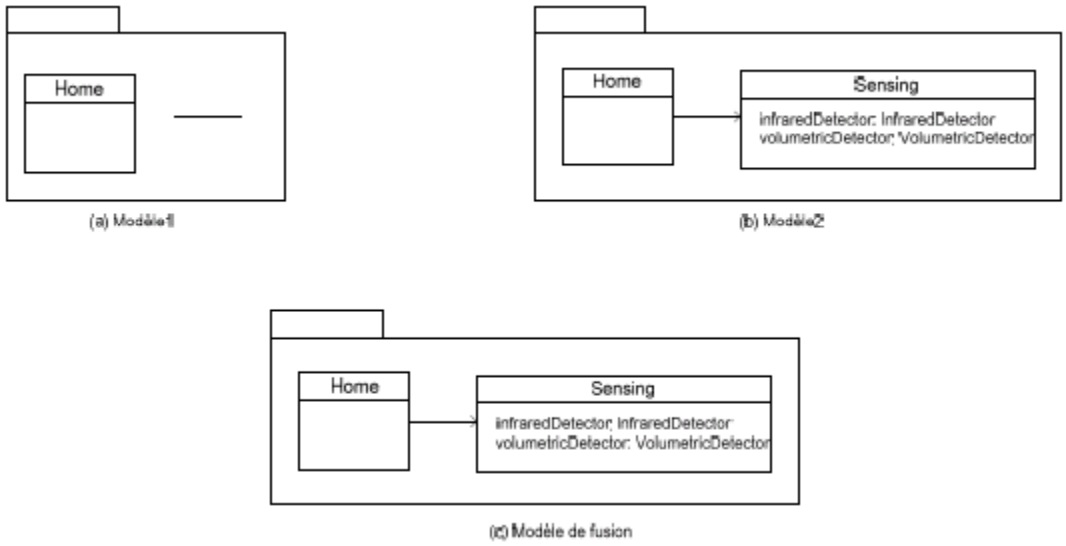

Figure 2.1 : Exemple de fusion basée sur les aspects [2]

Dans ce travail, les auteurs mènent un raisonnement local sur les modèles durant la fusion. La variabilité nřest pas traitée à ce niveau, ce qui explique lřabsence de gestion de lřinformation de variabilité associée aux éléments des modèles à fusionner. De même pour les contraintes de variabilité qui ne sont pas traitées durant le processus de fusion proposé.

Lřapproche de [9, 10] représente beaucoup de similitudes avec lřapproche précédente. Dans cette proposition, un modèle appelé *theme* est créé pour chaque exigence du système. Ces *themes*, équivalents aux modèles dřaspects et primaires, représentent différentes vues du modèle global. Le modèle intègre est obtenu par composition de ces *themes*. Dans lřapproche *Theme*, des relations de composition spécifient comment les modèles doivent être composés par identification des éléments correspondants ainsi que comment ils sont fusionnés. Le mécanisme de fusion est utilisé en se basant sur la comparaison des noms des éléments à fusionner. La description de lřutilisation de ce critère de comparaison manque de précision.

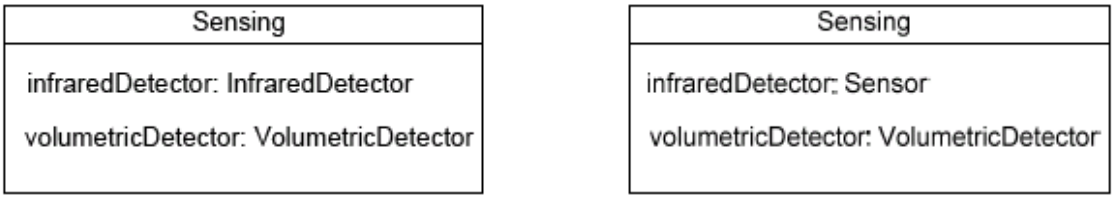

Figure 2.2 : Exemple de correspondance des classes Sensing [2]

Elle reste dépendante de lřutilisateur de lřapproche. Cependant, nous rappelons que la comparaison des éléments à fusionner en se basant sur leurs noms demeure un critère faible et permissif comme nous lřavons illustré pour lřapproche dřAcher dans la figure 2.2. De plus comme pour lřapproche précédente, les auteurs raisonnent localement sur les modèles et ne traitent pas la variabilité. La fusion des modèles ne traite donc ni lřinformation de variabilité des éléments à fusionner ni les contraintes de variabilité qui leurs sont associées. De même que

l'approche précédente, le raisonnement sur les modèles durant la fusion reste local. La variabilité n'est pas traitée à ce niveau, ce qui explique l'absence de gestion de l'information de variabilité associée aux éléments des modèles à fusionner. De même pour les contraintes de variabilité qui ne sont pas traitées durant le processus de fusion proposé.

L'utilisation des techniques de développement par aspect des logiciels dans le contexte des lignes de produits logiciels prend de plus en plus d'ampleur. En effet, les différentes exigences de la ligne de produits sont modélisées sous forme d'aspects (qui sont des modèles) et d'un modèle de base. Pour obtenir un produit de la ligne de produits logiciels, le modèle de base est composé avec les aspects qui représentent les exigences du produit souhaité. Les mécanismes de composition à ce niveau se basent sur l'aspect structurel des modèles composés. Cependant le raisonnement de ces approches au niveau composition reste local et ne traite donc ni l'information de variabilité des éléments structurels ni les contraintes de variabilité. De même la sémantique de composition n'est pas évoquée par ces approches.

Approches dirigées par les modèles annotatrices de variabilité

Certaines approches de développement de lignes de produits logiciels utilisent des annotations pour identifier les éléments variables dans le modèle. Nous distinguons les approches qui représentent les lignes de produits logiciels par le biais de modèles UML [11]. Plusieurs approches dirigées par les modèles utilisent des stéréotypes UML pour annoter les éléments variables qui appartiennent aux modèles de lignes de produits logiciels [1, 12, 13, 14, 15, 16]. Par exemple, [12,14] propose un profil UML pour permettre l'expression de la variabilité dans les modèles UML qui représentent des lignes de produits logiciels.

Ce profil indique que tout élément du modèle peut faire l'objet d'un point de variation. Un point de variation localise la variabilité dans un modèle et ses différentes réalisations décrivent plusieurs variantes. Le point de variation contient plusieurs variantes qui déterminent les caractéristiques de la variabilité. Une variante est toujours attachée à un point de variation. Le point de variation implique plusieurs valeurs étiquetées pour déterminer le moment de la réalisation du point de variation et la cardinalité des variances ainsi que le nombre de variances qui lui correspondent. Le stéréotype «*optional*» indique qu'un élément peut être supprimé dans le modèle dérivé.

Dans le travail de [14] en plus de l'utilisation du modèle de caractéristiques, cette approche utilise des extensions d'UML pour modéliser la variabilité au niveau structurel grâce à deux mécanismes à savoir l'optionalité et la variation ainsi qu'au niveau comportemental moyennant trois mécanismes qui sont l'optionalité, la variation et la virtualité.
Au niveau structurel (aspect statique) par exemple deux stéréotypes sont proposés:

- L'optionalité: par l'utilisation du stéréotype «*optional*» ce qui signifie que certaines propriétés sont optionnelles dans certains produits et donc elles peuvent être supprimées. L'optionalité n'est pas modélisée au niveau des associations entre les classes car une classe optionnelle signifie que toutes les associations auxquelles elle participe sont aussi optionnelles. L'optionalité d'une classe s'étend sur tous ses attributs et ses opérations, idem pour un paquetage optionnel dont l'optionalité s'étend sur tous ses éléments.

- La variation: par l'introduction des stéréotypes «*variation*» et «*variant*» associés

respectivement aux classes abstraites et aux sous-classes concrètes. Au niveau comportemental (aspect dynamique): diagrammes de séquence

- L'optionalité: par l'introduction du stéréotype «*optionalLifeline*» qui s'applique sur les messages reçus et envoyés des objets optionnels dans le diagramme de séquence. Le stéréotype «*optionalInteraction*» spécifie l'optionalité des interactions.

- La variation: par l'introduction du stéréotype «*variation*» appliqué sur une interaction qui référence un ensemble de sous-interactions stéréotypées «*variant*».

- La virtualité: par l'introduction du stéréotype «*virtual*» qui une fois appliqué sur une interaction, signifie que le comportement de celle-ci peut être redéfini par une autre interaction de raffinement associée à un produit particulier. Le comportement de l'interaction virtuelle sera remplacé pendant la dérivation de produit par le comportement de l'interaction de raffinement associée au produit.
Dans [12] lřauteur sřest basé sur les composants. On ne parle plus de modèle de caractéristiques mais plutôt d'arbre de composants dont le composant racine représente le système et les sous composants représentent les éléments constitutifs du système. Elle utilise un seul stéréotype «*variable*» pour marquer les éléments variables de type paquetage, classe, attribut, opération, transition et interaction.

Au niveau statique, l'approche traite les diagrammes de classes, au niveau comportemental (dynamique), les diagrammes de séquences et machine à états. L'approche introduit un modèle de décisions qui remplace le rôle du diagramme de caractéristiques et qui est construit à partir du modèle de famille de systèmes et des spécifications de l'ensemble des systèmes visés. Pour chaque décision le concepteur doit choisir une résolution qui affecte alors le modèle de famille de systèmes sous-jacent et peut renvoyer l'utilisateur vers de nouvelles décisions.

Lřapproche SEQUOIA (précédemment nommée [18] utilise le diagramme de caractéristiques pour la capture des exigences. Elle offre le moyen d'exprimer la variabilité au niveau du modèle UML de la ligne de produits logiciels grâce au stéréotype « *VariableElement*». Les contraintes de variabilité sont exprimées pour identifier les produits logiciels valides qui peuvent être obtenus à partir du modèle de la ligne de produits logiciels.

Nous citons à titre dřexemple une contrainte dřimplication entre deux éléments variables: *Implication (elem1, elem2)* : une contrainte de type *Implication* spécifie que la présence de lřélément *elem1* dans un produit valide implique la présence de lřélément *elem2* dans le même produit valide de la ligne de produits et inversement.

TABLEAU 2.1 : APPROCHES DE MODELISATION ET DE COMPOSITION DES LIGNES DE PRODUITS [2]

Enfin, un des grands avantages et non le moindre des lignes de produits est sa conception réfléchie et optimisée qui permet de réduire nettement les coûts de production tout en accroissant considérablement la qualité de chaque produit issu dřune LPL [25, 16, 74, 75]. De ce fait, les LPL sont adoptées par les plus grandes industries de la téléphonie mobile comme Nokia qui, en 2012, arrivait à produire 25 à 30 modèles de téléphone en un an ou des industries aéronautiques ou automobiles.
Après avoir étudié les bases, notamment la définition, de lřapproche par la ligne de produits, nous allons maintenant entrer en profondeur dans la connaissance des LPL en analysant leurs différents principes de gestion.

2.2. Les principes de gestion des lignes de produits

La gestion des lignes de produits détermine la qualité dřune gamme de produits. Pour créer une LPL digne de ce nom, deux sortes dřingénierie sont utilisées : lřingénierie de domaine et lřingénierie dřapplication [87, 88, 89]. Nous allons dřabord définir lřingénierie de domaine et ses objectifs avant de nous tourner vers lřingénierie dřapplication [1, 2, 25].

2.2.1. L'ingénierie de domaine

i. Définition

Comme nous lřavons mentionné un peu plus haut, lřingénierie de domaine ou « Domain Engineering » concerne la construction et le développement des « assets » utiles pour la création des logiciels. Les « assets » en question sont des éléments réutilisables. Lřingénierie de domaine est la première étape du processus de création et dřélaboration dřune ligne de produits car elle permet de réaliser la réutilisation, fondement même des LPL. Cette phase est exceptionnellement conçue afin de développer les éléments pour la réutilisation.
Elle comporte trois activités :
- Activité dřanalyse de domaine ou *Reference Requirements*
Elle consiste à apprendre tout ce qui concerne le domaine de la LPL à mettre en place, cřest-à-dire à détecter les similarités et les variabilités présentes dans chaque produit, par exemple FODA.

- Activité de conception de domaine ou Reference architecture
La conception de domaine est lřétape réservée à la réalisation de ce quřon appelle « architecture logicielle générique » Cette architecture logicielle générique déterminera chaque produit puisquřelle servira de référence à sa conception. Elle contient, généralement, les caractéristiques propres à lřélaboration de la LPL, à savoir ses constituants, ses connecteurs et les difficultés rencontrées lors de sa mise en place. Lřactivité de conception de domaine doit impérativement être effectuée après avoir analysé le domaine car lřarchitecture définie durant cette phase sera explicitement rapportée vers lřarchitecture logicielle générique.

- Activité dřimplantation du domaine ou Production Plan
Lřimplantation du domaine est la phase succédant à la conception du domaine. Une fois lřanalyse et la conception du domaine achevées, on pourra alors procéder à lřimplantation du domaine.
Cette dernière consiste à transposer lřarchitecture générique mentionnée précédemment vers le domaine à concevoir de manière à obtenir des « composants » ou « assets » essentiels à lřingénierie dřapplication.

Lřingénierie de domaine est donc la phase primordiale à la bonne mise en place dřune ligne de produits. Tout ce qui touche les actions primaires pour la conception des LPL est effectué durant cette étape. En outre, lřingénierie de domaine permet aussi de déterminer les noms attribués à chaque produit ou « scope ».
Après cette définition détaillée de lřingénierie de domaine, composant essentiel de la gestion des lignes de produits, nous allons maintenant déterminer ses différents objectifs.

ii. Objectifs

Lřingénierie de domaine est la phase clé dans la conception des lignes de produits. Elle permet dřanalyser le domaine, de concevoir lřarchitecture générique propre à ce dernier et dřimplanter tous ces éléments dans le domaine, avant de léguer les rennes à lřingénierie dřapplication.
En général, lřingénierie de domaine est indispensable pour :
« Définir la variabilité autorisée et ses limites, Capitaliser les artefacts dřingénierie (exigences, modèles, code, test, etc.), Gérer les évolutions en maintenant la cohérence globale des produits, Gérer les défauts et leurs impacts. ».

Elle permet dřétablir dřavance les principales caractéristiques de chaque produit qui va former la ligne de produits. La LPL repose donc essentiellement sur sa réussite, ce qui amène à la considérer avec sérieux et à ne pas la sous-estimer. Pour ce faire, des professionnels en la matière devraient être mobilisés et sollicités pour garantir une ingénierie de domaine optimale et impeccable.

La raison dřêtre de lřingénierie de domaine est de permettre la gestion de la LPL dans son ensemble, cřest-à-dire la gestion commune de tous les modèles et non la gestion individualisée. Cette phase permet de mettre en place les éléments relatifs aux exigences de tous les usagers des logiciels conçus. En même temps, cette étape permet de caractériser les objectifs que chaque modèle de produit renferme. Dans cette optique, lřingénierie des domaines est cent pour cent focalisée sur les éléments réutilisables qui définiront les points communs entre chaque logiciel formé.

2.2.2. L'ingénierie d'Application

i. Définition

Lřingénierie dřapplication ou « Application Engineering/ Development with reuse » est la phase utilisant les résultats de lřingénierie de domaine dans le but de construire la ligne de produits. Elle permet la réutilisation de ces éléments pour les convertir en nouveaux logiciels. Lřingénierie dřapplication est également appelée « dérivation » pour sa faculté à « dériver » les éléments réutilisables conçus dans lřingénierie de domaine en les convertissant en logiciels.

Cette phase sřappuie sur la réutilisation des éléments réutilisables ou assets pour développer chaque modèle de la LPL.
Elle renferme quatre étapes successives :

- Lř « Application Requirements »

Lř « application requirements » permet dřidentifier les variabilités chez chaque produit et de déterminer les exigences du logiciel final conçu à partir des « assets » ou « core assets » .

- Lř « Application Design »

Lřapplication design est lřétape consistant à arranger lřarchitecture générique en fonction des variabilités et des besoins spécifiques liés à la conception des nouveaux logiciels.

- Lř « Application Coding »

Lřapplication coding est en liaison directe avec les différents besoins spécifiques requis par les nouveaux logiciels. Dès que ces besoins sont identifiés, lřapplication coding est en charge de former de nouveaux composants leur correspondant.

- Lř « Application Testing »

Cette étape consiste à tester les éléments réutilisables dans la phase domaine dřapplication pour attester de leur compatibilité avec les nouveaux composants conçus en phase dř « application testing » et éviter dřéventuelles anomalies.

Nous pouvons constater à travers les différentes phases formant lřingénierie dřapplication quřelles sont échelonnées par degré dřimportance et quřelles dépendent les unes des autres. A présent, nous allons découvrir les différents objectifs de lřingénierie dřapplication.

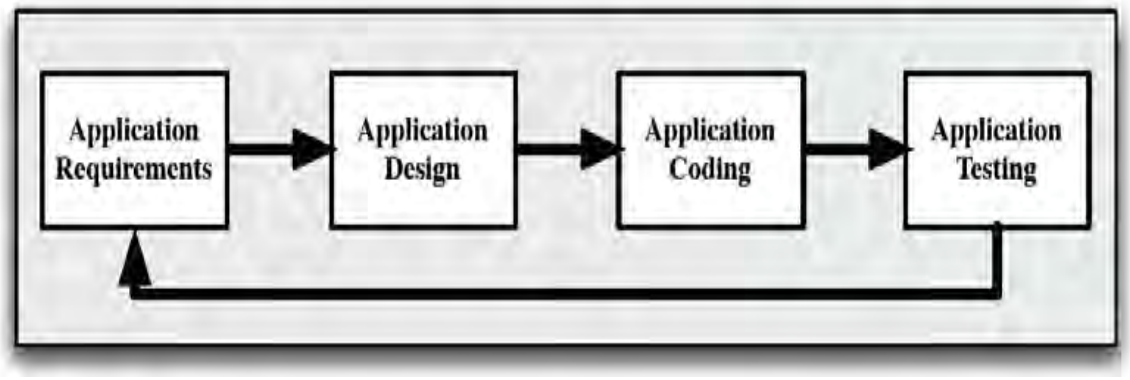

Figure 2.3 : Processus de développement logiciel classique [1]

ii. Objectifs

Lřingénierie dřapplication est le domaine étudiant les propriétés des nouveaux logiciels et qui les met en pratique. Pour quřelle soit menée à bien, lřexpert chargé de lřingénierie de domaine doit optimiser son travail. Les objectifs de lřingénierie dřapplication sont :

- La conception de logiciels performants dotés de variabilités et de commonalité à la fois à lřaide des éléments réutilisables ou assets fabriqués dans lřingénierie de domaine,

- La mise en place de nouveaux composants réutilisables en adéquation avec les éléments réutilisables déjà en main permettant dřassouvir les besoins spécifiques que les logiciels pourraient avoir,

- Lřachèvement des logiciels finaux

En somme, on peut constater une interdépendance entre lřingénierie de domaine et lřingénierie dřapplication. En même temps, la mise en place de ces deux processus est complexe, du point de vue des « variabilités » qui constituent actuellement les contraintes les plus conséquentes dans la conception de nouveaux logiciels.
Pour une gestion optimale des lignes de produits, il vaut donc mieux perfectionner ces deux ingénieries et respecter à la lettre leurs étapes. Maintenant que nous avons défini les principes de gestion des LPL, nous allons nous tourner vers la notion de variabilité et de commonalité.

La visée de lřingénierie dřapplication est avant tout de produire les produits demandés grâce aux assets ou artefacts développés dans lřingénierie de domaine. Elle représente donc une étape dépendante de celle de lřingénierie de domaine. Dans cette optique, elle se donne pour

mission de réutiliser le plus dřéléments possibles pour un modèle bien particulier de la LPL et dřétudier en profondeur les similarités et variabilités présentes dans chaque produit afin de mener à bien la construction des produits ou dérivation.

La figure ci-dessous résume le processus de développement dřune ligne de produits logiciels

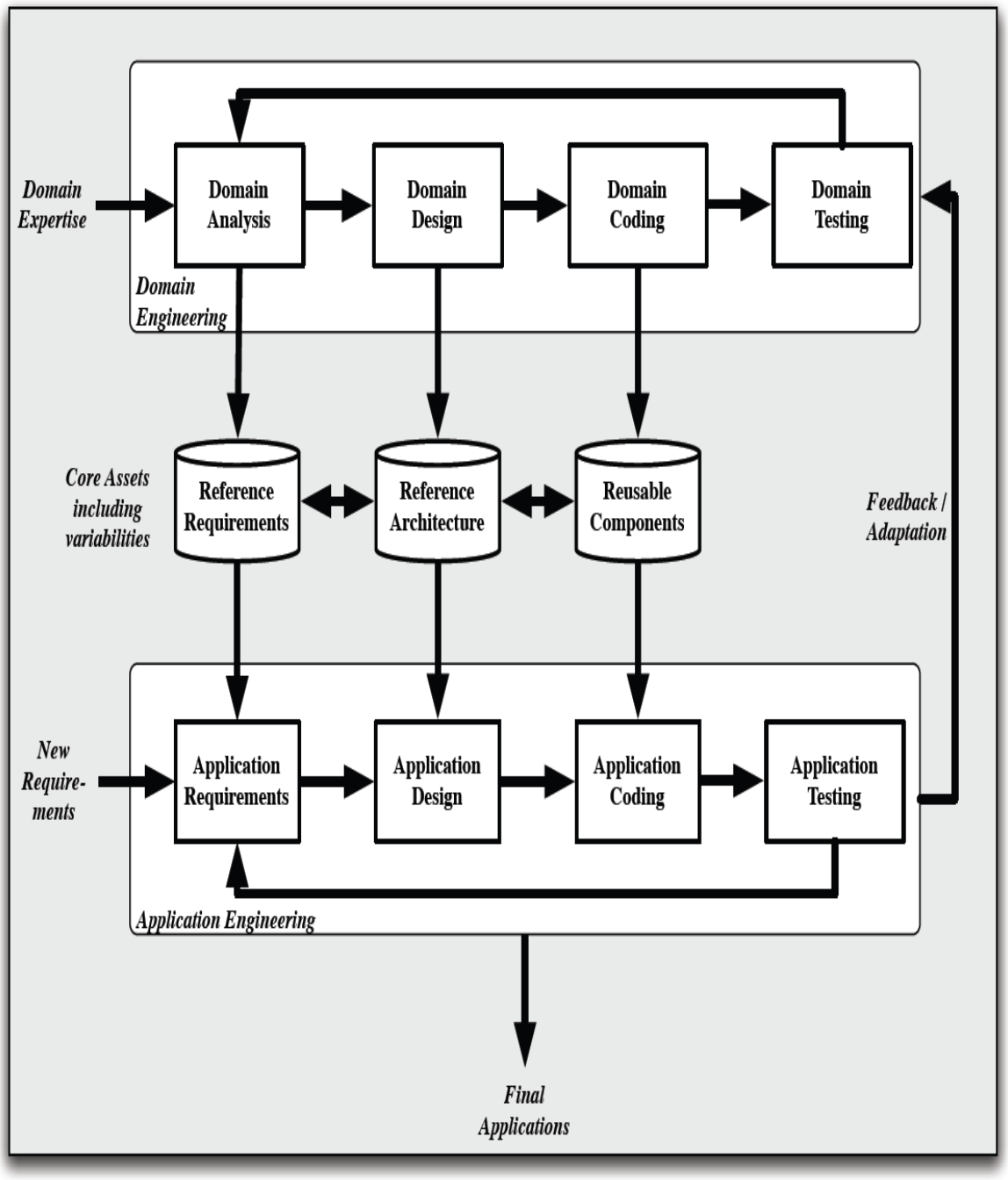

Figure 2.4 : Processus de développement d'une ligne de produits logiciels [1]

2.2.3. Notion de Variabilité et de commonalité

Comme nous lřavons répété plusieurs fois dans ce travail, le concept des lignes de produits puisent leur particularité dans la commonalité et la variabilité. Ces deux notions déterminent une mise en place de lignes de produits conforme aux règles.

La commonalité ou similarité « regroupe « *l'ensemble des hypothèses qui sont vraies pour tous les produits, membres de la ligne de produits* » Cřest un des éléments clés permettant de concevoir des logiciels issus dřune seule famille de produits. En général, ces similarités sont souvent faciles à détecter puisquřil sřagit des éléments de base tels que le volant ou les portières dans une voiture. En dřautres termes, ce sont des composants quřon ne peut pas séparer du produit, des éléments présents dans chaque produit formant une ligne de produits.

La variabilité, elle, « regroupe lřensemble des hypothèses montrant comment les produits membres de la ligne de produits diffèrent »
Nous allons entamer cette sous-partie par la présentation de la caractéristique ou feature en nous basant sur lřexemple de FODA.

2.2.4. Notion de Caractéristique (Feature) : FODA

Par définition, la caractéristique ou Feature est, selon [19], *"tout aspect important et distinctif ou caractéristique visible par les diverses parties prenantes".* La feature est obtenue à partir des analyses de la variabilité menée en phase de lřingénierie de domaine. Elle permet donc de différencier un produit dřun autre. Elle prend forme de diagramme avec plusieurs niveaux comportant la variabilité pour chaque produit (les éléments qui différencie un produit dřun autre) comme lřatteste le diagramme de FODA.

La caractéristique ou « feature » est un ensemble dřéléments logiciels déterminant un contexte particulier. La feature collecte les exigences imposées par les logiciels finaux. Elle a pour but de trouver les exigences réutilisables par les logiciels finaux citées ci-dessus afin de limiter leur nombre.

En temps normal, les features se décomposent en plusieurs sous-features afin dřobtenir une « feature finale » capable de se conformer à un composant logiciel réutilisable.
Exemple de features : le domaine des véhicules :
- Carrosserie
- Moteur Essence,
- Moteur électrique
- Moteur diesel
- Vitres manuelle
- Vitres automatique
- Climatisation

En plus dřavoir défini la notion de caractéristique, [19] a érigé un modèle de caractéristique désormais célèbre et usité partout dans le monde : le diagramme de FODA ou Feature Oriented Domain Analysis. Ce diagramme hiérarchisé a pour objectif de présenter les points communs et les variabilités dans un logiciel. Dans le FODA, on trouve une caractéristique racine qui est la ligne de produits elle-même. Puisque le FODA prend la forme dřun arbre, la caractéristique racine qui est la LPL se divise en plusieurs autres parties ou caractéristiques qui représenteront les commonalités et les variabilités.

Le FODA est la feature la plus utilisée et la plus connue. Il permet également de déterminer les éléments communs et variables dans un produit. Il prend la forme d'un diagramme ressemblant à un « arbre » constitué de nœuds représentant les particularités ou caractéristiques du domaine et d'arcs, une sorte de branches traduisant les relations existant entre les caractéristiques. Dans cette optique, trois catégories de caractéristiques sont visibles dans le FODA :

- La première est celle des caractéristiques obligatoires, c'est-à-dire les composants ou éléments qui doivent impérativement être présents dans chaque produit. Les éléments obligatoires sont les constituants mêmes d'un produit, ils permettent de le définir et de le distinguer des autres. En termes d'exemple, nous pouvons citer un ordinateur qui doit impérativement être muni d'un écran qu'il soit ordinateur portable ou ordinateur de bureau.
  L'écran définit l'ordinateur qui ne pourrait pas fonctionner sans lui, il fait partie des composants essentiels de ce derniers et lui permet de fonctionner correctement et d'être utilisé comme il se doit.

- La seconde est celle des caractéristiques optionnelles qui, comme son nom l'indique, n'est pas obligatoire à tous les produits, c'est-à-dire que ces caractéristiques sont visibles chez certains produits et absentes chez d'autres. Les caractéristiques optionnelles sont des options souvent chargées du confort de l'utilisateur du produit ou destinées à le distinguer des autres, à le valoriser et à le promouvoir.

  Toujours dans l'exemple de l'ordinateur, on pourrait prendre comme option la présence d'un filtre superposé à l'écran de l'ordinateur de bureau. Tous les ordinateurs de bureau ne sont pas munis de ce filtre et on peut s'en passer, il incombe à l'utilisateur de le placer sur l'écran afin de démarquer cet ordinateur des autres de la même série et d'y ajouter donc une fonction supplémentaire qui, ici, est le filtrage des rayons X émanés par l'ordinateur afin de se protéger contre les maladies qu'ils peuvent engendrer (maladies des yeux, etc.).

- La troisième est celle des caractéristiques alternatives qui sont des éléments alternatifs c'est-à-dire que leur présence n'impacte en rien l'utilisation d'un produit ou son fonctionnement. Les caractéristiques alternatives sont donc des caractéristiques de rechange. Dans l'exemple d'un ordinateur de bureau, on pourrait prendre comme composants alternatifs les écrans de l'ordinateur qui pourraient être des écrans plats ou LCD ou des écrans classiques CRT ou LED.

Le FODA est une des méthodes les plus efficaces dans la détection des variabilités et des composants communs dans les produits constituant une ligne de produits spécifique. Il consiste en une analyse approfondie du domaine du produit en vue de dégager les domaines obligatoires, optionnels ou alternatifs. L'exemple illustratif du FODA est le diagramme de FODA pour une voiture. La voiture en question fait partie d'une série ou ligne de produits. De ce fait, elle possède certaines similarités et différences vis-à-vis des autres voitures membres de cette même LdP.

Le FODA identifie la voiture comme étant le domaine ou le produit. En étudiant de près son aspect, ses composants ou ses options, on note qu'elle doit obligatoirement avoir une

carrosserie, un moteur et des vitres. Il sřagit ici des caractéristiques obligatoires quřelle doit posséder. En allant plus en profondeur, on constate, lors de lřanalyse, que la voiture comporte aussi une climatisation, ce qui nřest pas le cas des autres.

La climatisation est alors une caractéristique optionnelle car elle est facultative et nřest présente que chez certaines voitures. Une fois que ces caractéristiques obligatoires et optionnelles auront été identifiées, on procède à lřidentification des caractéristiques alternatives. Dans le cas présent, il peut sřagir du moteur qui peut fonctionner de manière électrique ou qui doit être alimenté par une essence ou qui sera un moteur diesel. Il se peut également que ce soit au niveau des vitres qui peuvent être manuelles ou automatiques.
Pour résumer, le FODA présente donc les caractéristiques dřun produit en dégageant les caractéristiques obligatoires, optionnelles ou alternatives. La figure suivante résume le FODA utilisé pour le cas dřune voiture.

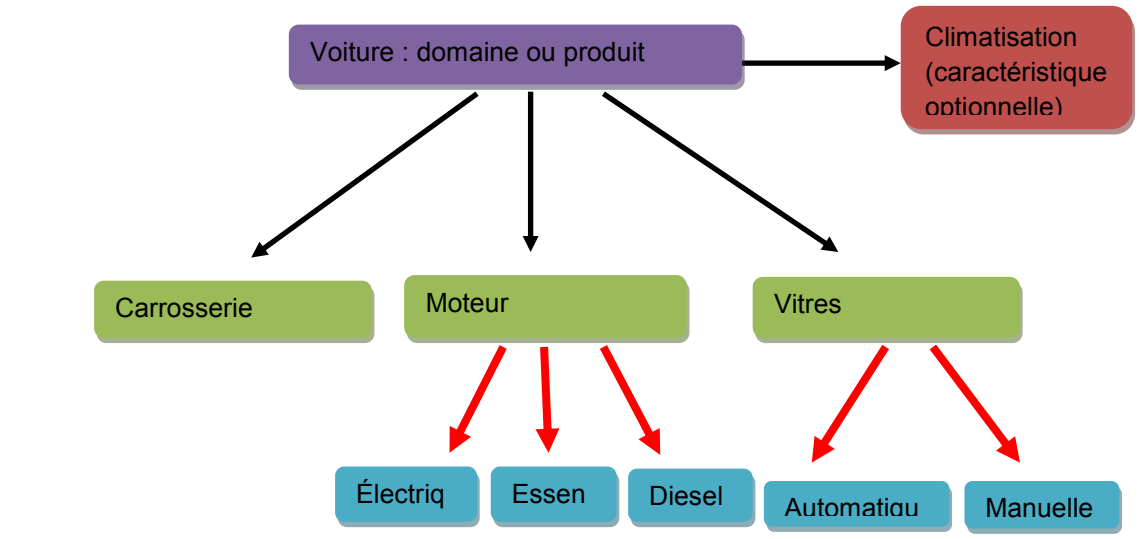

Figure 2.5 : Exemple d'un diagramme de caractéristiques FODA [14].

A part le FODA, nombreuses autres caractéristiques ont également vu le jour : FORM ou Feature Oriented Reuse Method , Feature RSEB et Generative Programming. FORM a la particularité de divise une LPL en plusieurs « couches » permettant de connaitre dřautres points de vue concernant la modélisation de la LPL et de prendre en compte dřautres critères tels que la technologie ou lřenvironnement. La feature RSEB met en exergue la cohésion entre les modèles de caractéristiques et les modèles de conception. Plusieurs autres approches ont succédé lřapproche FODA et ont permis son extension. Le Generative Programming a visé de rendre la conception de la LPL automatique.

La composition dřune caractéristique de lignes de produits connait plusieurs approches. Le tableau que nous dresserons ci-dessous est entièrement retranscrit de [2]. Il présente les différentes approches de composition dřune caractéristique selon plusieurs auteurs.

Approche à base de règles de transformation de modèles

Dans [22] les auteurs ont motivé le besoin de faire évoluer les lignes de produits logiciels et plus particulièrement les modèles de caractéristiques. Dans ce travail, les auteurs proposent dřétendre la notion traditionnelle de transformation pour les lignes de produits logiciels. La transformation de modèles est satisfaite lorsque lřensemble des systèmes valides du modèle de caractéristiques reste le même ou évolue. Pour ce faire, ils proposent un ensemble de règles de transformation qui considèrent les spécificités du modèle de caractéristiques. La figure 2.6 présente un exemple de règle de transformation.

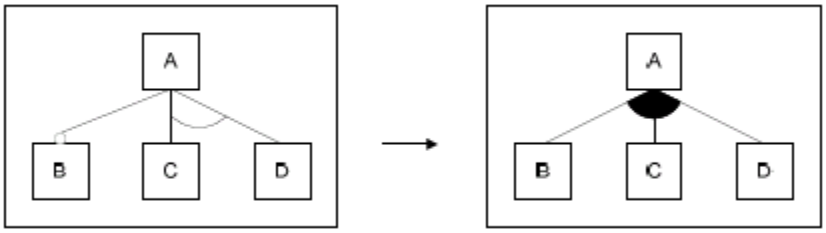

Figure 2.6 : Règle de transformation des variables et xor en or [2]

Cette règle prend en entrée un modèle de caractéristiques incluant la caractéristique variable B et les caractéristiques C et D reliés par une contrainte de type ***xor***. Ceci veut dire que les systèmes valides sont ABC, AC, ABD et AD. La transformation de ce modèle donne un modèle de caractéristiques où les caractéristiques B, C et D sont reliés par une contrainte de type ***or***. Lřensemble des systèmes obtenus à partir de ce modèle inclut lřensemble des systèmes valides obtenus à partir du premier modèle.

La proposition présentée dans ce travail ne se limite pas à la transformation des modèles de caractéristiques. Les auteurs suggèrent aussi lřutilisation des règles de transformation dans la fusion des modèles de caractéristiques. Cependant, les règles de transformation proposées ont aussi besoin dřêtre étendues pour traiter les différents cas rencontrés dans la fusion. Ces règles traitent principalement les contraintes de variabilité sans intérêt particulier à la structure des caractéristiques ou leurs informations de variabilité.

De plus, les règles proposées se limitent à gérer des contraintes de types *and*, *or* et *xor*. Les contraintes transversales de types *implication* et *équivalence* ne sont pas traitées par les règles proposées, malgré leur importance et la fréquence de leur utilisation dans le contexte des lignes de produits. Les règles de fusion de contraintes ne sont pas génériques, elles restent limitées aux types de contraintes spécifiées. Des précisions supplémentaires sont aussi nécessaires pour expliquer les critères de comparaison des caractéristiques à fusionner ainsi que les propriétés sémantiques de la fusion.

Approche à base de règles visuelles de fusion de modèles

Inspirés par le travail [22], dans [23] les auteurs proposent un catalogue de règles visuelles pour la fusion des modèles de caractéristiques. Chaque règle est composée de deux parties ; la partie gauche représente les deux patrons des modèles de caractéristiques à fusionner en entrée (les pré-conditions). La partie droite représente le patron du modèle de caractéristiques résultant de la fusion (post-conditions). La figure 2.7 donne un exemple de ces règles.

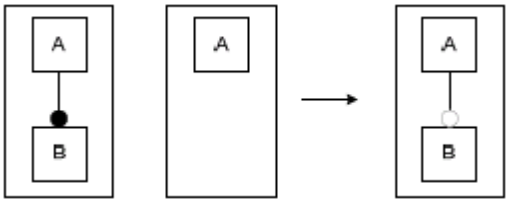

Figure 2.7 : Règle de fusion des caractéristiques [2]

Les règles de fusion proposées permettent d'obtenir un modèle de caractéristiques dont l'ensemble des systèmes valides inclut au moins l'ensemble des systèmes valides obtenus à partir des modèles de caractéristiques en entrée. Ceci implique que le modèle de caractéristiques résultant permet d'obtenir des produits qui ne sont valides pour aucun des modèles d'entrée. La sémantique de la fusion manque de précisions pour les règles proposées. En effet, pour une sémantique donnée de la fusion (par exemple en mode union ou en mode intersection), le catalogue de règles doit être trié ; certaines règles sont maintenues pour la fusion alors que d'autres doivent être mises à jour selon la propriété sémantique choisie.

La règle présentée dans la figure 2.8, montre la fusion de deux patrons de modèles de caractéristiques. Le premier contient deux caractéristiques A et B ayant une contrainte associée de type *implication*. Le deuxième inclut deux caractéristiques A et B ayant une contrainte associée de type *équivalence*. Le résultat de leur fusion donne un modèle de caractéristiques qui inclut les deux caractéristiques variables A et B permettant ainsi toutes les combinaisons possibles de A et B. Pour une fusion en mode union par exemple, cette contrainte permet d'obtenir des systèmes valides qui n'appartiennent à aucun des modèles d'entrée. Le système qui contient la caractéristique A est un système valide obtenu à partir du modèle de fusion. Cependant, ce système n'est valide pour aucun des modèles de caractéristiques en entrée. Cette règle doit alors être mise à jour pour satisfaire la propriété union de la fusion.

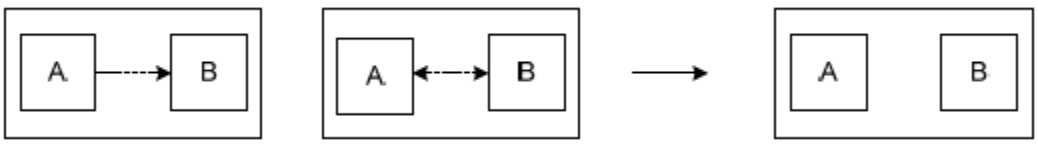

Figure 2.8 : Règle de fusion des caractéristiques transversales [2]

Les auteurs proposent des règles de fusion pour spécifier l'information de variabilité des éléments résultant de la fusion, comme l'exemple de la règle illustrée dans la figure 2.7. Ils proposent aussi un catalogue de règles pour la fusion des contraintes, comme l'exemple dans

la figure 2.8. Cependant, l'énumération de ces règles selon les types de contraintes n'offre pas de généricité et limite la fusion aux types de contraintes identifiées. De plus, le catalogue de règles contient 30 règles qui doivent être vérifiées selon la sémantique de fusion.

Approche à base d'opérateurs de fusion et d'agrégation de modèles

Dans [93] les auteurs proposent un ensemble d'opérateurs pour la composition de modèles de caractéristiques. Nous distinguons plus particulièrement les deux opérateurs suivants :

- Opérateur de fusion : cet opérateur permet de fusionner des parties communes de modèles de caractéristiques pour obtenir un modèle de caractéristiques intègre. L'opérateur de fusion utilise le nom des caractéristiques comme critère de correspondance. Les caractéristiques ayant le même nom sont alors fusionnées. Cependant, ce critère reste étroitement lié au contexte de modèle de caractéristiques. Il ne présente pas de problèmes de fusion pour ce type de modèles qui est généralement utilisé pour représenter les lignes de produits à un niveau d'abstraction élevé.

Ceci est loin d'être similaire pour des modèles de nature différente comme par exemple les modèles UML et qui représentent un niveau d'abstraction moins élevé que les modèles de caractéristiques. Comparer les éléments de modèles selon un seul critère qui est leurs noms est relativement facile à implémenter comme critère de correspondance. Cependant, ce critère peut être très permissif dans certains cas. Par exemple, comparer les attributs de classes en se basant sur leurs noms comme critère de correspondance peut impliquer des problèmes lors de la fusion si les types associés à ces attributs sont incompatibles.

La figure 2.9 montre l'exemple des caractéristiques *Sensing* appartenant à deux modèles de caractéristiques différents. Selon la proposition [93], deux éléments des modèles peuvent être fusionnés s'ils ont le même nom. Nous déduisons donc que *Sensing* de modèle1 correspond à *Sensing* de modèle 2. De même, *infraredDetector* et *volumetricDetector* de modèle1 correspondent respectivement à *infraredDetector* et *volumetricDetector* de modèle2. Par conséquent les éléments *Sensing*, *infraredDetector* et *volumetricDetector* de modèle1 peuvent être fusionnés avec les éléments respectifs *Transport*, *infraredDetector* et *volumetric Detector* de modèle2.

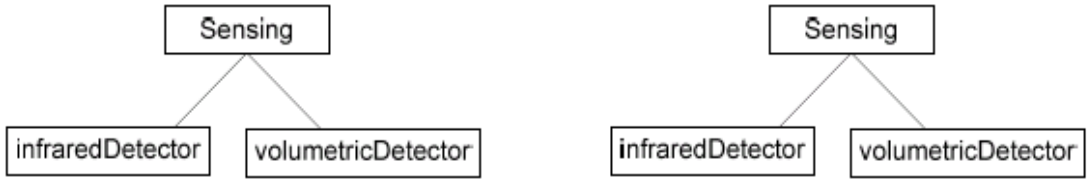

Figure 2.9 : Exemple de correspondance des caractéristiques Sensing [2]

Nous avons ensuite représenté les mêmes éléments dans un modèle UML illustré dans la figure 2.9. En utilisant le même critère de comparaison, nous constatons que les classes *Sensing* de modèle1 et modèle 2 correspondent (même nom). Les attributs *volumetricDetector* de modèle1 et modèle2 correspondent aussi et possèdent des types compatibles, contrairement aux attributs *infraredDetector* qui correspondent selon le critère du nom mais qui

ont des types incompatibles (*InfraredDetector* et *Sensor*).

Dans ce cas, la fusion des éléments *infraredDetector* peut engendrer des problèmes à cause de lřincompatibilité de leurs types. Les caractéristiques dans chaque modèle de caractéristiques sont reliées par des contraintes permettant ainsi de déterminer lřensemble des systèmes valides. Dans lřexemple de la figure 2.10, les caractéristiques *InfraredDetector* et *volumetricDetector* sont liées par une contrainte de type ***or*** comme le montre le modèle de la figue 2.10(a), alors que les mêmes caractéristiques sont liées par une contrainte de type ***xor*** dans le modèle de la figure 2.10(b).

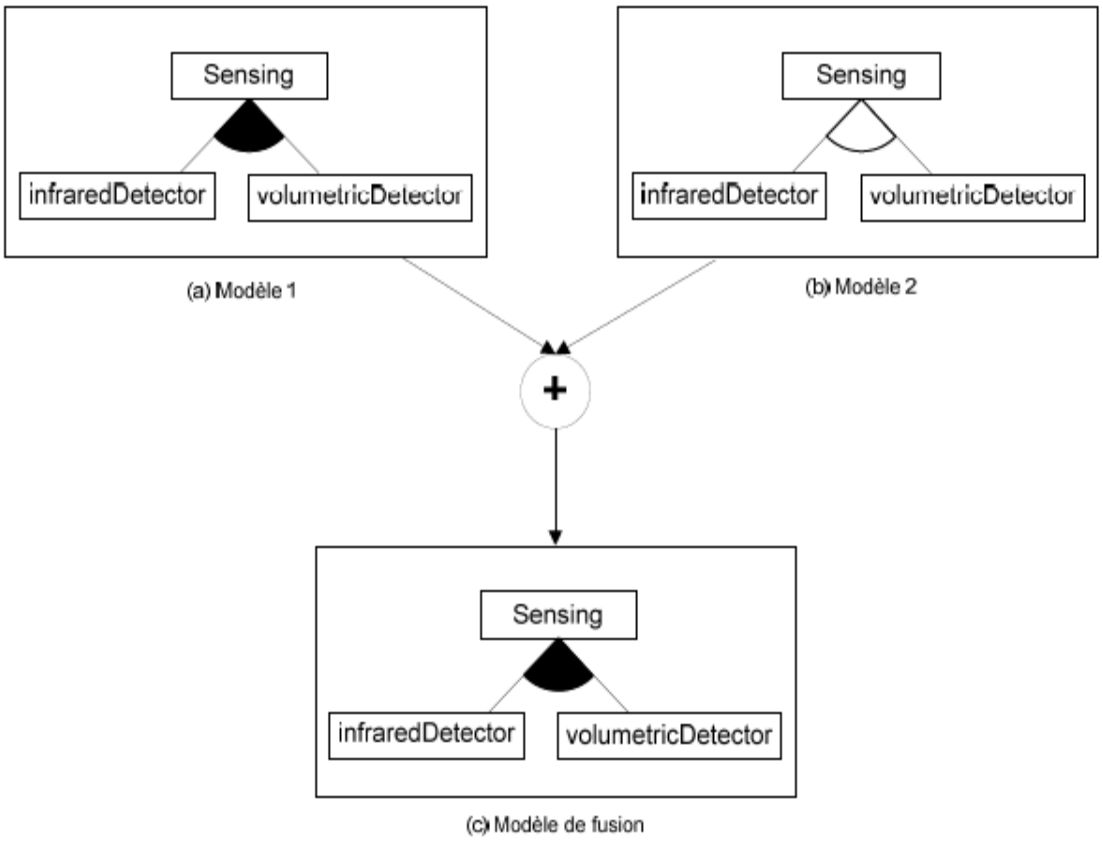

Figure 2.10 : Exemple de fusion des contraintes [2]

Pour décider de la contrainte résultante de la fusion des deux contraintes précédentes, les auteurs proposent un ensemble de règles de fusion. Ces règles calculent à partir des contraintes des modèles fusionnés les nouvelles contraintes reliant les caractéristiques du modèle de fusion. Il sřagit de calculer la contrainte prédominante des deux contraintes à fusionner selon la sémantique de fusion choisie (c'est-à-dire union ou intersection). Par exemple dans la figure 2.10. (a) les contraintes à fusionner sont de types respectifs ***or*** et ***xor***. La règle de fusion dans ce cas identifie la contrainte résultante de type ***or*** comme le montre le modèle résultant de la figure 2.10. (c).

Sensing Infrared Detector: Infrared Detector
Volumetric Detector: Volumetric Detector

Bien que les règles proposées permettent de fusionner les contraintes, elles restent limitées aux contraintes de types *and*, *or* et *xor*. Les contraintes transversales qui existent entre les

caractéristiques ne sont pas traitées. Par exemple les contraintes de type *implication* ne sont pas gérées durant la fusion. Ce type de contraintes spécifie que la présence dřune caractéristique dans un système valide implique la présence dřune ou plusieurs autres caractéristiques dans le même système valide.

De même pour les contraintes de type *équivalence* qui ne sont pas traitées durant la fusion. Ce type de contraintes spécifie la coprésence de toutes les caractéristiques dans un même système valide, si lřune dřentre elles est présente dans ce système et inversement. Contrairement à la fusion des contraintes, la fusion des caractéristiques qui appartiennent aux modèles dřentrée nřest pas réalisée de manière explicite. Elle est incluse dans le traitement de fusion des contraintes sans précisions supplémentaires.

- ***Opérateur d'agrégation :*** cet opérateur est utilisé pour produire de nouveaux modèles de caractéristiques en reliant dřautres modèles existants par des contraintes transversales. La proposition dřagrégation se limite à rassembler les arbres de caractéristiques en entrée sous une nouvelle caractéristique racine du modèle dřagrégation résultant. Dans lřexemple de la figure 2.11, lřagrégation des modèles de caractéristiques InHomeSecurity et Alarm se résume simplement à créer une nouvelle racine source, Security, qui regroupe les deux modèles dřentrée en regroupant leurs caractéristiques racines.

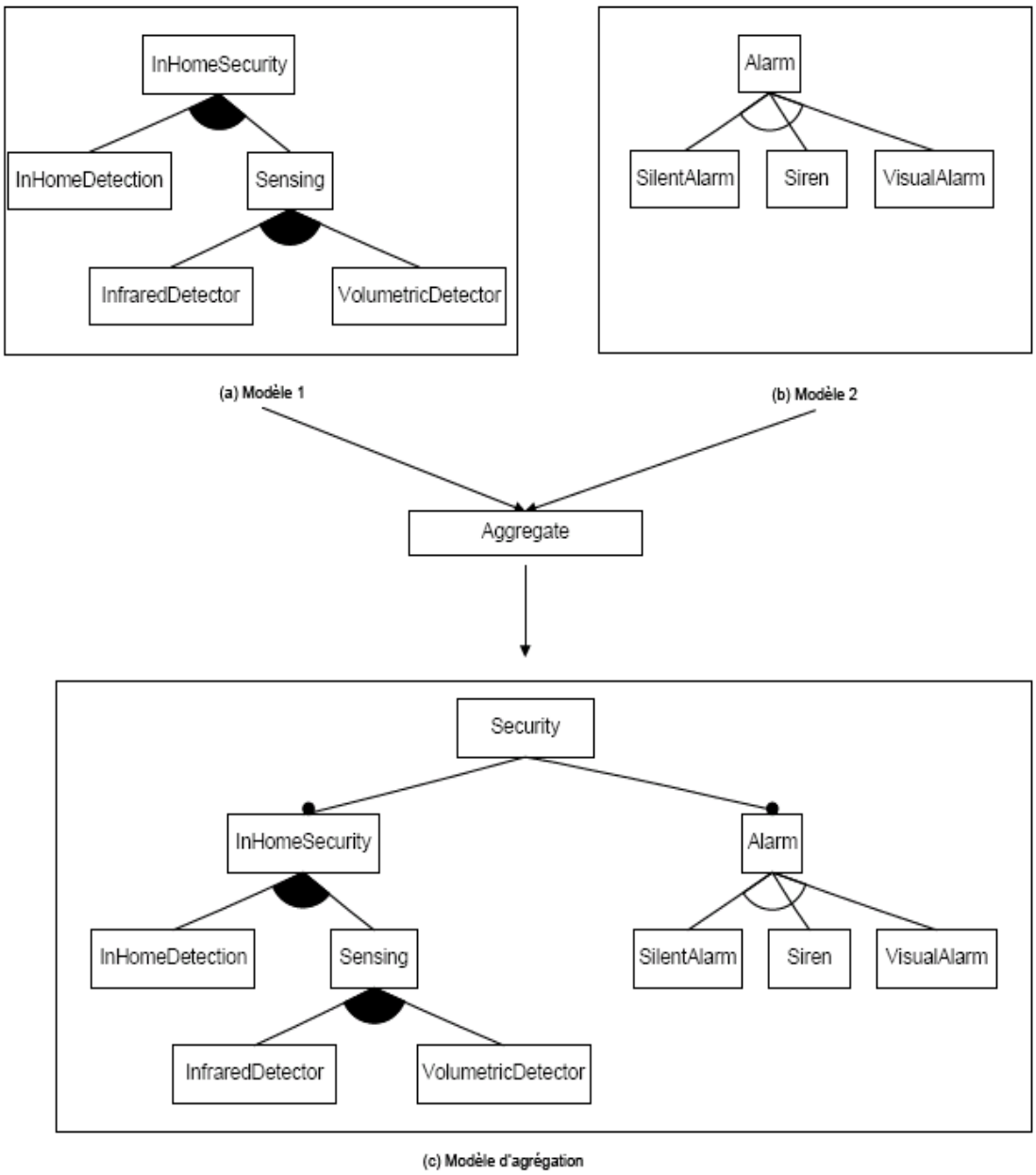

Figure 2.11 : Exemple d'agrégation de modèle de caractéristiques [2]

Dans le contexte des modèles de caractéristiques, on ne soulève pas de questions sur la faisabilité de connecter des caractéristiques variables appartenant aux modèles dont ils réalisent lřagrégation. De même pour la gestion de la variabilité de ces caractéristiques ainsi que les contraintes qui permettent de les relier. Toutefois, ces questions représentent des points dřinterrogation dans le cas dřutilisation de lřopérateur dřagrégation pour dřautres modèles comme par exemple des modèles UML. La compatibilité entre les éléments des modèles dřentrée, la gestion de la variabilité des éléments ainsi que les contraintes sont parmi les principaux problèmes à résoudre.

Approche selon la superposition de codes

Dans [94] [95] les auteurs proposent d'organiser les éléments structurels des caractéristiques en utilisant le mécanisme de FST (Feature Structure Tree). La fusion des caractéristiques revient alors à fusionner leurs FSTs correspondantes. La proposition de la fusion se base sur le mécanisme de superposition. La superposition permet la fusion des fragments de codes correspondants aux différentes caractéristiques. C'est un processus de composition récursive d'arbres par composition des noeuds qui se situent au même niveau en partant de la racine.

Les figures 2.12 et 2.13 montrent les modèles de caractéristiques et les fragments de code qui leurs sont correspondent respectivement. La figure 2.12 représente l'arbre de caractéristiques de la ligne de produits *InHomeSecurity* du fragment de code à gauche dans la figure. La ligne de produits inclut une détection intérieure représentée par la caractéristique *InHomeDetection* et la caractéristique de détection de mouvement *Sensing* qui inclut un détecteur infrarouge *InfraredDetector* et un détecteur volumétrique *VolumetricDetector*. Les différentes caractéristiques représentent des éléments du code correspondant.

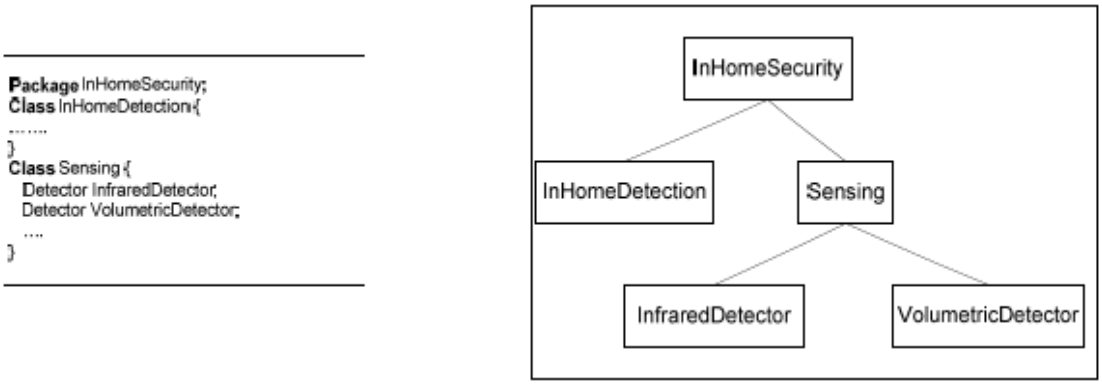

Figure 2.12 : Code et modèle de caractéristiques de InHomeSecurity 1 [2]

De même, la figure 2.13 représente l'arbre de caractéristiques de la ligne de produits *InHomeSecurity* du bout de code à gauche dans la figure. La ligne de produits inclut une détection intérieure représentée par la caractéristique *InHomeDetection* et de la caractéristique de détection de mouvement *Sensing* qui inclut un détecteur infrarouge *InfraredDetector* et un détecteur de 160 degré *160DegreeDetector*. Les différentes caractéristiques représentent des éléments du code correspondant.

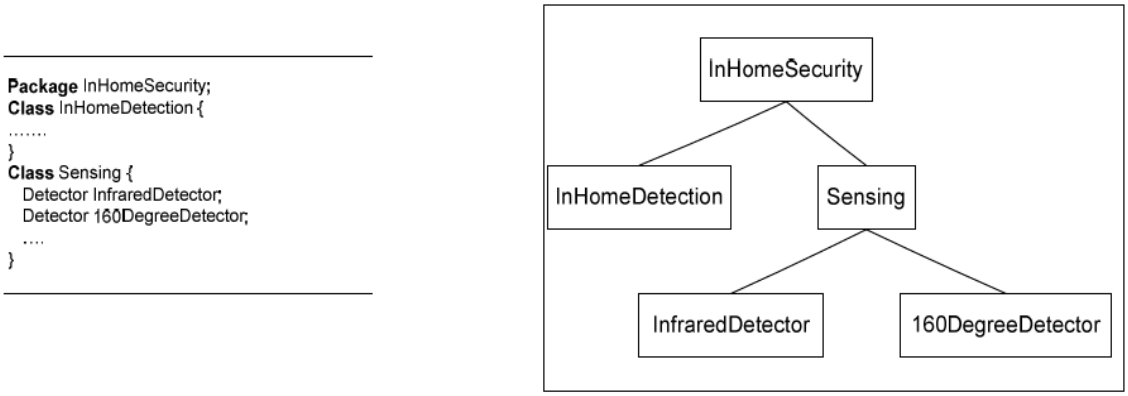

Figure 2.13 : Code et modèle de caractéristiques de InHomeSecurity 2 [2]

En s'appuyant sur le mécanisme de superposition, la composition des fragments de code de la figure 2.12 et la figure 2.13 implique la composition par superposition des modèles de caractéristiques correspondants. La figure 2.14 montre le résultat de cette composition au niveau code et au niveau modèle de caractéristiques.

Un modèle de caractéristiques est « une hiérarchie de caractéristiques avec variabilité » [24]. Il peut donc être représenté par un FST avec variabilité. Cependant, la proposition ne prend pas en considération la variabilité des caractéristiques qui appartiennent aux modèles à fusionner.

L'information de variabilité n'est donc pas gérée durant la fusion. De même, la proposition ne traite pas la fusion des contraintes associées aux modèles d'entrée. Les auteurs proposent de fusionner des modèles UML non annotés par la variabilité en utilisant le mécanisme de superposition. Leur critère de correspondance entre les éléments à fusionner se base sur le nom, le type ainsi que la position dans le modèle.

Cependant, ce critère représente une faiblesse dans certains cas. Prenons par exemple le modèle de structure composite où des classes composites incluent des propriétés de natures différentes comme les ports, parts et connecteurs par exemple. Nous reprenons l'exemple de *Sensing* présenté dans la figure 2.7. Supposons que les modèles à fusionner consistent en deux classes composites nommées *Sensing* dont chacune contient deux propriétés *InfraredDetector* et *volumetricDetector*. Le premier modèle représente la propriété *volumetricDetector* par une *part* typée par une classe *VolumetricDetector*. Le deuxième modèle représente la même propriété *volumetricDetector* par un *port* typé par une classe *VolumetricDetector*. En se basant sur le critère de correspondance mentionné, c'est-à-dire le nom, le type et la position des éléments, les propriétés *volumetricDetector* des deux modèles correspondent bien qu'elles soient représentées différemment (*part* et *port*). Ceci implique des problèmes dans leur fusion.

TABLEAU 2.2 : APPROCHES SUR LA COMPOSITION DES CARACTERISTIQUES [2].

Une fois que nous aurons fait le tour des features, nous allons nous focaliser un peu plus sur la notion de point de variation.

2.2.5. Notion de Point de variation

La variabilité pose souvent problème aux concepteurs souhaitant développer une ligne de produits. En effet, il apparait plus compliqué de trouver des points de différenciation entre plusieurs produits issus d'une même LPL. Cette variabilité est étudiée et jugée avant la livraison du logiciel, c'est-à-dire qu'elle doit être définie et identifiée au préalable.

Pour pouvoir obtenir une bonne variabilité, il faut savoir rivaliser d'inventivité et trouver des idées ou des concepts novateurs à un grand nombre afin de les répartir entre les différents logiciels, permettant à leurs utilisateurs de détecter ce qui rend leur logiciel différent de celui des autres, dès lors qu'ils proviennent de la même LPL.

La variabilité dispose de deux dimensions :
- la dimension espace ou variabilité spatiale : variation entre plusieurs produits de la même famille

- la dimension temps ou variabilité temporelle : variation dans le temps des produits dřune version à une autre.
Selon [25] la variabilité temporelle est "lřexistence de différentes versions dřun artéfact qui sont valides à des moments différents" et la variabilité spatiale représente "lřexistence dřun artéfact sous différentes formes en même temps". La variabilité spatiale est la plus côtoyée et la plus présente dans la conception des lignes de produits.

A noter que le principe de la réutilisation suppose quřun logiciel est utilisé plusieurs fois dans la conception de produits de la même famille, dřoù lřimportance de la variabilité qui peut prouver quřil sřagit bien de deux produits différents avec plusieurs similitudes. On peut prendre comme exemple de variabilité les options supplémentaires insérées dans les portables qui les différencient des autres.
La variabilité est également étudiée dans lřingénierie de domaine. Dans [26] les autres la définissent comme suit : « Un point de variation identifie un ou plusieurs emplacements auxquels la variation peut se produire » En dřautres termes, ce sont les différences présentes chez chaque logiciel dans une même ligne de produits. Il est constitué par ce quřon appelle les « variants ».

On note souvent la présence des variabilités ou points de variation lorsque le produit en question détient plusieurs caractéristiques qui permettent de ne pas le confondre avec les autres produits issus de la même LPL que lui. Ces éléments peuvent être des petits détails tels que les options ou les fonctionnalités supplémentaires incluses dans les téléphones portables.

Lorsquřune option est présente chez un modèle mais absente dans dřautres alors quřils proviennent dřune seule et même famille de produits, on dit quřil détient une variabilité. Cette variabilité est définie durant la phase dřingénierie de domaine et est définie par Weiss et Lai comme étant "une hypothèse sur la façon dont les membres dřune famille peuvent se différencier entre eux". La variabilité représente les besoins spécifiques des utilisateurs, cřest-à-dire quřen différenciant chaque modèle des autres, elle permet de satisfaire les besoins des usagers selon sa nature ou son type. A contrario, les similarités prouvent lřappartenance dřun produit à une ligne de produit avec des caractéristiques communes bien identifiées.

La conception dřune ligne de produits repose en grade partie sur a variabilité qui permet aux produits dřêtre similaires tout en restant différents. Elle représente dřailleurs une contrainte majeure dans le façonnement de la LPL puisquřelle doit être appréhendée durant la phase de modélisation et nécessite une gestion qui nřest assurée que par les professionnels en la matière.
Après avoir fait le tour de la question concernant les points de variation, les commonalités et les caractéristiques propres aux lignes de produits, une brève présentation des objectifs de la mise en place de ces dernières sřimpose.

2.3. Les objectifs de la mise en place des lignes de produit

Plusieurs entreprises ou institutions attestent de lřefficacité des lignes de produits, surtout en termes de production et dřatteinte dřobjectif. En effet, les objectifs que les lignes de produits se fixent sont toujours en accord avec ceux de ces entreprises.

Les lignes de produits ont différents objectifs, les plus importants sont ceux de minimiser les coûts de production des logiciels et dřaugmenter leur production en se servant dřanciens modèles. Grâce à ce concept, plusieurs grandes entreprises ont déjà su remonter la pente face aux périodes de crise en adoptant les lignes de produits et le concept a été adopté tant par les banques que par certains gouvernements tels que le Canada.

La ligne de produits vise également la réutilisation de certains éléments et évitent donc les gaspillages inutiles. Elles sont mises en place afin dřaccroître le taux de bénéfice des entreprises et de faciliter leur processus de production de par son fonctionnement à la chaine.
Dřautres objectifs peuvent encore être sur la ligne de mire de ces LPL, mais pour lřheure, nous allons parler des avantages et des inconvénients de ces dernières.

2.4. Avantages et inconvénients des lignes de produit

Les lignes de produits logiciels ont une solution proposée par le génie logiciel. Elles sont connues pour leur efficacité mais présentent également quelques failles que nous allons discuter ci-dessous.

2.4.1. Avantages sur le plan économique

Les lignes de produits offrent des avantages dans tous les domaines : au niveau de la production, de la concurrence, de lřéconomie, etc. Ceux que nous allons relater prochainement concernent exclusivement les bienfaits que la ligne de produits peut apporter sur le plan économique.

- **Meilleure capitalisation de la connaissance en production**

Les lignes de produits ont été adoptées dans les entreprises peu après leur apparition. Elles interviennent directement dans le cœur dřactivité de ces dernières en touchant à la part de production et dřentrée de revenu. Leur visée est dřabord purement économique avec une baisse considérable du montant de production. De ce fait, les entreprises qui les adoptent profitent donc dřune entrée fructueuse dřargent tout en économisant et en réduisant les sorties dřargent en matière de production industrielle.
En même temps, les lignes de produits représentent une solution et une alternative à long terme permettant de créer divers produits ou logiciels en même temps. Ce qui permettra dřavoir de nombreux logiciels avec les mêmes propriétés et des variabilités en un temps calculé et permettra de calculer le rendement moyen en logiciels et ainsi de rendre régulière la production. Ceci peut faire éviter les crises et les contraintes liées à lřinflation des coûts des matières premières. Cependant produire les mêmes produits lors des baisses du marché ou lorsque la tendance varie et que les produits en question ne sont plus en vogue risque de faire perdre beaucoup de bénéfices. Il faut alors analyser constamment le marché.

- **Amélioration de la qualité, diminution des coûts de conception et de production**

Les lignes de produits présentent des avantages combinés tels que lřamélioration de la qualité, la diminution des coûts de conception et de production. En effet, lors de la conception dřune

LPL, les critères relatifs à la commonalité et à la variabilité sont scrupuleusement surveillés, dřoù la qualité toujours supérieure des produits proposés.

Et bien quřil soit produit en masse, ce produit aura la chance dřêtre de qualité supérieure vu les séries de contrôle quřil subira en vue dřattester de cette qualité.
Les entreprises adoptant les approches par la ligne de produits sont surtout avantagées par les multiples avantages que ce type de procédé leur permet, surtout en termes dřéconomie. En effet, les lignes de produits ont un mode de fonctionnement continu, cřest-à-dire que les entreprises réalisent une production à la chaine, mais de manière plus sophistiquée et évoluée. Cette production à la chaine évoluée se caractérise surtout par un meilleur suivi de chaque produit par les ingénieries de domaine et dřapplication.

Les produits sont alors conçus de manière innovante avec des contrôles et des suivis réguliers et optimaux. En général, les productions massives manquent en qualité, ce qui nřest pas le cas des produits membres dřune ligne de produits.
Le processus de réutilisation sur lequel repose toute conception de ligne de produits lui permet également de favoriser une réduction du coût de production du fait que les matières et ressources requises pour la production diminuent avec les composants réutilisables ou assets qui font la plus grosse part du travail. De ce fait, les moyens financiers destinés à la production sont de moins en moins sollicités, alors que les produits fabriqués sont plus nombreux et plus qualitatifs quřauparavant. Dřoù la nécessité de suivre de plus près lřapproche par les lignes de produits et de sřen servir pour faire fructifier la production et lřoptimiser, avec un coût de moins en moins onéreux.

- **Soutien à l'innovation**

La ligne de produits est également à vocation novatrice, cřest-à-dire quřelle favorise, par le désir incessant de créer des modèles uniques et similaires en même temps, lřesprit de création et dřinvention. Dans le but de veiller à ce que tous les produits ne soient pas complètement identiques, leurs concepteurs intègrent, par exemple, des fonctionnalités de plus en plus révolutionnaires aux matériels et outils quřils conçoivent. On peut citer en exemple les systèmes de reconnaissance qui deviennent de plus en plus performants et requièrent une reconnaissance digitale à lřaide dřune empreinte de la main, dřun scan des yeux ou dřune reconnaissance vocale.

Les lignes de produits sont donc constamment partenaires de nouvelles idées et de nouvelles créations, rendant chaque produit conçu différent des autres par les options quřil détient. Ce sont ces options qui suscitent le soutien à lřinnovation car elles sont toujours performantes afin dřéviter de se laisser battre par les concurrents, de fidéliser les clients et de répondre à leurs attentes de plus en plus compliquées et ardentes.

Il convient de souligner dans cette sous-partie que pour mieux déjouer la concurrence et se rapprocher de la clientèle en présentant des concepts, des idées ou des façons de voir novateurs, les acteurs dans la conception des lignes de produits doivent avoir les compétences requises à ces tâches.

- **Réduction du délai de mise sur le marché**

Le délai de mise sur le marché (ou time-to-market) est le terme caractérisant le temps quřil faut pour concevoir et réaliser un projet avant de le lancer sur le marché. Cřest donc une extension

du temps de production. Le délai de mise sur le marché est un des critères de sélection dans la production puisquřil impacte fortement sur les bénéfices et revenus que la vente dřun produit peut générer. En dřautres termes, savoir gérer le délai de mise sur le marché permet aux entreprises dřaccroître la rentabilité dřun produit et de gagner une longueur dřavance par rapport aux concurrents.

La ligne de produits permet de réduire considérablement le délai de mise sur le marché puisquřelle favorise la production en groupe dřun produit, cřest-à-dire quřon ne se focalise plus sur la conception dřun seul produit pendant une période donnée pour ensuite concevoir un autre produit avec les mêmes caractéristiques que lui selon le même délai imparti. La ligne de produits réduit le temps de production dřune gamme de produits et permet dřéconomiser ce temps au profit dřune mise en marché plus tôt que si on avait conçu les produits un à un.

Une fois ces avantages économiques des lignes de produits relatés, nous allons maintenant nous concentrer sur les avantages globaux de la mise en application de lřapproche par les lignes de produits.

2.4.2. Avantages sur d'autres plans

Dans cette sous-partie, nous allons évoquer brièvement les avantages que les lignes de produits peuvent procurer en dehors des bienfaits économiques.

- **Favorisation des découvertes et innovations**

Comme nous lřavons déjà mentionné un peu plus haut, les lignes de produits sont favorables aux découvertes et aux innovations du fait quřelles incitent à la création et aux idées nouvelles grâce aux variabilités qui doivent figurer dans chaque produit. Pour illustrer notre hypothèse, prenons lřexemple des téléphones mobiles Nokia qui, dřannée à lřautre, intègrent une langue nouvelle faisant passer le nombre de langues disponibles dans ces petits appareils à plus de 60 à lřheure actuelle.

De ce fait, leurs utilisateurs dans des pays spécifiques peuvent donc se servir de leurs portables en utilisant leur propre langue au lie dřêtre obligés de se servir de la langue standard qui est lřanglais.

En outre, on peut aussi prendre lřexemple des innovations et découvertes en matière de téléphonie mobile comme la possibilité de recharger leur batterie à lřénergie solaire, leur faculté à indiquer la géo localisation des utilisateurs grâce à leur fonctionnalité GPS, au développement de plusieurs applications telles que Facebook, Instagram ou Whatsapp qui sont des réseaux sociaux permettant aux gens du monde entier de se communiquer, de sřéchanger des informations, des photos, et de se parler via un téléphone mobile.

- **Optimisation de l'effort de développement et de test par rapport à la diversité de l'offre de produits**

Les lignes de produits logiciels sont également réputés pour leur faculté à accentuer, voire même à optimiser lřeffort de développement et de test par rapport à la diversité de lřoffre de produits. Chaque produit membre dřune ligne de produits est donc certifié de qualité puisque des tests effectués en ingénierie de domaine et en ingénierie dřapplication permettent de

vérifier leurs compétences, de détecter des anomalies ou dřéventuels défauts de construction et dřy remédier avant le lancement sur le marché.

En dřautres mots, puisque les produits constitutifs de la LPL sont toujours en grand nombre et quřils sont presque identiques, ils requièrent alors une vérification plus minutieuse et des tests successifs et répétés afin de garantir la qualité de chacun dřentre eux.

- **Optimisation de la gestion des défauts en permettant que la correction d'un défaut constaté sur un produit bénéficie aux produits « similaires »**

De même, on remarque que la similarité des produits membres dřune LPL renforce la gestion des défauts en permettant que la correction dřun défaut sur un produit bénéficie aux produits similaires. Chaque produit issu dřune même LPL est donc soumis aux mêmes tests et à la même vérification afin de garantir sa conformité aux autres et de corriger ses défauts par rapport à ses « semblables ».

Dans cette optique, on note plus dřeffort de la part des entreprises dans la correction des défauts relatifs à chaque produit du fait quelle sont soucieuses de se démarquer des concurrents et de garder une image positive dřelles-mêmes aux yeux de la clientèle.

Nous avons vu dans cette sous-section que les lignes de produits ont un effet positif sur la production, la mise en marché, la vente, etc. dřun produit fabriqué en plusieurs modèles détenant chacun ses particularités et ses points communs. Malgré lřexistence des ces avantages indéniables, les LPL ont aussi des points faibles et plusieurs contraintes et besoins influent négativement sur leur mise en place.

2.5. Les contraintes et besoins liés à sa mise en place

2.5.1. Les contraintes liées à la définition d'une architecture globale au niveau du domaine

Les lignes de produits ont un niveau dřexigence élevé en ce qui concerne lřarchitecture globale. En général, cette architecture doit être la même pour tous les produits. Il sřagit de lřarchitecture générique qui sera la base de toutes les architectures de tous les produits membres dřune seule et même ligne de produits. Sa mise en place pose souvent problème pour les concepteurs de la ligne de produits puisque les similarités entre les produits sont plutôt faciles à établir que les variabilités.

En effet, il faut savoir juger dřune fonction qui sera bien accueillie par le public, qui devra être différente de celles des concurrents et qui devra surtout être novatrice et propre au produit. En général, les complications majeures apparaissent souvent au niveau de la conception de cette architecture globale rendant la LPL difficile à mettre en place.

2.5.2. Les contraintes au niveau du pilotage du domaine pour maintenir la cohérence des produits

Il en est de même pour le pilotage du domaine pour maintenir la cohérence des produits. Les contraintes à ce niveau sont surtout liées à la modélisation des variabilités, une des principales difficultés rencontrées lors d la modélisation, de la composition et de la conception des modèles membres dřune même ligne de produit. Ces contraintes sont surtout identifiées dans les variabilités spatiales.
Les contraintes liées à lřingénierie de domaine relèvent du souci de trouver des éléments ou des options qui pourraient différencier chaque produit des autres. Cette contrainte est majeure dans la mesure où toute la conception même de la LPL repose sur la faculté même du concepteur à dénicher des points de variation et des éléments différents à insérer dans chaque modèle.
Après avoir identifié les contraintes liées à la mise en place des LdP, nous allons maintenant nous tourner vers trois exemples dřoutils de gestion de lignes de produits : Requitim, Gears et Pure Variants.

2.6. Les outils de gestion de lignes de produits

2.6.1. Requitim (Cortim) ou REquirements Management Toolkit by CorTIM

Requitim est un logiciel développé par CORTIM. Il permet de faciliter la gestion des exigences dans la conception des lignes de produits en se basant sur un processus bien défini. *Les outils REQUITIM sont des add-ons du logiciel de gestion des exigences Rational DOORS®, et sont indépendant des uns des autres.* (site officiel http://www.requitim.com/).
Requitim dispose de deux outils principaux :

- Data Model Editor et
- Requirements Management.

Le Data Model Editor permet de voir la globalité des données dřun projet avec des tâches spécifiques :

- Identifier si les modules DOORS® sont reliés par le bon type de lien et dans le bon sens
- Filtrer la visualisation sur un type de lien
- Ouvrir un module DOORS® depuis lřinterface du « Data Model Editor »
- Détecter la présence de liens du type « DOORS Link »
- Maîtriser la complexité dřun projet DOORS® (http://www.requitim.com/?mod=Data_Model_Editor)

La figure ci-dessous est un « aperçu des compétences de ce logiciel » qui, placé sur un module, est capable de visionner tout lřaspect de lřenvironnement qi lřentoure et dřen définir chaque détail.

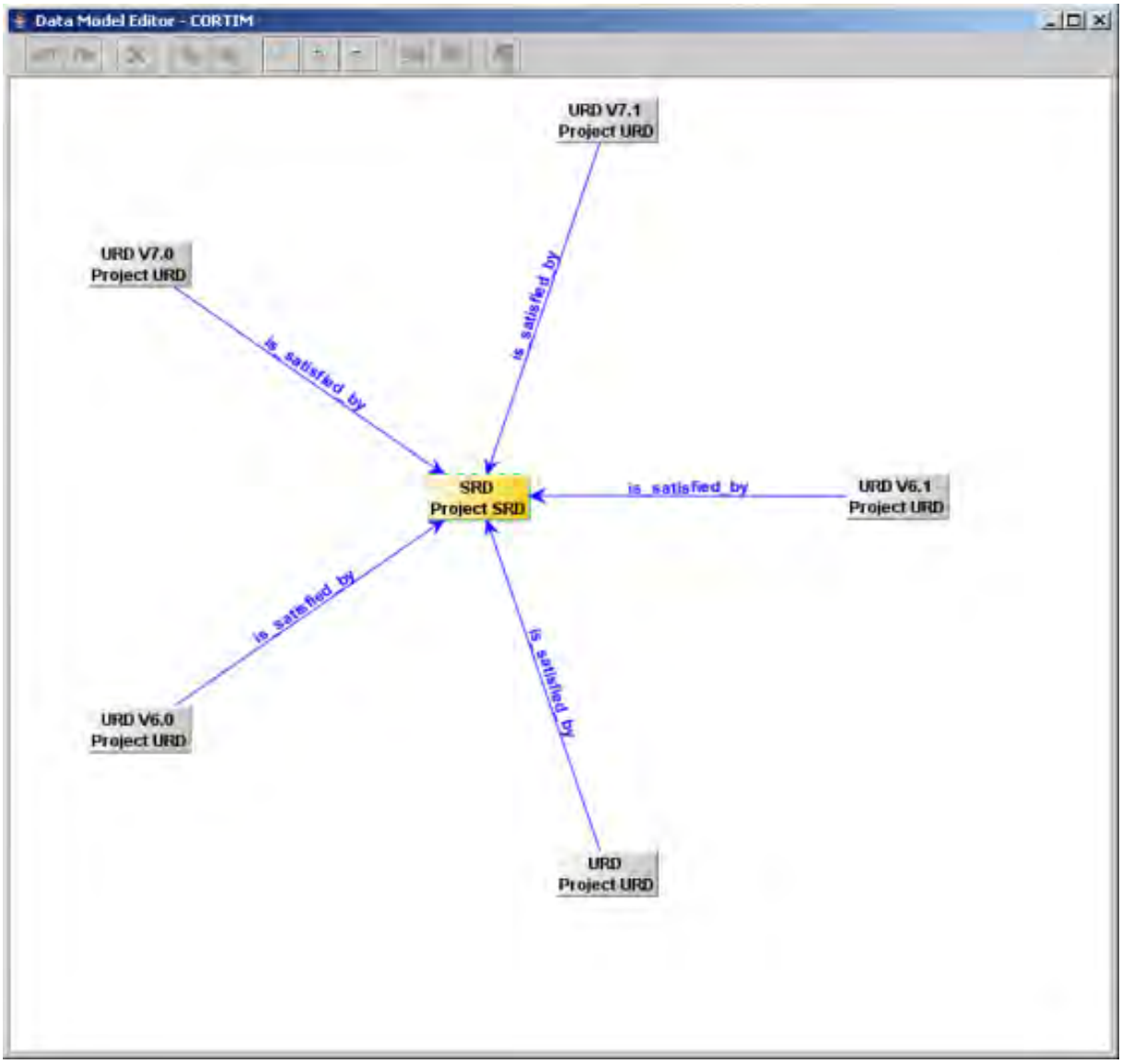

Figure 2.14 : Source : site officiel Requitim :

http://www.requitim.com/?mod=Data_Model_Editor

Le « Requirements Management », tout comme le « Data Model Editor », est un outil complémentaire à DOORS®. Son rôle dans la gestion de la ligne de produits est de gérer les exigences en termes dřévolution et de circuit dřapprobation en soutenant le processus mis en place à cette fin durant la conception dřun modèle. Cet outil facilite le travail du concepteur qui est alors exempté de tâches complexes et longues telles que la *numérotation des exigences, le changement de maturité ou le versionnement.* (http://www.requitim.com/?mod=Requirements_Management)
Requitim est aussi efficace grâce à sa capacité à manipuler les exigences, son rôle principal. La figure suivante présente les actions quřil utilise pour optimiser la manipulation des exigences :

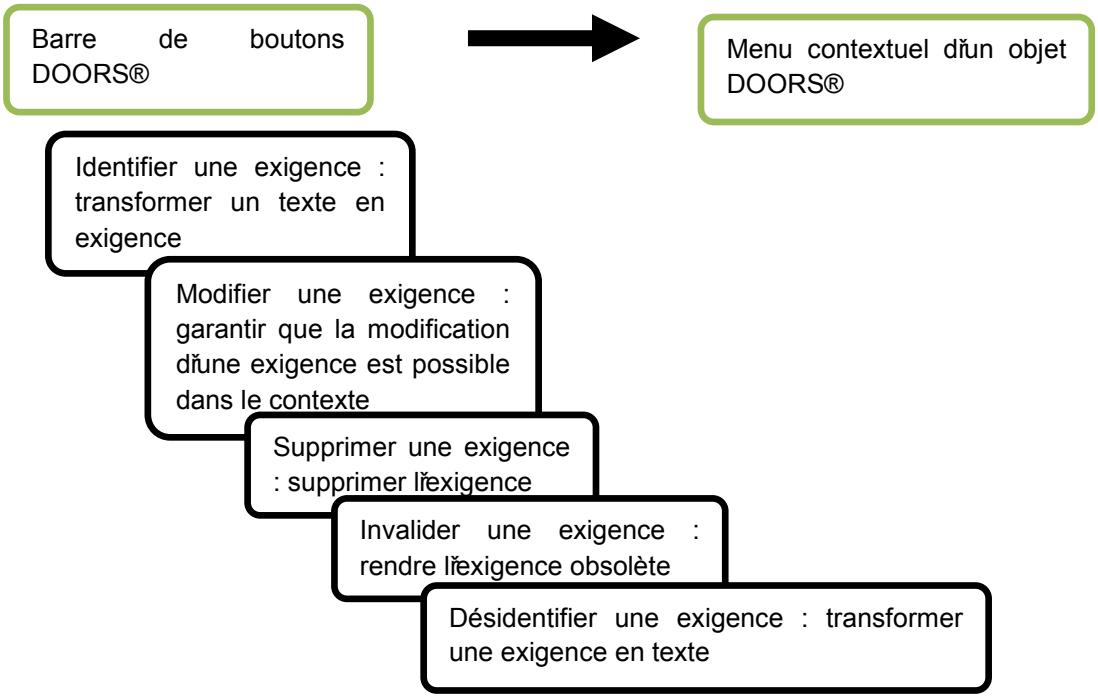

Figure 2.15 : Actions du Requirements Management. Source : *http://www.requitim.com/?mod=Requirements_Management*

En somme, le « Data Model Editor » est un outil de visualisation des données d'un projet, utilisé pour visualiser l'environnement de conception d'un modèle de produit, tandis qu'un « Requirements Management » est un outil de management des exigences, conçu pour gérer et manipuler les exigences dans un modèle.

2.6.2. Gears (BigLever Software Gears)

Gears est un outil d'ingénierie, un logiciel BigLever's Gears Product Line Engineering Tool and Lifecycle Framework permettant d'automatiser la ligne de produits en rendant automatique la conception d'une multitude de modèles basée sur la réutilisation d'un asset plutôt que d'utiliser la configuration et la conception manuelle d'une nuée de modèle composant une LdP. Il permet de créer une ligne de production automatisée axée autour de trois éléments :

- **Configurable Assets** ou Assets configurables qui sont des systèmes configurables et des artefacts logiciels tels que le code source, les exigences, les modèles et les cas de tests conçus pour être partagés à travers le portfolio de la gamme de produits,

- **Feature Profiles** ou profils de caractéristique, un élément permettant de modéliser chaque produit dans le portfolio en termes de choix de fonctions optionnelles et variables spécifiées pour la ligne de produits,

- **Product Configurator** ou Configurateur de produit, un élément assemble et configure automatiquement les systèmes et les assets logiciels - y compris les

exigences, conception, développement et essais - guidés par des profils de caractéristiques des produits, afin de produire des produits du portfolio.

Etant un logiciel dřingénierie de ligne de produits, BigLever Software Gears présente des avantages de taille :

- Une augmentation de la portée de la diversité des produits à lřéchelle des différents produits qui peuvent être livrés de manière efficace dans une ligne de produits,
- Une réduction des coûts et des frais généraux par-développement de produits et des marges bénéficiaires plus élevées,
- Une réduction des délais de commercialisation pour les produits nouveaux et mis à jour, et une agilité accrue pour réagir aux nouvelles opportunités et à lřévolution des conditions du marché,
- Une augmentation de la qualité des produits, une réduction de la densité de défauts et une meilleure gestion des risques.

2.6.3. Pure Variants (Pure Systems)

Le Pure Variants ou Pure Systems est un logiciel permettant de gérer les variables. Ses objectifs sont :

- Une réalisation efficace de solutions logicielles sur mesure
- Une gestion des tâches complexes,
- Une transformation des exigences et contraintes en solutions,
- Résoudre automatiquement les conflits,
- Une application basée sur le logiciel Eclipse,
- Une obtention de la copie dřévaluation le jour-même et
- Une obtention dřune communauté de variables pures.(In : http://www.pure-systems.com/pure_variants.49.0.html)

Pure Variants permet le développement de logiciels de solution sur mesure. Il intervient dans le processus de développement dřun produit en lřintégrant parfaitement tout en restant indépendant du langage de programmation. Il décrit et gère efficacement toutes les parties de produits logiciels avec leurs composants, restrictions et conditions d'utilisation. Il crée un ensemble dřinformations pour créer des solutions automatiques à partir des caractéristiques choisies. De plus, il favorise un processus de développement de produit plus efficace, plus rapide et plus fiable.

2.7. Conclusion

Le premier chapitre de cette thèse intitulé « Etat de l'art des lignes de produits » est entièrement consacré à la compréhension de la notion de ligne de produits, des différentes ingénieries qui permettent la conception des modèles membres dřune LPL et une connaissance des outils de gestion de la LPL.
Nous avons vu dans ce chapitre quřune ligne de produits est composée de plusieurs produits qui sont appelés membres de la LPL. Ces modèles sont conçus en masse grâce à un processus complexe de configuration, de composition et de modélisation. Les modèles ou logiciels membres dřune LPL sont composés dřéléments similaires appelés commonalités ou similarité et dřéléments de différenciation appelés variabilités.

Les similarités sont des caractéristiques communes à tous les logiciels tandis que les variabilités sont des caractéristiques uniques dans chaque produit le permettant de se distinguer des autres. Le principe de gestion dřune ligne de produit repose donc essentiellement sur la gestion de ces variabilités qui attestent de lřunicité de chaque produit du fait que chaque modèle détient une particularité et des options adaptées à un genre spécifique dřutilisateur. Les ingénieries de la ligne de produit permettent de développer ces variabilités (ingénierie de domaine).

Cette ingénierie de domaine permet aussi de concevoir les assets, les éléments réutilisables qui serviront lors de la phase dřingénierie dřapplication. Pour détecter les éléments similaires et les éléments de différenciation dans un modèle ou dans un produit, on utilise le FODA, un diagramme en forme dřarbre permettant de dégager les différentes caractéristiques obligatoires, optionnelles et facultatives dans un produit. A noter que la conception dřune ligne de produits est également majoritairement basée sur la réutilisation.

Lřapproche par les lignes de produit permet une conception rapide, simplifiée et efficace de plusieurs produits en même temps avec un gain de temps considérable, un coût de production minimal et une qualité optimisée des produits. Le gain de temps se traduit par la réutilisation qui consiste à utiliser des composants réutilisables ou assets qui vont constituer les caractéristiques communes à tous les produits ou similarités. Le coût de production repose aussi sur la réutilisation qui évite de façonner un produit à partir de zéro et qui, en plus, facilite la production dřune multitude de produits simultanément.

Enfin, la qualité de production optimisée est traduite par la recherche constante dřinnovation due à lřobligation de trouver des variabilités pour chaque produit, un défi encore très mal géré dans la conception dřune ligne de produits. Tous ces avantages de la ligne de produits sont favorisés par des outils de gestion de la LPL, tels que le Pure Variants, le BigLever Software Gears ou le Requitim.

Après avoir longuement discuté de lřétat de lřart des lignes de produits, nous allons maintenant entrer en profondeur dans le développement de notre problématique en nous focalisant entièrement sur lřétat de lřart des méthodes de développement des applications web et la conception des architectures des systèmes dřinformations web ou SIW.

3. ETAT DE L'ART DES SYSTEMES D'INFORMATIONS WEB (SIW)

Ce chapitre est consacré aux systèmes dřinformation Web. Pour mener à bien notre étude, il importe de définir préalablement ce quřon entend par système dřinformation avant de nous focaliser sur lesdites SIW.

Actuellement, lřinformation se répand et se diffuse partout grâce aux applications web que ce soit grâce aux méthodes Intranet, Extranet ou Internet. Lřère que nous vivons incite grandement les entreprises spécialisées dans tous les domaines, y compris dans celui du développement dřapplication web, de se servir du Web pour promouvoir leurs activités et pour marquer leur visibilité par rapport aux concurrents. On note actuellement la recrudescence dřapplications et de systèmes censés favoriser le développement interne de nombreux groupes industriels tels que ceux dédiés à lřimportation et à lřexportation des données en ligne.

Le Web 2.0 est lřapplication web qui a fortement révolutionné lřusage des applications web dans les industries. Elle a permis de rendre possible lřutilisation de plusieurs serveurs au lieu de se servir de plusieurs ordinateurs à la fois. De plus, elle a aussi contribué à développer et à concevoir les mêmes systèmes que ceux des Desktops jadis utilisés en guise dřapplication web. Grâce au développement et à la mise en marché en masse du Web 2.0, les applications Web ont pris dřassaut le marché en étant de plus en plus évolutives et complexes [27].
Le développement web est un concept découvert depuis plusieurs décennies, mais le terme nřa été officialisé quřen 1998, lors de la conférence dřACM sur les Hypertextes et les hypermédias. Lřingénierie du développement web rejoint la discipline du développement de logiciel avec plus de modernité et de technologie.

Selon [28]: «L'application Web est un ensemble de formulaires et de pages HTML générées dynamiquement (au moins en partie), et qui constitue une application métier. L'exemple le plus simple est celui d'une horloge: en cliquant sur un lien ou un bouton on obtient en retour l'heure courante. Cette dernière bénéficie d'avantages indéniables tels qu'un déploiement beaucoup plus aisé qu'en mode client serveur et un accès universel à moindre coût. L'interface privilégiée est celle du client léger, mais on pourra lui associer d'autres technologies si des interfaces encore plus riches sont nécessaires. La pierre angulaire d'une application Web est le serveur d'applications. ».

Les applications mises sur le web et basées sur celui-ci sont diverses. [28] décrit les applications web de la manière suivante : « les applications web résultent de lřutilisation de sites et de systèmes web » et définit un environnement à la fois technologique et virtuel du fait que le web est quand même un concept virtuel. Ainsi, on parle de système web, une infrastructure permettant dřéchanger des informations dřun ordinateur à un autre ou de plusieurs ordinateurs à dřautres. Il est composé dřéléments essentiels tels que :

Ŕ le réseau, qui relie physiquement les ordinateurs,
Ŕ un poste client doté dřun navigateur, i.e. un logiciel permettant dřafficher localement les documents hébergés sur dřautres ordinateurs du réseau,
Ŕ un poste hôte sur lequel un service (ou démon), appelé serveur Web, permet de localiser et de retourner les informations demandées par le client au moyen de requêtes [29].

Le lien hypertexte permet de faire fonctionner une application web. En effet, il pourvoit à la mise en relation des différentes pages web entre elles via un réseau de liens. Dans une application web, on trouve également un serveur dřapplication web qui permet aux utilisateurs dřeffectuer dřautres tâches en dehors de la simple navigation.

A noter cependant que les applications web présentent une différence par rapport aux sites web. Dans [30] les auteurs font état de cette différence en attribuant deux dimensions au site web : celles dřinformation et de navigation, et dřajouter une troisième dimension en plus de ces deux là à lřapplication web qui est la dimension dřopération. Dans [28], lřauteur appuie cette hypothèse en renchérissant sur la faculté opérationnelle et sur lřaspect manipulable de lřapplication web. Il la décrit comme étant plus polyvalente et plus complexe, avec une possibilité dřeffectuer des tâches et opération de création, de développement ou de suppression qui ne sont pas faisables sur un simple site web.

Cette complexité et cette polyvalence des applications web se traduisent par exemple par le fait de pouvoir sřenregistrer en tant quřutilisateur via un nom dřutilisateur et un mot de passe sur un serveur grâce à la fenêtre ou au volet dédié à cet effet sur la page dřapplication web. Cet enregistrement pourvoit au stockage et à la sauvegarde de données sur le serveur, un fait improbable via un site web. Lřavancée du temps a également permis aux applications web de se démarquer et dřévoluer, dřoù lřapparition en masse dřapplications web spécialisées dans le fonctionnement et la gestion dřentreprises.

En appui aux travaux de [28], une autre classification des applications web émerge [31]. Elle sous-entend aussi la complexité des applications web en référence à la gestion de données et aux offres de services par les sites web via lřapplication proprement dite. Il sřagit ici de la complexité des données et de la complexité des services qui ont permis à [31] de soulever quatre catégories :

Ŕ les sites de présence sur le Web sont caractérisés par un faible degré de complexité tant en terme de données que de services. Leur objectif concerne principalement la mise en ligne d'informations, à des fins publicitaires par exemple, ou résultant d'une volonté de diffuser des informations diverses (activités d'une association, présentations de manifestations, informations personnelles, etc.). Généralement constitués d'un petit nombre de pages, ces sites sont principalement construits à l'aide d'éditeurs HTML et ne s'appuient quasiment pas sur des techniques de développement ad-hoc.
Ŕ les sites "catalogues" ou à forte intensité de données (data-intensive Web sites) sont caractérisés par le fait qu'ils publient une masse importante de données, selon une structure hypertexte complexe. Ils n'offrent en revanche pas ou peu de services.
Ŕ les sites orientés services ont vocation à fournir des services spécifiques tels que ceux offerts par les moteurs de recherche ou les applications de messagerie électronique. De tels sites peuvent reposer sur des bases de données de taille conséquente, mais la structure des données et de l'hypertexte reste simple. La complexité est donc davantage liée aux applications sous-jacentes qui garantissent les services, et c'est en général autour de ces aspects que s'oriente le développement de tels sites.
Ŕ les Systèmes dřinformation basés sur le Web (SIW) constituent la quatrième catégorie de sites Web. Ils combinent la mise à disposition et la gestion de données complexes avec des services interactifs sophistiqués. Les applications de commerce électronique et les SI des organisations entrent dans cette catégorie.

TABLEAU 3.1 : CLASSIFICATION DES SITES ET APPLICATIONS BASEES SUR LE WEB DřAPRES [28] ET [31].

Cependant, la fiabilité des applications web est souvent mise à lřépreuve, notamment à cause du développement et des exigences de plus en plus complexes du marché, de la compétition et des opérations de transactions monétaires appuyées par des fraudes et des arnaques en tous genres. En outre, Dans [97] on détecte une anomalie chez certaines applications web mal conçues et déficientes du point de vue navigabilité, traçabilité et fonctionnalité. En outre, les applications web et les applications traditionnelles présentent des différences indéniables surtout en termes dřarchitecture, de qualité, de contrôle et de maintenance [29].

Dans [98] lřauteur détecte deux aspects pertinents dřune application web : les aspects statiques et faciles et les aspects dynamiques plus complexes regroupant les serveurs web et les bases de données serveurs Open DataBase Connection ou ODBC ou Java DataBase Connection (JDBC).

La complexité du développement et de la gestion des applications web a conduit à opérer des séries de tests performants en vue de valider les applications Web. Hélas, ces tests ne sont pas toujours de haut niveau et manquent souvent de bonne finalisation. Ces tests doivent être effectués à lřaide dřoutils spécifiques compliqués à manipuler et souvent mis à lřépreuve face à la dimension statique des applications web [99].

3.1. Le concept de SIW

Dans [32] les auteurs définissent le SIW comme étant un système dřinformation basé sur les technologies web. A cet effet, plusieurs auteurs [29,32] considèrent lřintranet, lřextranet, le commerce électronique et le site Web externes formant un système d'information comme faisant partie de la grande famille des SIW.

3.2. Portrait des SIW

Le Web a émergé au début des années 1990, suscitant une révolution dans le domaine de la diffusion et du traitement dřinformation. Le Web est une application reposant entièrement sur Internet, ce qui lui permet de disposer dřun espace dřinformation de la même envergure quřInternet. Au début de sa mise en place, le web a été centré sur le concept dřhypertexte : un principe simple *d'organisation non-linéaire des contenus informationnels hébergés sur des machines distantes et accessibles par de multiples chemins de navigation* [29].

Outre lřhypertexte, on distingue aussi lřhypermédia. Ces concepts diffusent et présentent les informations sous forme de textes, de vidéos ou dřimages, sous-entendant une implication directe de lřutilisateur du point de vue de ses besoins et exigences. En général, le concept dřhypermédia est le plus utilisé par les applications web pour présenter telle ou telle information. Ce concept permet aussi de traiter les informations à travers lřinterface Web. Ce sont ces applications web qui forment les systèmes dřinformation web puisquřelles sont basées sur le web [32,35].

Les systèmes dřinformation peuvent être traditionnels ou SIW, cřest-à-dire basés entièrement sur le web. Les SIW requièrent lřintervention dřacteurs humains tout comme le déploiement de ressources matérielles et techniques et dřoutils spécifiques.

De plus, il doit contenir des règles de fonctionnement, de données et de procédés pour acquérir, mémoriser, transformer, rechercher, communiquer et restituer ces données [29]. On parle alors de SIW ou Système dřinformation web lorsque les SI dépendent du Web dans leur mise et place et leur exploitation. Le concept dřhypermédia permet à un système dřinformation dřêtre diffusé par plusieurs chemins sur le Web, impliquant la circulation facile et répétée dřune information qui peut être obtenue en empruntant plusieurs chemins sur Internet.

Concevoir une SIW nřest pas facile et nécessite lřintervention des personnes habilitées à le faire, cřest-à-dire de professionnels de lřinformation. Ces derniers doivent prendre en compte deux dimensions essentielles dans la conception des SIW :

- Lřactivité dřorganisation dřinformation qui constitue le support de navigation et qui ne doit pas être négligée puisquřelle permet de présenter lřinformation sur le Web,

- Lřactivité consacrée spécifiquement aux aspects fonctionnels qui est primordiale dans la mesure où elle permet de concevoir les services que le SIW doit respecter.

Les SIW sont différents des SIT ou Systèmes dřInformation Traditionnels. Les SIT sont considérés comme étant classiques et non mis à jour du fait quřils ne dépendent pas dřInternet. Ils sont alors plus compliqués à utiliser et ne sřadaptent plus vraiment à lřépoque technologique dřaujourdřhui. Le fait que les SIT ne soient pas fondés sur le Web implique alors quřils ne se focalisent pas prioritairement sur la navigation et sur la présentation sur Internet tout comme les SIW. A une époque où le règne dřInternet dans le monde entier est incontestable, on note une dégradation de lřusage, de lřutilité et de la pertinence des SIT non basés sur le Web au profit des SIW. En effet, ils ne permettent pas diffusion rapide et presque sans limite des informations comme cřest le cas pour le SIW.

Les SIW dotés dřune conception dřhypermédias, par exemple, est un exemple bien typique de la différence flagrante entre le SIT et le SIW. Si le SIT ne sřintéresse presque pas à la navigation et aux outils dřInternet, le SIW, par les hypermédias, se concentre grandement sur lřagencement des pages Web, à la conception de liens entre elles en termes de navigation et à lřergonomie des pages avant de les présenter. Dans [100] on met en évidence une étude portant sur lřattention portée sur la présentation des pages en matière de SIW et souligne lřimportance de cet acte.

Les différences entre SIT et SIW sont aussi flagrantes du côté des utilisateurs et de la faculté de ces SI de sřouvrir à leurs utilisateurs. En effet, on constate que les SIT détiennent un nombre plus restreint dřusagers que les SIW du fait des informations diffusées nřimporte où dans le monde grâce à Internet. Cependant, les SIW tels que lřintranet peuvent tout aussi bien cibler leurs utilisateurs et les limiter en termes de nombre. On constate alors que les SIW et les besoins et attentes des utilisateurs sont également interdépendants.
Les SIW profitent dřune accessibilité qui les avantage grâce au Web et disposent dřun public mixte et hétéroclite. Ainsi, ces usagers ont chacun des objectifs, des connaissances, des préférences, des conditions dřaccès aux SIW et des compétences différents et spécifiques, ce qui incite les SIW à prendre en considération les exigences et besoins des utilisateurs en priorité.

A cet effet, on parle aussi de capacité dřadaptation des SIW [29] vis-à-vis des attentes des usagers. Cette adaptation a pour but de rassurer et de fidéliser lřutilisateur, de lui montrer que la

présentation de la page et lŕinterface quŕil visite ont été conçues spécifiquement en accord avec ses goûts et ses exigences.

Les SIW sont considérés comme un système dŕinformation composé par plusieurs réseaux. Dans [36], lŕauteur identifie les extranets, les intranets et Internet comme formant un seul réseau dŕaffaires. Un SIW dispose dŕun modèle général constitué par quatre dimensions et trois phases de développement présentées dans le tableau ci-dessous :

Les dimensions des SIW :

• une dimension informationnelle qui fait référence à l'information contenue dans le SIW et aux tâches reliées (organisation, repérage, etc.);

• une dimension technologique qui comprend l'ensemble des aspects technologiques du SIW comme les langages de programmation utilisés ou l'architecture technologique;

• une dimension "interface graphique" qui est composée des aspects graphiques du SIW tels que la mise en page, la signature graphique, etc.;

• une dimension "interaction système-utilisateurs" qui rend compte des moyens pris ŕ autres que l'interface graphique ŕ pour gérer l'interaction des utilisateurs avec le système comme, par exemple, de la formation à l'utilisation du SIW ou la personnalisation du système à des profils d'utilisateurs.

Les phases de développement des SIW :

- une phase de planification stratégique où les buts du SIW ainsi que les politiques générales gouvernant les entrées, le traitement et les sorties sont définis;
- une phase de conception et d'opérationnalisation où le SIW est conçu, développé, mis en place, maintenu à jour et évaluer;
- une phase de gestion des ressources où ce qui est nécessaire au soutien des tâches du SIW est défini sur les plans humain, technologique et financier, et où s'effectue la coordination de la technologie, des données et des ressources humaines afin d'atteindre les objectifs du SIW.

N.B.
Il est à noter qu'un SIW ne passe pas nécessairement par ces trois phases de manière linéaire. Son développement peut être itératif et le faire passer par les différentes phases de manière répétée et sans respecter d'ordre précis.

TABLEAU 3.2 : DIMENSIONS ET PHASES DE DEVELOPPEMENT DES SIW. [29]

3.3. Systèmes d'Information basés sur le Web

Comme nous lŕavons déjà signalé précédemment, les SIW sont basés sur le Web. Dotés des concepts dŕhypermédias et dŕhypertextes, les SIW offrent une gestion de lŕinformation fondée

sur un réseau « nœud » reliés entre eux par des liens. Ces nœuds sont les hypermédias et les hypertextes. Lřhypertexte est un réseau de nœud servant à produire des textes, cřest-à-dire que lřinformation nřest composée que de textes.

Lorsque les informations sont illustrées non seulement pas des textes, mais aussi par des images, des sons ou des vidéos présents simultanément sur une même page de présentation, on parle dřhypermédias. On constate alors que les sites Internet et les pages web actuelles sont touts composés dřhypermédias puisquřils comportent lřensemble de ces éléments.
Le concept dřhypermédia existe depuis plus dřun demi siècle mais nřa été popularisé et vulgarisé que grâce à deux évènements marquants [37] :

- Le lancement dřApple de lřoutil auteur Hypercard en 1987 qui est considéré comme un « organisateur dřinformations ». cet outil fonctionne avec les mêmes principes quřun concept hypermédia. Véritable innovation à cette époque, lřHypercard est doté de ce quřon appelle un HyperTalk, un outil créant des scripts permettant à lřusager de relier entre elles les informations quřil a classées en unités ou cartes.

  HyperCard comporte des textes, des sons et des images fixes ou animées à la fois, ce qui lui vaut une popularité immédiate. Dřailleurs, cet outil a été vendu en un million dřexemplaire lors de sa première année de mise sur le marché et a promu lřhypermédia auprès des personnes dotées de matériels informatiques et pouvant user de lřhypermédia pour créer leur propre application.

- La création du Web est le second facteur de popularisation de lřhypermédia. Lang considère que bien que lřhypermédia ne soit pas obligatoirement basé sur le Web, ce dernier à permis de le développer et de le promouvoir partout dans le monde [38].

3.3.1. Applications Web

Les applications web sont surtout utilisées en tant que « vitrine sur le web » [39] par les entreprises, les organismes, les institutions et les particuliers dans le but de promouvoir leur image et de lřaméliorer ainsi que de mettre en ligne leurs prestations. Dans cette optique, toute application est sujette à une conception complexe et optimale et à un souci de développement dřinterfaces. Lřinterface est la page que lřusager utilisera pour mener une action bien définie comme effectuer des achats en ligne, des emprunts de livres, etc.

Plus elle est qualitative, plus son usager réalisera avec succès les objectifs quřil sřest fixé en lřutilisant. Cette insatisfaction des clients conduit à une mauvaise publicité du concepteur de lřinterface, et donc à une mauvaise image de lřentreprise proposant les services.

Dans ce cadre, Thatcher et al. [39] expose un vote de lois en faveur du respect de la qualité des applications web dans plusieurs pays. Cette loi exige le respect des exigences minimales et dřobligation en termes dřapplication relatives à la conception et au développement des applications web. Dans cette mesure, la conception optimisée dřinterfaces [40] de qualité supérieure est devenue une exigence prioritaire pour les concepteurs dřapplications web. Cette conception optimisée est aussi relative aux exigences de plus en plus capricieuses en faveur de lřévolution grandissante des applications web dans lřespace et dans le temps et en tenant compte des attentes des utilisateurs.

Ivory et al. [39] développe des solutions afin dřassurer un suivi de développement ergonomique dřune application web en référence à lřoptimisation de sa conception. Il développe entre autres :

- Des démarches de conception centrées utilisateur,
- Des démarches de conceptions participatives,
- Une organisation de connaissances ergonomiques et
- Des méthodes dřévaluation.

Les applications web exigent la mise en place dřun processus de conception web complexe et uniformisé avec une mise à jour ponctuelle de la page pour rester dřactualité. Il sřagit de processus de conception classiques retranscrit de [39].

La navigation est un aspect important des applications Web [39] que nous ne retrouvons pas dans les systèmes interactifs classiques. Dans une application Web, nous ne parlons plus dř« utiliser » lřapplication mais de « naviguer » dans lřapplication. Si la navigation est mal conçue, lřutilisateur peut se perdre et passer du temps à chercher lřinformation voulue. Dans le pire des cas, lřapplication ne lui offrira pas la possibilité de revenir sur ses pas (auquel cas, lřutilisation de la fonction « retour arrière » du navigateur sera obligatoire). Par conséquent, la navigation peut causer des problèmes dřutilisabilité importants et **nécessite d'être prise en compte** dans une étape du processus de conception ;
Le **contenu informatif** est important pour les systèmes interactifs Web, contrairement aux systèmes interactifs classiques où le traitement des données est plus important. Ainsi, les informations sont fréquemment mises à jour (dřautant plus que ces mises à jour sont rapides et peu coûteuses). Le contenu dřune application Web peut être amené, dans certains cas, à évoluer plusieurs fois en une journée ceci afin dřy ajouter de nouvelles informations, dřy apporter des modifications sur le contenu existant, etc.

TABLEAU 3.3 : [39]

Pour peaufiner ce processus de conception, Scapin et al. [39] développent un processus de conception itératif pour le web basé sur ces phases :

- **Expression des besoins** Ŕ La phase Expression des besoins (ou Analyse des besoins) identifie les principaux buts des commanditaires, le contexte dřutilisation et les besoins. Elle comprend la collecte du contenu qui sera utilisé plus tard.

- **Spécification** Ŕ La phase de Spécification (ou Modélisation conceptuelle) produit des spécifications à partir du contexte dřutilisation et des besoins recueillis dans la phase précédente. Des modèles détaillés sont construits pour formaliser les besoins, comme, par exemple, la structure de navigation, les tâches utilisateur réalisables avec lřarchitecture de lřapplication et du site.

- **Conception** Ŕ La phase de Conception consiste à affiner les spécifications dřaprès leur contenu. A la fin, des modèles de navigation, de page et de données sont construits. Une spécification détaillée est produite afin de guider lřimplémentation de lřapplication Web.

- **Implémentation** Ŕ La phase dřimplémentation (ou Développement) correspond à la construction physique de lřapplication Web, à la production des pages HTML et à lřéventuelle intégration de contenus multimédias (son ou vidéo). A la fin de cette phase, toutes les pages ont un contenu, des liens et des éléments graphiques incorporés. Lřapplication est prête à être livrée. [39].

Cette conception optimisée entraine une évaluation performante de la qualité des applications web qui, selon Brajnik [39], nécessite une forte interaction entre deux types dřutilisateurs et le site web :

- **les utilisateurs finaux** qui sont surtout à la recherche dřinformation ou de produits à acheter. Ils ont un comportement qui est fortement affecté par leur culture, leur langage, leur niveau de connaissances antérieures dans le domaine et leur expérience sur le Web. Ils interagissent avec les sites Web via une couche de technologie quřils ignorent, qui nřest pas sous le contrôle du concepteur du site et qui comporte : les navigateurs, les protocoles, les plug-ins, les systèmes dřexploitations, les plate-formes, les dispositifs dřinteraction (écrans, multimédia…) et les connexions réseaux ;

- **les développeurs et les professionnels de la maintenance** qui essayent dřoffrir un site de qualité en maintenant le caractère opérationnel du système ou en lřaméliorant. Ils font face à une technologie qui évolue très rapidement et à un système qui a un cycle de vie très rapide. La maintenance dřun site Web se fait avec un rythme plus important que les autres produits logiciels à cause de la pression du marché et de lřabsence dřune barrière de distribution. En plus, la portée de la maintenance devient souvent tellement large quřune refonte totale du site est parfois nécessaire [41].

Les applications web doivent avoir une qualité optimisée [41]. Dans [41], les auteurs ont donc dressé un tableau représentant les Caractéristiques et sous-caractéristiques de la qualité dřune application web :

1. Fonctionnalité	- aptitude - exactitude - interopérabilité - sécurité - conformité
2. Fiabilité	- maturité - tolérance aux fautes - possibilité de récupération
3. « Utilisabilité »	- compréhensibilité - facilité dřapprentissage - attractivité
4. Rendement	- performance (temps) - ressources
5. Maintenabilité	- facilité dřanalyse - facilité de modification - stabilité

	- testabilité
6. Portabilité	- facilité d'adaptation - facilité d'installation - conformité - interchangeabilité

TABLEAU 3.4 : CARACTERISTIQUES ET SOUS-CARACTERISTIQUES DE LA QUALITE. [41].

Outre les caractéristiques définis par [41], plusieurs auteurs ont également mené des études sur la qualité des applications web.
Nielsen[39] propose une approche de la qualité des applications web entièrement focalisée sur l'utilisabilité, sans pour autant laisser de côté les autres critères. Ainsi, il définit une « *loi de Jakob sur l'expérience des utilisateurs* » dans laquelle il stipule que « *les utilisateurs passent la plus grande partie de leur temps sur d'autres sites que le vôtre, et c'est là qu'ils construisent leur expérience et apprennent à se faire une idée du mode de fonctionnement du web…* ».

Dans cette optique, Nielsen [101] offre des conseils pratiques et des instructions concernant l'utilisabilité des sites web, ces recommandations sont pour la plupart fondées sur ses propres recherches et ne comportent pas de dimensions quantitatives.

Certains travaux [102] ont également adopté l'utilisabilité pour traduire la qualité d'une application web, plus précisément d'un site web. Leurs recherches s'orientent vers la rédaction du contenu web qui doit se faire dont le style d'écriture doit être précis, objectif et exact. Leurs travaux se sont alors orientés vers les recherches de Nielsen grâce auxquels ils ont dressé une check-list ou liste de vérification comportant 203 éléments focalisés sur les utilisateurs des sites web. Cette liste de vérification contient des instructions issues des travaux de Nielsen et englobe des critères tels que :

- la facilité à retrouver l'information recherchée,
- la compréhensibilité de l'information,
- la capacité à répondre aux besoins de l'utilisateur, la capacité à présenter une information exacte et
- la présentation de l'information.

La capacité des sites web à remplir ces critères et à les appliquer permet d'affirmer qu'ils sont de qualité. Outre l'utilisabilité, ces deux auteurs préconisent également l'actualisation régulière du site web et de leur contenu ainsi que l'usage de techniques interactives. De ce fait, ils invitent les développeurs de sites web à échanger avec leurs utilisateurs dans les sites, comme c'est le cas pour les blogs ou tout simplement en incluant un espace dédié aux commentaires dans les pages contenant des articles.

Cette pratique permet de mettre en valeur l'utilisateur et de lui montrer que le site est bien à son service pour lui fournir ce qu'il attend. Pour détecter les besoins des utilisateurs, poser un questionnaire en ligne est une solution envisageable, à condition que les questions soient fermées et nécessitent donc une réponse catégorique entre oui ou non. Le but d'un tel questionnaire est de minimiser les réponses trop subjectives de la part des utilisateurs et de privilégier leur objectivité afin d'améliorer la qualité du site web.

Dans le cadre de l'utilisabilité, certains auteurs [102] mettent l'accent sur la présentation de la page web ou du site avec une langue et un style rédactionnel impeccables, un format adéquat,

etc. La liste d'évaluation qu'ils proposent permettra de recueillir des avis objectifs de la part des utilisateurs. Pour ce faire, les deux auteurs recommandent de poser des questions bien détaillées.

Les auteurs dans [41] ont également étudié la façon d'optimiser la qualité des applications web, notamment des SIW. Ils ont alors conçu un prototype expérimental d'évaluation quantitative des SIW en se basant sur les travaux de plusieurs chercheurs. A cet effet, ils ont orienté leurs recherches uniquement sur l'utilisabilité des SIW en s'identifiant aux recherches de [41]. Ils préconisent alors l'usage des métriques pour accentuer la qualité des SIW.

Pour d'autres les SIW sont un ensemble de *logiciels, de données, d'information et du design graphique,* de plus, ils caractérisent les SIW comme étant *l'expérience différente pour chaque visiteur.* En outre, ils proposent aussi d'optimiser la qualité des SIW en prenant en compte ces quatre dimensions [74] :

- l'utilisation
- la maintenance
- le développement et
- la technologie

Cornela Boldyreff [103] a également dirigé des recherches sur la qualité d'une application web. Dans cette mesure, elle a distingué les différences entre un site web, une page web et un composant : « *un site peut comprendre un certain nombre de pages, et une page peut être composée d'un ou de plusieurs composants…Nous croyons (les auteurs) que la présentation apparente à l'utilisateur est un facteur important du succès ou de l'échec d'un site Web. Ainsi, il est vital de comprendre l'apparence d'une page Web et non pas de la réduire simplement à ses composants*».

Boldyreff [103] et son équipe ont également identifié huit dimensions suivant lesquelles les sites web et les pages web peuvent être classifiés :

- la taille,
- le domaine,
- l'objectif,
- la technologie,
- la fonctionnalité,
- l'âge,
- le taux de changement, et
- la stratégie d'évolution. (Boldyreff, 2000).

Luis Olsina [104] est un autre chercheur ayant comparé et mesuré la qualité des applications web, surtout durant leur phase opérationnelle. Cet auteur préconise l'orientation vers le génie logiciel afin d'améliorer, d'analyser et de comprendre la qualité des applications web. Olsina rappelle la complexité des exigences et contraintes liées à la qualité des applications web qui requièrent.

Il recommande alors de mettre en place des processus et des dispositifs pour l'évaluation de la qualité des applications web en usant de l'ingénierie d'application web, mais aussi en usant d'outils appropriés à cette tâche.

A noter qu'Olsina a fait équipe avec l'Ecole d'ingénierie à UNLPam en Argentine pour approfondir ses recherches qui sont fondées sur les normes IEEE et ISO/IEC. Ces travaux

certifiés concernent majoritairement les instructions et conseils sur la qualité du logiciel et sur les métriques existantes. En 1998, du fait de leurs recherches, Olsina et son équipe ont conçu le Website Quality Evaluation Method afin dřanalyser les caractéristiques de la qualité dřun SIW ou Système dřInformation Web.

Ainsi, nous voyons donc que lřefficacité et la qualité dřune application web sont déterminatifs de leur réussite. De nombreux auteurs ont lancé des recherches en la matière et décrivent une approche centrée sur :

- Lřutilisateur et ses besoins, cřest-à-dire que les développeurs des applications web doivent prendre en compte à un certain degré les exigences et attentes des utilisateurs et façonner les SIW, les sites web, les pages web et leurs composants en fonction de ces derniers.
- La présentation, cřest-à-dire le format, le contenu, le style, la rédaction et tout ce qui concerne lřaspect visuel de la page web. Cette dernière est également à modeler suivant les attentes des utilisateurs sur certains points.
- Les développeurs web qui doivent mettre leur inventivité et leur imagination au profit dřidées nouvelles. A cet effet, ils doivent également mettre continuellement à jour le contenu des sites web et sřinformer sur les tendances du moment pour ne pas se laisser dépasser par la concurrence.

Outre ces critères, on remarque aussi que la qualité dřun SIW ou dřune application web repose aussi sur lřutilisation des métriques et des techniques certifiées telles que celles indiquées dans les normes IEEE et ISO/EP ou dřoutils tels que des logiciels issus du génie logiciel.

3.4. Systèmes d'Information Traditionnels et les systèmes d'Informations Web

3.4.1. Définition des Systèmes d'Information Traditionnels

i. Ergonomie des SIT

Un système dřinformation traditionnel est un moyen de communication, de sauvegarde, de partage et de diffusion dřinformations presque non basé sur le web. Ce partage et cet échange dřinformations se fait dans un milieu bien défini et est appuyé par des logiciels, des matériels, des données ou des procédures bien spécifiques.

ii. Origine et usage

Dans les années 80 jusquřaux années 90, les SIT étaient de plus en plus répandus dans les entreprises. Ils représentaient alors une certaine hiérarchie de lřentreprise et étaient utilisés surtout pour gérer le fonctionnement interne, pour améliorer lřéchange dřinformations entre les différents salariés et pour améliorer le processus de management interne. On remarque alors lřapparition de systèmes DSS (soutiens de décisions), de gestion dřinformation comme le MIS ou des SIT usités par la direction ou EIS. Ces SIT ont formé une pyramide, une forme très en vogue à cette époque.

3.4.2. Etat de l'art actuel

Actuellement, les SIT sont surtout utilisés en entreprise et présentent différents aspects. On parle dřERP ou Enterprise Resource Planning ou PGI (Progiciel de Gestion Intégré) dédié spécifiquement au bon fonctionnement de lřentreprise. Il existe aussi des SIT sous forme de CRM ou Customer Relationship Management ou gestion de la relation cliente (GRC).

Ce dernier prend en compte lřutilisateur qui fait partie des personnes à informer, tout comme les intervenants dans lřentreprise. Dřautres SIT tels que XRM ou eXtended Relationship management ou gestion de la relation au tiers, le SCM ou Supply Chain Management ou GCL (Gestion de la chaîne logistique), le HRM ou Human Resource management ou gestion des ressources Humaines (GRH), le PDM : Product Data Management ou Système de Gestion de Données Techniques (SGDT).

Actuellement, on note une tendance basculant vers lřinformation des utilisateurs en même temps que des intervenants dans la conception du SIT. De plus, les évolutions technologiques qui ne cessent dřaccroitre permettent une meilleure diffusion, une meilleure gestion et une meilleure sauvegarde des informations. On peut alors dire que nous assistons surtout à une information de plus en plus diffusée sur le web suivant des processus de plus en plus automatisés.
Cette orientation vers lřusager sous-entend une gestion du contenu, une conception optimale mais aussi une gestion accrue de lřaccès qui doit être plus ouvert, plus souple et élargi.

3.4.3. Comparaison entre les Systèmes d'Information Traditionnels et les systèmes d'Informations Web

Dřaprès les différents aspects du système dřinformation traditionnel ou SIT et du Système dřinformation web ou SIW. Nous pouvons conclure que ces deux concepts ont la même finalité qui est dřorganiser et dřarranger le fonctionnement dřune société ou dřune entreprise. Par conséquent, les finalités sont communes, seule la technologie diffère car le SIT est traditionnel et le SIW est révolutionnaire puisque basé sur le Web.

De ce fait, des études, notamment celles de Collette Rolland en 1988, ont permis de définir le SI en tant quřensemble composé de :

Ŕ « de collection de données, représentation partielles, en partie arbitraires mais nécessairement opératoires, dřaspects pertinents de la réalité de lřorganisation sur lesquels on souhaite être renseigné. Ces collections inter-reliées, aussi cohérentes que possible, sont mémorisées et communiquées dans le lieu, le moment et la présentation appropriées aux acteurs qui en ont lřusage ;

Ŕ de collections de règles qui fixent le fonctionnement informationnel. Ces règles traduisent où sont calquées sur le fonctionnement organisationnel. Parties intégrantes du SI, elles doivent être connues des acteurs qui utilisent le SI et leur sont nécessaires pour lřinterprétation et la manipulation des collections de données ;

Ŕ dřun ensemble de procédés pour lřacquisition, la mémorisation, la transformation, la recherche, la communication et la restitution des renseignements ;

Ŕ dřun ensemble de ressources humaines et de moyens techniques intégrés dans un système, coopérant et contribuant à son fonctionnement et à la poursuite des objectifs qui lui sont assignés ».

Ainsi, plusieurs théories ont commencé à sřintéresser de près au SIW comme une extension plus informatisée et technologique du SI puisque basée sur le Web. Ces études ont même mené à une certaine théorie destinée à savoir si un SI doté dřune application Web ne diffèrerait pas dřune SIW proprement dite. Quoi quřil en soit, la grande différence entre ces deux concepts repose surtout dans la technologie et dans les utilisations.

3.5. Les méthodes de conception des applications web

3.5.1. RMM: Relationship Management ModelWebML: Web Modeling Language

i. Aperçu général de RMM

Le RMM (Relationship Management Methodology) [46] est un outil de conception et de construction des applications hypermédia qui permet de mieux assurer la gestion des éléments dřinformation. Il est recommandé dans [46] grâce à sa capacité de structurer lřapplication des informations ainsi que de les modifier facilement. Il est par contre inutile de lřutiliser pour des applications faiblement structurées comme les applications artistiques ; cette approche est en revanche efficace pour les applications.

ii. Les étapes de la méthode

Lřobjectif principal du RMM réside dans le développement intégral du logiciel tout en suivant ses processus. On retrouve principalement trois étapes destinées à la conception et quatre autres pour le développement.
La première étape de la conception des applications web sřagit de représenter le domaine dřapplication à travers un modèle E-R. Lřétape suivante se soucie de concevoir les slices et de déterminer la façon dont le contenu des entités aux utilisateurs est représenté et quelles possibilités dřaccès sont offertes. Pour faciliter cette seconde étape, un diagramme de slices est définit selon chaque entité et celui-ci mettra en évidence ses propres représentations.

A partir de cette étape, on obtient un artefact de schéma E-R+ correspondant au modèle obtenu par lřétape précédente qui a permis à chaque entité dřêtre remplacée par son diagramme de slices .Cřest dans le système multimédia de présentation des informations ou lřhypermédia quřun slice devient un nœud ou une page HTML. La troisième et dernière étape de la conception consiste à définir les aspects navigationnels de lřhypermédia. Des éléments en rapports avec le modèle E-R sont remplacés par des liens, regroupements ou index grâce aux indications suivantes :

- Etablissement dřun lien bidirectionnel pour une relation (1.1)
- Choix dřun index ou une visite guidée dans le cas dřune relation (1 ; n)

Cřest à partir de ce stade que le modèle RMDM est défini et que les modalités dřaccès finales sont représentées.

En ce qui concerne le processus de développement, la première étape consiste à décrire une règle de transformation afin de transformer les éléments du modèle RMDM en objets de la plate-forme retenue pour le développement c'est-à-dire lřétablissement des index par des formulaires HTML. Lřétape suivante met en valeur la conception de lřinterface utilisateur et la description de lřapparence ainsi que la localisation de chaque élément du modèle RMDM.

La troisième étape du processus de développement est consacrée à la conception du comportement à lřexécution et à la prise de décision relative aux mécanismes de navigation et au suivi des actions de lřutilisateur. Le degré de volatilité des données est un critère de choix dřune génération dynamique des pages. Enfin vient la septième et dernière étape qui vise à construire et à soumettre des divers tests au système élaboré.

iii. Limites et extensions de la version originale de RMM

Selon Isakowitz[46, 47] et dřautres auteurs, le RMM présente des inconvénients dans la possibilité de modélisation et de démarche. Voici une brève présentation de ses deux limites :

i. Limite des slices et leur évolution : les m-slices

On constate que les pages Web à caractère complexes ne sont pas suffisamment modélisées par le RMM, le slice est peu flexible car il ne peut être utilisé quřà lřentité ce qui veut dire quřil est impossible que des informations venants de plusieurs entités se classent dans un seul et unique slice.

Dans [46] un m-slice est créé pour combler le non flexibilité des slices, « m » se référant aux poupées russes emboitables ou Matriochkas. Ces dits m-slices favorisent le regroupement des informations proviennent de diverses sources afin de constituer des unités dřinformations très complexes intégrant dřautres m-slices.

ii. Redéfinition du processus de conception

Le RMM se limite au niveau conceptuel de lřhypermédia qui suit une approche top-down. Le modèle E-R donne une forme initiale à lřhypermédia puis le sculpte à lřaide des diagrammes de slices évoqués auparavant. Lřinexistence de primitives de modélisations favorisant la construction de lřapplication par des éléments encore plus petits est selon [47] un problème majeur qui peut être contré par les m-slices construits à partir des attributs et composés par des nœuds de lřhypermédia. Toutefois, une approche de conception devient ascendante ou bottom-up. Bref, Le problème se révèle être la difficulté de production dřun modèle dřapplication représentant clairement lřhypermédia.

Un diagramme de redéfinition des étapes conceptuelles est proposé par les auteurs [47].Ce dernier est un diagramme dřapplication servant à remplacer le type de modélisation résultant de la troisième étape du processus de conception. Le processus de changement et remplacement est ici appréhendé de manière ascendante et descendante :

- La structure générale de lřapplication est mise en évidence à partir de la description de la première version du diagramme dřapplication, les unités du diagramme sont départagées en m-slices par une approche descendante.

- Les m-slices construisent un diagramme dřapplication tout en respectant les étapes spécifiques par une approche de type ascendante [47].

Les deux approches descendante et ascendante combinées permettent de comparer les deux diagrammes dřapplication et dřidentifier les problèmes tels lřoubli et suppression des m-slices Ainsi, à partir de ses deux approches le concepteur affine ces spécifications dřune manière approximativement successive pour obtenir un diagramme dřapplication conformes aux approches précitées.

iv. Conclusion sur RMM

RMM est à la fois une démarche de conception et de développement des hypermédias de structure à caractère stable et à contenu fréquemment modifié.
RMM est basé sur une image HDM et un modèle E-R le spécifie, quant à lřhypermédia, il est alors construit à partir du modèle RMDM grâce à la transformation des m-slices en nœuds et des relations en lien de navigation. Les apports de RMM par rapport à HDM, relèvent particulièrement dřune reproduction plus détaillée de lřutilisation des différents éléments du modèle. Néanmoins, RMM est la seule méthodologie qui met en exergue les diverses étapes de conception de lřhypermédia et la seule qui soit exécuter dans lřatelier RMCase ou Relationship Management Case Tool) [48] incluant les différentes intervalles du procédé.
Par ailleurs, RMM a fait lřobjet dřoffres qui veulent étendre les déterminations de liens de navigation dynamiques [49], aspects liés à la présentation [50, 51]. La version originale de RMM est insuffisante en adaptation car les m-slices ne sont pas utilisés à des fins dřadaptation à différents utilisateurs : ils décrivent seulement la conception de la représentation informatique des utilisateurs.

3.5.2. HERA

i. Aperçu général de Hera

Pour [50], les possibilités de précision proposées par RMM dans un concept présentation sont éphémères. Lřon reproche souvent au RMM de nřintroduire que quelques éléments de représentation dans le diagramme dřapplication ce qui peut biaiser le modèle de conception et freiner le processus même.

Le projet Hera [105] est selon les auteurs une étape de création de la présentation et des adaptations [51] tenant en compte des spécificités matérielles, des choix de lřutilisateur et de lřhistorique de ses navigations.

ii. Conception de la présentation

Elle se distingue par un niveau logique et un niveau représentatif. Au premier niveau, des spécificités de présentation plus évoluées que les liens de navigation dans RMM sont manifestées par le biais de définition de nouveaux éléments de modélisation. Au second niveau présentatif, la production dřun diagramme de présentation est vivement conseillée pour faciliter le rapport des deux niveaux.

Le niveau logique
Ici, le diagramme dřapplication est doté de renforcements sous forme de relations se situant à un niveau élevé dřabstraction et des liens de navigations et les relations dans le temps et dans lřespace entre les mi-slices.

Ces relations sont parfois simples c'est-à-dire quřelles ne proviennent que dřune seule source et sont parfois multiples ou provenant de plusieurs sources et cibles, en ce qui concerne cette

pluralité de sources, plusieurs dispositions sont proposées sur la base des opérateurs AND et OR comme la configuration multi and target qui préconise que franchir le lien vise à atteindre les cibles [50].

On remarque aussi lřinsertion des mi-slices dans dřautres et cette relation constitue ce quřon appelle un internal mi-slice ou *internal relationship.* Elle requiert une spécification du contenu m-slice si son lien a été pénétré ; cřest dans ce cas quřon évoque le terme de *preserving et vanishing source relationship* ou relation interne de préservation.

Les précisions au niveau logique exposent intégralement lřagencement de la présentation sans donner une représentation concrète et visible des sélections en termes dřorganisation navigationnelle, spatiale et temporelle.

Le niveau présentation
Comme ce qui a été dit précédemment, le diagramme de représentation permet la description de la conception de lřhypermédia .Les éléments obtenus durant le niveau logique deviennent par suite des éléments de présentation : les relations navigationnelles et lřexpression matérialisent les liens établis entre m-slices et régions qui sont proprement définies comme une collection dřattributs véhiculant lřinformation du domaine référant lřélément source dřinformation. Chaque région est agrégée à une zone rectangulaire de lřécran, déterminée par une grandeur et une position.

La liaison m-slice-région peut affecter un même m-slice à différentes régions, ou différents m-slices à une même région. Elle ne doit en aucun cas conduire à une séparation dřune unité significative quřun m-slice doit montrer.

Trois types de relations sont remarquées au niveau de la relation m-slice et régions qui ont chacun une morphologie et un caractère propre en liaison étroite avec sa réalisation : on trouve la relation de navigation qui sert à représenter les hyperliens de navigation, puis la relation temporelle exprimant la notion de durée dans la présentation quřelle soit intrinsèque ou conceptrice, plus précisément, cette relation vise à déterminer la durée de lřaffichage de lřinformation.

En dernier lieu, nous pouvons parler de la relation spatiale qui décrit lřemplacement des coordonnées dřune région cible par rapport à une région source. Notons que ces trois relations sont décrites par un indice de synchronisation.

La progression du processus de RMM résulte du [50] qui élabore une spécificité de lřhypermédia. Cřest au niveau logique du processus que sont incorporées les unités de modélisation de lřhypermédia et cřest au niveau spécifique que les éléments relatifs à la présentation sont pris en charge. Ces niveaux évitent que les choix de présentation ne soient extraits lors de la phase dřinstallation.

iii. Conception des adaptations

La conception des adaptations est proposée dans [51], elle détermine lřapparence des m-slices et concrétise la relation entre elles. Lřadaptabilité et lřadaptativité sont les deux types dřadaptation envisagés. Lřadaptabilité vise à exploiter des informations concernant les capacités dřaccès aux dispositifs comme la taille de lřécran, la capacité de mémorisation ainsi que la vitesse du réseau et la mise en page que lřutilisateur souhaite.

Lřadaptabilité se basant sur la définition dřun profil décrit en utilisant le vocabulaire CC/PP [106]. Ce profil propose les caractéristiques matérielles et logicielles du navigateur en question et procède de manière à stocker le choix personnel de lřutilisateur ou le User Preference. Lřaugmentation des m-slices décrits de condition à évoquer ce qui est accepté pour afficher les m-slices. Lřadaptability condition noue un lien entre le profil et la catégorisation des m-slices. Elle fait en sorte que le m-slice soit intégré dans une page HTML lorsque les conditions dřassociations sont acceptées par le dispositif de sortie.

Lřadaptativité quant à elle est une adaptation se focalisant sur lřhistorique de la navigation dřun utilisateur. Les informations concernant les activités de lřutilisateur sont regroupées et mises à jour au fur et à mesure quřil soit actif. Le modèle de lřutilisateur définit un ensemble de pairs attributs_ valeurs correspondant au concept et à la valeur que lřutilisateur y attribut. Les conditions dřadaptativité font partie de la valeur minimale dřintérêt accordée par rapport à lřutilité de présentation du m-slice, elles évaluent également la valeur dřun concept selon un utilisateur en fonction des pages quřil a visitées. Lorsquřun lien est consulté par lřutilisateur, le concept augmente automatiquement de valeur et doit être considéré par le concepteur.

iv. Conclusion sur Hera

Hera est une étape importante dans la conception de lřhypermédia, elle facilite le passage du niveau logique à la phase de réalisation finale ou phase dřimplémentation. Les capacités de modélisation selon Hera se classent en deux catégories bien distinctes :

- Lřadaptabilité comme analyse statique de la présentation antérieure de la navigation
- Lřadaptativité come analyse dynamique c'est-à-dire le changement de représentation navigationnelle [51].

Ces deux modes déterminent la visibilité et lřaspect des m-slices, néanmoins lřadaptativité ne recourt quřà lřaspect afin de réduire les inconvénients du RMM.

3.5.3. WEBML

Le web modeling language (WebML) [28, 44, 45, 68] est une codification graphique ayant une syntaxe ou forme textuelle XML des applications web complexes. Il permet en effet de sculpter les données, les pages qui la composent, la technique de navigation entre les pages, la présentation et lřindividualisation par utilisateur. Le WebML procède par :

- Lřassemblage des besoins
- La conception des données
- La création de lřhypertexte
- La conception de la présentation
- La conception des utilisateurs et groupes dřutilisateurs
- La phase de conception de la personnalisation

La phase de collecte des données vise à déterminer les objectifs principaux du site par rapport aux besoins de lřutilisateur, à identifier les utilisateurs cibles et à personnaliser le site.

La phase de collecte des données sert à créer le modèle de structure qui définit les liens, le domaine et données dřapplication comme les entités, les attributs et les différentes composantes. Elle nřest réalisable que grâce à un diagramme de type UML et un modèle XML.

La phase de création de l'hypertexte consiste à construire un modèle de conception et un modèle de navigation par l'architecture de l'application web. Le premier modèle construit les nœuds en termes de page. Le second quant à lui décrit comment les pages et unités vont être liées entre elles et comment l'utilisateur pourra naviguer facilement à travers ces unités. En vue de cette facilité de navigation, deux liens contextuels et non contextuels sont à utiliser, ils permettent le transport d'information de l'unité initiale jusqu'à l'unité d'objectif.

La phase de conception de la présentation est une phase durant la quelle on assemble les feuilles de styles ou layout ainsi que la présentation spécifique du contenu. On distingue deux types de feuilles de styles : les non typées qui sont indépendantes au contenu, applicables sur n'importe qu'elle page et les styles typées qui sont spécifiques au contenu, applicables que sur une seule page.

La conception des utilisateurs et groupes d'utilisateurs décrit et spécifie les groupes ou types d'utilisateurs selon leur traits communs, , caractéristiques définies grâce aux propriétés des concepts du modèle structurel. C'est durant cette phase que l'on tente de délimiter **le profil utilisateur.**

Le profil utilisateur est structuré en interne c'est le cas du regroupement des attributs sous forme de composants. Le mot profil est ici associé à l'identité personnelle de l'utilisateur, à ses activités, les pages qu'il a dernièrement visitées, les items visités et les achats effectués. Ce profil est évolutif par rapport à la navigation.

La phase de conception de la personnalisation personnalise la page en fonction des données du profil utilisateur. On distingue deux types de personnalisations : d'abord la personnalisation déclarative parallèle au comportement de l'utilisateur, ensuite la personnalisation procédurale qui consiste à catégoriser l'utilisateur pour le personnaliser et le différencier des autres par exemple **le client privilégié.**

Etapes	Collecte des besoins	Conception des données	Conception de l'hypertexte	Conception de la présentation	Conception des utilisateurs et groupes d'utilisateurs	Conception de la personnalisation
Artefacts produits	Objectifs du site, utilisateurs cibles, exemples de contenu, guides de styles, personnalisation, contraintes	Entités, attributs, composants, liens entre entités	Modèle de composition (nœuds, page, unité de contenu), Modèle de navigation (liens contextuels, liens non contextuels	Feuilles de styles typées, Feuilles de styles non typées	Profil utilisateur	Entités, Attributs, Composants (pour la personnalisation déclarative) et Règles métier (pour la personnalisation procédurale)

TABLEAU 3.5 : ARTEFACTS PRODUITS DANS LE CYCLE DE VIE SELON LA METHODE WEBML [68]

WebML est utilisé dans l'approche de spécification par étapes consécutives et s'inscrit dans des travaux HDM Lite [42] qui adapte les propositions du HDM à la généralité spécifique du web. A part ce HDM-Lite, on retrouve aussi le HDM é Lite W313 (Web-based Intelligent Information Infrastructure) [43] qui oriente le concept de dérivation et de personnalisation.

Web ML fait des représentations graphiques associées à une syntaxe XML afin de mieux présenter le site. Les articles concernant le webML que nous avons étudiés [33,44,45], révèlent qu'il existe une variation du nombre de dimensions.

Certaines dimensions sont à la base de l'approche, car elles sont constamment présentes dans la littérature sur WebML et aboutissent aux modèles structurels, de déviation, d'agencement, et de navigation. Les deux derniers modèles se classent dans le modèle hypertexte qui à lui seul arrive à définir la typologie de l'hypertexte. Une quatrième dimension de gestion des contenus est décrite dans le langage WebML par les extensions des modèles de composition et de navigation.

i. Le modèle structurel

Il permet de structurer les contenus inclus dans le site, ses éléments peuvent être des relations et entités qui désignent des connexions similaires entre les éléments. Une entité peut être hiérarchisée et peuvent avoir un identifiant unique représenté par un attribut nommés et typés. Les relations binaires expriment le rôle d'une entité source à une entité cible qui peuvent être influencées par une cardinalité que le WebML impose pour assure cette traduction de rôle.
Bref, le modèle structurel s'appuie sur une concordance et correspondance d'Entité /Relation. Les auteurs préconisent à la fois une approche objet pour satisfaire les contraintes liés à ce modèle.

ii. Le modèle de dérivation

Obtenu à partir du modèle structurel, il consiste à rajouter dans le schéma préétablit des informations nécessaires à la redéfinition du regroupement des données ainsi que l'augmentation de la faculté d'expression. Il est exprimé par le biais des requêtes permettant de construire des représentations supplémentaires dont le contenu est définit intentionnellement. Les dérivations servent à transférer les attributs d'une entité vers une autre, à définir des caractéristiques calculés, à créer des entités découlées regroupant des objets aux propriétés communes. Un langage inspiré du chemin OQL nommé WebML-OQL, est défini pour cela.

iii. Le modèle de composition

Le modèle de composition est utilisé pour délimiter le nombre de pages qu'un site peut constituer ainsi que les informations dont les pages sont dotées. Ce modèle offre une unité graphique et un contenu symbolisant une page de l'hypermédia. Ces unités aident à l'accomplissement de la détermination des pages grâce aux multiples possibilités qu'elles détiennent (possibilité d'intégration, de représentation des instances par instance, les visites guidées).

iv. Le modèle de navigation

Le prochain modèle nous informe sur la formation de l'hypertexte à partir de la liaison des pages du modèle de liaison en s'appuyant sur les deux variantes suivantes :

- Les liens contextuels permettant de promouvoir une relation reflétant le schéma de la « sémantique d'application » entre les unités. Ce lien a pour rôle de porter l'information de l'unité source vers l'unité de destination tout en déterminant les objets à transporter vers cette source. Les objets transportés dépendent fortement de l'unité source, c'est le cas du Data Unit dont le contexte est l'identifiant de l'instance montrée dans l'unité au moment de l'activation du lien .Pour un Filter Unit par contre on parle de valeurs d'attributs de l'utilisateur conformément au formulaire associé.

- Les liens non contextuels est un moyen de relier librement les pages dans le site, librement veut ici dire qu'il y a indépendance des unités qu'elles émanent et de la sémantique associée aux concepts des unités. Prenons l'exemple d'une page du site reliée à la page d'accueil par un lien non contextuel.

v. Extensions apportées aux modèles de composition et de navigation

Pour supporter les modes d'utilisation à part la consultation des informations, des concepts pour la saisie de données et les initiatives ont été ajoutés à WebML [107].Les extensions dont nous parlons ici consistent à l'ajout, la modification et suppression des données. Le modèle de composition regroupe deux types d'unités appelées Data Entry Unit et Operation Unit ; le modèle de navigation lui, propose des liens spécifiques aux deux unités.

L'unité Data Entry Unit collecte, dans des champs précis les valeurs données en entrée qui seront ensuite utilisées comme paramètres opérationnels. La valeur d'un champ appelée constante est extraite d'une base d'informations par l'utilisateur. Une propriété appelée « activating » est ensuite ajoutée aux liens du modèle afin d'indiquer que le lien résultant d'une telle unité active et requiert une opération. Un lien d'activation possède également des caractéristiques fondamentales favorisant le transfert des valeurs de paramètres du Data Entry Unit vers l'Operation Unit.

Inversement, une Operation Unit invoque une opération externe à l'application ou une opération pour la création, la modification et la suppression de contenu, et la création et la suppression de relations. Deux types de lien sont attachés en sortie d'une opération : d'une part , on a le OK-Link et d'une autre part on a KO-Link, qui indiquent vers quel élément du site aller si l'exécution de l'opération est en échec ou en succès.

vi. La définition de la présentation

La présentation est l'apparence des pages du site définie dans le précédent modèle de composition [44]. Les auteurs envisagent de recourir à des feuilles de styles communs indépendantes du contenu des pages et/ou de style granulaire plus fine qui s'appliquant à des notions spécifiques Pourtant, dans [45], on ne fait plus référence à un véritable modèle conceptuel spécifique à la présentation, mais à des dispositifs d'application de feuilles de styles XSL à des fichiers XML qui sont porteurs du contenu du site. Le modèle de présentation se synthétise alors à la précision des feuilles de style à relier aux fichiers de contenu.

vii. La personnalisation

Deux entités désignées Group et User [107] constituent le modèle Web ML, elles décrivent respectivement, un profil utilisateur ayant une certaine particularité et un utilisateur individuel. Ces entités sont reliées aux autres par des relations, structurées et agencées. Les entités

Group et User sont pour gérer lřaccès privée et personnel. Et pour contrôler cet accès, il est primordial de distinguer les utilisateurs enregistrés qui font partie dřune communauté définie selon leur besoin et les non enregistrés qui appartiennent à la classe everyone. Le site view ou la vision dřun site qui sont des constructions à partir du modèle structurel est défini selon le dispositif dřaccès et lřoptique de diffusion qui peut être soit un HTML ou un WML.

Pour ce qui est de la personnalisation, une vue du site peut être adaptée selon le profil dřun utilisateur, la personnalisation est donc appliquée de façon procédurale ou déclarative.

La personnalisation déclarative fait appel à des concepts dřentités et dřattributs qui se définissent selon les informations spécifiques à lřutilisateur. Le système trie, regroupe et récupère les informations de chaque utilisateur pour les utiliser lors de la spéculation du contenu des unités relatives aux notions découlés. Le calcul du prix dřun article, tient donc compte du pourcentage de remise accordé à lřutilisateur (information recueillie dans son profil), est réalisé de manière déclarative. La personnalisation procédurale repose sur une base XML proposée par WebML qui écrit les règles métiers représentées par un triplet événement-condition à remplir et lřaction à entreprendre permettant le calcul et le stockage de données de lřutilisateur.

viii. Processus de conception avec WebML

Le processus de conception nécessite la mobilisation de plusieurs techniciens compétents dont un expert des données pour la construction du modèle structurel, un architecte de lřapplication Web pour concevoir les pages, les unités et liens de navigation, un architecte du style Web pour les aspects relatifs à la présentation du site, et un administrateur Web pour la définition des utilisateurs, des groupes et de leur droits.

Tout dřabord, on procède par la collecte des besoins dřapplications par la détermination des objectifs du site dont les contenus attendus et dřéléments de personnalisation. Notons que WebML ne consiste pas à donner des propositions relatives aux besoins. Ensuite vient la seconde étape de conception des données par lřexpert des données qui détermine le modèle structurel et les dérivations essentielles.

Puis, on procède par la conception de lřhypertexte de manière générale en évaluant le nombre de visiteurs du site selon les groupes dřutilisateurs et leur dřaccès. Lřarchitecte met en évidence chaque page et leur unité composante ainsi que les liens. Lřarchitecte procède par calcul approximatif la première vue du site et adopte un technique « copier-coller ».

Lřhypertexte est ensuite élaboré à partir dřune conception précise sur la définition des pages crées durant la précédente étape.

Des index et filtres qui facilitent lřaccès aux informations sont introduits avec lřajout de différents chemins dřaccès par la suppression ou fractionnement des pages. Lřadministrateur web joue un rôle dřintégrateur dřaspects personnels et travaille le modèle structurel et sa dérivation pour y apporter des modifications car elle est jugée nécessaire.
Lřétape finale correspond à la de présentation, et elle nřest abordée que lorsquřil nřest plus nécessaire de retravailler les quatre modèles et quand lřensemble des pages dřune vue du site est stable.

ix. Conclusion sur WebML

- Notation pour la spécification conceptuelle des applications Web selon la dimension du contenu, sa gestion et la composition des pages.
- Mise en œuvre des capacités d'adaptabilité et d'adaptativité grâce à la description des règles et la recherche d'informations
- Approche modèle reposant sur le langage et s'appuyant sur un ensemble de concepts disponibles sous forme de représentations graphiques basées sur XML
- Démarche classée dans le Web Ratio Site Development Studio qui est un outil de conception commercial.

3.5.4. HDM: Hypertext Design Model

Ce modèle se base sur le modèle entité-relation se prolongeant avec une organisation hiérarchisé. HDM il est à la fois un outil d'identification des composants atomiques hypermédia et un critère d'assemblage des structures, ce modèle se soucie en effet de concevoir les aspects en large et de décrire l'agencement de l'hypertexte en ne tenant pas compte de la mise en œuvre spécifique [50]. L'entité représentée par les structures d'information est une information indépendante pouvant être relayé et supprimée de l'application.

Les entités sont assemblées dans des types d'entité comme le « Traviata » et l'Aida » qui appartiennent à l'entité « opéra musical ».

Une des caractéristiques les plus remarquables de HDM est l'admission de différentes catégories de liens qui tiennent compte des différentes relations qui se présentent entre les éléments d'informatifs, tels que la capture des relations de domaine et des règles de navigation ou plus exactement le mode de déplacement de l'utilisateur sur la structure hypermédia. Des liens de perspective nous aide à associer ses unités à un composant, prenons l'exemple du "Le baptême de Crist/italien-texte" et "Le baptême de Crist/italien-image" qui sont associés à la composante "Le baptême de Crist".

Bref, le modèle HDM est un groupement d'entités et de liens réels obéissant à des règles définies par le schéma HDM même. L'hypertexte est par conséquent constitué de données de navigation dont les guides et cartes de visites. Des liens se situant dans l'hyperbase ne sont pas mis en exergue dans le schéma mais peuvent recourir à un ajout dans l'application afin de donner un repère d'entrée pour l'utilisateur.

HDM et HDM2 permet la description du schéma mais ne fournit que quelques informations sur les étapes de conception et développement de l'hypermédia. Ils ne font que favoriser une méthode de développement proposée par HDM .Toutefois, c'est sur ce modèle que se fonde la gestion relationnelle RMM et l'Object-Oriented Hypermédia Design Model OOHDM.

3.5.5. WSDM: Web Site Design Method

i. Aperçu général de WSDM

La conception d'un site web WSDM [34] se différencie par la mise en valeur des souhaits du public cible. Au lieu de considérer les informations comme source de conception, WSDM spécule l'intérêt des utilisateurs : par opposition aux approches HDM, RMM, OOHDM-96, la

proposition devient une approche justifiée par lřexistence des utilisateurs et visiteurs dřun site. Il est primordiale dřétudier les attentes des cibles concernées afin dřaméliorer et de répondre à leurs besoins.

Pour concevoir le WSDM, on doit tout dřabord passer par la « Mission Statement » qui vise à annoncer la mission tout en répondant aux questions suivantes : quels sont les objectifs du site et qui sont les cibles ? Les réponses obtenues de ces questions feront découler la classification et la caractérisation du public cible ou proprement dite la spécification des utilisateurs.

La conception se divise en trois phases : modélisation objet et domaine, modélisation fonctionnelle puis conception navigationnelle. Après la mission Statement, on passe à la conception en vue de lřimplémentation pour terminer le WSDM par la phase dřimplémentation du site dans lřenvironnement HTML.

ii. Modélisation du public

La modélisation du public se subdivise en deux périodes. La classification des utilisateurs désignés dans [45] est menée à partir de lřétude des édits de mission du site lors de la première étape. Des classes utilisateurs sont donc identifiées et elles regroupent les utilisateurs qui ont une similarité de besoins en informations et en fonctions user requirements. Notons quřun utilisateur ne peut pas faire partie de plusieurs classes. [45] propose une approche arrangeant les catégories dřutilisateurs en hiérarchie.
De par ces catégories générales existent les sous-catégories ou perspectives se traduisant en sous- classes car elles partagent les mêmes usability requirements comme la langue utilisée, la compétence et les préférences utiles pour décrire les classes des utilisateurs [34].

iii. Modélisation conceptuelle

Modelisation Objet Et Modelisation Du Domaine

Elle intéresse la définition plus formelle des Usability Requirements évoquées dans les classes utilisateurs et perspectives par une notation OMT ou E-R. Une réunion dřobjets conceptuels ou objects chunk est élaborée pour chaque Usability Requirements, elle montre les différents objets et leurs relations. Ces chunks sont classifiés en un modèle appelé User Object Model (UOM) qui attribue la représentation du domaine dřapplication pour une classe utilisateur particulière.

A partir de chaque UOM, un Modèle Objet Perspective (Perspective Object Model Ŕ POM) est déterminé lorsquřil est juger nécessaire de donner des versions différentes de types dřobjets conformément aux spécifications des perspectives. Le modèle du domaine est acquis lors dřune étape de modélisation qui se synchronise à une réunion de tous les UOM.

Conception Navigationnelle

Durant la conception navigationnelle, on essaye de produire un modèle de navigationnelle. La seconde étape de la modélisation conceptuelle vise à produire le modèle de la navigation composée de navigation tracks décrivant la possibilité de cheminement à travers lřinformation. Ce modèle est composé de pistes de navigation dites navigation tracks qui décrivent les possibilités de progression à travers les données. Une association de navigation ou navigation chunk est spécifique selon chaque User Requirements. Les pistes de navigation modélisent le site web.

Modélisation en vue de l'implémentation

Elle repose sur la définition des pages qui constituent le site en structure et apparence externe. WSDM recommande de suivre des règles de conception pour aboutir à un site cohérent, convivial, attrayant. Par exemple, une page doit contenir des informations biens agencées pour que lřutilisateur ne soit pas dans un cas de surcharge, le concepteur doit aussi savoir guider son utilisateur par des points et signes de repérage.

iv. Conclusion sur WSDM

- Approche centrée sur lřutilisateur qui détermine ses besoins identifie des classes utilisateurs qui « indiquent » ces besoins.
- Description des variantes de classes utilisateurs pour alléger la modélisation des publics cibles.
- Inexistence de véritable modèle de lřutilisateur. Les informations des différents utilisateurs sont renvoyées au niveau du modèle de données.
- Construction des modèles objets, montrant ainsi une représentation du domaine dřapplication en concordance avec les particularités des variantes de classes.
- Lřadaptation du WSDM crée un hypermédia correspondant aux exigences des différentes catégories dřutilisateurs
- Processus à vision statique de lřadaptation.

3.5.6. OOHDM: Object-oriented Hypermedia Design Method

OOHDM (Object-Oriented Hypermedia Design Method) [17, 20, 40] est une approche basée sur des modèles pour finaliser des applications hypermédias. Il revêt quatre étapes dont la Modélisation conceptuelle, Conception de la navigation, Conception de lřinterface abstraite et Implémentation.

Dans la phase de *Modélisation conceptuelle*, le domaine dřapplication est modélisé selon les principes de lřapproche orientée objet avec une notation similaire à UML. Lřobjectif dans cette phase est de capturer la sémantique relative au domaine. Le résultat de cette étape est un ensemble de diagrammes de classes appelé řřmodèle conceptuelřř et représentant les différents concepts du domaine et les liens quřil peut y avoir entre eux.

Dans la phase de *Conception de la navigation*, les nœuds et pages de navigation, les contextes de navigation ou ensemble de nœuds de navigation ainsi que les chemins de navigation sont identifiés à partir du modèle conceptuel. Cette identification se fait dřaprès le profil de lřutilisateur final et lřanalyse de différents scénarios. Le produit résultant de cette phase est un diagramme de contexte représentant les contextes de navigation.

Dans la phase de *Conception de l'interface abstraite*, les objets de lřinterface perçus par lřutilisateur sont spécifiés au moyen de lř ADV (Abstract Data Views) [21], une notation absolue pour les concepts de lřinterface dřune application hypermédia. Ainsi chaque élément de lřinterface est décrit par sa structure, ses rapports avec les objets de navigation, et ses effets face aux événements extérieurs.
Dans la phase dřimplémentation, lřapplication finale est codée selon lřADV de la phase précédente. Les artefacts produits utilisés dans la méthode OOHDM sont présentés dans le Tableau 3.6.

	Modélisation Conceptuelle	Conception de la navigation	Conception de l'interface abstraite	Implémentation
Artefacts produits	Classes, sous systèmes, relations, attributs, perspectives	Nœuds, liens, structures d'accès, contextes de navigation, transformations navigationnelles	Objets de l'interface abstraite, réponses aux événements externes, transformations de l'interface	Pages Web

TABLEAU 3.6 : ARTEFACTS PRODUITS DANS LE CYCLE DE VIE SELON LA METHODE OOHDM [20]

i. Aperçu général de OOHDM

La méthode OOHDM (Object Oriented Hypermedia Design Methodology) [17][20] s'inspire tout comme RMM du modèle HDM. Elle se particularise toutefois des méthodes précédentes par l'usage d'une démarche de conception orientée objet. Les premiers articles sur OOHDM [17] révèlent que l'approche objet de renvoi est OMT [109]. Les prochaines articles laisse les auteurs dire que UML [108][109] est acceptable de par sa qualité de standard [53] et parce que leurs offres pour l'analyse des besoins [54] s'en inspirent.

Pour les auteurs, l'approche objet est essentiellement indispensable dans le cas d'applications informationnelles sujettes à des transformations ou simultanément des calculs complexes sont réalisés. Ils proposent par conséquent d'exprimer les actions correspondantes par la méthode de classes. Quatre méthodes de conception font l'objet d'OOHDM mais dans [54], les auteurs proposent une autre étape prolongeant le processus en amont des quatre autres.

Cette étape est consacrée à la collecte des besoins. OOHDM se base sur des spécifications permettant de connaitre les besoins des utilisateurs. L'activité de conception tourne autour de la relation entre l'utilisateur et le système de navigation. Le processus initial de OOHDM recourt à une modification: les étapes sont analogues, mais la démarche et les spécifications produites sont plus précises ce qui est le plus remarqué durant la modélisation conceptuelle et navigationnelle.

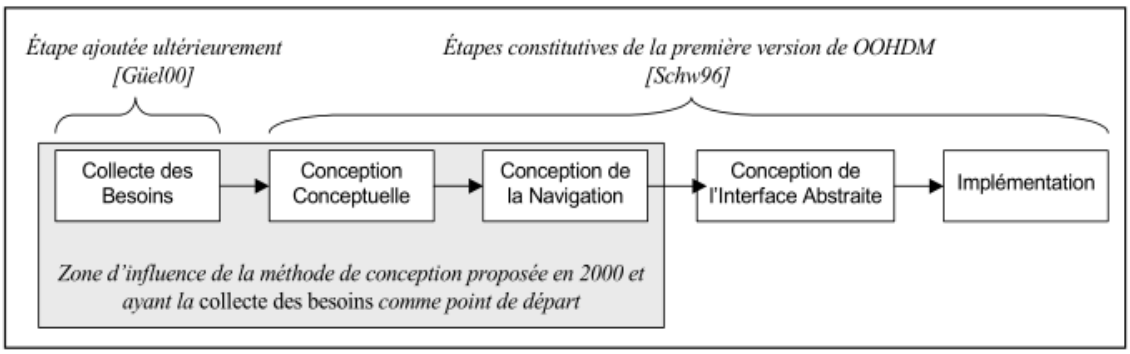

Figure 3.1 : Les étapes OOHDM dans sa version 2000. [29]

ii. Collecte des Besoins

Lřidentification des besoins permet par la suite de concevoir le schéma conceptuel et navigationnel cřest pourquoi cette étape est dřune importance majeure. La collecte des besoins est réalisée selon une succession de cinq étapes :

- Recherche des rôles au sens dřacteur dans UML et ainsi que les responsabilités qui doivent être tenues par lřapplication ;

- Catégorisation de scénarios ou descriptions narratives des tâches par les utilisateurs selon les fondements proposés dans [55].

Ŕ Spécification des cas dřutilisation [26], chacun regroupant les scénarios décrivant la même responsabilité.

Ŕ Spécification des Diagrammes dřinteraction Utilisateur (User Interaction Diagram Ŕ UID) [57] qui capturent les interactions observables, entre un utilisateur et une application.

Ŕ Validation ou acceptation des cas dřutilisation et diagrammes dřinteraction utilisateur à travers un entretien entre le concepteur et les utilisateurs impliqués.

iii. Modélisation Conceptuelle

La modélisation conceptuelle établit le modèle conceptuel de lřapplication selon lřUML. Elle vise à appliquer des règles de modification UID pour acquérir un diagramme de classes représentatif du domaine dřapplication. Ces normes, détaillées dans [57], se fondent essentiellement sur les techniques courantes de rationalisation des schémas définis dans [58].
Dřun point de vue général, le processus consiste, pour chaque UID, à trouver à partir des éléments qui le composent, les classes, les attributs de classes, et relations sémantiques entre classes.

Modélisation de la Navigation
Priorisée dans les deux versions OOHDM,elle est caractérisée par son essai de construction de modèle de navigation .Il sřagit notamment de faire une différence majeure par rapport aux propositions antécédentes et de considérer que les objets obtenus par navigation ne sont pas ceux du modèle conceptuel mais dřautres qui ont été élaborés par des mécanismes de vue ou à partir de quelques objets conceptuels. Le modèle navigation dans lřhypermédia sera obtenu à lřissue de différentes phases décrites ci-dessous.

Modelisation De La Navigation Correspondant à une tâche
Selon chaque fonction, les cas dřutilisation, scénarios, UID et du modèle conceptuel sont étudiés en vue de construire un contexte navigationnel [17]. Ces contextes sont un moyen dřorganisation de lřespace de navigation par la définition des unités constitués dřinformations, de liens et dřéventuels autres. Ils disposent dřune structure de navigation interne comme la séquentielle et lřindex, ainsi que dřun point dřentrée et des index reliés.

Les contextes navigationnels les plus ordinaires décrivent des réunions, en fonction des caractéristiques que les éléments doivent satisfaire et en fonction dřobjets issus dřune unique classe. Des explications graphiques propres aux contextes sont données dans [20] et lřidée générale est proposée dans [54] pour pouvoir établir ces contextes.

Production du diagramme de contexte navigationnel de l'application
Cette phase consiste à élaborer un seul diagramme à partir des contextes navigationnels décrits pour chaque fonction. L'union des contextes est menée à bien et d'éventuels ajustements sont entrepris afin d'obtenir un diagramme adéquat reflétant les besoins des utilisateurs. Ce diagramme doit encore être affiné car il ne possède pas tous les traits propres à la navigation dans l'hypermédia.

Spécification du modèle de navigation de l'application
La spécification repose sur un ensemble de références parmi lesquels figurent le diagramme final de contexte navigationnel et les cartes de précisions issus des étapes ultérieures.
Il y a tout d'abord un établissement du schéma des classes navigationnelles ou les nœuds qui décrivent le contenu à travers lequel l'utilisateur .Un nœud est une vue sur une ou certaines classes du modèle conceptuel, il possède des éléments obtenus à partir des liens de passation d'un nœud vers un autre.

Ces nœuds sont obtenus par des relations conceptuelles établissant des relations nouvelles. Un nœud peut être impliqué dans différents contextes. Il est jugé nécessaire d'exprimer un certain nombre de donnés qui sont complémentaires au contexte.

Pour cela, des classes spécifiques, appelées InContext [17], sont définis pour chaque nœud et chaque lien dans lequel il apparaît. Une classe InContext a une structure similaire à celle du patron de conception « décorateur » de Gamma. Quand un nœud est de même structure au patron décorateur est obtenu dans un contexte, les informations InContext correspondante s'additionnent aux informations du nœud de base.

Après, vient l'établissement d'une table de correspondance conceptuelle-navigationnelle. Dans le modèle de conception, une classe de base est choisit pour chaque nœud et l'on procède par comparaison des attributs et des méthodes.

iv. Conception de l'interface abstraite

Quand la structure de navigation est déterminée, les apparences des interfaces font l'objet de sujet. OOHDM se réfère à l'approche ADV (Abstract Data View [21] dans la description de l'interface utilisateur pour décrire l'interface utilisateur. Les ADV sont des objets interface associés aux objets du modèle OOHDM comme les nœuds et index et dont les attributs définissent les caractéristiques en termes organisationnel et d'apparence. Certains attributs arrangent la liste des événements que l'utilisateur peut provoquer et auxquels l'ADV doit prêter attention. L'aspect dynamique tel que le comportement à constituer par rapport à un événement sont décrits par des ADV-charts, résultant des statecharts.

v. Implémentation

C'est lors de cette étape que les objets conceptuels, de navigation et d'interface décrits par les différents modèles d'OOHDM sont traduits en fonction du milieu informatique choisi pour l'installation. Une ébauche concernant les obstacles et techniques recommandées pour établir les différentes communications est proposée dans [20].

vi. Conclusion sur OOHDM

La méthode OOHDM (version proposée en 2000 [54] est déterminée par un processus en cinq étapes débutant par un examen des tâches que les utilisateurs peuvent effectuer dans

lřapplication. Il introduit une dimension fonctionnelle dans les méthodes antérieures. Les besoins fonctionnels sont exprimés par des scénarios.

Ces spécifications font partie des étapes de modélisation conceptuelle et navigationnelle et chaque étape est définit par un schéma objet qui lui est spécifique : diagramme de contexte navigationnel, diagramme de classes navigationnelles et InContext. La détermination de l'interface permet de décrire les objets qui composent lřinterface de lřapplication à partir des unités de navigation. En définitive, la dernière étape se résume à la correspondance entre les éléments d'interface et ceux de l'implémentation.
OOHDM se différencie des approches antérieures par une approche de conception objet. Le degré de concordance offert entre les niveaux conceptuel et navigationnel favorise la réalisation de la méthode à un modèle conceptuel objet .Néanmoins, dans OOHDM lřadaptation permet surtout de choisir la version hypermédia que lřutilisateur veut, en fonction du groupe dřacteurs auquel il appartient. Dans les cartes de précision, les groupes (employés, dirigeants, etc.) auxquels sřadresse le contexte navigationnel fait référence sont mentionnés [54].

3.5.7. UWE: UML-based Web Engineering

UWE (UML-based Web Engineering) [64, 69] est une méthode servant à concevoir les applications Web. Elle se base sur une extension de la notation UML appelée « profil UML » selon la terminologie de lřOMG pour élaborer plusieurs formes de lřapplication Web tels que la navigation ou la présentation. Cette méthode regroupe quatre activités de modélisation qui sont lřanalyse des besoins, la modélisation conceptuelle, la modélisation de la navigation et la modélisation de la présentation.

La phase **d'*analyse des besoins***, détermine quels seront les différents utilisateurs de lřapplication ainsi que les différentes actions qui devront être faites. Elle permet donc dřidentifier les divers types dřutilisateurs de lřapplication ainsi que les fonctionnalités correspondant à chacun dřeux.

La phase de ***modélisation conceptuelle*** **utilise** les éléments de la phase ultérieure pour servir de base à lřélaboration UML. Cette phase a pour objectif principal la production du modèle de domaine en tenant compte de la navigation, la présentation et lřinteraction ainsi que les deux aspects modélisation de navigation et de la présentation qui seront nécessaires aux phases suivantes. La phase de ***modélisation de la navigation*** spécifie à la fois les objets visités par navigation et la manière de les atteindre. Le modèle de lřespace de navigation est un diagramme UML constitué de classes pénétrables identifiées par le stéréotype « navigational class ».

Des liens de « navigabilité directe » permettent le passage dřune classe vers une autre et cette structure de navigation transforme le modèle de lřespace de navigation et lřenrichissant avec des éléments de plus haut niveau comme lř « index », « visite guidée », « requête » et « menu ». Ces éléments permettent une structuration navigationnelle en spécifiant comment accéder à chaque page. La phase de ***modélisation de la présentation*** décrit comment les différentes structures de lřinformation décrit à travers accessible à travers lřespace de navigation sont présentées à lřutilisateur.
A chaque classe navigable et à chaque structure de navigation sont associées une ou plusieurs classes de présentation qui comprend des éléments basiques tels que řimageř, řancreř,

řcollections dřancresř. Ainsi, chaque structure de navigation est influencée par un modèle template pour sa présentation.
Les artefacts produits et utilisés dans la méthode UWE sont présentés dans le tableau suivant :

Etapes	Analyse des besoins	Modélisation conceptuelle	Modélisation de la navigation	Modélisation de la présentation
Artefacts produits	Cas dřutilisation en UML	Diagramme de classes UML	Diagramme de classes UML enrichis avec des classes et des structures de navigation	Classes de présentation (ou template)

TABLEAU 3.7 : ARTEFACTS PRODUITS DANS LE CYCLE DE VIE SELON LA METHODE UWE [69]

i. Aperçu général dřUWE

Classée dans lřapproche pour lřamélioration et le développement des applications hypermédias fondés sur le web, UWE (UML-based Web Engineering) [Koch00a] est une méthode qui couvre tout le cycle de développement des applications en cinq phases**: la création ou conception, l'élaboration, la construction, la transition et la maintenance**.

Les quatre premières reprennent et ajustent celles du processus Unifié (PU) [26] à partir duquel le processus proposé dans UWE. La cinquième phase élargi le PU par la prise en compte dřactions liées à la confirmation et la gestion dřun système qui, après sa création nécessite des ajustements au niveau du contenu, la présentation ou encore lřagencement de lřhypermédia.

Dans le cadre de cet état dřart, nous ne décrirons plus la totalité du processus, mais notre étude sera plutôt axée sur lřétude dřUWE et les propositions relatives à lřabstract et à la conception de lřapplication. Nous mentionnerons également que lřapproche repose sur un méta-modèle, appelé Munich Reference Model [62], formalisant les caractères et principes de fonctionnement et progression des applications hypermédias adaptatives. La description de ce méta-modèle nřest toutefois indispensable si lřon veut comprendre exactement la proposition décrite.

UWE propose une explication et une méthode [62]. Lřexplication présente offre des règles de modélisation qui combinent les notations dřUML [UML99] et des extensions spécifiques dřun profil UML[63,64]. A travers un processus de création itératif et incrémental, la méthode permet lřanalyse des besoins, la modélisation conceptuelle, la conception de la navigation et celle de la présentation, ainsi que la modélisation des utilisateurs et des adaptations.

Les dimensions abordées donnent lieu à six modèles, représentés par des paquetages. Les relations de dépendance entre les packages révèlent les repères visibles existant entre les processus permettant dřaboutir à ces modèles.

ii. Analyse des besoins

Lřanalyse des besoins dans UWE se veut être une approche établie sur les cas dřutilisation et consiste à identifier et satisfaire les besoins fonctionnels des différents utilisateurs auxquels doit répondre lřapplication.

Les entités de création utilisées à cette étape sont ceux proposés par UML. Les notions d'acteurs, de cas d'utilisation, de relations d'héritage, d'utilisation et d'extension entre eux, ainsi que les mécanismes de paquetages et de vues sont repris et conservent la sémantique décrite par UML. Les notations graphiques sont également celles d'UML.

C'est lors de cette étape que la démarche conduit les propositions communes vers les processus par les cas d'utilisation. Il s'agit en général ainsi que les activités qu'ils entreprennent. Ces activités sont révélées par des cas d'utilisation auxquels sont reliés les acteurs. Si nécessaire, des relations d'utilisation et d'extension sont définies entre certains cas d'utilisation, ainsi que des relations d'héritage entre acteurs.

iii. Modélisation Conceptuelle

Cette étape vise à établir le modèle conceptuel du domaine d'application en se référant aux besoins et décrit le diagramme UML qui organise les concepts du domaine sur les UML suivants : classes, associations et paquetages. L'exactitude des deux derniers éléments permet aux classes du modèle conceptuel UML d'avoir une subdivision optionnelle s'ajoutant aux attributs et opérations. Ce compartiment ou subdivision est fait pour recevoir des variantes qui forment des autres informations utilisées pour adapter le contenu de la classe en fonction de chaque profil utilisateur.

La démarche de la modélisation conceptuelle consiste à identifier les classes et à spécifier pour chacune ses attributs et opérations, elle veut aussi relier les classes par des associations et à remarquer d'éventuels problèmes exprimés en OCL (Object Constraint Language).

iv. Modélisation de la Navigation

La modélisation de la navigation comprend en fait deux étapes, conduisant à l'élaboration de deux modèles. D'abord, un modèle de l'espace de navigation permet de décrire quels objets atteints par la navigation. La structure de navigation exprime comment ces objets sont atteints dans l'application. Nous pencherons à l'étude successive des deux modèles et les étapes associées.

Modelisation de l'espace de Navigation

L'espace de navigation est modélisé grâce aux classes navigation et aux associations navigation qui sont tous deux des entités du UWE.

La première présente une catégorie dont une classe navigation représente une classe dont les applications sont atteintes par l'utilisateur lorsqu'il navigue dans l'hypermédia. Les caractéristiques d'une classe navigation sont celles de la classe conceptuelle homologue. Il est pourtant possible d'omettre certains traits de la classe conceptuelle lorsqu'ils conviennent au modèle de navigation c'est-à-dire qu'il n'est plus utile de les concrétiser dans l'hypermédia.
Il est aussi faisable de donner à la classe navigation des caractéristiques qui n'appartiennent pas à la classe conceptuelle correspondante. Ces attributs doivent dérivés à partir d'une classe du modèle conceptuel qui n'apparaissant pas dans le modèle de navigation. Ils sont évalués à partir d'une expression OCL.
La seconde association navigation exprime une éventualité d'accès direct à une classe navigation cible à partir d'une classe navigation source. Cette classe navigation est représentée sur le schéma par une flèche dirigée et stéréotypée par l'expression « direct navigability ». Le sens de cette flèche montre la direction de l'association navigation, qui est uni-directionnelle. On parle par contre de bidirectionnalité lorsque l'association navigation être parcourue dans les

deux sens .En effet, toute extrémité fléchée dřune association navigation est dotée dřun nom de rôle et dřune abondance qui exprime respectivement le rôle de la classe cible.

Les associations du modèle conceptuel sont modifiées en associations navigation quand les classes du modèle conceptuel quřelles relient possèdent des classes navigation dans le modèle de lřespace de navigation. Des contraintes en OCL permettent de délimiter des limitations sur le modèle de lřespace de navigation qui prend par la suite une forme de diagramme de catégorie UML en mettent en relief des éléments statiques de deux types.

Les associations navigation se reflètent dans le modèle de lřespace de navigation dřUWE, La façon dont cette navigation est réellement maintenue dans lřhypermédia est décrite par le modèle de la structure de navigation que nous allons à présent décrire.

Modélisation de la structure de navigation
Un modèle de concrétisation des faisabilités de navigation dans le modèle de navigation est produit grâce à la modélisation de la structure de navigation. Des primitives dřaccès définissent le modèle de structuration à partir du modèle dřavant. La structuration recourt à trois étapes bien distinctes.

Les primitives dřaccès que sont les index, les visites guidées et requêtes augmentent le modèle de navigation, la primitive dřaccès introduite par UWE exprime une requête en OCL et est modélisée par une classe stéréotypée ou index, guide tour et query.

La démarche spécifique à UWE procède par le suivi des règles de modification du modèle source, ces règles sont imprécises concernant certains aspects .Le modèle de structure est modifié par le biais de lřajout dřun quatrième type dřaccès ou menu qui est un intervalle de classe liant une classe MenuItem avec une composition. Les menus sont des accès directs aux classes hétérogènes contrairement à lřindex qui nřoffre accès quřà un seul type de classe ou classe navigation.

Pour introduire des menus au modèle de la structure de la navigation, on associe un menu à une classe navigation initiale. La liaison de ces deux types est réalisée par une composition. Les rôles des associations navigation sont alors les items du menu et ce sont eux qui sont en liaison avec la cible.

La modélisation de la structure de navigation se termine par lřajout des propriétés aux différents éléments du modèle pour spécifier et différencier des contraintes en vue dřune lřadaptation adéquate. Nous pouvons citer comme exemple la propriété direct guidance attribuée à une visite guidée doit indiquer le meilleur nœud à choisir en fonction de lřutilisateur. Il en est de mêmes pour la propriété annotated en concordance avec un index ou menu et qui devra circonscrire les mécanismes des éléments.

v. Modélisation de la Présentation

Elle est principalement composée dřun ensemble de vues montrant la forme et le contenu des nœuds de lřhypermédia en décrivant la façon dont lřutilisateur et les nœuds sont en concordance. UWE propose deux étapes pour la modélisation de la présentation : le premier concerne la conception des storyboards et le second le flot de présentation.

La création de storyboards repose sur trois éléments décrits par le profil UWE dont une classe stéréotypée « UI View » ou encore User Interface View , Une instance de Vue Interface

Utilisateur ou VIU qui permet de grouper les unités qui doivent être présentées de manière simultanée. Lřélément suivant est une « presentation class », qui divise une VIU en unités basiques Le troisième élément est une classe abstraite ou User Interface Element associée à une classe présentation par une composition.

Cette dernière classe se spécialise en diverses classes stéréotypées comme le « text », « image », « video », « audio », « form » ou « anchor ». Ces deux dernières classes sont considérées comme des éléments dřinteraction cela veut plus exactement dire que la classe « anchor » est attachée à un lien de navigation, la classe « form »soumet lřinformation, déclenche lřévénement .Ces éléments ont une représentation graphique singulière.

vi. Modélisation des utilisateurs

La modélisation des utilisateurs a pour but de déterminer les spécificités à retenir pour concevoir des profils utilisateurs et instaurer des liens entre eux et leurs relations avec les entités de modèle représenté dans un diagramme de classes.
La construction du modèle utilisateur requiert la recherche des caractéristiques adaptées au domaine dřapplication en vue de lřadaptation. Par exemple, une classe Type de Paiement indiquant le mode de paiement de lřutilisateur est justifié en lřapplication de commerce électronique mais nřa pas forcément la même valeur et intérêt dans dřautres cas. Des classes peuvent sřadapter et sřassocier avec les autres.

vii. Modélisation de lřAdaptation

Lřadaptation dřUWE est fondé sur un concept de règle modélisé par une classe de type « rule» et lié par une relation dřune classe « action » avec une classe « condition » qui sont liées à des classes des modèles utilisateur. Des règles dřadaptation spécifient comment le contenu, navigation, et présentation doivent être adaptés.

Les règles dřacquisition décrivent la manière de sřapproprier les informations de lřutilisateur et de mettre à jour son modèle en décrivant ses comportements examinés par le système à partir de la classe « User Behaviour » Les règles conditions et actions précitées sont exprimées en langage naturel formel avec OCL.

Des diagrammes de collaboration sont élaborés pour mettre en évidence les enchaînements aléatoires des règles. Bref, cette démarche reprend les spécifications dans les multiples modèles en les exprimant sous forme de règles.

3.5.8. Comparaison entre les différentes méthodes de développement des applications Web

Evaluer les sites web est une tâche encore très peu connue et pratiquée. La recherche des mensurations pour lřévaluation quantitative de la qualité des applications Web nřen est quřà ses premiers termes. Il est devenu essentiel pour les industries de développer ces applications en mettant au point des principes et méthodes dřapproches spécifiques.

Il est vrai que des guides de style, principes et techniques de conception spécifiques aux applications Web, apparaissent par une rapidité incroyable. Pourtant, très peu de travaux ont été effectués et publiés sur lřévaluation quantitative de la qualité de cette application.

Lřimportance de la qualité de ces applications et leurs interfaces utilisateurs nřest plus à démontrer. Les concepts définis pour le développement des connexions humain-ordinateur pour des applications conversationnelles traditionnelles sont pertinents et satisfaisants pour des applications Web. Quoique, les caractéristiques générales de ces applications sollicitent les considérations additionnelles.

La qualité des interfaces peut avoir un impact important sur la réussite de lřapplication même au plan. Donc, pour réaliser un bon système web tant dans la structure que le contenu, il est tout dřabord nécessaire dřétablir une manière qualitative évaluative des sites web.

Lřémergence du Web, à partir des années 90, a constitué un grand changement permettant de considérer et réaliser différemment la diffusion et le traitement de lřinformation. En tant quřapplication de lřinternet, le Web offre un espace large dřinformations sans limite offertes par cette infrastructure. Au commencement, sa création découle des travaux autour du concept dřhypertexte dont il reprend les fondements organisationnels non-linéaire des contenus de données hébergés dans des machines éloignées accessibles grâce aux multiples chemins de navigation.

Les hypertextes type image, son, vidéo ou encore multimédia, ont vocation à présenter les informations utiles au site web qui sont présentées par une approche hypermédia. Ces applications constituent un SI ou système dřinformation basé sur le web désigné par SIW [32,35]. Un SIW forme un ensemble constitué de ressources humaines et de moyens techniques, de règles de fonctionnement, de données et de procédés pour acquérir, mémoriser, transformer, rechercher, communiquer et restituer ces données [32].

Le Web constitue alors une base pour la création des systèmes. Un élément différenciant lřapproche SIW par rapport à une approche SI influence concrètement le processus de conception. Les SIW mettent en œuvre, comme les hypermédias, une organisation non-linéaire dřinformations reliées entre elles par plusieurs chemins, ces derniers multipliant les possibilités dřaccès à une même information [29]. La conception des SIW doit, en conséquence, porter sur les deux dimensions suivantes :

- Une activité dřorganisation des informations en vue de la présenter car jugée comme base de navigation, il est interdit de lřignorer dans le processus de développement des SIW.

- Nécessité dřune activité courante aux aspects fonctionnels ou à la conception des services auxquels doit répondre le SIW, est nécessaire.

Les applications Web présentent certaines caractéristiques et applications classiques différentes des logiciels traditionnels. Ces caractéristiques environnementaux tels les approches ainsi que la vitesse de développement des applications les rendent différents et compliqués par rapport au développement des logiciels traditionnel. Les aspects liés à la sécurité de ces applications est plus important et plus critique que ceux dédiés aux applications classiques [27].

Par les caractéristiques distinguant les SIW des SIT, le recours aux logiciels traditionnels de conception est difficile et inadéquat. En effet, les aspects de présentation et de navigationn ne sont pas prioritaires dans un processus de développement de SI non basés sur le Web. Par contre, les méthodes de conception dřhypermédias ont particulièrement attiré à la structuration

des pages, à lřélaboration des liens entre elles pour la navigation, et à leur apparence lors de la présentation. Nonobstant, la conception dřun SIW nřest pas une activité banale ni une simple de manipulation de médias et de création de présentations.

En outre, [100] met en évidence combien cela est observé dans les faits à travers une étude visant à évaluer les applications sur le Web [100].Les approches de conception dřhypermédias nřappréhendent pas de façon systématique le milieu fonctionnel du système. En effet, le public cible dřun SIW est beaucoup plus large que celui dřun SI traditionnel, même sřil est toutefois possible de limiter lřaccès dřun SIW à une population spécifique en ayant recours à des politiques de sécurité comme lřintranet.

Dans le cas dřun SIW ouvert, la population dřutilisateurs est plus large et plus diversifiée cette pluralité des utilisateurs se constate à plusieurs niveaux relativement similaires par exemple à leurs buts, leurs préférences en terme de présentation, leurs connaissances.
Dřune autre manière, comparées aux applications traditionnelles, elles présentent plusieurs caractéristiques propres tout en étant de taille et de complexité de plus en plus élevées [41]. En constatant que ces applications évoluent très précipitamment, même si la plupart des sites Web sont pauvrement conçus et présentent des déficits sur les plans aussi bien de la navigabilité, de la fonctionnalité que de la traçabilité [41].
Dans la plupart des cas, les sites sont confus, trop lents et ne répondent pas aux besoins des utilisateurs. Des études récentes montrent que 90% des sites ne satisfont pas les objectifs de qualité préalablement établis par les concepteurs [103]; et ceci pour différentes raisons :

- Technologie sous-jacente non conçue pour les applications en temps réels ;
- Technologies du concepteur non équitable à celles de lřutilisateur
- Développement des sites Web amateurs focalisant trop seulement lřaspect visuel des interfaces. Il en résulte des sites lourds, difficiles à utiliser, et ne répondant pas aux besoins des usagers ;
- Manque de standards dřingénierie et techniciens associés à la production de sites Web de haute qualité.

Dans lřoptique de choisir entre les différentes méthodes de développement des applications Web, elles sont comparées par évaluation de leur base de critère dont la modélisation de lřutilisateur, dimensions de lřadaptation, modélisation et mise en œuvre des adaptations, et capacités dřadaptation du SIW conçu [29].

La modélisation des utilisateurs permet de définir si lřutilisateur peut être lřobjet de représentation dans les approches étudiées Le tableau suivant nous en dira beaucoup plus

NOM DU CRITÈRE	DESCRIPTION
Existence	Exprime s'il existe ou non un modèle décrivant spécifiquement les utilisateurs.
Granularité	Décrit le niveau auquel sont représentés les utilisateurs, i.e. groupe, individu, les deux.
Caractéristiques :	Précise les informations qui permettent de représenter un groupe ou un utilisateur.
Connaissances spécifiques au Domaine d'Application	Indique si les connaissances de l'utilisateur en la matière peuvent être représentées dans le modèle (i.e. niveau de compétence, d'expérience en informatique, etc.).
Connaissances Autres	
Conditions d'Exploitation	Indique si les conditions (matérielles ou logicielles) dans lesquelles l'utilisateur exploite l'application sont représentées.
Fonctionnalités	Indique si une représentation des fonctionnalités (objectif, buts) exploitées par l'utilisateur est donnée.
Préférences :	Indique s'il est possible pour l'utilisateur d'exprimer ses préférences et à quel propos :
Contenu	Représentation d'un concept, granularité de l'information souhaitée
Structure de l'hypermédia	Composition des nœuds en pages et gestion des liens
Présentation	Charte graphique personnalisée
Acquisition	Exprime comment sont collectées les informations à propos des utilisateurs
Droits sur le modèle Utilisateur	Exprime les droits de l'utilisateur à l'égard des informations le concernant (i.e. Aucun / Lecture / Écriture)

TABLEAU 3.8 : CRITERES DřEVALUATION RELATIFS A LA MODELISATION DES UTILISATEURS [29].

La vision synthétique de lřévaluation des méthodes en regard de la modélisation de lřutilisateur est récapitulée dans le tableau ci-dessous. Les cases vides dans le tableau montrent que RMM et OOHDM nřont pas de modèle en la matière et ne sont donc pas évaluées sur les autres critères .Les autres propositions utilisent un modèle des utilisateurs ou un modèle à part entière pour WSDM, Hera, UWE, alors que WebML intègre la représentation des utilisateurs dans le concept dřinformations.

NOM DU CRITÈRE		RMM	OOHDM	WSDM	HERA	AWIS-M	UWE	WEBML
Existence		non	non	oui	oui	oui	oui	En tant que partie du modèle conceptuel
Granularité				groupe	individu	groupe (individu)	individu	groupe / individu
Caractéristiques	Connaissances spécifiques au Domaine d'Application			(oui)	non	oui, à définir	oui, à définir	oui, à définir
	Connaissances Autres			non	non	oui, à définir	oui, à définir	non
	Conditions d'Exploitation			non	oui	non	non	non
	Fonctionnalités			oui	non	oui	non	non
Préférences	Contenu			non	non	non	non	non
	Structure de l'hypermédia			non	non	non	non	non
	Présentation			(oui)	oui	non	oui, à définir	Non
Acquisition				explicite	explicite / implicite	explicite / implicite	implicite	implicite
Droits sur le modèle Utilisateur				non précisé	non précisé	non précisé	non précisé	non précisé

TABLEAU 3.9 : ÉVALUATION DES METHODES CONCERNANT LA MODELISATION DES UTILISATEURS. [29]

WSDM propose un modèle de représentation des groupes d'utilisateurs élaborés sur la base de leurs besoins en informations et données. Les ?uir reportés en italique pour les critères de connaissances spécifiques au domaine d'application et préférences de présentation traduisent que l'exploitation de telles informations est suggérée par les auteurs mais décrite cependant de façon informelle.

Une représentation des utilisateurs individuels dans Hera permet de spécifier les conditions matérielles et logicielles dans lesquels le système est exploité. Des considérations en termes de présentation sont également représentées. Le modèle repose sur le vocabulaire CC/PP proposé par le W3C [106].

UWE représente les caractéristiques des utilisateurs en tant qu'individus par la création des classes appropriée dans le modèle. Seule la représentation des connaissances est envisagée. Les fonctionnalités ne sont pas considérées par le modèle c'et à dire que c'est dans la construction de la structure de l'hypermédia que les modalités sont définies. Des préférences de l'utilisateur liées à la présentation sont également exprimées dans les classes spécifiques.

Les utilisateurs sont représentés dans des classes de type structurel grâce au Web ML, les groupes sont définis sur la base de fonctions communes à plusieurs utilisateurs, qui peuvent être plus compliqués qu'une simple navigation. Les utilisateurs sont principalement caractérisés par des informations conformes au domaine qui sont exploitables dans des règles de gestion métier : l'application cible de Web ML étant un outil de commerce électronique, aide à classer les individus par des données telles que le taux de remise qui lui est accordé.

Pour ce qui est de l'acquisition des informations caractéristiques d'un modèle utilisateur, les procédures ne sont pas clairement évoquées dans les approches. Pour WSDM, le concepteur construit explicitement les informations concernant les fonctionnalités de chaque groupe.. Notons que Dans Hera, une partie des informations sur le profil doit être donnée explicitement par le concepteur ou l'utilisateur.

Il est utile de souligner qu'aucune offre ne mentionne clairement les droits de l'utilisateur à l'égard du modèle le concernant. Notamment, la lecture et la rectification des caractéristiques et préférences par l'utilisateur lui-même devraient être autorisées même si de telles informations sont automatiquement déterminées par le système.

Dimensions adaptées
Les critères d'évaluation de cette rubrique permettent d'identifier les dimensions qui font l'objet d'adaptation dans les approches présentées.

NOM DU CRITÈRE	DESCRIPTION
Contenu	Exprime s'il existe des adaptations possibles du contenu présenté à l'utilisateur.
Structure	Exprime s'il existe des adaptations possibles de la structure de l'hypermédia, i.e. des nœuds et de leur composition.
Navigation	Exprime s'il existe des adaptations possibles de la navigation dans l'hypermédia, i.e. des liens et structures d'accès.
Présentation	Exprime s'il existe des adaptations possibles de la présentation dans l'hypermédia, i.e. de la mise en page et de l'apparence des éléments.
Fonctionnalités	Exprime s'il existe des adaptations possibles concernant les fonctionnalités offertes par l'hypermédia, i.e. adaptation de l'espace fonctionnel et guidage.

TABLEAU 3.10 : CRITERES D'EVALUATION RELATIFS AUX DIMENSIONS ADAPTEES [29].

Pour chacun des critères, nous indiquons dans le Tableau 3.10 les éléments de modélisation proposés ou les techniques employées à ces fins. Notons que nous mentionnons les aspects qui, selon nous, pourraient être utilisés à des fins d'adaptation même si, dans certaines propositions (RMM, OOHDM), l'adaptation n'apparaît pas clairement supportée (notamment en raison de l'absence d'un modèle utilisateur). Une case vide traduit l'absence de proposition en la matière.

Les éléments de modélisation, les techniques, les aspects des critères sont proposés, selon des propositions RMM et OOHDM, une adaptation n'est toujours pas supportée (case vide dans le tableau).

Dans RMM, Hera et OOHDM, les adaptations en termes de contenu, structure et navigation dans l'hypermédia sont mélangés car le modèle de l'hypermédia est fortement couplé au modèle du domaine. UWE et Web ML proposent une autre représentation du domaine qui expose les possibles adaptations de contenu tout en définissant une vue du site selon les groupes d'utilisateurs.

UWE ajoute le compartiment 'variant' à la description des classes UML alors que Web ML adopte un modèle de dérivation défini sur le modèle informatif. Il s'agit aussi de vues mais, à la différence d'OOHDM, ce modèle est indépendant de toute représentation hypermédia. En termes de structure et de navigation, UWE intègre des propriétés dans le modèle de structure de navigation qui en contraignent les spécifications.

Des modèles (User Object Model et Perspective Object Model) qui présentent la vision du modèle de données de chaque groupe d'utilisateur décrient par WSDM ne garantie pas l'uniformité des descriptions avec les nœuds de l'hypermédia. Les adéquations concernant la navigation sont en fait induites par les deux modèles ci-dessus : les pistes de navigation reflètent les possibilités de cheminement dans les éléments du modèle.

WebML définit une vue du site distinctif à chaque groupe mais également spécifique à chaque méthode d'accès du groupe. Des feuilles de styles XLST adéquates sont définies, traitant ainsi une configuration d'adaptation de la présentation. Hera adopte une approche plus simple en définissant des conditions de présentation sur les m-slices et en exploitant celles-ci selon les préférences des utilisateurs et les conditions d'exploitation dans différents processus d'adaptation.

Bref, OOHDM, WSDM et Web ML sont des propositions d'adaptation des fonctionnalités du SIW. Ces approches ont en commun une construction de différentes versions de l'hypermédia des groupes d'utilisateurs. En conséquence, les pages auxquelles peuvent accéder les utilisateurs constituent un espace restreint qui répond à leurs besoins informationnels et fonctionnels.

<table>
<tr><th>CRITÈRE</th><th>RMM/HERA</th><th>OOHDM</th><th>WSDM</th><th>AWIS-M</th><th>UWE</th><th>WEBML</th></tr>
<tr><td>Contenu</td><td rowspan="3">(RMM)
m-slice</td><td rowspan="3">Navigational contexts, Navigational classes et classes Incontext</td><td>User Object Model et Perspective Object Model</td><td rowspan="2">Élément Atomique de Navigation, Nœud de Navigation Adaptable</td><td>Classes Variantes</td><td>Modèle de dérivation</td></tr>
<tr><td>Structure</td><td></td><td rowspan="2">propriétés exprimées dans le modèle de structure de Navigation</td><td rowspan="2">une vue du site pour chaque groupe</td></tr>
<tr><td>Navigation</td><td>Navigation track</td><td>Directive de Navigation</td></tr>
<tr><td>Présentation</td><td>(Hera)
m-slice, adaptability et adaptativity conditions
et update rules</td><td></td><td></td><td></td><td>dépend des spécifications du modèle utilisateur</td><td>une vue du site pour chaque groupe et dispositif d'accès</td></tr>
<tr><td>Fonctionnalités</td><td></td><td>Navigational contexts, Navigational classes et classes Incontext définis en fonction des tâches</td><td>User Object Model et Perspective Object Model définis en fonction des besoins</td><td>Unité Sémantique de Navigation définies en fonction des objectifs</td><td></td><td>Une vue du site pour chaque groupe</td></tr>
</table>

TABLEAU 3.11 : EVALUATION DES METHODES CONCERNANT LES DIMENSIONS ADAPTEES [29].

Modélisation et mise en œuvre de l'adaptation

A travers cette étape, nous essayerons d'identifier s'il existe une phase d'analyse des besoins en matière d'adaptation et si les méthodes ont ou non recours à un modèle conceptuel d'adaptation.

Ce tableau synthétise les résultats de cette analyse. On remarque l'existence d'une case vide indiquant l'absence de proposition. Dans WSDM, AWIS-M et UWE, une phase d'analyse vise à déterminer les besoins fonctionnels des utilisateurs et des besoins en matière d'adaptation

dans la mesure où elle conduit à la détermination de différents groupes dřutilisateurs en fonction desquels différentes versions de lřhypermédia seront élaborées. Toutefois, lřanalyse fonctionnelle ne suffit pas à déterminer réellement des attentes relatives aux capacités dřadaptation du système.
En outre, OOHDM et WSDM ne présentent pas de modèle conceptuel de lřadaptation et Web ML ne propose pas de modèle spécifique. La spécification déclarative correspond à des caractérisations introduites dans les classes utilisateurs qui permettent dřémaner le modèle de dérivation. La spécification procédurale est exprimée par des règles événement Ŕ condition Ŕ action. Il est important de préciser à quelle structure la règle est liée. Hypothétiquement, elle est intégrée dans les descriptions des unités du modèle de composition.

Dans Hera, les éléments pour lřadaptabilité sont définis dans le modèle conceptuel de lřhypermédia sous forme de conditions sur les m-slices. Elle repose sur un modèle spécifique de règles. Lřhypermédia délivré à lřutilisateur est construit en fonction de ces spécifications. Dřune part, les éléments qui satisfont les conditions décrites dans le modèle conceptuel sont retenus pour composer les pages.

Cřest sur la base des spécifications quřil convient à construire lřhypermédia selon une approche descendante : les Unités Sémantiques de Navigation (USN) sont sélectionnées en fonction de la catégorie de lřutilisateur, puis les Nœuds de Navigation (NN) en fonction du profil de la catégorie dřutilisateur, et enfin, les Éléments de Base (EB) en fonction du degré dřappartenance de lřutilisateur à ce profil. Enfin, UWE propose un modèle spécifique de lřadaptation qui exprime sous forme de règles spécifiques insérées dans les autres modèles.

CRITÈRE	OOHDM	WSDM	WEBML
Analyse des besoins en matière d'adaptation		Analyse des besoins fonctionnels – détermination de groupes (Mission Statements)	
Modèle conceptuel de l'adaptation			Pas de modèle particulier mais spécification de règles de personnalisation déclarative et procédurale
Technique de mise en œuvre de l'adaptation	Choix de la version correspondant au groupe parmi différentes versions prédéfinies	Choix de la version correspondant au groupe parmi différentes versions prédéfinies	Choix de l'hypermédia adapté à chaque groupe (site view)

CRITÈRE	HERA		AWIS-M	UWE
	Adaptabilité	Adaptativité		
Analyse des besoins en matière d'adaptation			Analyse des besoins fonctionnels – détermination de groupes	Analyse des besoins fonctionnels – détermination de groupes (use cases)
Modèle conceptuel de l'adaptation	Conditions incluses dans le modèle conceptuel de l'hypermédia	Modèle spécifique de règles d'adaptation	Décrit par le modèle mathématique de l'hyperespace	Modèle d'adaptation défini sur la base des spécifications ajoutées aux autres modèles (classes Rule, Condition, Action, User Behaviour)
Technique de mise en œuvre de l'adaptation	Recherche des conditions satisfaites au moment de la présentation	Règles événement – condition – action traitées par un moteur d'adaptation	Construction dynamique d'un hypermédia adapté à chaque groupe par sélection des USN et des NN et par assemblage des EB en fonction de l'utilisateur	Implémentation d'un moteur d'adaptation

TABLEAU 3.12 : CRITERES D'EVALUATION RELATIFS A LA MODELISATION ET MISE EN ŒUVRE DE L'ADAPTATION [29].

Capacités d'adaptation du SIW conçu

Ces derniers écrits ont pour buts d'évaluer les capacités d'adaptation que SIW peut atteindre à partir des méthodes étudiées. Notons que les définitions retenues pour les différentes capacités d'adaptation combinent les définitions de l'adaptabilité et de l'adaptativité présentées auparavant.

NOM DU CRITÈRE	DESCRIPTION
Adaptation minimale	Cette capacité correspond aux systèmes qui permettent de choisir une version parmi plusieurs définies au préalable
Adaptabilité	Cette capacité est définie comme l'aptitude du système à réagir à des demandes explicites d'adaptation de la part de l'utilisateur
Adaptativité	Cette capacité est définie comme l'aptitude du système à observer le comportement de l'utilisateur et à procéder en conséquence à des adaptations.

TABLEAU 3.13 : CRITERES D'EVALUATION RELATIFS AUX CAPACITES D'ADAPTATION DU SIW [29]

La combinaison de ces deux approches apparaît à travers les deux dimensions. La simplification des définitions retenues est liée au fait que seuls les deux cas de processus d'adaptation sont considérés.

Les constats établis grâce aux précédentes rubriques sont présentés dans le Tableau 3.13. Les conventions utilisées dans le tableau sont les suivantes : le caractère « » indique que le SIW présente la capacité d'adaptation associée, le caractère « » traduit que cette capacité n'est pas observable dans le système, l'association de ces deux caractères « / » exprime que la présence de cette capacité est discutable. La capacité d'adaptation minimale est observable dans tous les SIW conçus avec les méthodes présentées.

D'une part, il est évidemment nécessaire que l'installation veille à ce que la programmation respecte les modèles et les principes. D'autre part, la seconde condition porte sur l'obligatoire identification de l'utilisateur et du groupe auquel il appartient afin de pouvoir choisir la version

du SIW que le concepteur devra lui présenter. RMM, OOHDM et WSDM proposent uniquement de choisir la version qui correspond à lřutilisateur.

CRITÈRE	RMM	OOHDM	WSDM	HERA	WEBML	AWIS-M	UWE
Adaptation minimale	+	+	+	+	+	+	+
Adaptabilité	–	–	–	+ / –	+ / –	+ / –	+ / –
Adaptativité	–	–	–	+ / –	+ / –	+ / –	+

TABLEAU 3.14 : EVALUATION DES CAPACITES DřADAPTATION DES SIW CONÇUS AVEC LES METHODES ETUDIEES [29].

Pour les capacités dřadaptabilité du SIW, lřévaluation est plus atténuée. Les raisons à cela sont principalement liées, dřune part à lřabsence de description de lřutilisateur sur sa propre représentation dans le système, et, dřautre part, aux préférences de lřutilisateur prises en compte par les modèles. Dřune manière générale, Hera et Web ML, et UWE présentent certaines caractéristiques pouvant laisser supposer que les SIW offrent des capacités dřadaptabilité. Toutefois, aucune description ne positionne clairement ces approches en termes dřadaptabilité.
Le développement web officialisé depuis 1998 concerne les pages HTML et le développement de logiciel avec les technologies modernes virtuelles. Lřapplication des ces technologies permet de mieux échanger les informations entre plusieurs ordinateurs et de coordonner les tâches des trois composantes web dont le réseau qui lie entre eux, les ordinateurs puis le poste client qui affiche les documents hébergés sur un autre ordinateur et enfin un poste hôte qui sert à localiser les informations demandées par le client. Lřapplication web se diffère des sites web par sa capacité dřinformation et de navigation et opération et sa facilité de manipulation et création.

Dans [31] lřauteur soulève quatre grandes catégories par rapport à la complexité des applications web : dřabord, les sites de présence sur le web caractérisés par un faible degré de complexité en diffusant les informations, ces sites sont construits à partir des éditeurs HTML. Ensuite, les sites catalogues à forte densité de données en publiant une quantité importante de données. Puis les sites orientés services qui sont les moteurs de recherche et enfin, et enfin les SIW qui mettent à disposition la gestion des données jugés complexes avec des services interactifs modernes.

Les méthodes de conception des applications web se diversifient en plusieurs étapes selon les outils en autres le RMM Relationship Management Model qui a pour principale fonction la conception des hypermédias et la gestion des éléments informatifs, elle a une grande capacité de structuration et modification des informations cřest pourquoi les auteurs le qualifie comme efficace et crédible.

Les étapes du RMM sont la conception des applications web, la conception des slices et la conception du contenu des entités, la construction des diagrammes de slices. Il convient utile de réexpliquer les limites de la méthode RMM qui réside dans son incapacité à modéliser les pages web complexes ainsi que la non flexibilité du slice et la conception de lřhypermédia. Pour remédier à ses limites [46] préconise les diagrammes et les m-slices construits à partir des attributs.

Le processus de changement et de remplacement sont appliqués de manière ascendante et descendante qui, combinés aident à comparer les diagrammes dřapplications et à identifier les problèmes durant la phase de conception.

A part RMM, les applications web disposent aussi de la méthode HERA pour la création de la présentation et des adaptations en prenant en compte des critères individuels de lřutilisateur et les derniers sites quřil a visité. La conception dans un projet HERA se fait en deux niveaux dont la manifestation des liens de navigation et la production des diagrammes de présentation. Dans cette méthode, la liaison des m-slices et de la région peut affecter un mêmes slice dans différentes régions. Cette liaison se base sur trois relations dont la relation de navigation, les relations temporelles, les liens hyperliens de navigation.

La conception des adaptations spécifie lřapparence des m-slices en matérialisant leur relation, les adaptations en HERA sont lřadaptabilité et lřadaptativité qui mettent en œuvre et conçoit les dimensions de mémorisation et la taille de lřécran. Bref, HERA est une étape de passage dřun niveau logique vers une phase de finalisation. Elle est une étude à la fois statique et dynamique de la présentation et changement de la navigation.

Le Web ML quant à lui est une codification qui se veut graphique et se basant sur le langage XML des applications complexes. Il sert à structurer les données et les pages. Cette phase procède par lřassemblage des besoins, la conception des données, la création de lřhypertexte, la conception de la présentation, la conception des utilisateurs, et enfin la phase de conception de la personnalisation.
La collecte des données détermine les objectifs du site par rapport aux utilisateurs, la création du modèle de structure ; la phase de création de lřhypertexte construit les modèles de navigation et les nœuds et aussi la mise en relation des pages. Pour ce qui est de la phase de conception de la présentation, elle assemble les feuilles de styles et présente de manière adéquate les contenus ; la conception des utilisateurs est faite pour décrire le profil utilisateur ou identité personnelle qui profite dřun UML et élaborer des données lui correspondant.

En finale la phase de conception de la personnalisation comme son nom lřindique est une étape de personnalisation de la page dřaprès les données de lřutilisateur. Elle sřaccomplit suivant le comportement de lřusager encore appelée personnalisation déclarative parallèle ainsi que la catégorisation de lřusager à correspondre à un groupe de personnes ayant des traits dřinformations communs.

Le Web ML est un des travaux HDM Lite en adaptant ses propositions à la généralité spécifique du web. Lřun de ses points forts est cette représentation graphique grâce à lřXML. En applications web, on remarque plusieurs dimensions à la base du Web ML aboutissant aux modèles de structure, dřagencement, et de navigation.

Le modèle de dérivation consiste à augmenter la faculté dřexpression et à rajouter dans le diagramme des éléments pour redéfinir les données. Ce modèle sert à transférer les attributs dřune entité vers une autre et à créer des entités.

Le modèle structurel façonne les éléments dans le site directement en relation avec les entités qui sont hiérarchisées et représentées par des attributs nommés ou typés. Le modèle structurel sřappuie sur un lien et une communication entre lřentité et la relation.

Le modèle de composition sert à délimiter le nombre de pages et informations quřun site peut englober, il propose un contenu symbolisant lřhypermédia qui va déterminer les pages.
Le modèle de navigation qui constitue les liens textuels et non contextuels forme lřhypertexte et relie librement les pages en les rendant indépendantes des unités.

Pour améliorer les éléments saisis dans le site web, des initiatives [70] ont été prises et ceci dans le cadre de pouvoir mieux gérer, modifier, ajouter et supprimer dřautres sous-éléments ou données.

3.6. Les techniques, méthodes et outils Internet, Intranet, Extranet

Pour se faire un site web doit recourir à lřemploi des techniques et méthodologies tel lřinternet, lřextranet et lřintranet. Ces trois techniques sont reconnues pour leur capacité de diffusion des informations dans le temps et lřespace. On assiste aujourdřhui à une forte émergence dřapplications et de systèmes collaboratifs de gestion de contenus en ligne, de syndication, dřimportation et dřexportation de données. En plus de développer ces nouvelles applications, les entreprises spécialisées dans le développement dřapplications Web doivent maintenant faire face aux demandes accrues de pseudo migration des applications Desktops vers le Web. Par conséquent, des milliers dřapplications de différents domaines sont déployées sur le Web.
Les applications traditionnelles Desktop tendent à perdre leur valeur à cause de lřapparition du Web. Les utilisateurs sont beaucoup plus à lřaise avec la navigation par le biais dřun navigateur, et la conception des interfaces est désormais possible avec le web. Le web permet aux administrateurs du système de ne gérer que quelques serveurs contrairement à lřépoque du Desktop. Le concept de Web.2.0 a augmenté le degré de complexité des applications web mais a ouvert un chemin électronique grâce aux sites web administratifs et gouvernementaux.

3.6.1. Intranet

Par définition, lřintranet est un réseau informatique privé à l'intérieur d'une organisation et qui utilise les technologies Web (protocoles et applications TCP/IP) [110]. Centre francophone d'informatisation des organisations, Office de la langue française (OLF), s.d.). Cet outil permet de gérer, de partager et de répertorier les informations et les connaissances, de communiquer et de collaborer et de faciliter lřaccès lřéchange et lřaccessibilité aux informations en utilisant les différentes interfaces et les données dřapplications.
Intranet est donc un système dřinformation web permettant de communiquer et de collaborer en toute liberté en profitant dřun échange fluide et cohérent dřinformation via le web. Lřopérationnalité, la qualité et lřefficacité dřun intranet repose sur son contenu qui doit être optimisé, sa technologie qui doit être actuelle et performante et de sa « culture informationnelle positive ».
Le bon fonctionnement dřun outil intranet nécessite également lřintervention de professionnels et le déploiement de ressources financières et budgétaires de choc Les intervenants humains sont utiles pour la conception du contenu, la gestion dřintranet et les interférences avec les utilisateurs.

3.6.2. Extranet

Lřextranet est un réseau informatique constitué des intranets de plusieurs entreprises qui communiquent entre elles, à travers le réseau Internet, au moyen d'un serveur Web sécurisé [77]; Centre francophone d'informatisation des organisations ; whatis.com, s.d.)., définit quatre rôles principaux des extranets :

- le partage d'information sous la forme de publication électronique,
- la communication stratégique entre les compagnies et leurs bureaux distants ainsi que les travailleurs mobiles,
- l'accès à des applications clés de traitement de l'information comme le suivi de la production et des commandes, et
- la sécurité des échanges .

En général, lřextranet est surtout utilisé par les professionnels et les entreprises qui souhaitent échanger des informations entre elles par le biais dřun réseau électronique. Etant surtout utile en affaire, il nécessite alors des compétences en processus dřaffaires. De plus, ses concepteurs doivent avoir une grande connaissance des systèmes de sécurité réseautiques. Cependant, les technologies dont il use sont similaires à ceux utilisés pour lřinternet et pour lřintranet.

Pour résumer, on peut aisément dire que les intranets et les extranets qui usent dřune même et seule technologie : le web. Cependant, on note leur portée différente et leurs usages souvent opposés.

3.6.3. Comparaison entre Intranet, Extranet et Site web externe

Certains auteurs et chercheurs contestent lřappartenance du site web externe à la famille des SIW. Cependant, [110] intègre ce dernier dans les SIW et établit une comparaison entre les intranets, les extranets et les sites web externes que nous allons représenter dans le tableau suivant.

Extranet

Les définitions d'extranet sont très similaires et se synthétisent ainsi :
Extranet : Réseau informatique constitué des intranets de plusieurs entreprises qui communiquent entre elles, à travers le réseau Internet, au moyen d'un serveur Web sécurisé [77].

Les extranets répondent à quatre tâches principales : (1) le partage d'information sous la forme de publication électronique, (2) la communication stratégique entre les compagnies et leurs bureaux distants ainsi que les travailleurs mobiles, (3) l'accès à des applications clés de traitement de l'information comme le suivi de la production et des commandes, et (4) la sécurité des échanges [36]. Une de leurs utilisations principales est le support du commerce électronique interentreprises. Les clients pour lesquels le système est développé sont centraux à ce type de SIW. La planification initiale de l'extranet devra donc tenir compte de cette clientèle mais aussi des objectifs de l'extranet et de sa portée. Bien que les technologies utilisées soient les mêmes que pour les sites Web externes et les intranets, le déploiement effectif des technologies extranet demande en plus une compréhension approfondie des processus d'affaires [36,77,110].

La littérature sur les tâches et les compétences professionnelles requises pour la mise sur pied

d'un extranet est encore plus mince que celle sur le commerce électronique. Aucun texte n'a été trouvé qui identifie explicitement les différentes tâches et compétences professionnelles. Il faut noter que puisque les extranets sont considérés comme une des principales avenues pour le commerce électronique, il est possible qu'ils n'aient pas fait l'objet d'un développement spécifique à ce niveau et relèvent plutôt de la littérature sur le commerce électronique. Certains auteurs utilisent effectivement les deux concepts indistinctement. Mais les caractéristiques générales d'un extranet peuvent servir à identifier les grandes familles de compétences professionnelles requises [77].

Partageant avec les autres SIW les mêmes technologies Web, les extranets vont posséder les caractéristiques de base qui se retrouvent dans le contexte des sites Web externes sur les plans technologique et informationnel ainsi qu'au niveau de l'interface graphique et de l'interaction système-utilisateurs. Le contexte des extranets donne une importance accrue aux compétences professionnelles technologiques reliées à la sécurité des réseaux. De plus, comme pour les intranets, il est possible de rencontrer un besoin d'intégration de systèmes existants qui demandent des compétences professionnelles spécifiques tant opérationnelles que de gestion. Finalement, les utilisateurs étant au cœur des extranets, le processus de définition des besoins est particulièrement important ainsi que la compréhension des processus d'affaires [36].

Intranet

L'intérêt pour les intranets est de plus en plus grand et les organisations les adoptent rapidement [110]. L'intranet est présenté comme un outil qui permet :

• la gestion, le partage et le repérage de l'information et des connaissances;

• la communication et la collaboration [36]; car l'intranet peut être utilisé comme collecticiel [59]; à des coûts beaucoup moins élevés que certains produits commerciaux ;

• la facilitation de l'accès à l'information en offrant une interface intégrée aux différentes bases de données et applications.

Cependant, un écart entre ces utilisations anticipées (collecticiels, réseaux locaux, bibliothèques virtuelles, systèmes de gestion des documents et SIW) et les utilisations actuelles qu'en font les organisations est observé. Le manque de modèles de développement d'intranets peut expliquer en partie cette difficulté [36].

i. Dimension technologique

Phase de la planification stratégique

Le développement de normes techniques supervisé par un comité technique Web est un élément qui ne ressort que dans la littérature sur les intranets ainsi que la définition des mécanismes d'accès, de copies de sécurité et d'archivage [36]. Les tâches liées aux stratégies de sécurité et à la définition des types de fichiers n'apparaissent pas dans la littérature sur les intranets.

Phase de la conception et de l'opérationnalisation

L'intranet ayant entre autres comme objectif de supporter le travail interne, une dimension "intégration des systèmes existants" s'ajoute au niveau de la conception et de l'opérationnalisation. L'univers plus contrôlé et fermé de l'intranet crée un environnement

propice au développement et à l'implantation d'outils de recherche [36], activités moins présentes dans la littérature sur les sites Web externes.

Phase de la gestion des ressources

Les écrits sur l'intranet ajoutent un volet de gestion des ressources humaines "technologiques" que sont les webmasters [127] ainsi que celui de la gestion des outils de recherche [126];[154]. La gestion des risques est une tâche absente pour les intranets.

ii. Dimension informationnelle

Phase de la planification stratégique

Les tâches nouvelles trouvées dans la littérature pour la planification stratégique de la dimension informationnelle des intranets sont l'identification des besoins informationnels ainsi que le développement des politiques, normes et guides de style. Ces activités, prises en charge par des individus comme le rédacteur ou l'éditeur ou par des comités comme un conseil Web, demandent des compétences professionnelles liées au développement de politiques, à l'identification de besoins ainsi qu'à la communication orale et écrite. L'élaboration des règles de navigation, présente dans les écrits sur les sites Web externes, est absente pour les intranets [36].

Phase de la conception et de l'opérationnalisation

À la phase de la conception et de l'opérationnalisation, la littérature sur les intranets présente de nouvelles tâches d'analyse du contenu ainsi que de traitement du contenu, d'extraction et d'accessibilité des connaissances [36].

Les tâches de vérification de la qualité de l'information, du repérage et de l'élaboration de la navigation ne se retrouvent pas dans la littérature consultée sur les intranets.

Phase de la gestion des ressources

Seule la dimension de la gestion des ressources diffère sensiblement car elle est beaucoup plus détaillée au niveau de l'intranet qu'au niveau des sites Web externes. La gestion des ressources pour la dimension informationnelle des intranets ajoute les tâches de gestion des ressources humaines et des normes informationnelles, de gestion de l'organisation de l'information, de gestion de la révision du contenu, de la coordination des éditeurs et de la gestion des échéances de rédaction ainsi que de la gestion des activités de veille [36].

iii. Dimension "interface graphique"

Seule la phase de la conception et de l'opérationnalisation de la dimension "interface graphique" d'un intranet est abordée dans la littérature consultée tandis que, pour les sites Web externes, la phase de planification stratégique est aussi présente. L'intégration des bases de données et applications patrimoniales existantes propres au contexte des intranets crée un besoin pour le développement d'interfaces spécifiques. Cependant, les aspects plus graphiques comme le développement de logos et de bannières sont moins présents, l'importance d'accrocher" visuellement les utilisateurs étant peut être moins grande dans le contexte de la clientèle plus "captive" des intranets.

iv. Dimension "interaction système-utilisateurs"

Phase de la conception et de l'opérationnalisation

La dimension "interaction système-utilisateurs" sur le plan de la conception et de l'opérationnalisation est abordée par les auteurs au niveau du développement et de la prestation de formations. Ces formations couvrent un plus grand éventail de thématiques que celles présentées dans la littérature pour les sites Web externes. Les sujets des formations mentionnés dans les écrits sont la création de pages Web, l'évaluation de sites Web externes, la recherche de documents sur l'intranet, ainsi que l'utilisation générale de l'intranet. La sensibilisation ainsi que l'intégration des caractéristiques des utilisateurs n'apparaissent pas dans les écrits sur les intranets [36]

Phase de la gestion des ressources

Au niveau de la gestion des ressources, la présence d'un coordonnateur pour la formation est observée [36,77,110]

TABLEAU 3.15 : [36].

4. LIGNES DE PRODUITS AUX SERVICES DES SYSTEMES D'INFORMATIONS WEB

Ce chapitre traite lĭutilisation des lignes de produits logiciels (LPL) dans Systèmes dĭnformation Web.

4.1. Architecture de ligne de produits pour le design des applications web

Au cours des dernières années, les systèmes Web progressent allant de la collection des pages statiques non évolutives vers des applications complexes. La pertinence économique et la complexité croissante de ces applications nécessitent des techniques modernes convenables et des modèles qui peuvent offrir un meilleur retour sur les temps de développement et les facteurs de qualité. Par contre, les méthodes traditionnelles sont souvent basées sur les simples compétences des programmeurs individuels et n'appliquent pas toujours les principes du génie logiciel et les apports des techniciens web.

Comme les applications Web ont souvent des comportements similaires, mettre lĭaccent sur le déplacement de la conception d'applications simples et individuelles vers celle de la famille de systèmes est un moyen efficace pour poursuivre la réutilisation du logiciel et sa pérennité. En particulier, les systèmes basés sur le Web peuvent être considérés comme des produits logiciels issus d'une infrastructure et des biens communs qui capturent lĭabstraction spécifique dans le domaine, par exemple le panier, la caisse, et l'enregistrement de l'utilisateur dans un système de vente en ligne.

La réutilisation de logiciels est une vision simple mais puissante qui vise à créer des systèmes de logiciels artefacts existants plutôt que de construire des systèmes en partant du début de la conception. Exploiter les points communs entre les membres d'une famille de produits ainsi que les familles de produits similaires devient un moyen efficace pour poursuivre la réutilisation du logiciel et assurer son efficacité. Une ligne de produit logiciel se compose généralement d'une architecture de lignes de produit ou Product Line Architecture (PLA), un ensemble de composants réutilisables et un ensemble de produits dérivés des ressources partagées en commun.

Chaque produit hérite en architecture de la PLA, sollicite et configure un sous-ensemble des composants de la ligne de produit et contient généralement un code de produit spécifique permettant une meilleure catégorisation et différenciation. Le potentiel majeur pour la PLA est de simplifier la conception et l'entretien des familles des programmes et de répondre aux besoins des applications hautement personnalisables de manière rentable.

En utilisant une ligne de produits logiciels, les développeurs sont en mesure de se concentrer sur des questions spécifiques sur chaque produit plutôt que sur les questions qui sont communes à tous les produits [65,74,76,86]. En résumé, une PLA est un modèle pour une

famille d'applications dépendantes qui vise à une réutilisation des applications pour pouvoir assurer une durabilité et une efficacité. Le concept clé du développement des familles est la variabilité du système, comme la capacité destinée à dériver divers produits à partir de la famille de produit.
La variabilité est caractérisée et associée aux points de variation, c'est-à-dire que des lieux de conception et une mise en œuvre sont nécessaires pour atteindre un intervalle de fonctionnalité. Chaque point de variation a un ensemble de variantes qui expriment sa variabilité, les points ne sont pas liés jusqu'à une variante particulière et sont sélectionnés, jusquĭà ce quĭils doivent être liés à cette variante.

Lĭassociation à chaque point de variation est un temps de liaison, durant lequel la résolution du point de variation a lieu. Les Temps de liaison typiques sont la configuration et la réalisation de l'architecture ainsi que de la sélection des composants de paramétrage. La configuration des composants de paramétrage vise à instaurer les différents paramètres tandis que le temps de démarrage et d'exécution constitue lĭimplémentation du site.

La Manipulation de la variabilité est une rude tâche du fait que les différences entre les produits d'une même famille de produits peuvent être décrites en termes de fonctionnalités et de caractéristiques de chaque produit. Une caractéristique est une unité logique de comportement qui est spécifiée par un ensemble d'exigences fonctionnelles et de qualité.

Par conséquent, les caractéristiques réalisent un moyen de faire abstraction des exigences et cela encore grâce à la catégorisation. Les exigences sont liées aux caractéristiques d'une relation *n*-to-*n*. La modélisation d'entité est une approche importante pour la capture des points communs et des variabilités dans les familles de systèmes et les lignes de produits.

Plusieurs méthodes ont été proposées pour identifier les modèles, parmi elles figure lĭanalyse de domaine FODA (ou Feature Oriented Domain Analysis) qui est souvent désignée comme l'une des plus émergentes et fiables car elle fournit à la fois un processus pour déterminer les caractéristiques communes et les variables d'instances du concept ainsi que leur interdépendance et une notation pour les représenter dans des modèles longs constitués de longs diagrammes dĭinformations avec quelques données supplémentaires telles que les courtes descriptions sémantiques de chaque fonction, les contraintes mais surtout la dépendance par défaut de règles.

Les produits et éléments au sein d'une famille de produits sont généralement développés dans les étapes qui ont tendance à être asynchrone et indépendantes, c'est à dire une ingénierie de domaine et une ingénierie d'application en cours d'exécution en même temps. Cet asynchronisme est dû au fait que les produits sont des milieux dĭédification et dĭidentification ; cĭest plutôt à partir de la famille de produits que lĭon procède à lĭexécution.

L'ingénierie de domaine implique, entre autres, l'identification des points communs et des différences entre les produits membres de la famille et la mise en œuvre d'un ensemble de logiciels partagés de telle sorte que les points communs peuvent être exploités économiquement, tout en préservant en même temps la possibilité de faire varier les produits ou de les conserver.

Au cours de cette phase de variation, les points de variation sont conçus et un ensemble de variantes est associé à chacun d'eux pour effectuer les tâches ensemble. Les différents procédés d'ingénierie du domaine sont des composants logiciels fondamentalement

réutilisables et dont les configurations et analyses des modèles de conception sont exigées. En général, aucun produit réutilisable ne dénomme une ressource réutilisable.

Pendant l'ingénierie d'application, des produits individuels sont dérivés de la famille de produits et construits en utilisant un sous-ensemble des artefacts logiciels partagés. Il est dĭautant plus nécessaire de créer des actifs spécifiques aux produits supplémentaires ou de remplacement.

Cĭest dans cette phase que chaque point de variation est lié à une variante spécifique choisie dans lĭensemble de ses variantes associatives comme nous a montré lĭétape précédente. Les étapes mentionnées ci-dessus constituent deux cycles de développement indépendant relatif c'est-à-dire un développement du point de vue de la réutilisation connu sur le terme technique de produit dĭinstanciation et de la ligne de produit elle-même. Une mise à jour des applications est maintenue par lĭinstitut de génie logiciel.

Peu de travaux relatent lĭimplication des lignes de produits dans la conception des systèmes dĭinformation. Cependant, les lignes de produits sont une approche très répandue dans les applications web dĭaujourdĭhui. Dans la littérature on trouve les travaux de [76]. Ils traitent une architecture de lignes de produits pour les applications web.

Actuellement, les applications Web sont de plus en plus utilisées dans des environnements similaires pour accomplir des tâches similaires et rendre uniformes et équitables les éléments de conception de lĭhypermédia. Le partage d'une infrastructure commune et la réutilisation des actifs de déploiement et le développement des services récurrents qui se répètent plusieurs fois peuvent être considérés comme un avantage en termes d'importance économique et de qualité globale.

Ainsi, il peut être opportun de concevoir des applications Web en tant que membres d'une famille de produits et de les classer hiérarchiquement dans une catégorie englobant toutes les applications relatives à la création de site web.

4.2. KORIANDOL, une PLA spécifiquement conçue pour le web

Selon le modèle d'organisation fourni par Koriandol [76], une application Web peut être considérée comme une association de plusieurs pages. Une application Web est un produit consistant en un certain nombre de pages qui sont hiérarchisées dĭaprès leur contenu et leur ordre de priorité. Chacune d'elles recueille un contenu de façon dynamique fourni par les composants sélectionnés.

Comme il est mentionné précédemment, les composants sont génériques, c'est-à-dire qu'ils sont conçus pour capturer les points communs et les variabilités dĭune classe de comportement dans un domaine d'application. Par exemple en tant que fonctionnalités typiques d'un système de vente en ligne, le panier, le catalogue, et la caisse sont réalisés à différents degrés de complexité. Chaque méthode offre une autre fonctionnalité et les valeurs admises pour ses paramètres formels sont représentatifs pour les variantes de fonctionnalité dont la sélection résout un point de variation à un seul bond.

De manière analogue, les méthodes choisies dans une page sont liées une fois que tous leurs points de variation aient été résolus. Un composant est un composant lié dès quŭne partie de sa méthode est choisie dans le produit.

Comme il est dit ci-dessus, les composants de Koriandol incarnent aussi une intégrité dans le mécanisme de la manipulation de la variabilité dřapplication dynamique qui permet aux ingénieurs de lier dynamiquement chaque point de variation à la variante appropriée. Selon les sélections de l'utilisateur précédent, le mécanisme révèle à travers l'introspection applicable les points de variation et leurs variantes associées. Ceci permet de naviguer et dřopérer un modèle de fonctionnalité sur le composant.

La composante englobe deux méthodes dřapplication qui sont le résumé dŭun article affichant une liste de nouvelles et d'un article de contenu respectif. Elle est représentée en tant que solution de rechange pour les sous-composants du composant, on ne peut donc sélectionner quŭun composant à la fois. Le résumé à son tour, a une obligation de sous-fonction qui spécifie le nombre de nouvelles à résumer, et une autre en option qui applique un filtre de catégories.

Une correspondance entre la notation de longs diagrammes et de widgets HTML dont des cases à cocher ou des boutons radio a été définie. En exploitant cette cartographie, la gestion de la variabilité et les mécanismes peuvent être générés automatiquement à partir des spécifications déclaratives. Des diagrammes de fonction spécifiques à un domaine ont été proposés afin de fournir un texte commode de représentation.

Le temps de liaison typique pour le composant et la méthode de sélection composent la configuration d'architecture. Dans cette approche, la détermination de la variabilité peut être réalisée à la fois pendant la phase de post-déploiement ou période dřexécution ainsi que pendant la phase d'instanciation, ce qui rend les applications largement reconfigurables.

Lŭune des principales contributions de Koriandol depuis cette approche est dřaméliorer les capacités de gestion post-déploiement de la variabilité de la phase d'ingénierie d'application [76]. En outre, au cours de lřédification du produit dřinstanciation, la variation non révélée précédemment peut offrir une variabilité supplémentaire à l'application des ingénieurs. Théoriquement, cela signifie qu'ils ont la possibilité de décider du nombre de variabilité à exploiter une tâche de conception qui est généralement limitée à l'ingénierie de domaine.

L'architecture proposée peut être utilisée pour développer une large gamme d'applications en fonction des domaines dont les abstractions et leurs relations sont capturées par marque actifs. Bien entendu, la nature des produits dépend nécessairement des composants disponibles et au cas où certains comportements manquent, le concepteur devrait se soucier de leur analyse, leur conception et leur réalisation.

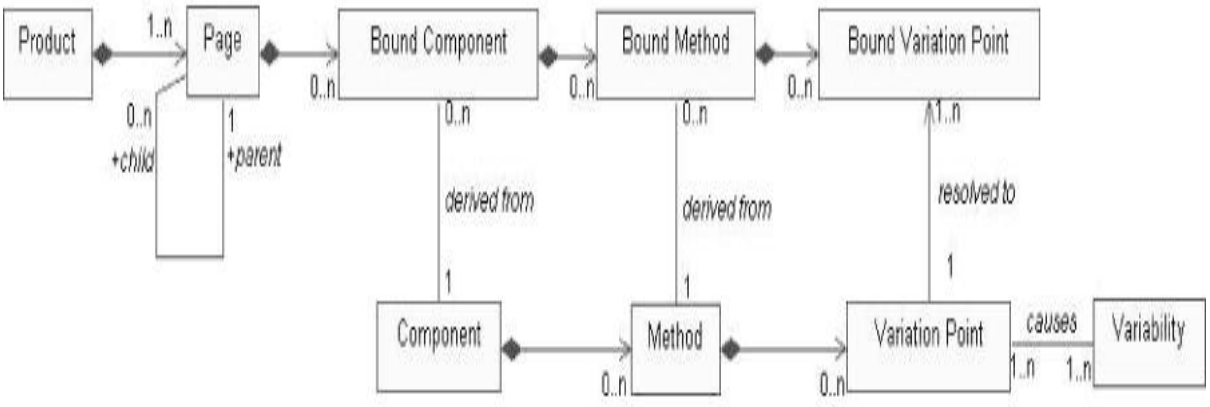

Figure 4.1 : Associations among the assets in a product [76]

4.2.1. Outil de support

Le système Koriandol compose des modules de logiciels spécialisés qui réalisent l'infrastructure commune du produit de l'architecture de la ligne décrite dans les sections précédentes [76]. De plus, certains modules auxiliaires, principalement pour la gestion et la manipulation de configuration sont aussi fournis. Les unités de base suivantes sont parmi les plus importantes:

- la gestion
- le module de configuration,
- le terme de l'environnement en temps et
- le moteur de présentation.

Quand une demande de l'utilisateur arrive, le module d'exécution interprète grâce à WRT la spécification de la page correspondante. En d'autres termes, il valide les privilèges et exigences des utilisateurs afin d'accéder à la page et, au cas où les utilisateurs ont accès libre à la page. Le module identifie les composants qui doivent être invoqués pour servir cette demande particulière. Chaque fonctionnalité de la page est récupérée par une composante, c'est à dire un appel de méthode pour le composant sélectionné et configuré au cours de la variabilité pour la détermination de l'instanciation de la page.

La méthode renvoie le code XML qui est passé au moteur XSLT, un modèle qui génère du code HTML au moyen de feuilles de transformation. Une fois tous les composants et les demandes réalisés, les segments HTML obtenus sont mis ensemble et renvoyés à lĩutilisateur.

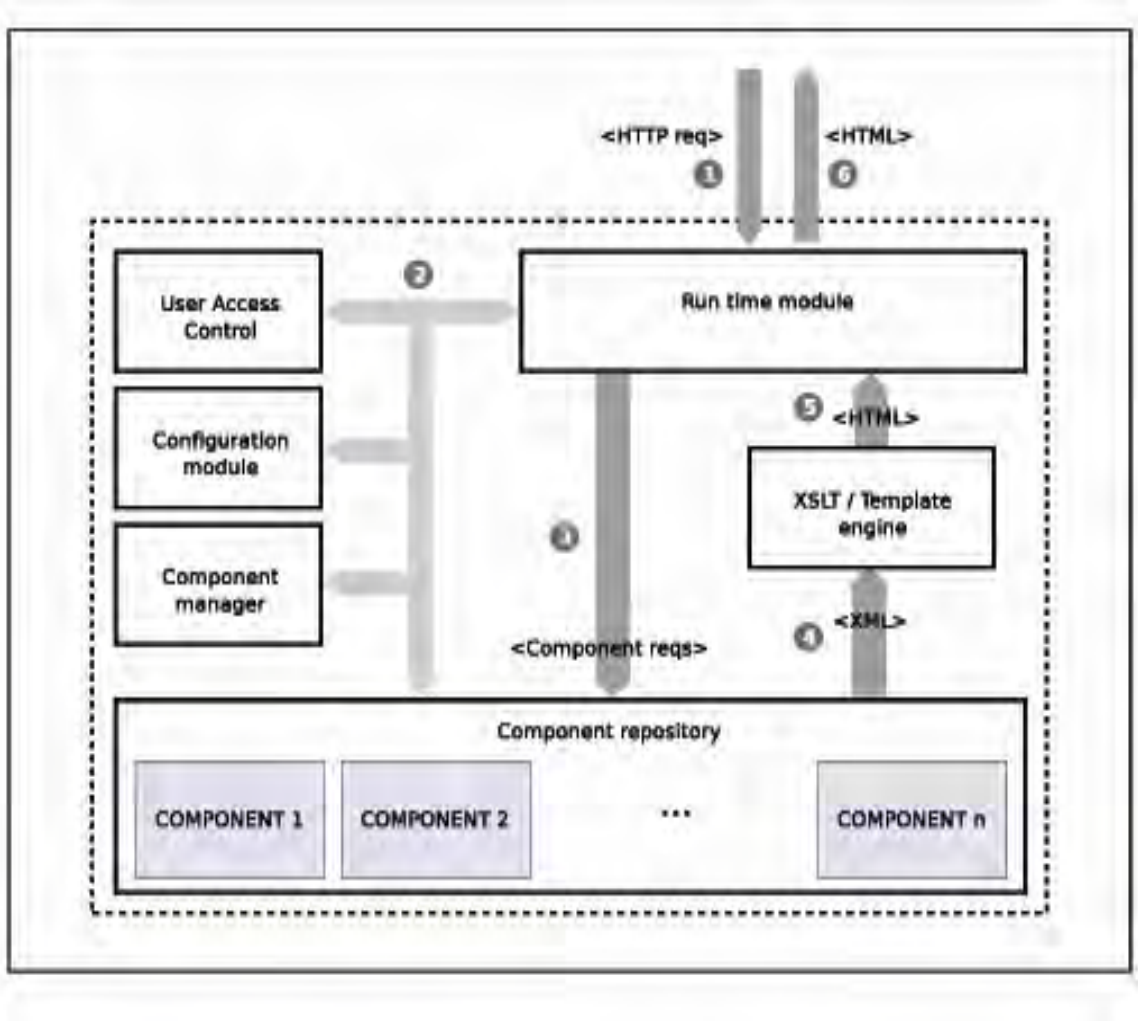

Figure 4.2 : architecture de Koriandol [76]

i. Management/Configuration du module

Le management du module de gestion prévoit un soutien administratif des tâches tel que l'utilisateur, les composants et la compatibilité du produit entre autres. Ce sont des fonctionnalités assez habituelles mais qui requièrent la gestion et la configuration des composants. En fait, chaque composant doit être enregistré et validé avant dĭêtre utilisé, suivant des prescriptions établies proposées par Koriandol comme une spécification d'interface. Le mécanisme de détermination de la variabilité est inclus dans une telle spécification afin d'accorder les possibilités de configuration du système.
Le système aide le concepteur de l'application dans les décisions d'enregistrement et dĭentrée des directives, comme la préparation et la mise en option à utiliser dans une page. Cette opération et d'autres semblables qui ne sont pas décrites en raison de limitations de l'espace peuvent être réalisées au moyen dĭune assistance interactive. Une fois qu'un composant est sélectionné, le système est capable de récupérer les fonctionnalités disponibles à travers la réflexion.

Ce ne sont pas toutes les méthodes qui peuvent être consultées au cours de cette étape mais seulement celles qui ont mis en œuvre une fonction, la plupart des autres sont soit utilisées localement pour le composant ou directement invoquées par le système. Une fois qu'une

méthode est sélectionnée, le système met à la disposition du concepteur de l'application la partie inférieure de la forme obtenue par une autre méthode de la composante et présente tous les paramètres de la conception à offrir afin de déterminer les coordonnées désignant la variante souhaitée.

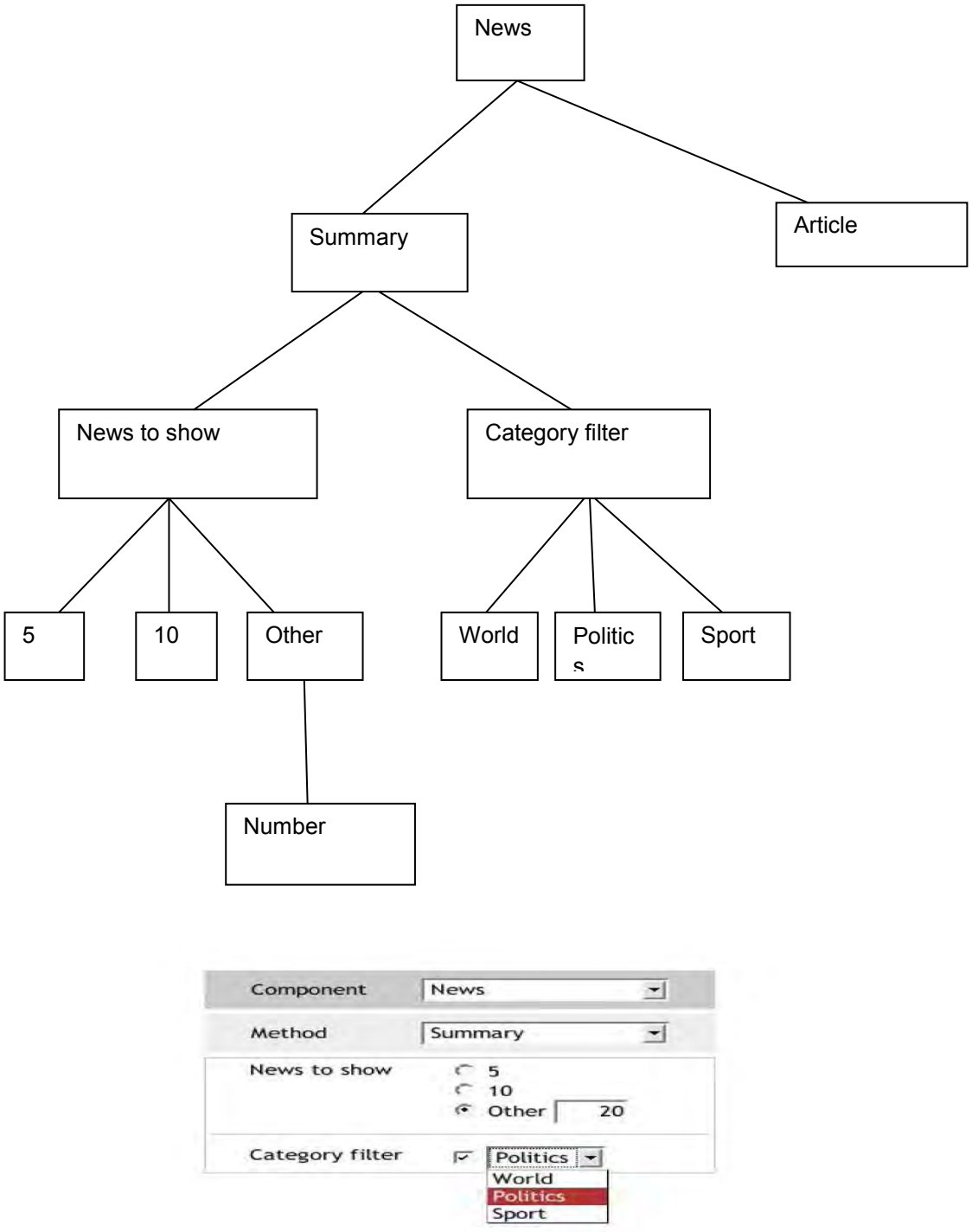

Figure 4.3 : Feature diagram for a news component [76]

ii. Environnement Run-time

La portée de l'environnement d'exécution coordonne dĭune part toutes les activités entre les modules une fois la demande arrivée de l'extérieur et, d'autre part, récupère les fonctionnalités des contenus et services dynamiques qui sont demandées par les pages étant servies. Ces fonctionnalités sont dynamiquement récupérées par l'identification des méthodes de composant et la transmission des informations pertinentes qui sont évaluées selon les directives données par le concepteur lors de la détermination de la variabilité de cette instanciation spécifique.

Présentation du moteur

Chaque composant est capable de produire des contenus qui sont structurés et donnés en XML. Les contenus sont livrés indépendamment de tout aspect de présentation qui porte principalement sur l'aspect d'une application. Le système fournit un moteur de présentation qui permet au concepteur d'associer à chaque fonction et chaque méthode dans un composant une autre feuille de style de transformation XSL ou encore un HTML.

Les feuilles de transformation XSL sont hiérarchiquement disposées selon la relation parentale entre les actifs, par exemple une feuille de style associée à un produit peut être substituée par une autre associée à une baisse des actifs en la hiérarchie, telle que celle associée à un composant ou une fonction. Ainsi, le concepteur de composants peut prendre des décisions importantes indépendamment des contraintes éventuelles qui pourraient être imposées par les aspects de présentation.

Exemple

Les applications Web développées en utilisant le système Koriandol sont composées de services et de contenus transmis dans les pages qui sont arrangées hiérarchiquement. L'outil discuté dans la section précédente fournit les installations pour créer et organiser de manière cohérente les exigences de l'application en cours d'élaboration.

La page d'accueil implique trois composantes dont les fonctions de recherche de produits, une vitrine, et un formulaire d'authentification, respectifs. Chaque fonctionnalité est instanciée avec des paramètres qui sont spécifiques à lĭexploitation intégrée dans la détermination de la variabilité de la réflexion mécanique. Par exemple, le composant fournissant lĭinstallation de la vitrine peut avoir un certain nombre de points de variation, comme les catégories et les fréquences, qui sont utiles pour les produits et pour affiner les stratégies des fournisseurs. Dans ce cas, le système Koriandol fait une introspection du composant et récupère une forme similaire qui est utilisée pour lier une telle variation avec l'une des variantes disponibles [74,76,86].

Illustration à l'aide du cas du site de commerce en ligne Amazon.fr

En prenant le cas dĭun site de commerce en ligne, on peut très bien illustrer cette affirmation. Les sites de vente en ligne doivent être présentés sur des interfaces ou des pages web. Toutes ces pages sont reliées par un lien hypertexte, un peu comme le concept du référencement. Lĭidée, pouvoir voguer dĭune page à une autre sans avoir à ouvrir le volet recherche mais juste en cliquant sur le lien hypertexte dĭun mot, dĭune phrase ou dĭune expression entière.

La page de produits à vendre à ligne doit à tout prix être vendeur et accrocheur. La conception dĭune page passe avant tout par son design. Ce design doit être le plus convaincant possible, sans toutefois tendre vers le ridicule. A cet effet, le designer web use dĭapplications et surtout dĭune stratégie de ligne de produit.

Dans cette optique, il utilise le lien hypertexte comme étant une caractéristique commune renvoyant vers une autre page similaire à la précédente dans sa configuration mais différente du point de vue contenu. Cette seconde page présente alors des variabilités pour être différenciée de la première.

Plusieurs pages constituent un site ou une application web. Ici, lĭapplication web en question est un système dĭnformation web puisque le site informe sur des produits, leur prix et leurs caractéristiques en usant du web et dĭnternet. Lorsque la page web est ouverte à tous, elle nĭest alors protégée par aucun mot de passe et ne nécessite pas de manipulations fastidieuses pour être consultée. Il suffira à son utilisateur de surfer sur Internet, de se référer au lien menant à la page quĭil souhaite accéder et de commencer ses activités.

Cependant les variabilités que nous avons précédemment évoquées stipulent la présence dĭéléments neufs dans la conception des pages, cĭest-à-dire que toutes les pages web sont similaires dans la mesure où elles appartiennent au même site. Leur similitude réside dans leur architecture, par exemple dans la présentation des pages qui est la même quĭil sĭagisse de la page dĭaccueil ou de celle renfermant les articles à vendre, la page de contact, la page de renseignements, la page de services additionnels, etc.

Les variabilités sont alors illustrées par des variantes et des options telles que le volet contenant lĭaccessibilité de lĭutilisateur à une interface personnelle ou à un espace spécialement dédié à ses activités dans le site. Ce volet est souvent aperçu sur la page dĭaccueil du site et requiert la connexion de lĭusager à son compte personnel pour profiter dĭavantages réservés aux membres du site. Il sĭagit ici dĭune option qui ne sera pas présente dans toutes les pages web et qui sera donc vue comme étant une « différence » par rapport au contenu des autres pages.

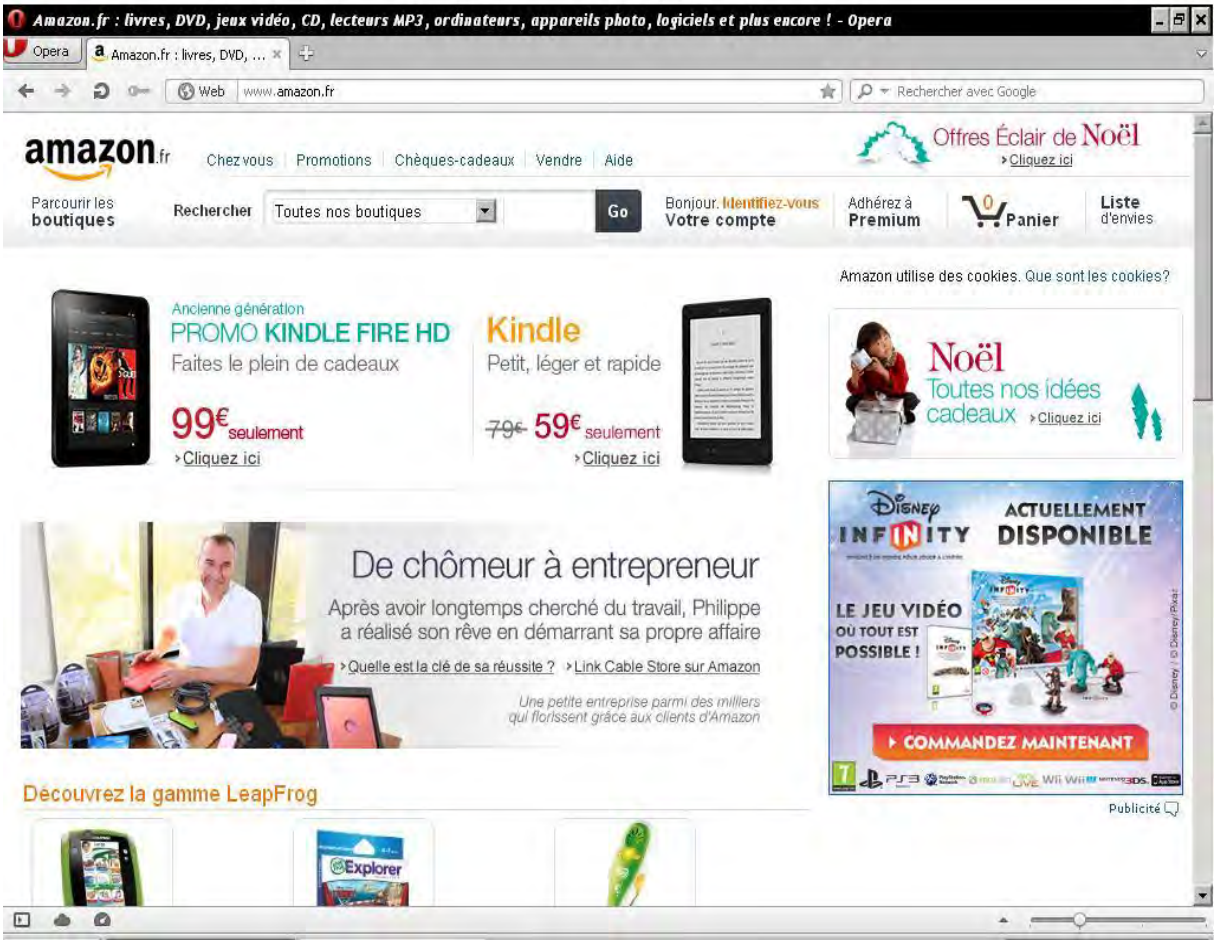

Figure 4.4 : Page d'accueil et de présentation du site amazon.fr

Outre la page dĩaccueil, chaque page constituant le site dĩe-commerce contiendra des différences tout en étant similaires dans leur architecture de base. On peut citer, en exemple, la page de renseignement sur le site ou la société ou le particulier qui propose des ventes en ligne à travers le site. A cet effet, cette page est éditée par une personne travaillant au sein de la société chargée de la promouvoir ou dĩaméliorer son image.

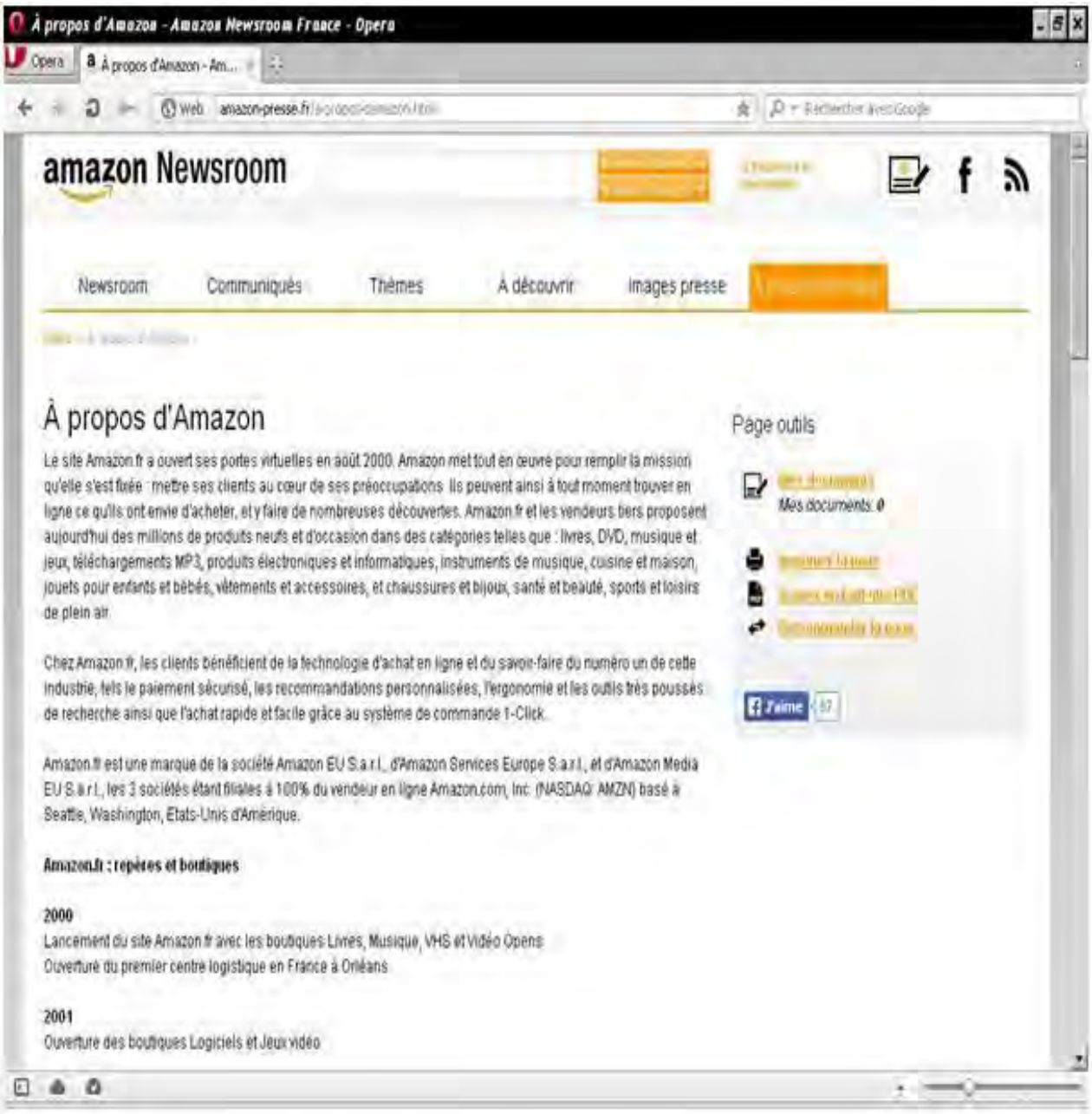

Figure 4.5 : Page de renseignement du site de commerce en ligne amazon.fr

Outre la page de renseignement, on peut aussi trouver une page de contact dont le contenu et lĩagencement architectural diffèrent également des autres pages. Cette dernière contiendra les coordonnées des concepteurs ou des hébergeurs du site, ou de la société quĩil représente.

La page de présentation des produits est tout à fait différente aussi tout en conservant des traits communs avec les autres. On peut par exemple trouver sur cette page les différentes offres disponibles, les promotions, le prix des articles, leur état et des options supplémentaires comme le panier dans lequel lĩusager du site pourra stocker ses articles préférés pour faciliter son choix lors du prochain achat en ligne.

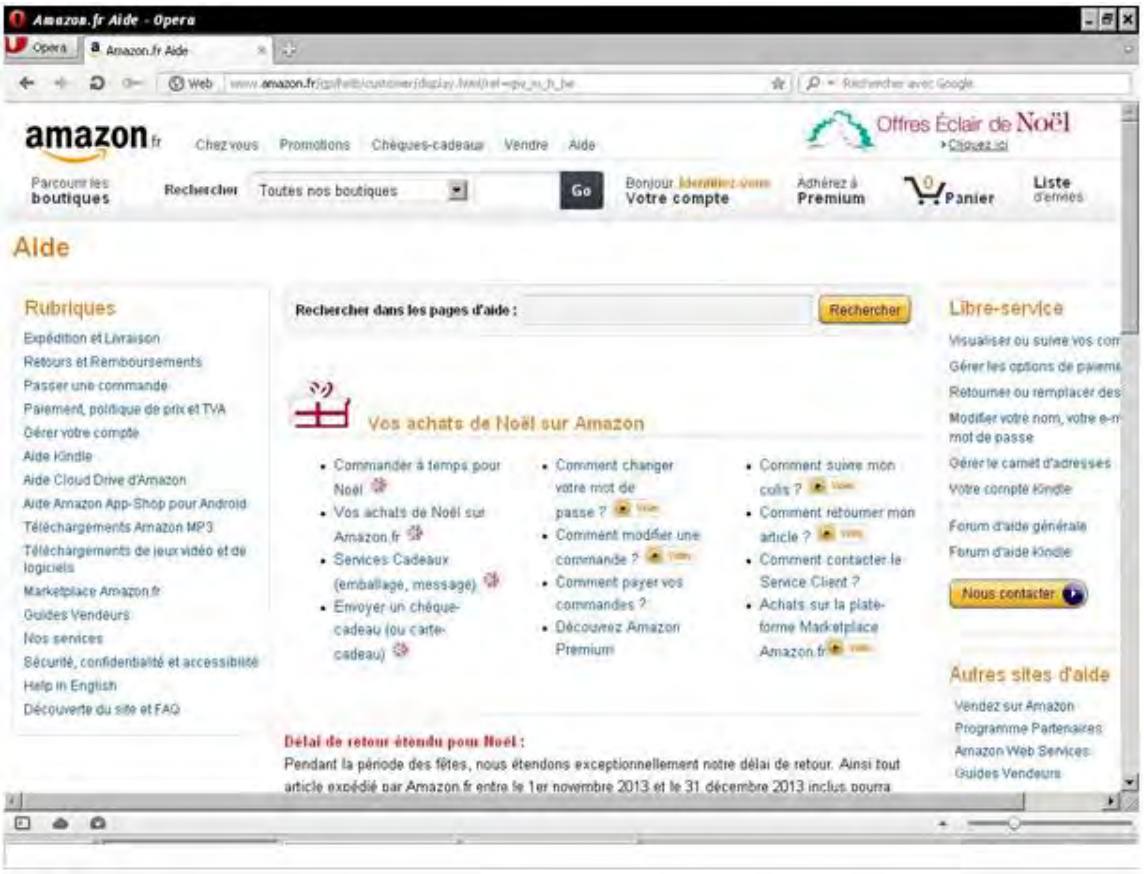

Figure 4.6 : Page d'aide d'amazon.fr

Il va sans dire que chacune de ces pages doit être conçue par des professionnels compétents car leur contenu est unique alors que leur présentation, leur design et leur architecture sont conçues sur une même base et de la même manière. Ces professionnels sont des développeurs dĭapplication web formés et expérimentés conscients des exigences et des besoins des clients et qui orientent la présentation de la page dans ce sens. En même temps, ils doivent aussi prendre en considération les types dĭarticles à vendre et leur usage pour ne pas dévier dans la conception des pages.

Leur but est donc dĭattirer les clients et de leur indiquer rien quŭavec le design de la page web ce quŭils proposent ou produisent. Pour améliorer cette présentation, certains sites nĭhésitent pas à proposer un petit volet « boîte à idées » dans les pages afin de recueillir les avis des clients, de mener une étude approfondie en se basant sur eux et dĭapporter les modifications nécessaires.

Lĭinterface réservée aux concepteurs web et au développeur est différente de celle qui est visible du côté de lĭusager du site, pourtant, il sĭagit bien des mêmes pages. De ce fait, la manipulation par les concepteurs et par les utilisateurs est tout à fait différente. Les concepteurs les manipule de sorte à ce quŭils accèdent aux données centrales et confidentielles du site pour pouvoir insérer les nouvelles offres, les nouveaux prix ou pour modifier le contenu ou le design de la page. Lĭutilisateur entre dans le site et se connecte sur son compte pour effectuer les activités quŭil a lĭhabitude de faire en visitant le site.

Lĭutilisateur nĭa donc pas accès à la manipulation du site et nĭa aucun pouvoir de changement direct sur celui-ci. On peut, par exemple, prendre le cas dĭun site de vente de livres en ligne. Si les utilisateurs aperçoivent les offres, les titres des ouvrages, les prix et tous les détails nécessaires à lĭachat, les concepteurs, eux voient bien au-delà puisquĭils sont chargés dĭimplémenter ces données dans les pages.

Par conséquent, les concepteurs doivent aussi se mettre un peu à la place des visiteurs du site pour détecter leurs préférences ou leurs besoins et orienter les offres et le design des pages en fonction de ces éléments. Les concepteurs ont un rôle clé puisquĭils sont chargés de mettre à jour les informations contenues dans le site, de les supprimer ou dĭen insérer des nouvelles au fil du temps.

Les utilisateurs nĭont pas dĭinfluence directe sur la conception des pages. Cependant, il faut avouer que les concepteurs agencent et arrangent la disposition des composants du site afin de les rendre cohérents mais surtout pour faciliter la recherche des clients. En prenant en compte cette affirmation, on peut partir de lĭexemple où le concepteur trie les différentes offres et les différents produits selon leurs caractéristiques, leur type, leur usage, leur âge et leur utilité en fonction de la saison.

Après avoir effectué ce triage, ils se livrent alors à lĭédification du site en classant les différents produits suivant leurs caractéristiques. En voici un exemple :

Pour que les utilisateurs dĭun site de vente en ligne de livre, le site qui la propose peut classer les livres suivant leur caractéristique et créer des étagères virtuelles. Tous les livres seront alors agencés selon :

- leur titre,
- leur date de publication,
- le thème ou le sujet,
- le public ciblé par la lecture,
- le genre littéraire,
- le prix,
- les notes et commentaires attribués par les lecteurs/les critiques dĭautres auteurs ou de journalistes sur les livres,
- etc.

Lorsque lĭutilisateur visitera le site, il pourra donc trouver dĭemblée ce quĭil cherche grâce à cet agencement ordonné et structuré.

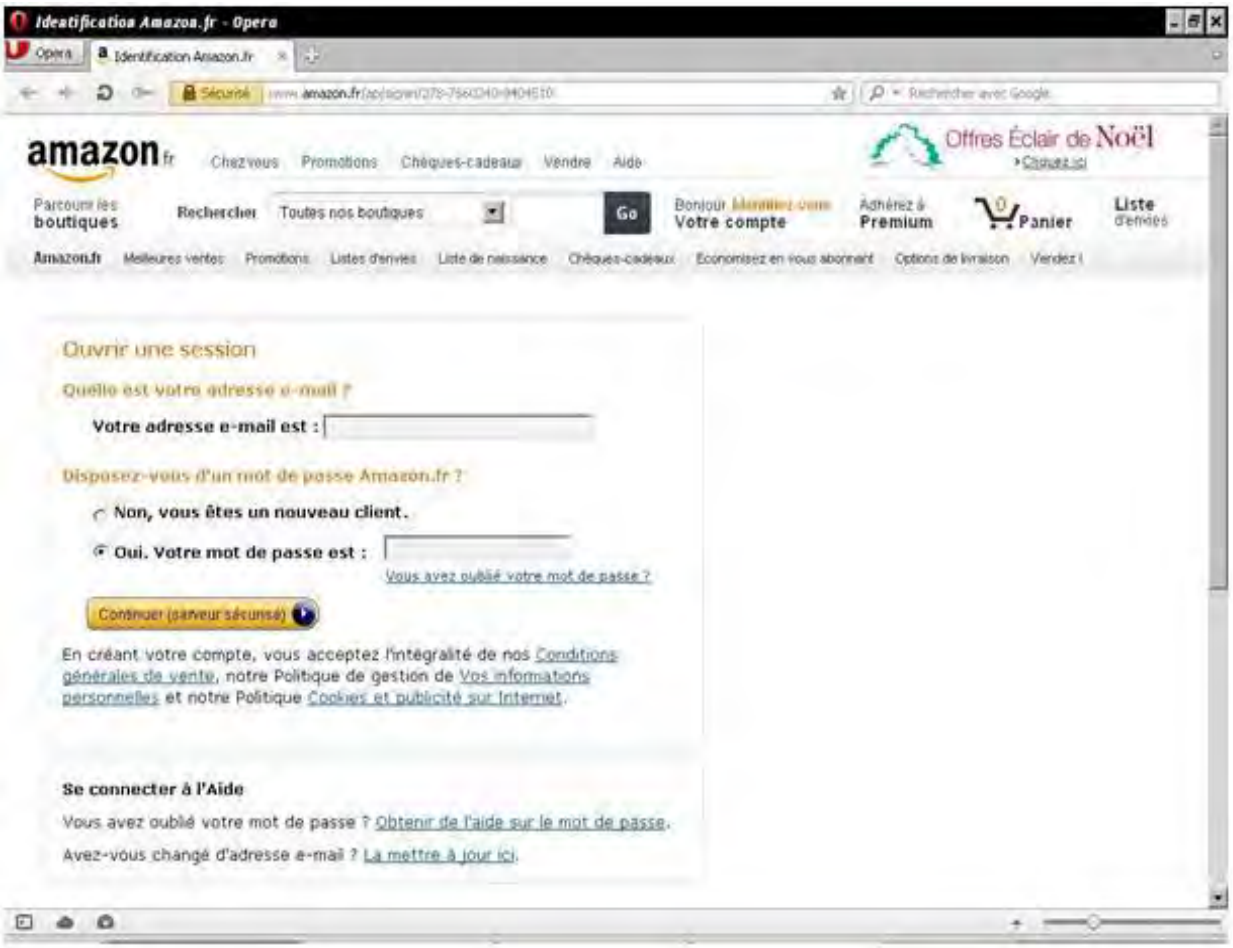

Figure 4.7 : Page de connexion au compte ou d'ouverture de session sur amazon.fr

En prenant lĩexemple du site de vente en ligne Amazon.fr, nous voyons clairement que les quatre pages présentées dans les figures que nous avons choisies sont similaires dans leur architecture. Sur chacune dĩelles, le logo du site apparait, on note aussi lĩusage de la couleur jaune orangée dans toutes les pages, nuancée avec le bleu et le noir. Le fond de toutes les pages a une même couleur : le gris. Il sĩagit ici des similarités, ce sont éléments sont présents sur toutes les pages et permettent de reconnaitre dĩun trait les pages du site.

Trois des pages que nous illustrons ici contiennent les options supplémentaires :

- accessibilité au compte sur amazon.fr
- adhésion à premium,
- panier,
- liste dĩenvies.

A priori, la plupart des pages du site contiennent ces options affichées en haut à droite de chaque page. Seulement, il sĩagit ici dĩune variabilité statique puisquĩelle nĩapparait pas sur toutes les pages, comme cĩest le cas pour la page de renseignement du site amazon.fr. Outre ces variabilités, on note aussi des variabilités représentées par le contenu de chaque page. En effet, chaque page a son propre contenu, ce qui peut constituer une variabilité.

Cette variabilité est alors illustrée par un titre différent pour chaque contenu :

- pour la page de renseignement, le titre est : « A propos dĭAmazon
- pour la page dĭouverture de compte ou de session dans le site, le titre est : « Ouverture de session. »
- pour la page dĭaide, le titre est : « Aide »

Le fait de donner un titre différent pour chaque page permet de se référencer à son contenu et de se rendre compte à première vue que toutes les pages ne sont pas unanimement identiques. Le titre choisi doit refléter le contenu et doit être disposé de la même manière dans toutes les pages pour informer le client sur ce quĭil trouvera dans la page. Ces titres peuvent donc être pris comme des variabilités.

Les lignes de produits au service du SIW sont donc traduites par la conception de plusieurs pages web pour un site web. Les pages web sont architecturées de la même manière avec un logiciel spécifique et présente des points communs comme le logo ou la couleur dans les exemples que nous avons présentés ci-dessus. Elles possèdent aussi un contenu et un titre diversifiés en fonction de ce que la page doit présenter. Il sĭagit ici des variabilités.

Le système des lignes de produits basé sur la réutilisation est très visible dans ce SIW. En effet, on voit très bien que les pages web sont les produits ou les modèles. Il sĭagit ici dĭapplications web. Les concepteurs ou développeurs web conçoivent alors ces pages autour dĭun seul principe : la réutilisation, cĭest-à-dire la possibilité dĭuser dĭun ou de plusieurs éléments appelés assets ou composants réutilisables.

Ces assets, définis en ingénierie de domaine, sont des éléments pouvant être :

- des logiciels,
- une interface,
- un logo, une couleur de page,
- une dimension,
- une structure,
- un design,
- une forme,
- etc.

Grâce à ces assets, on peut voir que toutes les pages appartiennent à la même famille, dans le cas de lĭexemple que nous avons choisi dans ce chapitre, il sĭagit de la couleur de la page et du logo du site qui est Amazon.fr. Toutes les pages détenant ces mêmes caractéristiques sont donc issues de la même famille de produits. Le but de ces similarités est de permettre aux utilisateurs du site de reconnaitre immédiatement les pages qui le composent.

Outre ces similarités, les pages disposent également de variabilités représentées par la différence dans les titres ou dans les contenus des pages web. Ces variabilités décrivent lĭunicité des pages, cĭest-à-dire quĭelles permettent de voir dĭun trait que toutes les pages ne sont pas intégralement identiques bien quĭelles soient similaires.

Grâce à ces variabilités et à ces similarités, on peut aisément comprendre la conception des pages qui repose essentiellement sur lĭapproche des lignes de produits. Afin de détecter, on use généralement des features ou caractéristiques telles que FODA.
Si nous concevons un diagramme de FODA pour le site Amazon.fr, on peut lĭélaborer comme suit (figure 4.8): Les différentes pages sont ici les produits ou modèles.

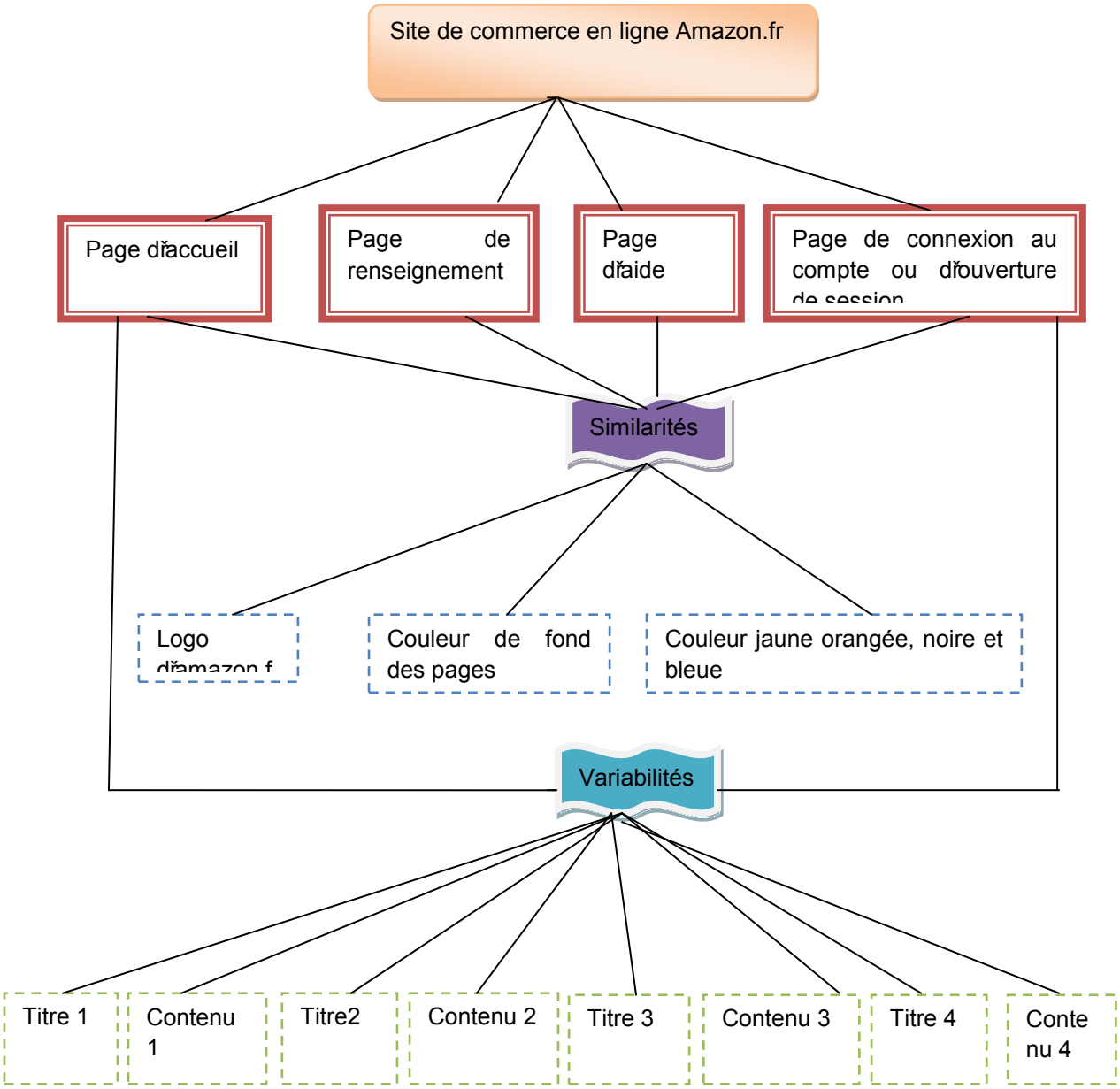

Figure 4.8 : Diagramme de FODA pour le site Amazon.fr

Les similarités ou caractéristiques obligatoires sont le logo du site, la couleur de fond des pages et la couleur jaune orangée, la couleur noire et la couleur bleue utilisées dans le site.
Les variabilités sont le titre et le contenu.

4.3. Les avantages des LPL dans la conception des SIW

Comme nous lĭavons déjà vu auparavant, les lignes de produits prodiguent des avantages variés, surtout dans le domaine des SIW en effet, user des lignes de produits pour concevoir un

système dĭnformation web est une pratique de plus en plus répandue bien quĭelle ne fasse lĭobjet que de peu de publications ou dĭétudes sérieuses.

Pour rappel, les LPL, Lignes de produits Logiciels, permettent une composition de logiciels ou de produits appelés modèles qui sont similaires sur certains points et opposés sur dĭautres le principe de le LPL est la conception massive dĭun même produit présentant des traits communs et des traits distinctifs durant une durée déterminée, suivant une qualité optimisée et un coût de production inférieur.

Dans les SIW, on peut dire que les LPL pourvoient donc au développement dĭoutils de diffusion dĭnformation qui peuvent être des sites internet ou des applications web. Ces SIW sont conçus en se basant sur les LPL, cĭest-à-dire quĭils contiendront plusieurs éléments de composition et de modélisation obtenus à partir de la réutilisation. Dans cette optique, chaque SIW membre dĭune gamme de produits contiendra des éléments de ressemblance et des éléments de différenciation.
Le premier avantage que nous constatons est celui de la vitesse de production qui augmente en même temps que le nombre des produits. La LPL pourvoit alors à la conception dĭune famille ou dĭun ensemble de logiciels, de produits, dĭapplications web ou de systèmes dĭnformation web. Cette multitude de produits créés permet une diffusion plus fluide et plus rapide des informations et donc une grande efficacité des SIW.

Cette efficacité repose cependant sur la mise en application et le suivi régulier et pointilleux des démarches et processus de conception des LPL. A cet effet, passer par lĭingénierie de domaine et par lĭingénierie dĭapplication est non seulement évident mais aussi obligatoire. Plus les exigences et contraintes rencontrées lors de ces ingénieries sont maitrisées et dépassées, plus la qualité des LPL est assurée, plus lĭefficacité des SIW est garantie.

Outre la production, on constate aussi des avantages du point de vue du temps. Ces bienfaits se traduisent par une minimalisation du délai de production et par le respect de ce dernier. De ce fait, les lignes de produits sont utiles pour les SIW tels que les sites internet. En effet, elles favorisent la conception de plusieurs pages à la fois, ce qui permet de suivre la tendance en matière de chiffres et dĭobjectifs à atteindre. Cela permet aussi de tenir face à la concurrence et dĭactualiser en permanence le contenu des sites web.

A part la réduction du temps de production, on peut également assister à une réduction du coût de production. Une telle baisse du coût de production se traduit par la réutilisation qui empêche la conception à zéro des pages web. En dĭautres termes, on peut créer plusieurs pages à partir dĭun seul modèle qui détient lĭarchitecture de base de tous les produits. Cette stratégie permet de ne plus se focaliser sur une seule page à la fois mais plutôt de créer plusieurs modèles en se basant sur les caractéristiques obligatoires quĭelles doivent contenir.

Ces pages contiendront les éléments similaires évoqués ci-dessus et présenteront un contenu différent. Le coût de la production se voit diminué grâce à la possibilité de créer plusieurs pages en un trait sans avoir à recommencer le processus à zéro.

Nous avons vu dans ce chapitre que les lignes de produits peuvent contribuer à la conception et au développement des applications web ou des SIW. Un système dĭnformation web est une application basée sur le web dont les vocations sont orientées vers la diffusion, le stockage, la sauvegarde, lĭéchange et la manipulation dĭnformations via le web.

Le développement en force du web actuellement a permis le développement des SIW, si bien quĭils peuvent à présent cerner plusieurs domaines tels que la santé, les études, les recherches scientifiques, etc. Ainsi, la communication, les échanges dĭnformation, les demandes de renseignements et le contrôle des flux dĭnformation sont facilités par les SIW qui nĭont pas de frontières que ce soit en termes dĭespace ni de temps.

La conception des SIW se fait donc en suivant les règles de conception des LPL. Cela signifie quĭen prenant lĭexemple dĭun site web, on peut alors concevoir plusieurs pages web en même temps.

Cette technique pourvoit à la production massive des pages web membres dĭune seule famille ou dĭun seul site. Ces pages web possèdent des spécificités telles que les similarités qui sont des caractéristiques communes à chaque page comme la dimension ou la couleur de fond (cf. lĭexemple sur le site de commerce en ligne Amazon.fr). Elles détiennent aussi des éléments de variabilité tels que le contenu, les titres, etc.

Le but des similarités est de permettre aux utilisateurs de reconnaitre une page web et le site qui la promeut rien quĭen la regardant. Les variabilités permettent de distinguer toutes les pages dĭune même famille dĭapplications web et de prouver quĭil sĭagit bien de pages différentes. Dans ce contexte, lĭélaboration dĭune interface utilisateur et dĭune interface pour le développeur est requise.

Ce chapitre dédié à lĭétat de lĭart des LdP dans les SIW a permis de constater que les SIW peuvent être grandement avantagés par lĭadoption de la technique des LdP dans leur conception. Pour illustrer cette affirmation, lĭexemple du Koriandol, un outil de développement, de déploiement et de maintenance des familles dĭapplications web.
Ce genre dĭoutil permet de créer plusieurs applications web en même temps avec un coût de production minimal et une qualité maximale. La production massive et le coût de production qui diminue sĭexpliquent par la réutilisation, un processus permettant de créer plusieurs applications en usant de composants réutilisables. Cela pourvoit à ne plus créer une application à partir de zéro, ce qui permet de concevoir rapidement plusieurs SIW à moindre coût du fait que les matières et ressources requises sont aussi diminuées.

La qualité des SIW conçus à partir des LPL se traduit par le souci de trouver des variabilités afin de distinguer toutes les applications membres dĭune seule ligne, gamme ou famille. Cela se fait en effectuant plusieurs études et en optimisant le développement de chaque application.

Une fois que nous avons vu cet état de lĭart des LPL dans les SIW, nous allons maintenant illustrer notre approche dans les chapitres suivants.

5. UNE APPROCHE LPL POUR LE DEVELOPPEMENT D'APPLICATIONS WEB

5.1. Introduction

Les exigences des applications web impliquent une grande diversité de différents services et fonctionnalités, mais aussi une grande complexité dans leur définition. L'ingénierie d'applications Web implique l'utilisation de nombreuses technologies différentes afin de satisfaire tous les besoins des utilisateurs [80, 90, 91, 92].

D'autre part, la pertinence économique et la complexité croissante de ces applications nécessitent lŕemploi de méthodes, techniques et modèles qui offrent de meilleurs facteurs en matière de temps de développement et de coût [80]. Par conséquent, le besoin dŕutiliser des méthodes adéquates et efficaces d'ingénierie dŕapplications web se fait de plus en plus sentir afin dŕaboutir à des applications robustes.

Les Lignes de Produits Logiciels (LPL) sont des familles de logiciels qui partagent des fonctionnalités communes, où chaque membre se distingue à travers des variantes sur certaines fonctionnalités [25]. Ces dernières années, lŕingénierie LPL est l'une des approches les plus prometteuses visant à réduire les coûts de développement, ainsi quŕà améliorer la qualité des familles de produits logiciels [25,84].
En tant que cadre de développement établi, bénéficiant de support méthodologique considérable, l'objectif principal de l'ingénierie LPL est le développement agile et rapide des applications logiciel, profitant des artefacts réutilisables, déterminés tout au long du cycle de vie de développement du modèle du domaine de ces applications [84].

Une ingénierie LPL nécessite de décrire les produits en utilisant des modèles de variabilité, sous forme de modèles de features qui contiennent seulement les particularités et les relations entre eux. Un feature est considérée comme une caractéristique du système qui est observable par l'utilisateur final [84]. Un modèle de feature décrit les caractéristiques qui sont présents dans tous les produits, appelées features de base, et celles qui sont spécifiques à certains produits du domaine, appelées caractéristiques variables.

D'autre part, dans les applications Web, nous trouvons souvent des comportements de plus en plus similaires, partagées par les différents produits, utilisés dans des environnements similaires, accomplissant des tâches similaires [25,64,85,86].

Le partage d'une infrastructure commune qui constitue le cœur de chaque membre (produit) du domaine, et la réutilisation des artefacts qui peuvent être fournis de façon à déployer les services récurrents, sont toujours profitables à la réutilisation du logiciel et son développement planifié, menant vers une diminution considérable de l'effort nécessaire au développement d'une application Web. C'est pourquoi en mettant l'accent sur la conception non pas dŕun produit unique, mais des familles de produits conduit à une forte réutilisation du logiciel [92].

Dans ce chapitre, nous présentons une méthode de développement en lignes de produit logiciel web, visant à améliorer la productivité, le temps de mise sur le marché; lŕeffort de maintenance, ainsi que la qualité du logiciel. Nous montrons, en particulier, comment modéliser la variabilité dans les lignes de produits logiciels dans le cadre du domaine des applications web. A cet effet, nous définissons un modèle UML multi-vues de représentation de la variabilité. Tout au long des trois phases de développement, à savoir, le modèle des besoins, le modèle dŕanalyse, et le modèle de conception, nous présentons les mécanismes de variabilité, particulièrement au niveau du modèle structurel (données) et du modèle de navigation.

En génie logiciel, le concept de lignes de produits logiciels (LPL) est un nouveau paradigme, où les produits logiciel partagent des caractéristiques communes. Il s'agit d'un ensemble de produits qui partagent un ensemble commun d'exigences, mais chaque produit possède son propre ensemble de caractéristiques spécifiques. L'ingénierie en lignes produits logiciels simplifie essentiellement la conception et la maintenance des familles de programmes et répond aux besoins des applications hautement personnalisables, d'une manière très rentable [4]. La finalité est d'obtenir les avantages désirés en termes de productivité, coût; effort, et délais de commercialisation [25,74,87].

5.2. Lignes de produits logiciels

Le concept de LPL est un paradigme de génie logiciel pour le développement du logiciel. Ce concept est important dans le sens de la promotion de la réutilisation de logiciels, conduisant à une plus grande productivité et un degré élevé de qualité [25,88].

En profitant dŕune réutilisation de logiciels à grande échelle, l'approche LPL pour le développement de logiciel peut générer des améliorations quantitatives et qualitatives importantes de la productivité, les délais de commercialisation et la satisfaction des clients [81]. Son succès grandissant provient de sa capacité à offrir aux entreprises les moyens d'exploiter leurs produits logiciels communs pour réaliser des économies de production [16].

Un logiciel au sein d'une ligne de produits logiciels a souvent des fonctionnalités spécifiques qui ne sont pas communs à tous les autres membres au sein de la gamme de produits. Ces fonctionnalités spécifiques sont appelés «features-variantes " dans une ligne de produits logiciels. Lŕemploi du paradigme LPL implique la modélisation des features-variantes [88].

Un feature est une propriété du système qui est pertinente pour certaines parties prenantes. Il est utilisé pour capturer des points communs ou de discrimination entre les produits d'une LPL. Les fonctionnalités sont présentées dans une représentation arborescente, appelée diagramme de features, avec une notation spécifique pour chaque catégorie de variabilité (obligatoire, alternative et optionnelle). Un modèle de feature fait référence à un diagramme de features annoté des informations supplémentaires telles que les dépendances entre les features. Il représente la variabilité au sein d'une famille de système d'une manière abstraite et explicite.

Une ingénierie LPL englobe la création et la gestion des familles de produits pour un domaine particulier, où chaque produit de la famille est dérivé d'un ensemble commun des assets de base, par application d'un ensemble de règles prescrites [180]. En utilisant une ligne de produits logiciels, les développeurs sont en mesure de se concentrer sur des questions spécifiques sur les produits plutôt que sur les questions qui sont communes à tous les produits [16].

Le concept clé pour le développement de la famille de produits logiciels est la variabilité du système, considéré comme étant la capacité à dériver divers produits logiciels à partir de la famille de produits [88]. La variabilité est la capacité à modifier ou adapter les produits logiciels [16]. Elle fait référence feature-variante (option, points de variation et variantes) d'une LPL [74]).

La variabilité est non seulement utilisée pour définir une LPL et la dérivation de produits, mais aussi pour gérer l'évolution des logiciels en anticipant certains types de variabilité, de construire un système de telle manière qu'il soit préparé à introduire des changements prédéterminés [89]. Le fait de s'appuyer sur les points communs entre les membres d'une famille de produits logiciels ainsi que sur les familles de produits similaires, constitue un moyen efficace de la réutilisation du logiciel [25].

En général, les features sont représentés schématiquement sous la forme dŕune arborescence (diagramme de feature). Les nœuds du diagramme montrent comment certaines fonctionnalités sont composées de fonctions de niveau inférieur, qui peuvent être obligatoires (toujours présentes), en option, choisies au moment de la construction, ou bien chargées au moment de l'exécution, à des points de variation déterminés.

Diverses relations existent entre ces fonctionnalités, telles que la généralisation, l'agrégation, l'utilisation et la dépendance mutuelle. Une ingénierie LPL est divisée en deux parties: Ingénierie de domaine et ingénierie d'application. Elles constituent deux cycles de développement relativement indépendantes, à savoir le développement pour la réutilisation, signifiait que le développement de la ligne de produit lui-même, et le développement par la réutilisation, aussi appelé instanciation de produit.

L'ingénierie de domaine implique, entre autres, d'identifier les points communs et les différences entre les entités du domaine de produits. Ce sont des artefacts produits du processus d'ingénierie du domaine, exprimant les exigences réutilisables et configurables, des modèles dŕanalyses et des modèles de conception, mais aussi des modèles dŕimplémentation. En général, n'importe quel artefact produit du processus est réutilisable. Il est considéré comme étant une ressource réutilisable [16].

Contrairement à lŕingénierie du domaine qui couvre une gamme de produits inter-reliés, l'ingénierie d'application porte sur la dérivation d'un certain produit, en utilisant un sous-ensemble des objets logiciels partagés. Ce résultat est obtenu grâce à la réutilisation des objets de domaine, en exploitant la variabilité de la ligne de produit [25].

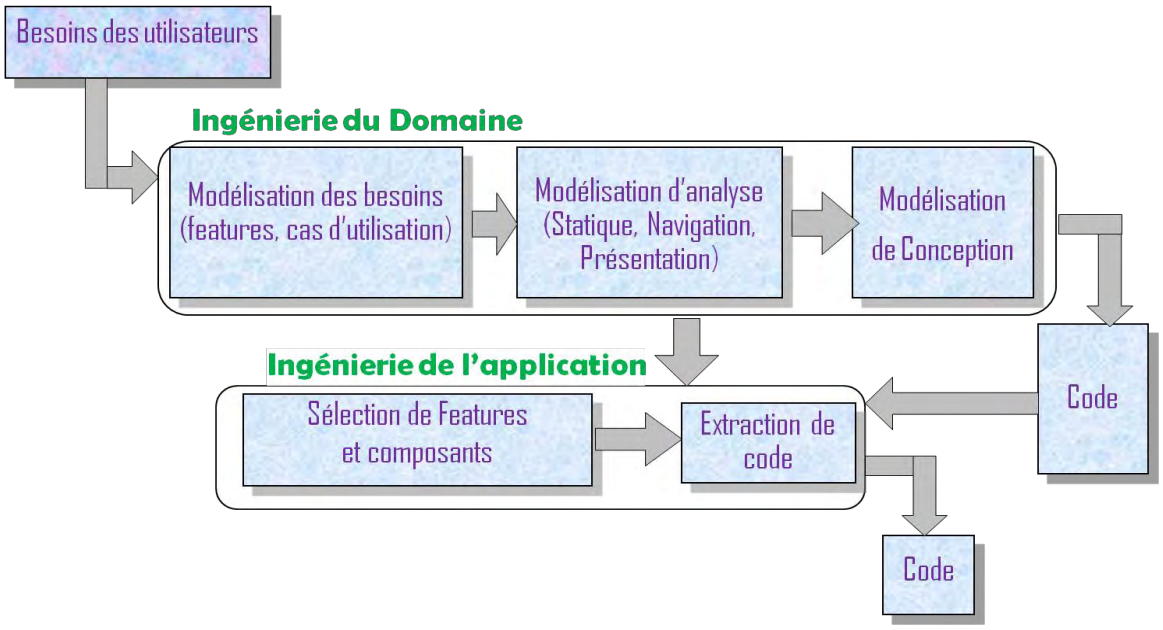

Figure 5.1 : Processus d'ingénierie LPL du Web.

Dans cette étape, chaque point de variation, tel que défini à l'étape précédente, est lié à une variante spécifique, sélectionnée à partir de l'ensemble des variantes qui lui sont associées. Une configuration de produit donne une liste des fonctions sélectionnées et non sélectionnées selon le modèle de la variabilité de la famille de produits. Cette configuration de produit avec la fonction de mapping des features avec les modèles est utilisée pour calculer la mise en œuvre du produit en réutilisant des objets de domaine existants [89].

5.3. Ingénierie LPL du Web

Les applications Web peuvent être considérées comme étant des produits logiciels issus d'une infrastructure commune et des artefacts qui capture les spécificités de produits logiciels, extraites du modèle du domaine, par exemple, enregistrement de l'utilisateur, panier virtuel, et paiement dans un système de vente en ligne [52,82,86].

Naturellement, notre processus d'ingénierie LPL d'applications Web (Figure 5.1) est réalisé en deux grandes phases: ingénierie de domaine et ingénierie d'application. Dans la première phase, l'analyse des points communs et les différences entre produits fait apparaitre le modèle des features, le modèle dŕanalyse, le modèle de conception, ainsi que le modèle dŕimplémentation.

Le but de l'ingénierie de domaine est de développer des modèles qui peuvent être utilisés dans le développement de produits pour un domaine donné. Comme le montre la Figure 5.1, on définit trois étapes dans l'ingénierie de domaine: modélisation des exigences, modélisation de l'analyse et modélisation de conception. Le modèle des features est le produit dŕune activité qui commence à lŕétape de modélisation des exigences et peut être affiné lors des étapes suivantes.

5.3.1. Modèle des besoins

Dans notre méthode d'ingénierie LPL du Web, les besoins à collecter ne sont pas lŕapanage dŕune application, mais ils concernent toute une famille d'applications (ingénierie de domaine). Nous commençons par la modélisation des exigences où les modèles de features et cas d'utilisation sont construits (du point de vue de l'utilisateur final).

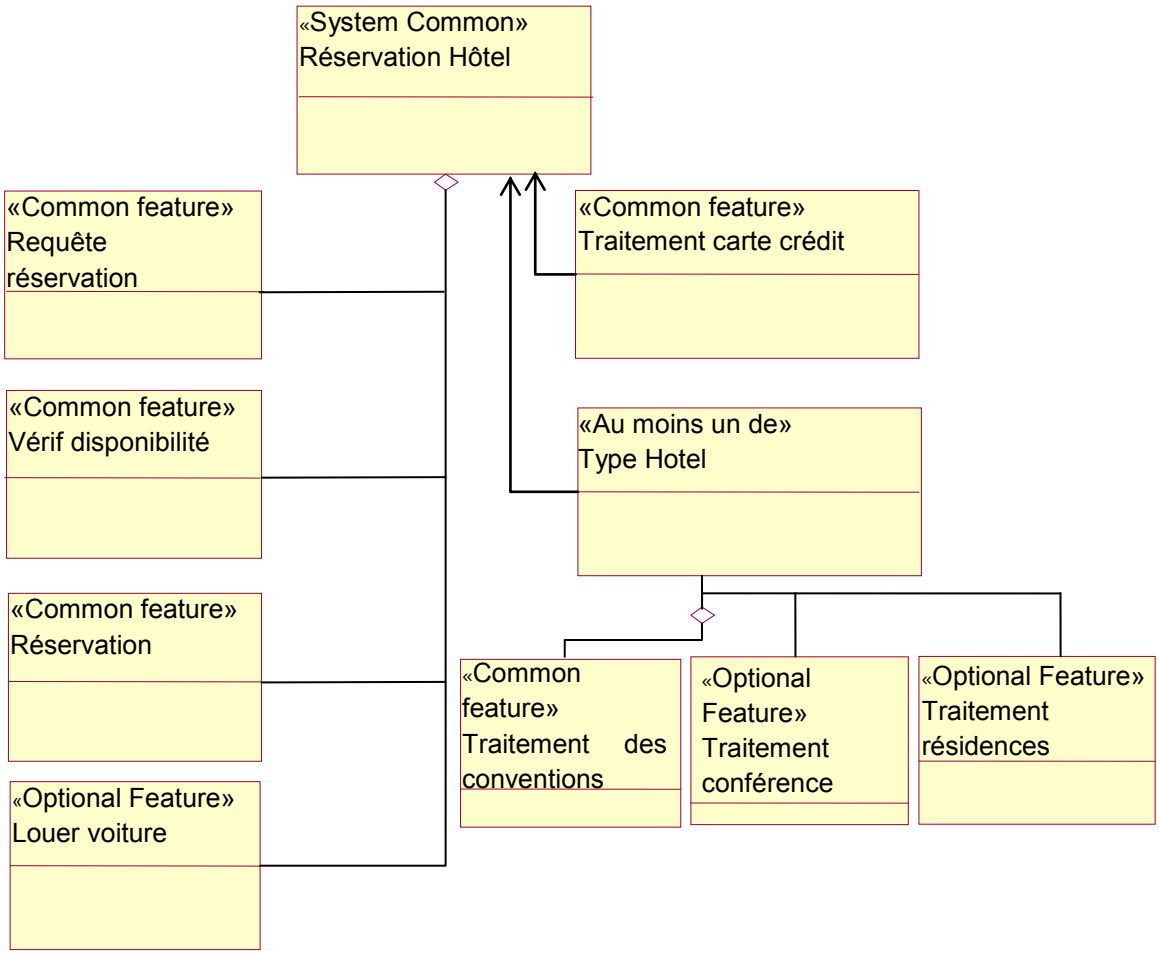

Figure 5.2 : Un exemple de modèle de features d'un système de réservation d'hôtel.

Le modèle structurel (modèle des entités) exprime un point de vue unificateur pour la modélisation de la variabilité dans les lignes de produits logiciels [89]. Les features peuvent être incorporés dans UML en utilisant le concept de méta-modélisation [16,89], dans lequel les features sont modélisés comme des méta-classes et les stéréotypes sont donnés pour différencier les caractéristiques <<commun>>, <<en option>>, <<par défaut>> et <<fonction>> «alternative».

Un exemple d'un modèle de features est donné pour un système de réservation d'hôtel à la figure 5.2, qui représente le système commun, des features communs, features optionnels, ainsi que des features par défaut. Les features communs sont les suivants: «Demande de

réservation», «Vérifier la disponibilité», «faire une réservation» et «Confirmer la réservation». Le feature "au moins une" est le "type dŕhôtel". La fonction par défaut est le «traitement conventionnel». Les features optionnels sont «traitement de la conférence" et "traitement résidentiel".

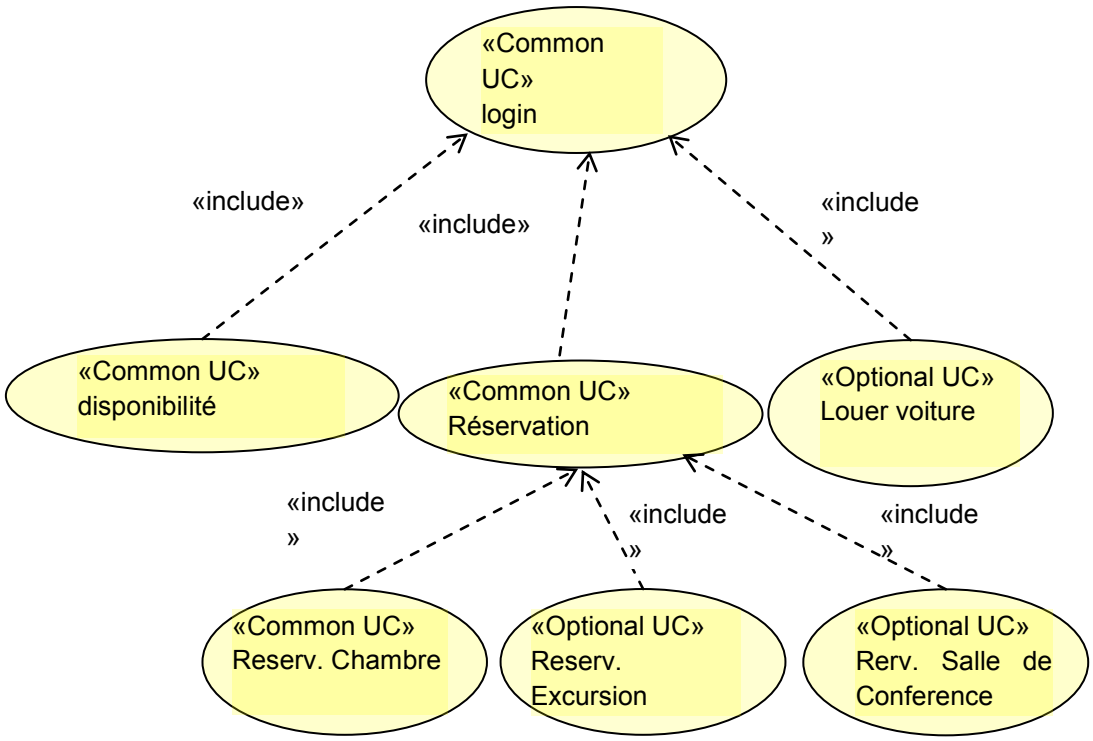

Figure 5.3 : Cas d'utilisation avec variabilité.

Les exigences fonctionnelles d'un système sont définies en termes de cas dŕutilisation et acteurs [5]. Pour un seul produit, tous les cas d'utilisation sont nécessaires afin de faire apparaitre de façon plus complète les besoins des utilisateurs. Dans une ligne de produit logiciel, une partie seulement des cas d'utilisation, qui sont appelés cas d'utilisation communs, sont tenus par tous les membres de la famille [16] [89]. Dŕautres cas d'utilisation peuvent être facultatifs, en ce sens qu'ils sont exigés par certains membres mais ils peuvent ne pas être exigés par dŕautres membres de la famille.

Nous utilisons le modèle de cas d'utilisation UML, les stéréotypes <<UC commune>> pour cas d'utilisation commune, << UC option >> cas d'utilisation en option, ou << UC autre >> pour un usage autre cas. Dans le modèle de cas d'utilisation pour le LPL de la réservation dŕhôtel (Figure 5.3), le cas dŕutilisation "louer une voiture", ainsi que le cas dŕutilisation «réserver une salle de conférence" sont des cas d'utilisation facultatifs. Il est à noter que la variabilité peut également être insérée dans un cas d'utilisation par des points de variation, qui spécifient des emplacements dans le cas d'utilisation où la variabilité peut être introduite [16, 26,89].

Le but de l'analyse par cas d'utilisation est d'obtenir une bonne compréhension des exigences fonctionnelles [26], alors que le but de l'analyse des features est de différencier les points communs et la variabilité. Les cas d'utilisation et les features se complètent mutuellement. Les cas d'utilisation sont mappés à des features facultatifs et/ou alternatifs, respectivement, tandis que les points de variation dans les cas d'utilisation sont aussi mis en correspondance avec les features [16].

Cependant, avec la modélisation des cas dŕutilisation, il est plus difficile d'obtenir des dépendances entre cas d'utilisation que dans le cas dŕun modèle de features où la dépendance peut être capturée de façon explicite [89]. La même variabilité peut être répétée dans plusieurs cas d'utilisation, mais elle est capturée de manière plus efficace par les features.

5.3.2. Modèle d'analyse

La modélisation d'analyse a pour but de représenter la fonctionnalité en utilisant une représentation orientée objet. Dans cette phase, le concepteur représente les principales structures du système et la façon dont ils collaborent afin de réaliser la fonctionnalité du système. Dans UWE [64,73], la fonctionnalité de l'application Web est représentée par trois modèles: les modèles de contenu, de navigation et de présentation. UWE est une méthode de développement et modélisation des applications web utilisant UML comme notation, étendu par des stéréotypes afin de représenter quelques spécificités du web. Elle fournit au développeur des lignes directrices pour dériver le modèle de navigation et le modèle de présentation à partir de modèle de contenu. UWE est une approche orientée objet basée sur le standard UML. UWE spécifie un cycle de vie complet de conception pour les applications web. Il prévoit un certain nombre de modèles et de transformations de modèles dans le cadre du MDA de l'OMG [64]. Dans notre approche, nous avons suivi l'UWE dans le modèle d'analyse en définissant le modèle de contenu et le modèle de navigation tout en ajoutant les variabilités dans le cadre dŕune approche LPL.

i. Modèle structurel avec variabilités

Le modèle structurel (ou modèle de l'entité) vise à construire un modèle de domaine en essayant de prendre en compte aussi peu que possible les liens de navigation et les aspects de présentation. Pour une application Web les cas d'utilisation commun et le modèle de features pourraient être la base de l'identification des entités à la base du modèle structurel.

Nous pourrions utiliser une approche progressive pour identifier les classes. Tout d'abord, nous identifions les entités «actives» dans le système. Les acteurs identifiés dans le cas d'utilisation seront considérés comme des candidats de choix pour les considérer en tant que classes potentielles. Ensuite, nous identifions les entités du domaine de système ("passives"). Il est à noter que dans les cas orientés données, les entités du domaine correspondent aux objets gérés, et non pas aux activités des utilisateurs [98]. Le modèle structurel utilise le diagramme de classes UML qui va déterminer la structure du système, les classes et leurs relations (comme association, agrégation, ...).

Lors de la modélisation des lignes de produits logiciels, chaque classe peut être classée en fonction de sa caractéristique de réutilisation, en utilisant les stéréotypes «entité commune» pour une entité commune, «entité option» pour l'entité en option, dans le diagramme de classes représentant le modèle de contenu ou le modèle de fonctionnalité.

ii. Modèle de navigation avec variabilité

L'étape suivante dans le processus de développement est la conception du modèle de navigation. Le modèle de navigation définit les objets qui peuvent être visités dans une application Web [52,60]. En outre, il définit la disponibilité des objets.

Dans le processus de construction du modèle de navigation, le développeur prend des décisions cruciales de conception, en déterminant les chemins de navigation qui sont nécessaires pour assurer la fonctionnalité de l'application [60]. Les décisions sont fondées sur les exigences, le modèle conceptuel, le modèle de cas dŕutilisation et le modèle de navigation que l'application doit satisfaire.

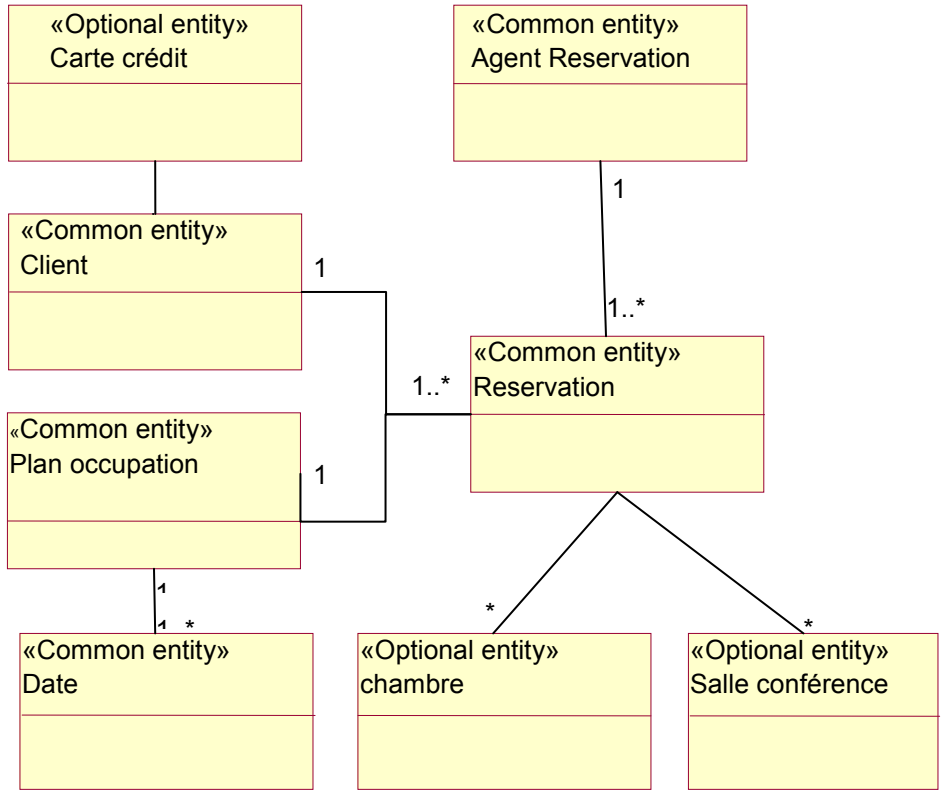

Figure 5.4 : Modèle structurel (des entités) avec variabilité.

Le modèle de navigation se base sur le modèle structurel comme point de départ, ce qui pourrait être étendu à d'autres associations [60]. Généralement, ces nouvelles associations doivent être ajoutées pour la navigation directe pour éviter une longueur supérieure à une unité dans le chemin de navigation. Toutefois, il se pourrait que certaines classes conceptuelles ne soient pas un objectif à visiter lors de la navigation, dans ce cas, il y a lieu de fusionner les points de navigation résultats de ces classes.

La navigation à l'intérieur de l'application Web se produit le long des associations, qui sont utilisées pour décrire la relation entre les classes de navigation. Ces associations apparaissent comme des liens hypertextes dans l'interface utilisateur. Nous pourrions trouver des lignes

directrices utiles et notations graphiques décrits en détail par l'UWE, tels que; «index», «query», et «menu».

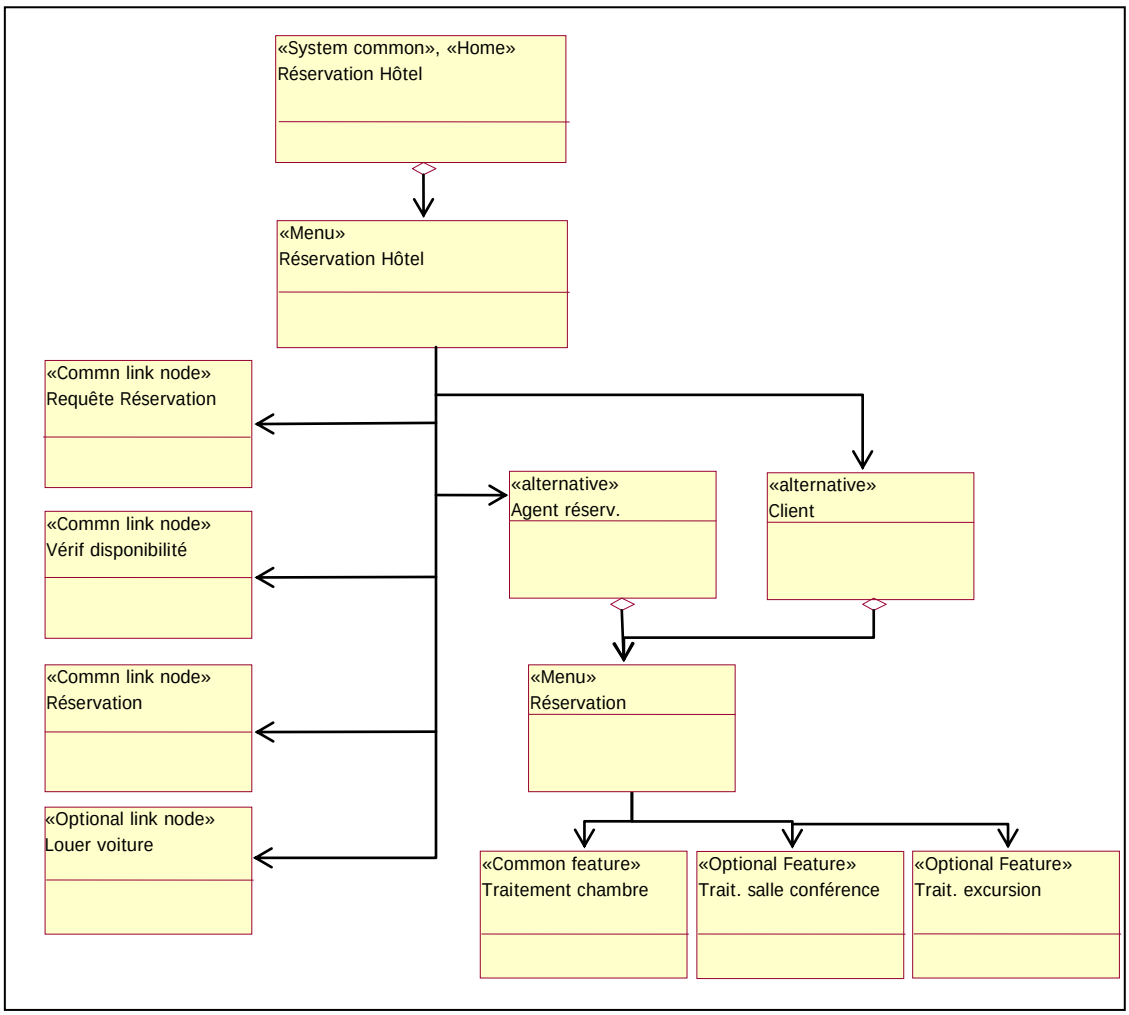

Figure 5.5 : Modèle de Navigation avec variabilité.

Nous pourrions concevoir des modèles appropriés si nous étendons notre modèle de navigation avec ces notations, et ce nouveau modèle pourrait servir de base du système de navigation générée automatiquement. Au contraire à UWE, nous n'introduisons pas de nouveaux éléments graphiques dans les diagrammes UML, mais, nous prenons de nouveaux attributs étendus, à des stéréotypes pour décrire leur fonctionnalité dans la navigation. Cette approche pourrait être vue sur la figure 5.5, correspondant au modèle de contenu illustré par la figure 5.4 et le modèle de features illustré par la figure 5.2.

iii. Modèle de conception

Dans la phase de modélisation de la conception, nous déterminons les architectures possibles où les composants logiciels réutilisables et leurs configurations sont construits. Avec la stratégie de la ligne de produit logiciel, une architecture logicielle sert de cadre pour la composition et l'adaptation des composants de domaine. L'architecture logicielle peut être développée à travers les activités de la division des spécifications de logiciels de haut niveau et la répartition des spécifications divisées en différentes composantes architecturales, compte-tenu des

exigences non fonctionnelles telles que la sécurité, la performance, la fiabilité, etc. [16, 26, 89]. En utilisant les architectures logicielles, les composants de domaine sont configurés et adaptés pour les lignes de produits. Enfin, le code peut être généré automatiquement en fonction de l'architecture du logiciel sélectionnée.

5.3.3. Mapping des features à l'architecture et dérivation

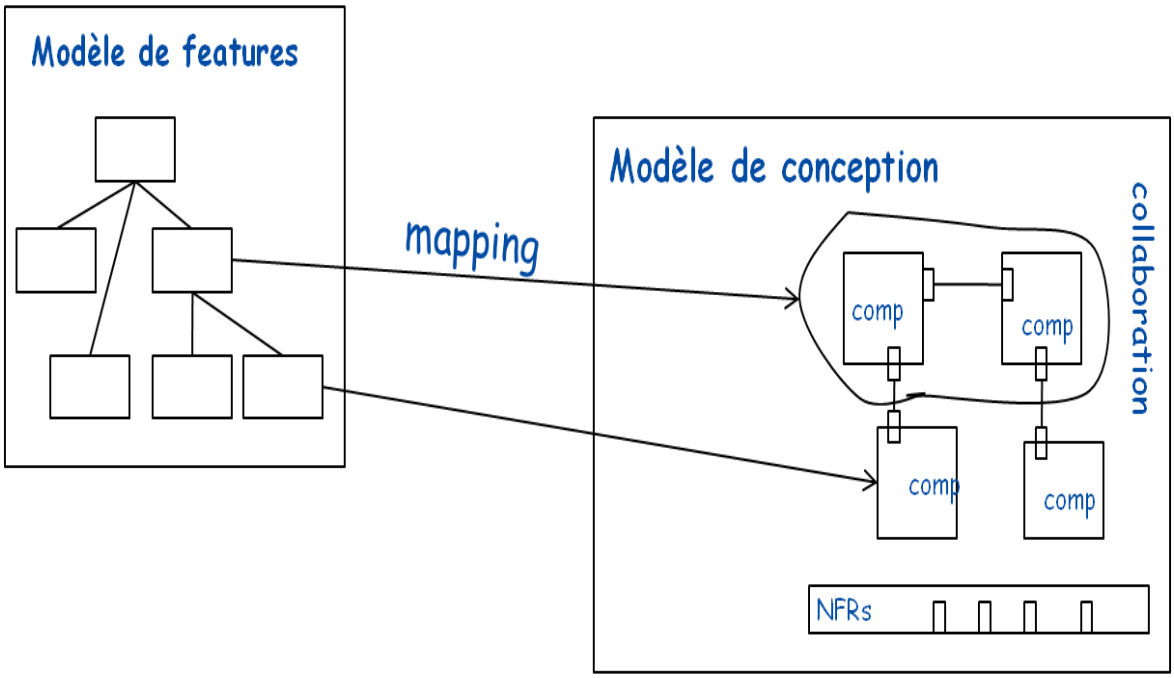

Figure 5.6. Mapping des features.

Les features sont réalisés par lŕarchitecture du système conçu. Le mapping des features se fera compte-tenu de la remarque suivante : un feature est réalisé par une collabroation de composants ou de NFRs (besoins non fonctionnel). Par exemple; le feature « reservation dŕune suite » sera réalisé par la collaboration des composants «chambre suite », « réservation », « client » et « interface reservation ». le feature « reservation chambre simple », sera réalisé par la collaboration des composants «chambre simple», « réservation », « client » et « interface reservation ».

La figure 5.6 illustre le lien entre les features et les collaborations dans une architecture de système conçu.

La dérivation dŕun produit se fait de façon simple et directe, par dŕabord la sélection de features du produit, qui vont nous déterminer les collaborations impliquées; Ces dernières nous donneront les composants impliqués mais seules les interfaces concernées par ces collaborations seront sélectionnées.

5.4. Conclusion

Dans ce chapitre, nous avons présenté notre principale contribution à l'ingénierie LPL du web, au moyen dŕune approche pour la modélisation de la variabilité dans les lignes de produits logiciels pour le web. En effet, nous avons proposé une modélisation multi-vues de la variabilité au moyen de diagrammes UML. Tout au long du processus de modélisation, nous avons défini les modèles de features, leur façon de les obtenir à partir des besoins des utilisateurs, les cas d'utilisation avec variabilité lors de la modélisation des exigences,. Lors de la modélisation de lŕanalyse, nous avons défini le modèle structurel (modèle des entités) et le modèle de navigation.

Lors de la conception, nous avons défini le modèle de l'architecture avec les variabilités, ainsi que la façon de les dériver à partir des modèles obtenus lors des phases précédentes.
Enfin, nous avons illustré et démontrés notre méthode à travers une étude de cas sur le domaine de la gestion des hôtels, clarifiant et expliquant les activités de du processus LPL pour lŕingénierie du web.

6. UNE ARCHITECTURE DE SYSTEME WEB POUR LA SANTE INTEGRANT LE CONCEPT DE LPL

6.1. Introduction

Le principal but de notre système web pour la santé (SWS) est de fournir des services de monitoring de santé, à toute personne et à tout moment, surmontant les contraintes de l'endroit, du temps et de lřespace. Développer un SWS revient à entreprendre la conception, le développement et lřévaluation des solutions de prise en charge médicale intégrant les TICs et la mobilité. Ainsi, notre système de monitoring sřappuie sur la technologie des réseaux sans fil et de lřinformatique mobile. Plus généralement, il relève de lřinformatique ubiquitaire.

Lřinformatique ubiquitaire est un concept qui incorpore tout système de calcul intégré dans un environnement de telle manière que l'interaction homme-machine, interactions entre périphériques mobiles ou les ordinateurs devienne extrêmement naturelle et l'utilisateur puisse obtenir les types multiples de données d'une façon totalement transparente [89]. Lřinformatique ubiquitaire intègre presque chaque aspect de notre vie courante ; aux hôpitaux, les urgences et situations critiques, l'industrie, l'éducation, etc. L'informatique ubiquitaire signifie donc lřutilisation de moyens informatiques (fixes ou mobiles) à tout moment, à tout endroit et à lřaide de tout dispositif multimédia.

Grâce aux nouvelles technologie de lřinformation et de la communication (TIC), anisi que lřinternet, l'informatique mobile apporte de nouvelles classes de dispositifs de calcul, de stockage et de communication sans fil extrêmement puissants, fiables et robuste, devenus omniprésents dans la vie quotidienne. PDAs, smartphones, et tablettes, rendent l'accès à l'information facile et efficace pour chacun n'importe où à tout moment.

Lřintégration de l'informatique mobile aux systèmes de santé est sans cesse croissante et apporte continuellement de nouvelles solutions et innovations dans ce domaine, particulièrement dans le monitoring à distance de patients à maladies chroniques (e-santé).

Les progrès technologiques récents dans les capteurs biologiques, les circuits intégrés de basse puissance, et les communications sans fil ont permis le développement de composants embarqués communicants, à très bas prix, miniaturisés, légers et intelligents. Ces composants sont capables de capter, de traiter, et de communiquer un ou plusieurs signes vitaux. Ils peuvent être reliés par des réseaux sans fil personnels ou de corps pour le monitoring de lřétat de santé de patients même à distance et lřétat.

Notre système e-santé mobile (SWS) consiste en lřutilisation d'outils, équipements et services mobiles basés sur les TICs. Il intègre des domaines technologiques tels que la télémédecine, l'informatique médicale, l'imagerie médicale, le monitoring à distance, etc.

6.2. Architecture technologique

Le SWS rassemble les données physiologiques du patient par les biocapteurs. Les données sont acheminées au PC ou à la cellule phone/PDA d'un patient. Après une analyse de situation critiques et détermination possible dřalerte, ces dispositifs transmettent ces données au serveur pour l'analyse complète. Après que les données soient analysées, le serveur médical fournit les résultats au PC ou à la cellule phone/PDA du patient. Les patients peuvent prendre des mesures nécessaires selon ces résultats. Le SWS est constitué de trois composants essentiels:

1. Réseau local sans fil (RLSF)
2. Serveur local à domicile (SLD) accessible par le patient, son assistant ou un membre de sa famille
3. Serveur médical distant (SMD) accessible par le médecin ou tout autre autorisé.

La figure 6.1 illustre lřarchitecture de ce système ;

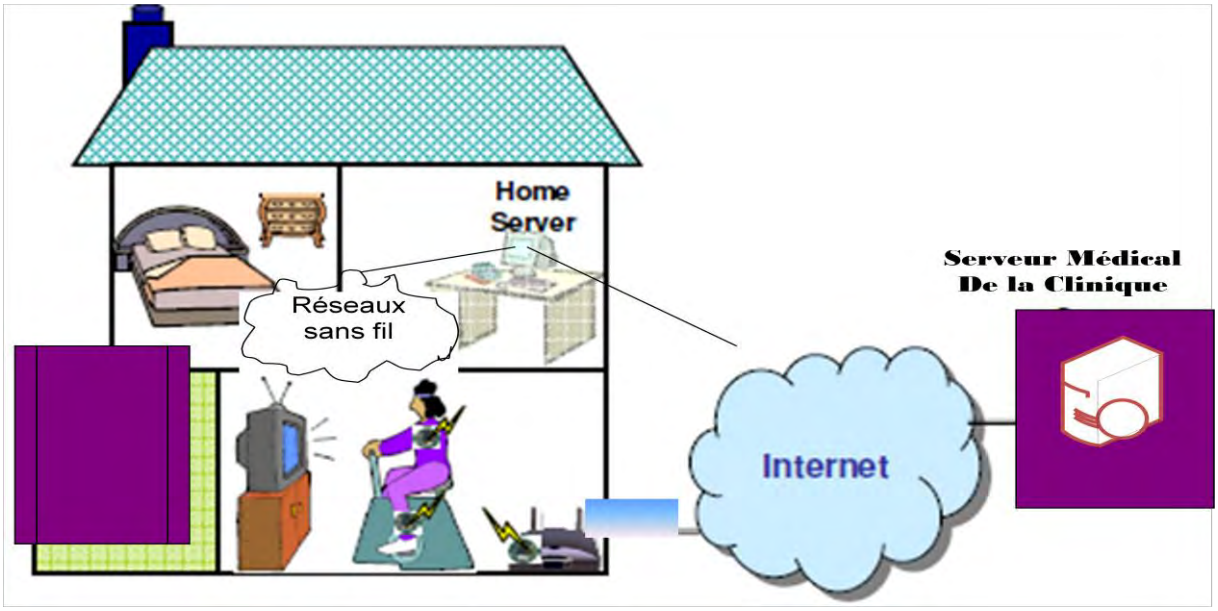

Figure 6.1 : Vue externe du système e-santé mobile (SWS).

6.2.1. Réseau local sans fil (RLSF)

Le réseau local sans fil permet la communication entre les appareils de mesure des paramètres physiologiques du patient et la cellule locale (smartphone, PDA, tablette, ou PC) [74]. La technologie réseaux Bluetooth est utilisée pour des raisons de disponibilité, de faible coût, dřefficacité et de simplicité.

6.2.2. Serveur local à domicile (SLD)

Le serveur local à domicile (SLD) du patient peut être un PC ou des périphériques mobiles telles que smartphone ou PDA. Nous proposons des périphériques mobiles parce qu'il sera plus

approprié que les utilisateurs utilisent à cet effet leurs périphériques mobiles avec lesquelles ils se sont familiarisés.

Les capteurs permettent de collecter de façon continue des données physiologiques du patient. Ils sont dotés de technologie de communication sans fil (Bluetooth), permettant dřacheminer lřinformation au smartphone. Ce dernier analyse ces données, détermine les éléments pertinents. Les patients participeront au processus de soins de santé par leurs périphériques mobiles et peuvent accéder ainsi à leur information de santé de n'importe où n'importe quand.
Le SLD collecte des informations et les analyse. Dans le cas de situations déclarées anormales, il alerte le patient et aussi le médecin au serveur distant (SMD). Ils intègrent les règles et logiques de traitement de façon à déterminer les situations et états du patient selon la tendance obtenue.

6.2.3. Serveur médical distant (SMD)

Du côté du médecin, un serveur est installé (au niveau du cabinet ou clinique). Cřest un serveur médical distant (SMD) auquel les dispositifs mobiles (tablette, smartphone, et même PC) du patient sont connectés par moyen de communication sans fil. A ce niveau, un grand nombre de données médicales sont traitées. Les résultats dřanalyse sont exploités, des suggestions peuvent être formulées aux médecins. De plus, l'information fournie par les spécialistes dans les scénarios critiques peuvent alimenter le serveur.

Le serveur médical distant (SMD) reçoit des données de tous les patients enregistrés, en provenance des SLDs. C'est l'épine dorsale de cette architecture entière. Il est capable de déterminer les seuils spécifiques des paramètres physiologiques de patients. A chaque fois qu'un médecin examine un patient, les résultats de l'examen et les informations sur les traitements éventuels sont stockés dans la base de données du SMD. Ce dernier intègre des traitements informatiques basés sur paradigmes de réseaux neuronaux, de réseaux Bayesiens, ou de modèles mathématiques de détection de changements. Les résultats de ces traitements sont à exploiter par le médecin mais aussi ils peuvent être envoyés au SLD pour une éventuelle consultation par le patient.

Le SMD garde lřhistorique des patients. Il peut intégrer n'importe quelle tendance de maladies pour le patient, localité de famille aussi. Il peut faire apparaître les variations de lřétat santé dues aux changements saisonniers, les épidémies, etc. Ces tendances sont exploitées et commandées et surveillées principalement par les médecins spécialisés.

Ces informations pertinentes concernant la santé générale des personnes peuvent être obtenues. Elle peut aider l'autorité à décider des politiques sanitaires.

6.3. Scenarios essentiels

Dans des unités de soins intensifs, il y a des dispositifs de contrôle continu des patients : les fréquences cardiaques, températures, etc. Ces paramètres sont surveillés sans interruption.
Les patients ayant une forte tension artérielle courir un risque majeur dans le cas où leur situation nřest pas traitée à temps, en raison du changement soudain des pressions. Les tensions artérielles changent soudainement et peuvent être dangereuses pour la vie du patient. Grâce à notre système SWS, elles peuvent être détectées à temps et des alertes de nature à y remédier sont possibles.

Ainsi, les paramètres physiologiques tels que la température, le rythme cardiaque, le niveau de glycémie, la tension artérielle, peuvent être continuellement mesurés, stockés au niveau du SLD et même envoyés au SMD.

Au niveau SMD, des études épidémiologiques peuvent s'effectuer à la suite de la signalisation par le serveur de l'existence de phénomène de maladies, aidant ainsi à arrêter leur propagation. En effet, a chaque fois qu'il constate que plusieurs personnes de la même localité sur une petite période ont la même maladie, il signale ce phénomène aux concernés de sorte que l'autorité puisse prendre une mesure immédiate.

6.4. Architecture topologique

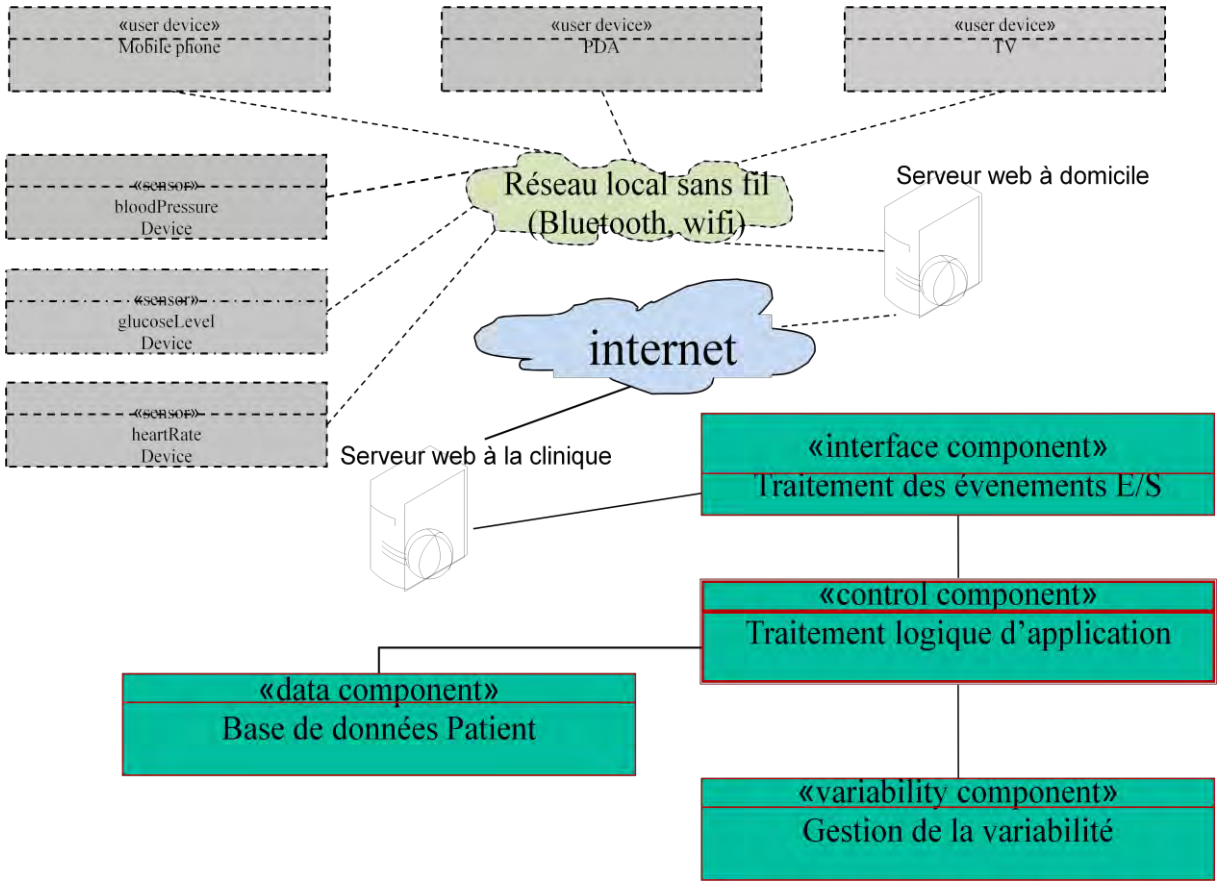

Figure 6.2 : Architecture fonctionnelle de l'SWS

L'architecture de système SWS est simple sans le système complexe ou l'architecture de communication [75]. Bien que l'installation de SWS soit tout à fait sophistiquée mais pour obtenir l'aide de tels biocapteurs intelligents de systèmes de contrôle de santé devez implanter ou avez porté au corps du patient.
Par ailleurs, la sécurité est une question importante dans notre système SWS. La lecture normale d'une mesure peut être changée et les patients cardiaques peuvent être affectés par le

résultat défectueux qui peut même leur causer une crise cardiaque sérieuse. Ainsi sans sécurité, le SWS ne peut être utilisé de façon fiable et sûre.
Les mécanismes de sécurité que nous avons intégrés dans notre application se présentent comme suit:

- Les données doivent être transmises sous la forme chiffrée.
- Authentification Basée Session.
- Sites Web exigeant le login:
- Vérification de lĭinformation sur la session
- Autorisation basée sur le contenu
- Les utilisateurs autorisés peuvent accéder aux enregistrements de données des patients
- Autorisation requise pour tout accès aux données des patients.

6.5. Conclusion

Dans ce chapitre nous avons présenté notre framework architectural pour la conception dĭun système web pour la santé. Ce système utilise un réseau de biocapteurs sans fil, pour la communication entre divers composantes locales (smartphone, PDA, serveur à domicile) et entre le serveur local et le serveur distant au niveau du médecin ou clinique.

Le système SWS permet de fournir aux utilisateurs des services dĭassistance médicale fiables et sécurisés tout en utilisant les outils, équipements et services mobiles surmontant la contrainte du temps et de lĭendroit.

7. CONCLUSION

Les lignes de produits sont un concept permettant de créer plusieurs modèles dǔn produit spécifique à la fois, des modèles identiques qui seront compris dans ce quǒn appelle une famille de produits.

Les caractéristiques principales dǔne ligne de produit est de renfermer des points ou traits communs et des variabilités. Les similarités sont modelées durant la phase de lǐngénierie de domaine qui permet de créer les assets ou les composants réutilisables. Les assets sont la clé de la conception des LPL puisquǔls sont la base de toute lǐarchitecture des différents modèles de la LPL.

Durant lǐétape de lǐngénierie de domaine, les éléments réutilisables seront conçus pour pouvoir apparaitre dans chaque produit formant une LPL afin dǐattester leur appartenance à cette ligne de produits. Les variables, les exigences et les contraintes des LPL seront développées dans lǐngénierie dǐapplication qui repose entièrement sur lǐngénierie de domaine.

Les LPL possèdent des atouts inqualifiables, surtout en termes dǐéconomie. En effet, elles permettent de concevoir durant un temps record un grand nombre de produits avec une qualité supérieure et un coût de production réduit. Toutes ces qualités de la LPL sont développées et promues durant les ingénieries citées ci-dessus et permettent au concepteur de faire dǐénormes économies et de ne plus être obligé de réaliser un produit à partir de zéro chaque fois quǔil en concevra un.

Dans cette thèse nous avons proposé un processus et un modèle de lignes de produits pour les systèmes dǐnformations web, dans le but de tirer profit des LPL qui contribuent réellement à la production et au développement des SIW et qui peuvent les améliorer dans une certaine mesure. Pour ce faire, nous avons défini une notation et modifié le processus de UWE afin de prendre en compte les concepts LPL.

De plus, nous avons défini un Framework de système web pour la santé, tout en intégrant le concept LPL. Ce qui lui permet des fonctionnalités plus consistantes, augmentant le degré dǐadaptabilité et réutilisabilité visant à réduire le coût et les délais de mise sur le marché.
Ce système proposé a été construit en respectant les règles de conception des LPL et présente donc des avantages de taille comme la production massive des applications en usant dǔn minimum de temps, de coût et de meilleure performance.

Comme perspectives de notre travail, nous comptons continuer ce processus dans au moins trois direction: La première direction consiste à appliquer notre approche sur dǐautres cas pratiques de façon à sǐassurer de sa consistance et de son lǐefficacité.

Après quoi, il serait utile de chercher à répondre à quelques questions importantes. En effet, face à un environnement numérique en perpétuel mouvement et en perpétuelle évolution, les LPL assureront-elles à elles seules la promotion et lǐévolution des SIW ou devront-elles être appuyées par dǐautres technologies ou même être supplantées par dǐautres techniques ?.

La deuxième direction consiste à adapter lĭenvironnement ArgoUWE de façon à prendre en charge les nouveaux concepts et éléments que nous avons introduits, et par là-même supporter notre approche.
Enfin, la troisième direction consiste à définir dans notre processus un mécanisme approprié, faisant le lien avec des outils de test et de vérification (model-checking).

8. BIBLIOGRAPHIE

1. Trigaux, J. (2009) : « Les lignes de produits logiciels : Réutilisation et variabilité », Publication périodiques de Samals, Juin 2009.
2. Ben Rhouma Aouina, T. (2012) : « Composition des modèles de Lignes de produits logiciels », Thèse de doctorat à Université Paris-Sud, Ecole doctorale dĭnformatique, université Paris sud, France, 29 nov. 2012..
3. Krueger, C. W. (1992) : « Software reuse », ACM Computing Surveys, vol. 24, no. 2, p. 131Ŕ183, juin 1992.
4. Nyßen, A. , Tyszberowicz, S. et Weiler, T. (2005) : « Are Aspects useful for Managing Variability in Software Product Lines? A Case Study », in Aspects and Software Product Lines: An Early Aspects Workshop at SPLC, 2005.
5. Siy, H. , Zand, M., et Winter, V. (2005) : «The Role of Aspects in Domain Engineering», In Aspects and Software Product Lines (An Early Aspects Workshop at SPLC-Europe), 2005.
6. Groher, I. et Voelter, M. (2007): « Expressing Feature-Based Variability in Structural Models », in Workshop on Managing Variability for Software Product Lines, 2007.
7. Groher, I. et Voelter, M. (2008) : « Using Aspects to Model Product Line Variability », in Workshop on Early Aspects: Aspect-Oriented Requirements and Architecture for Product Lines, SPLC, Limerick, Ireland, pp. 89-95, 2008.
8. France, R. , Fleurey F., Reddy, R. et al.(2006) : « Providing Support for Model Composition in Metamodels », in Enterprise Distributed Object Computing Conference, 2007. EDOC 2007. 11th IEEE International, 2007, p. 253-253.
9. Clarke, S. et Walker, R. J. (2001) : « Composition patterns: an approach to designing reusable aspects », in Proceedings of the 23rd International Conference on Software engineering, Washington, DC, USA, 2001, p. 5Ŕ14.
10. Clarke, S. (2002): « Extending standard UML with model composition semantics », in The Journal of Science of Computer Programming-special issue on Unified Modeling Language (UML) , vol. 44, no. 1, p. 71Ŕ100, juill. 2002.
11. UML 2.4.1 http://www.omg.org/spec/UML/2.4.1/.
12. Atkinson, C. , Bayer, J. et Muthig, D.(2000) : « Component-Based Product Line Development: The Kobra Approach », in First Product Line Conference (SPLC), Kluwer International Series in Software Engineering and Computer Science, 2000, p. 19.
13. Jena, I. et Clauß, M. (2001) : « Generic Modeling using UML extensions for variability». In Proceedings of OOPSLA Workshop on Domain-specific Visual Languages (2001), pp.11-18.
14. Ziadi, T., Hélouët, L. et Jézéquel, J.-M. (2004) : « Towards a UML Profile for Software Product Lines », in Software Product-Family Engineering, vol. 3014, F. J. Linden, Edition Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, p. 129-139.
15. Haugen, Ø., Møller-pedersen, B., Oldevik, J. et al. (2004) : « An MDA-based framework for model-driven product derivation », in the software engineering and applications, p. 709-714, 2004.
16. Gomaa H. (2005) : « Designing Software Product Lines with UML », in Proceedings of the 29th Annual IEEE/NASA Software Engineering Workshop - Tutorial Notes, Washington, DC, USA, 2005, p. 160Ŕ216.
17. Schwabe D., Rossi G., Barbosa S. (1996) : ŗSystematic Hypermedia Application Design with OOHDMţ, Proceedings of ACM Hypertext, 1996.

18. Tessier, P. (2005) : «Conception de modèles de familles de systèmes temps réel». 2005.
19. Kang, K. C. , Cohen, S. G. , Hess, J. A. et al.(1990) : « Feature-Oriented Domain Analysis (FODA) Feasibility Study », Technical report, CMU/SEI TR-21, USA, nov. 1990.
20. Schwabe D. & Rossi G. (1998) : ŗAn Object Oriented Approach to Web-Based Application Designŗ, Theory and Practice of Object Systems, 4(4), Wiley and Sons, New York, 1998.
21. Cowan D.D. & Lucena C.J.P. (1995) : ŗAbstract Data Views, an Interface Specification Concept to Enhance Design for Reuseŗ, IEEE Transactions on Software Engineering, 21(3), March 1995.
22. Alves, V. , Gheyi, R. , Massoni, T. et al.(2006): « Refactoring product lines », in Proceedings of the 5th international conference on Generative programming and component engineering, New York, NY, USA, 2006, p. 201Ŕ210.
23. Segura, S. Benavides, D., Ruiz-cortés, A. et al. (2008) : « Automated Merging of Feature Models using Graph Transformations » in Generative and Transformational Techniques in Software Engineering II, Berlin, 2008,p. 489-505.
24. Czarnecki, K. Peter Kim, C. H. et Kalleberg, K. T. (2006) : « Feature Models are Views on Ontologies », in Proceedings of the 10th International on Software Product Line Conference, Washington, DC, USA, 2006, p. 41Ŕ51.
25. Pohl, K. , Böckle, G. et Linden, F. V. D. (2005) : ŗSoftware Product Line Engineering: Foundations, Principles and Techniquesŗ. Birkhäuser, 2005.
26. Jacobson I., Booch G. & Rumbaugh J. (1999) : «The Unified Software Development Process», Addison-Wesley 1999.
27. Kadri R. (2010): «Une approche pour la Modélisation dŕApplications Web à base de Composants Logiciels », Thèse de Doctorat à lŕUniversité de Bretagne Sud.
28. Conallen J. (2002): « Modeling Web Applications with UML », 2nd Edition. Addison-Wesley Professional.
29. Villanova-Oliver M. (2002) : « Adaptabilité dans les systèmes dŕinformation sur le web : Modélisation et mise en œuvre de lŕaccès progressif ». Thèse pour lŕobtention du grade de docteur en informatique, Institut National Polytechnique de Grenoble
30. Baresi L., Garzotto F. & Paolini P. (2000): « From Web Sites to Web Applications: New Issues for Conceptual Modeling». In Proceedings of the International Workshop on The World Wide Web and Conceptual Modeling, co-located with the 19th International Conference on Conceptual Modeling, Salt Lake City (USA), October 2000.
31. Mecca G., Merialdo P., Atzeni P. & Crescenzi V. (1999): « The ARANEUS Guide to Web-Site Development», ARANEUS Project Working Report, AWR-1-99, March 1999.
32. Isakowitz, T., Bieber, M., & Vitali, F. (1998). Web Information Systems. Communications of the ACM, 41(7), 78-80.
33. Ceri S., Fraternali P. & Matera M. (2002), ŗConceptual modeling of data-intensive Web applicationsŗ. IEEE Internet Computing, à paraître en 2002.
34. De Troyer O.M.F., Leune C.J., WSDM (1998) : ŗAUser-Centered Design Method for Web Sitesŗ, Proceedings of the 7th International World Wide Web Conference (WWW7), Computer Networks, 30(1-7), April, 1998, pp. 85-94.
35. Takahashi K., Liang E.(1997) : ŗAnalysis and Design of Web Based Information Systemsŗ, Proceedings of the 6th International World Wide Web Conference (WWW6), Santa Clara, California, USA, April 1997.

36. Baker, S. (2000) : ŖGetting the Most from your Intranet and Extranet Strategiesŗ, The Journal of Business Strategy, 21(4), 40-43.
37. Lang, M. (2001) : ŖA Study of Practice in Hypermedia Systems Designŗ, Doctoral Consortium, European Conference on Information Systems, Bled, Slovenia, 24-26 June, 2001.
38. Bieber, M. & Vitali F. (1997) : ŖToward Support for Hypermedia on the World Wide Webŗ, IEEE Computer, 30(1), January 1997.
39. Xiong, J. (2008) : « Une méthode dĭnspection automatique de recommandations ergonomiques tout au long du processus de conception des applications Web ». Thèse de doctorat, Université Toulouse III - Paul Sabatier
40. Rossi, G.; Schwabe, D. (2006) : ŖModel-Based Web Application Development, In Web Engineeringŗ, Chapter 10, Springer-Verlag, pp. 303-333.
41. Malak, G., Belkhiter, N., Badri, M., & Badri, L. (2001) : « Évaluation de la qualité des applications Web », Rapport de recherche, Département dĭnformatique, Université Laval, Québec, Canada, Octobre, 48 pages.
42. Fraternali, P., Paolini P. (1998) : ŖA Conceptual Model and a Tool Environment for Developing More Scalable and Dynamic Web Applicationsŗ, in Proceedings of EDBTĭ98, Valencia, Spain, March 1998, pp. 421435.
43. Ceri, S. , Comai, S. , Damiani, E. , Fraternali, P., Paraboschi, S. and Tanca, L..(1999) : ŖXML-GL : A graphical language for querying and restructuring WWW data (1999) ŗ, In Proceedings of the 8th Int. WWW Conference, WWW8, Toronto, Canada, May 1999.
44. Ceri S., Fraternali P. & Bongio A. (2000) : ŖWeb Modeling Language (WebML): a modeling language for designing Web sitesŗ, Proceedings of the 9th International World Wide Web Conference (WWW9), Amsterdam, Netherlands, May 15 - 19, 2000.
45. Ceri, S., Fraternali, P., Bongio, A. & Comai, S. (2001) : « WebML User Guide, Politecnico di Milano », 2001. disponible à lĭadresse : http://Webml.elet.polimi.it/Webml/manuals/WebML_User_Guide.pdf
46. Isakowitz, T., Kamis, A. & Koufaris, M. (1997): ŖExtending the capabilities of RMM: Russian Dolls and Hypertext, Proceedings of the 30th Hawaii International Conference on System Sciences (HICSS-30), 1997.
47. Isakowitz, T., Kamis, A. & Koufaris, M. (1997) : Ř Reconciling Top-Down and Bottom-Up Design Approaches in RMMŗ, Proceedings of WITS-97, Atlanta, GA, USA, December 1997.
48. Diaz, A. & Isakowitz, T. (1995) : Ř RMCase: Computer-Aided Support for Hypermedia Design and Developmentŗ, International Workshop on Hypermedia Design, 1995.
49. Gabbrielli, M., Marchiori, M. & de Boer, F.S. (1998) : ŖDynamic Web sitesŗ, The Query Languages Workshop (QL'98), Boston, Massachussets, December 3-4, 1998.
50. Frasincar, F., Houben, G-J. & Vdovjak, R. (2001) : ŖAn RMM-Based Methodology for Hypermedia Presentation Designŗ, Proceedings of the 5th East European Conference on Advances in Databases and Information Systems (ADBIS 2001), LNCS 2151, Vilnius, Lithuania, September 25-28, 2001, pp. 323-337.
51. Frasincar, F. & Houben, G-J (2002) : ŖHypermedia Presentation Adaptation on the Semantic Web, in de Bra P., Brusilovsky P. & Conejo R., (Eds.)ŗ, Proceedings of the 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH 2002), Malaga, Spain, May 29 - 31, 2002, LNCS 2347, 2002, pp. 133-142.

52. Lou, B. (2014): ŖStudy on View-Oriented Navigation Modeling of Web Applicationsŗ, TELKOMNIKA Indonesian Journal of Electrical Engineering, Vol.12, No.3, March 2014, pp. 2356 ~ 236514
53. Rossi, G., Schwabe, D. & Lyardet, F. (1999) : ŖWeb application models are more than conceptual modelsŗ, in Proceedings of the World Wild Web and Conceptual Modeling'99 Workshop, ER'99 Conference, LNCS 1727, Springer, Paris, 1999, pp. 239-252
54. Güell, N., Schwabe, D. & Vilain, P. (2000) : ŖModeling Interactions and Navigation in Web Applicationsŗ, in Proceedings of the World Wild Web and Conceptual Modeling Workshop, ER'00 Conference, LNCS1921, Springer, Salt Lake City, 2000.
55. Caroll, J.M (1995) : ŖScenario-based Design: Envisioning Work and Technology in System Developmentŗ, John Wiley & Sons, 1995.
56. Martinez, J., Lopez, C., Ulacia, E. and Hierro, M. D. (2009): ŖTowards a Model-Driven Product Line for Web Systemsŗ, Proceedings of the Fifth International Workshop on Model-Driven Web Engineering (MDWE 2009), San Sebastián, Spain, June 22, 2009.
57. Vilain, P., Schwabe, D. & De Souza, C.S. (2000) : ŖUse Cases and Scenarios in the Conceptual Design of Web Applicationsŗ, Technical Report MCC 12/00, Departemento de Informática, PUC-Rio, 2000.
58. Ramez, E. and Shamkant, N. (1994) : ŖFundamentals of database Systemsŗ, (2nd ed.) Benjamin/Cummings, Redwood, CA, 1994.
59. Detlor, B. (1999) : Ř Utilizing Web Information Systems for Organizational Knowledge Work: An Investigation of the Information Ecology and Information Behaviors of Users in a Telecommunications Companyŗ, In Doctoral Research Forum, 62nd Annual Meeting of the American Society for Information Science. Washington, D.C.
60. Adamkó, A. (2006) : ŖUML-Based Modeling of Data-oriented WEB Applicationsŗ, Journal of Universal Computer Science, vol. 12, no. 9 (2006), 1104-1117, September 2006.
61. Hennicker, R., Koch, N. (2000) : ŖA UML-based Methodology for Hypermedia Designŗ, In Proceedings of the Unified Modeling Language Conference, UMLř2000, 2000, Evans and Kent S., Eds. LNCS 1939, Springer Verlag, pp. 410-424.
62. Koch, N., (2000) : ŖSoftware Engineering for Adaptative Hypermedia Systems Ŕ Reference Model, Modelling Techniques and Development Processŗ, Ph.D Thesis, Fakultät der Mathematik und Informatik, LudwigMaximilians-Universität München, December 2000.
63. Baumeister, H., Koch, N. & Mandel, L. (1999) : ŖTowards a UML extension for hypermedia designŗ. In France R. & Rumpe B. (eds), UML´99 Ŕ The Unified Modeling Language - Beyond the Standard, LNCS 1723, Springer Verlag, pp. 614-629, 1999.
64. Hennicker R. & Koch N., A UML-based Methodology for Hypermedia Design. In Evans A., Stuart S. & Selic B. (eds), UML'2000 - The Unified Modeling Language - Advancing the Standard, LNCS 1939, Springer Verlag, York, England, October 2000.
65. Van Gurp, J., Bosch, J., Svahnberg, M. (2001) : ŖOn the notion of variability in software product linesŗ, In Procs. Working IEEE/IFIP Conference on Software Architecture (WICSA'01), page 45,.
66. Patrick, K. Griswold, W. G. Raab, F. and Intille S. S. (2008) : ŖHealth and the mobile phone,ŗ Amer. J. Prev. Med., vol. 35, no. 2, Aug. pp. 177Ŕ181,.

67. Chenhui, Z., Huilong D., Xudong, L.(2008) : "An integration Approach of Healthcare Information Systems",Proc. IEEE Conf. on Biomedical Engineering and Informatics, pp.606-609.
68. Ceri, S., Fraternali, P. & Bongio, A. (2000) : "Web Modeling Language (WebML): a Modeling Language for Designing Web sites", Computer Networks, 33(1–6):137–157, 2000.
69. Koch, N. and Kraus A.(2002): "The expressive power of UML-based web engineering", In IWWOST, Springer, LNCS- 2548, pp. 105–119.
70. Loques, O., Sztajnberg, A. (2010) : "Adaptation Issues in Software Architectures of Remote Health Care Systems", in 2nd SEHC, Cape Town, South Africa, pp. 24–28.
71. Sergio, T. Carvalho,_S. T., Murta, L., Loques, O. (2012) : "Variabilities as First-Class Elements in Product Line Architectures of Homecare Systems", SEHC, June.
72. Sztajnberg, A. Rodrigues, A., Bezerra, L., Loques,O. Copetti, A. and. Carvalho S. T. (2009) : "Applying Context-aware Techniques to Design Remote Assisted Living Applications", Int. Journal of Functional Informatics and Personalized Medicine, vol. 2, no. 4, pp. 358–378.
73. Koch, N., Knapp, A.,Zhang, G. and Baumeister, H. (2008) : "UML-based web Engineering", chapter 7, pages 157-191. Human-Computer Interaction Series. Springer, 2008.
74. Clements, P., Northrop L.. (2002) : "Software Product Lines: Practices and Patterns", SEI Series in Software Engineering, Addison-Wesley, 2002.
75. Gomaa, H. (2004) : "Designing Software Product Lines with UML: From Use Cases to Pattern-based Software Architectures", The Addison-Wesley Object Technology Series, 2004.
76. Balzerani, L., Ruscio D. Di, Pierantonio, A., De Angelis G. (2005) : "A Product Line Architecture for Web Applications", ACM Symposium on Applied Computing 2005.
77. Preston, P., & McCrohan, K. (1998) : "A Strategy for Extranet Development for Professional Programs", International Journal of Educational Management, 12(4).
78. Al-hakim, L. (2007) : "Web and Mobile-Based Applications for Healthcare", Management, Hershey, PA: IRM Press,2007.
79. Kolitsi, Z. and Umpierrez, M. (2007) : "Personal Health Systems: Deployment opportunities and ICT research challenges",Conference report, February 12-13, 2007,Brussels, http://ec.europa.eu/information_society/ehealth.
80. Kimour M. T., Boudour R., Ghanemi S. (2012) : "Closing the gap between the requirements specification and the design model for embedded systems", Second International Symposium on Modelling and Implementation of Complex Systems (MISC 2012), Constantine, Algeria, may 2012.
81. Sidi Mohamed O. Moulaye Abdellahi, Mohamed Tahar Kimour, and Mbaye Sene (2012) : "Designing Web Embedded Systems for healthcare", in proc. of the 1st International Conference on Software Engineering and Technologies December 15-17, 2012. Hammamet, Tunisia.
82. Roger S. P. and David L. (2009) : "Web Engineering: A Practitioner's Approach", New York: McGraw-Hill. 2009.
83. Conallen, J. (2003) : "Building Web Applications with UML 2nd Edition, Addison Wesley, 2003.
84. Czarnecki, K., Antkiewicz, M. and Peter Kim, C. H. (2006): "Multi-level customization in application engineering". Commun. ACM, 49(12):60-65, 2006.
85. Ginegi A. and Murgesan S. (2003): "The Essence of Web Engineering", IEEE Multimedia, Vol. 8, no. 3, 2003

86. Balzerani, L., Ruscio D. D., De Angelis A. (2006): "Supporting web applications development with a product line architecture", journal of web engineering, vol. 5, no. 1 (2006) 025–042
87. Hallsteinsen S., M. Hinchey, S. Park, and K. Schmid (2008): "Dynamic software product lines" , Computer, vol. 41, no. 4, pp. 93–95, 2008.
88. Krueger C. W. (2006) : "New methods in software product line practice". Commun. ACM, vol. 49, no. 12, pp. 37–40, 2006.
89. Gomaa H.and Shin M.E. (2008) : "Multiple-View Modeling and Meta-Modeling of Software Product Lines", Journal of IET Software, Vol. 2, Issue 2, pp. 94-122, April 2008.
90. Sidi Mohamed O. Moulaye Abdellahi, Mbaye Sene, Mohamed T. Kimour , feature-oriented web applications engineering, Third International Symposium on Modelling and Implementation of Complex Systems (MISC'14), El-Oued, Algeria, November 29-30, 2014
91. Mohamed T. Kimour, Sidi Mohamed O. Moulaye Abdellahi, Mbaye Sene, An SPL Approach to Embedded Systems Engineering, International Conference on embedded systems in telecommunications and instrumentation (ICESTI), Annaba, Algeria, October 27 - 29, 2014.
92. Sidi Mohamed O. Moulaye Abdellahi, Mohamed T. Kimour, Mbaye Sene, (2014), A Product-Line Approach to Design an Embedded Web System for Healthcare. International Journal of Control Science and Engineering, Volume 2, Number 2, April 2014, ISSN 2328-2231, David Publishing Company, USA.
93. Acher, M., Collet, P., Lahire, P. et al. : « Composing Feature Models », in Software Language Engineering, vol. 5969, M. Brand, D. Gaševic, et J. Gray, Edition Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, p. 62-81.
94. Apel, S., Trujillo, F. J, S. et Kästner, C. (2009) : « Model Superimposition in Software Product Lines », in Proceedings of the International Conference on Model Transformation (ICMT), volume 5563 of LNCS, p. 4-19, 2009.
95. Apel, S. et Lengauer, C. (2008) « Superimposition: A Language-Independent Approach to Software Composition », in Software Composition, 2008, p. 20-35.
96. Mecca G., Merialdo P., Atzeni P. & Crescenzi V. (1999): « The ARANEUS Guide to Web-Site Development», ARANEUS Project Working Report, AWR-1-99, March 1999.
97. Fleming J.(1999) : « WEB navigation: designing the user experience», O'Reilly & Associates, 1999.
98. Yang, J.-T. et autre. (1988) : « A Toolset to Support Web Application Testing». International Computer Symposium (ICS'98), Taiwan, 1998.
99. Adnane G. (2006) : « Test des applications web: modélisation et génération de séquences de test basées sur le contrôle ». Université du Québec à trois-rivières.
100. Zelnick N., Nifty Technology and Non conformance (1988) : "The Web in Crisis, Computer", October 1998, pp. 115-119.
101. Nielsen J., & Molich R. (1990) : "Heuristic Evaluation of User Interfaces", ACM SIGCHI Bulletin, pp. 249-256.
102. Keevil B. (1998) : "Measuring the Usability of Your Web Site", http://www3.sympatico.ca/bkeevil/sigdoc98
103. Boldyreff, C., Gaskell, C., Marshall, A, & Warren, P. (2000). "Establishing a Measurement Programme for the World Wide Web", Août 2000. http://www.dur.ac.uk/cornelia.boldyreff/WEB-SEM/ias.html

104. Olsina, L., Godoy D., Lafuente, G., & Rossi, G. (1998). "Quality Characteristics and Attributes for Academic Web Sites", Web Engineering Workshop, Toronto, Canada.
105. Houben, G.J. (2000) : "Hera: Automatically Generating Hypermedia Front Ends for Ad Hoc Data from Heterogeneous and Legacy Information Systems", in Proceedings of the 3rd International Workshop on Engineering Federated Information Systems, Aka and IOS Press, 2000, pp. 81–88.
106. Klyne, G., Reynolds, F., Woodrow, C. & Ohto, H. (2001) : "Composite Capability/ Preference Profile (CC/PP)", Structure and Vocabularies, W3C, 2001, http://www.w3.org/TR/CCPP-struct-vocab.
107. Ceri, S., Fraternali, P., Bongio, A. & Maurino, A. (2000) : "Modeling data entry and operations in WebML", WebDB 2000. Dallas, 2000.
108. Booch, G., Rumbaugh, J., Jacobson, I. (1999) : "The UML user guide", Addison-Wesley,1999.
109. Rumbaugh, J., Jacobson, I., & Booch, G. (1999) : "The UML reference manual, Object Technology Series", Addison-Wesley, 1999.
110. Garman, N. (1996) : "The Inverted File: From the Internet to Intranets", Online, 20(3), 8. Adresse URL http://www.onlineinc.com/onnlinemag/MayOL/invert5.html
111. Sidi Mohamed O. Moulaye Abdellahi, Mohamed Tahar Kimour, and Mbaye Sene (2012) : "A product line approach to design an Embedded Web System for healthcare", in proceedings of the International Conference on Embedded Systems in Telecommunications and Instrumentation, November 5, 6 and 7 2012, Annaba Algeria
112. Sidi Mohamed O. Moulaye Abdellahi, Mohamed Tahar Kimour, and Mbaye Sene (2013) : "An Architecture Framework to Design a Healthcare Information System", in proceedings of the International Conference on Information Systems and Technologies, March 22, 23 and 24 2013, Tangier, Morocco.
113. Sidi Mohamed O. Moulaye Abdellahi, Mohamed Tahar Kimour, and Mbaye Sene (2013) : "An Architecture Framework to Design a Healthcare Information System", in Business Intelligence and Mobile Technology Research: An Information Systems Engineering Perspective. Cambridge Scholars Publishing, Newcastle upon Tyne, NE6 2XX, UK : Mohamed Ridda Laouar and Sean B. Eom, 2014, p. 214-225, ISBN (10): 1-4438-5507-3, ISBN (13): 978-1-4438-5507-5.

Nom et prénoms du Candidat : Sidi Mohamed Ould Moulaye Abdellahi MAKHTOUR

Titre de la thèse : Modèle et Processus de Lignes de Produits pour les Systèmes d'Information Web

Date et lieu de soutenance : le 23/07/2015 à l'UCAD

Jury :	Président : M. Hamidou DATHE	Professeur	UCAD, Dakar, Sénégal
	Membres : M . Joel RODRIGUES	Professeur	UBI, Covilhã, Portugal
	M. Mohamed Tayeb LASKRI	Professeur	UBM, Annaba, Algérie
	M. Abdourahmane RAIMY	Maître de Conf.	UCAD, Dakar, Sénégal
	M. Oumar DIANKHA	Professeur	UCAD, Dakar, Sénégal
	M. Mbaye SENE,	Maître de Conf.	UCAD, Dakar, Sénégal
	M. Mohamed Taher KIMOUR	Professeur	UBM, Annaba, Algérie

Résumé : Les systèmes d'information web sont des éléments cruciaux dans la gestion des entreprises. Les facteurs de productivité, coût et délais de mise sur le marché sont des paramètres clés lors de leur développement. L'ingénierie des lignes de produit logiciel est une nouvelle approche de génie logiciel ayant principalement pour buts de répondre au mieux à ces exigences.

Cette thèse présente un modèle et un processus de développement de systèmes d'information web, basé sur le concept des lignes de produit logiciel. Ce concept offre une grande souplesse et un haut degré de réutilisablité de nature à concevoir et réaliser efficacement ces systèmes. Basé sur UML et défini selon les trois vues de contenu, de navigation et de présentation, le modèle proposé est doté de règles de passage entre les features comme étant un moyen de spécification des besoins et l'architecture qui représente la conception du système.

Mots clés : Systèmes d'information Web, Lignes de Produits Logiciels, UWE, Features.

Abstract: Web information systems are crucial elements for enterprise management. Productivity factors, cost and time-to-market are key parameters for the development of such systems. Product-line engineering is a new approch in software engineering, especially aiming to efficiently achieve such requirements. This thesis presents a model and a development process of web information systems, based on the concept of software product lines.

This concept offers great flexibility and a high degree of reusability in order to effectively design such systems. Based on UML and defined according to three views of content, navigation and presentation, the proposed model offers transition rules between features as a requirements specification tool and the architecture that represents the system design.

Keywords : Web information system, Software product line, UWE, Features.