2

Le découpage d'un projet et les modèles de cycle de vie

2.1 LES PRINCIPES DU DÉCOUPAGE

Nous avons vu au chapitre 1 que le management de projet comporte trois aspects représentés dans le triangle Projet : la production, les ressources et le temps.
Une des premières responsabilités du chef de projet est de découper le projet pour pouvoir répartir dans le temps la production et les ressources.
Le découpage doit s'appuyer à la fois sur l'approche cartésienne de réduction de la difficulté[1] et sur l'approche systémique de prise en compte des liens entre les éléments.
Découper un projet consiste ainsi à identifier des sous-ensembles quasi autonomes, présentant les caractéristiques suivantes :
• chaque sous-ensemble du projet donne lieu à un résultat bien identifié ;
• la charge propre à chacun peut être évaluée ; • les contraintes d'enchaînement entre les sous-ensembles sont repérables : certains sous-ensembles peuvent être réalisés parallèlement, d'autres sont liés entre eux par des contraintes d'antériorité ;
• le découpage est fait à des mailles différentes, un sous-ensemble étant souvent à son tour décomposé.
On utilise deux grands critères pour découper un projet : l'un est temporel, l'autre structurel. Ces deux critères ne sont pas exclusifs.
Le *critère temporel* est utilisé dans la plupart des projets. Il permet de répartir le travail dans le temps : la décomposition fait apparaître une succession d'étapes et de phases. À chacune, on attache une date de début prévue et une date de fin visée.
La figure 2.1 en donne une représentation avec le formalisme UML
Un projet se compose de phases ; chaque phase comprend un certain nombre d'activités[1] ; une activité est définie par une ou plusieurs tâches à effectuer.
À chaque élément de décomposition on attache un résultat à atteindre, appelé livrable, qui peut faire l'objet d'un engagement contractuel.
L'ensemble ordonnancé des phases d'un projet s'appelle le **cycle de vie du projet**.
Le découpage temporel poursuit deux objectifs : il balise et il guide. En effet, chaque date représente un jalon permettant de marquer les points de décision du parcours. De plus, la distribution dans le temps reflète un cheminement intellectuel.

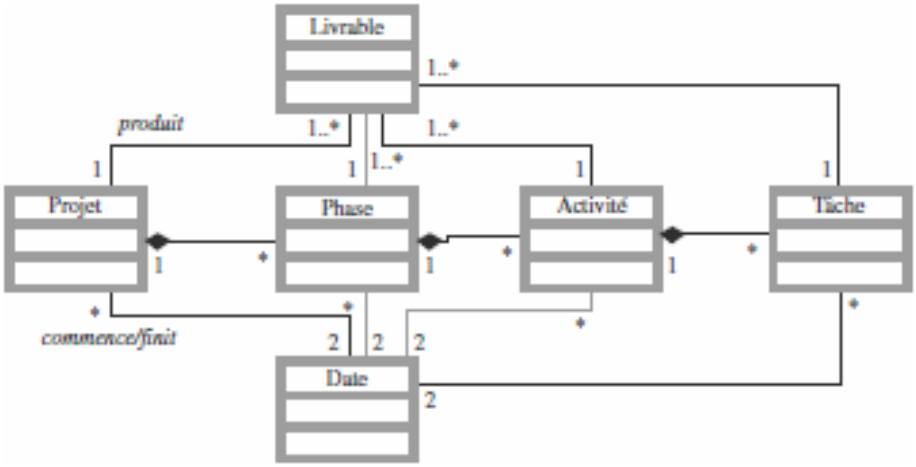

Figure 2.1 — Le découpage temporel d'un projet.

La plupart des méthodes de conception de système d'information proposent un tel cheminement, souvent appelé cycle de développement ou modèle de cycle de vie. Le découpage temporel est souvent de type descendant (*top-down*, du haut vers le bas), c'est-à-dire qu'il favorise :

• une visibilité croissante, car les résultats sont de plus en plus précis et la maille d'étude de plus en plus fine ;

• une progression réelle des travaux, dans la mesure où les résultats consolidés en fin d'une étape ne sont pas remis en question dans les étapes suivantes.

Dans tous les cas, le cheminement intellectuel permet de converger, c'est-àdire d'éviter de partir continuellement sur de nouvelles pistes. Il doit être économique, évitant de faire et défaire. Le client et le chef de projet ont tous deux intérêt à découper le projet dans le temps, car ainsi :

• *Le client peut valider et orienter le projet*.

Le découpage temporel permet au client de s'assurer progressivement que ce qui a été conçu traduit bien les objectifs généraux, de faire des choix, éventuellement de réorienter le travail. En général, la fin d'une phase se traduit par la livraison d'une fourniture contractuellement définie. La fin d'une activité donne lieu à la remise de produits intermédiaires.

• *Le chef de projet peut baliser le déroulement du projet*.

S'il a découpé son projet en tranches, le chef de projet peut effectuer une planification et en suivre pas à pas l'avancement. La fin de chaque tranche est comme un jalon où il regarde plus particulièrement s'il n'y a pas des signaux inquiétants concernant l'état de santé du projet.

Le *critère structurel* permet d'organiser le travail en se basant sur la structure du produit final : la décomposition fait apparaître les différents modules qu'il faut obtenir (figure 2.2).

L'utilisation de ce critère requiert une visibilité suffisante sur le résultat à produire.

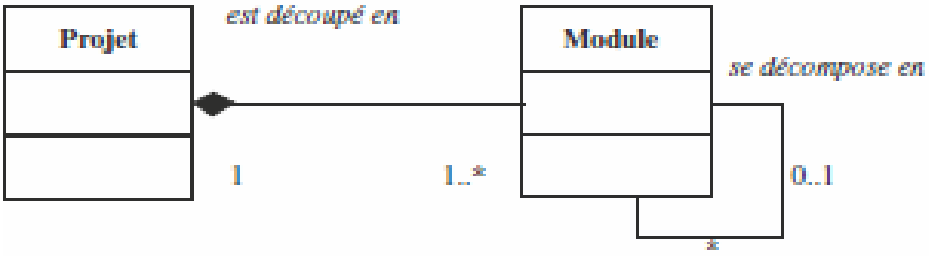

Figure 2.2 — Le découpage structurel d'un projet.

En plus du découpage temporel et surtout si le projet est de taille importante, on recourt au découpage structurel qui présente plusieurs avantages.
• *Maîtriser le projet*.
Le découpage conduit à des sous-ensembles cohérents d'une taille plus réduite et plus facile à maîtriser.
• *Répartir les responsabilités*.
L'autonomie des modules autorise leur répartition dans des sous-projets séparés, dont la réalisation est confiée à différents responsables ou éventuellement sous-traitée.
• *Réduire les délais planifiés*.
Certains modules indépendants sont développés en parallèle, ce qui permet d'avancer la date théorique d'achèvement du projet.
• *Avoir un développement incrémental*.
Pour différentes raisons (taille, budget, délais), on choisit parfois de développer un système d'information par versions successives (en général trois ou quatre), chaque version comportant un nombre croissant de modules par rapport à la précédente. Le découpage structurel est alors essentiel pour définir le contour de chacune d'elles.

2.2 LES DÉCOUPAGES NORMALISÉS

Les normes internationales proposent trois découpages : PBS, WBS et OBS.
Le découpage PBS, *Product Breakdown Structure* (structure de décomposition du produit), correspond au découpage structurel : ce sont les différents composants du produit final. Soit par exemple un progiciel de gestion des valeurs mobilières à développer (figure 2.3). Le PBS représente le découpage du progiciel en modules, chacun assurant une fonction spécifique. Trois principaux modules ont été identifiés :
un référentiel des différents titres (la base Valeur), la tenue de la comptabilité Titres (comptabilité) et la gestion d'un carnet d'ordres passés sur les marchés (ordres de Bourse).
Ce dernier module contient à son tour deux parties :
l'enregistrement des ordres (carnet d'ordres) et le traitement administratif des ordres effectivement passés (dénouement).
Le PBS est parfois appelé « structure du produit » ou « arborescence produit »,« *représentation des liens de composition entre les divers constituants d'un produit complexe »* [AFITEP, 2000].
Le découpage WBS, *Work Breakdown Structure* (structure de décomposition du travail), représente, sous forme d'une arborescence, les différents composants de travail nécessaires pour parvenir au résultat tel qu'il est décrit dans le PBS. Il s'appuie, en général, à la fois sur le critère structurel et sur le critère temporel.
L'AFITEP et l'AFNOR le définissent comme le « découpage hiérarchisé et arborescent du processus de réalisation en éléments plus faciles à analyser et à maîtriser, appelés lots de travaux ou tâches ». Il apporte une réponse aux deux questions : « Que doit-on faire ? Comment doit-on s'y prendre ? ».
En reprenant l'exemple ci-dessus, le WBS pourrait se présenter comme illustré à la figure 2.4.
L'on mène d'abord une conception générale sur l'ensemble sur domaine ; l'on poursuit le travail à travers trois sous-projets, dont on décidera, au moment de les planifier, de les mener ou non en parallèle.
Chacun comporte une phase de conception, puis un développement et une phase de tests.
Dans le cas du sous-projet Ordres de bourse, on a distingué deux phases de développement, correspondant à chacun des sous-modules Carnet d'ordres et Dénouement ; dans le cas du sous-projet Comptabilité, on s'appuie sur un progiciel existant qu'il s'agit de découvrir et paramétrer.
Enfin, les trois logiciels issus des sous-projets font l'objet d'une intégration.
L'AFITEP traduit le terme anglais WBS par « Organigramme des tâches » ou « OT ». L'IPMA considère cependant que l'OT est la représentation graphique du WBS, sous forme d'un diagramme tel celui que nous présentons au chapitre 4 (réseau PERT ou diagramme des antécédents). Le logiciel MS Project, dans sa
version 2003, utilise le terme OT dans cette même acception.
Le PMI propose pour WBS la traduction de SDP (Structure de découpage du projet),

Le niveau le plus bas de l'arborescence est appelé « lot de travail ». Chaque lot de travail pourra ensuite être
décomposé en activités pour une planification détaillée.
L'OBS, *Organisation Breakdown Structure* (structure de décomposition de l'organisation), reprend le WBS et fait apparaître les noms des personnes responsables de la production des différents éléments.

Dans notre exemple (figure 2.5), le chef du projet global a la responsabilité directe de la conception générale et de l'intégration. Les trois sous-projets sont conduits par d'autres personnes, les différents travaux sont affectés aux ressources du projet.

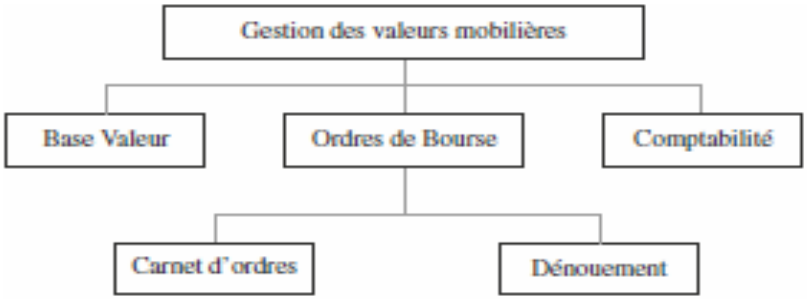

Figure 2.3 — Découpage PBS simplifié.

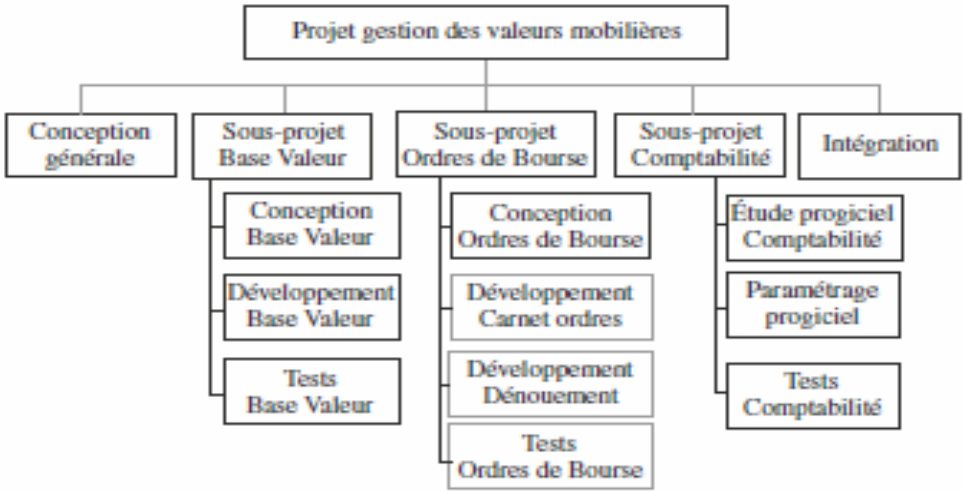

Figure 2.4 — Découpage WBS simplifié.

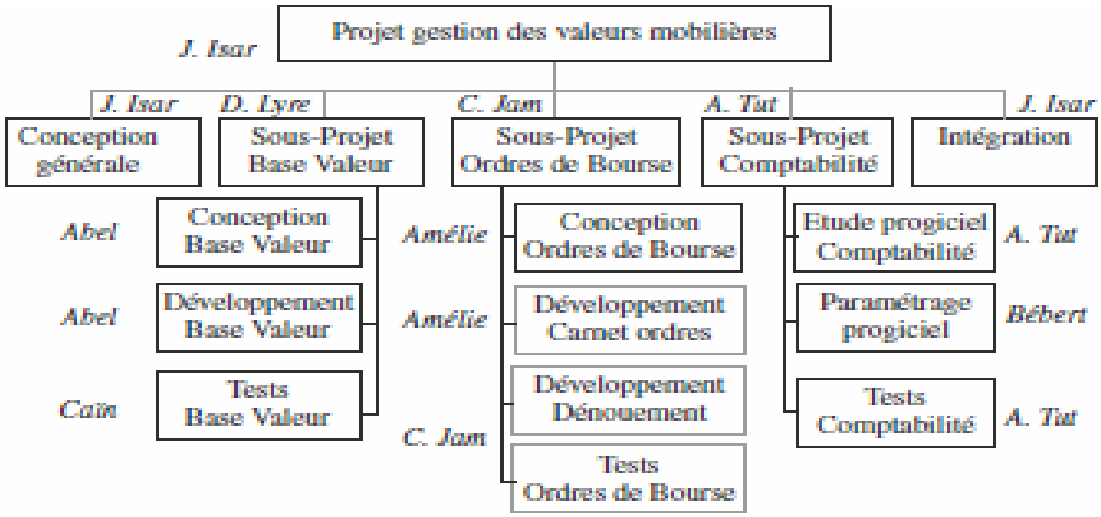

Figure 2.5 — Découpage OBS simplifié.

2.3 LE DÉCOUPAGE STRUCTUREL

Le découpage structurel repose sur la possibilité de découper le domaine en sous-ensembles quasi indépendants. Il peut être fait à plusieurs niveaux.
Un premier niveau est celui du découpage d'un système d'information en domaines. Un domaine peut être défini comme un sous-ensemble du système d'information global, qui possède en propre des informations — c'est-à-dire qu'elles sont créées et mise à jour dans le domaine — et des processus. Cela implique que souvent un domaine est transversal à plusieurs entités de l'organisation (site, département, service…).
Un second niveau de découpage est celui qui identifie des modules, comme dans l'exemple donné à la figure 2.3, illustrant le PBS. Chaque module peut être à son tour découpé en sous-modules. Il y a en général deux façons d'approcher ce découpage en modules : par la vision statique ou par la vision dynamique.
Le découpage par la vision statique s'appuie sur une macro-modélisation des entités, c'est-à-dire le repérage des principaux concepts d'information à l'aide d'un modèle Entité-Relation ou un diagramme de classes UML. À chaque entité principale correspond un module. Dans l'exemple du domaine Gestion des valeurs mobilières, le découpage en trois premiers modules découle de la vision statique (figure 2.6).
En effet, une modélisation des entités ferait apparaître trois principales classes :
Valeur mobilière, Ordre de bourse et Écriture comptable.

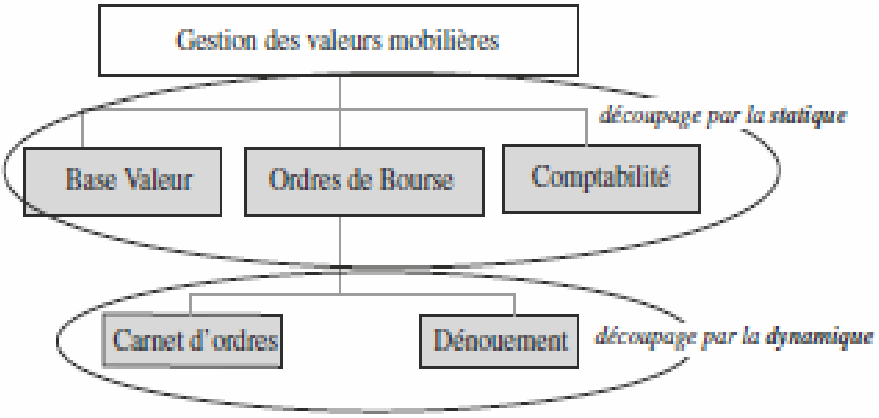

Figure 2.6 — Découpage structurel.

Le découpage par la vision dynamique s'appuie sur une identification des principaux processus du domaine. À chaque processus correspond un module.
Dans l'exemple du domaine Gestion des valeurs mobilières, le découpage du module Ordres de bourse en deux sous-modules découle de la vision dynamique (figure 2.6).
En effet, on a distingué deux processus : l'entrée d'un ordre dans le carnet et sa transmission à une société de bourse, en réponse à une demande d'un client ; le suivi du traitement de l'ordre, déclenché par la réception de l'avis d'opéré par la société de bourse.

2.4 LE CYCLE DE VIE STANDARD

2.4.1 Le contenu du cycle de vie standard

Le découpage temporel des projets industriels relevant d'une production unitaire est la référence initiale : c'est celui que l'on trouve souvent dans les ouvrages traitant de gestion de projet.
Ce cycle de vie standard se compose des phases suivantes :
• étude de faisabilité ;
• définition des solutions ;
• conception détaillée ;
• réalisation.
L'*étude de faisabilité* comprend des travaux d'analyse, des travaux de recherche, des études sur le terrain. Il s'agit de vérifier si le projet est techniquement réalisable. Par exemple, si l'on veut construire un immeuble, il faut vérifier que le terrain et le sous-sol le permettent, à un coût acceptable.
La *définition des solutions* donne une représentation précise de l'objectif à atteindre. Les solutions possibles sont étudiées de façon détaillée. Pour cela, on utilise différents moyens, par exemple faire des essais, élaborer une maquette, construire un prototype. Au terme de cette phase, une solution est alors choisie et l'on dispose de spécifications exactes.
La *conception détaillée* sert à préparer les contrats de réalisation. En effet, un grand projet industriel fait souvent intervenir des corps de métier différents.
Ces contrats contiennent les cahiers des charges pour les sous-traitants. Les spécifications techniques décrivent la mission et les moyens pour la réaliser.
Aucune ambiguïté ne doit subsister. Même si l'on ne recourt pas à des fournisseurs extérieurs, le principe du cahier des charges exhaustif est fondamental.
Les fournitures livrées à l'étape suivante sont acceptées sur la base de son contenu.
La *réalisation* est l'exécution des contrats par les sous-traitants, conformément aux cahiers des charges. Cette phase se termine en général par une procédure d'acceptation officielle.

Dans ce découpage de référence, la réalisation du projet passe par une définition complète, précise et détaillée de l'objectif. Les trois premières phases représentent généralement 10 % des efforts et des dépenses. Le management de projet (planification, organisation, suivi) porte essentiellement sur la phase de réalisation.

2.4.2 Les problèmes posés par le découpage standard

Le découpage temporel standard ne peut guère être utilisé tel quel dans un projet système d'information. D'abord la notion de cahier des charges y est déclinée à plusieurs moments.
En effet, la plupart des phases du cycle de développement peuvent conduire à un cahier des charges qui oriente le travail de l'étape ultérieure. On parle alors d'un « cahier des charges d'analyse » ou d'un « cahier des charges de conception » ou encore d'un « cahier des charges de réalisation ».
Ensuite, dans le découpage temporel standard, on postule la possibilité pour le client d'établir une description complète de ce qu'il attend. Or, dans notre domaine, l'élaboration des spécifications, c'est-à-dire la détermination des besoins et des solutions adéquates, est un problème majeur. Il y a souvent une construction progressive, qui s'appuie sur des allers-retours entre une solution de gestion et des possibilités techniques. Les besoins ne préexistent pas, ils émergent. Pour le chef de projet, les étapes d'analyse et conception sont risquées. Une part non négligeable du budget y est consacrée — jusqu'à 40 % — sans toujours atteindre la qualité espérée. Par conséquent, la gestion de projet commence dès le début du projet.
De plus, l'élaboration d'un cahier des charges de réalisation est un travail coûteux. En effet, l'écriture de spécifications souffre de l'absence de *composants réutilisables*. Par exemple, la définition d'un système de gestion des clients serait allégée si l'on pouvait réutiliser des fonctions standard :
(création d'un prospect, mise à jour d'une personne…) sans les reconcevoir et les décrire intégralement. Ce n'est malheureusement pas le cas. Dans le domaine industriel, le recours à la CAO (conception assistée par ordinateur), pour concevoir un nouveau produit, favorise l'utilisation de composants existants, élémentaires ou agrégés. On peut les modifier ou les assembler différemment, en obtenant automatiquement les nouvelles cotes. Les ateliers de génie logiciel, qui aident à concevoir un système d'information, offrent un méta modèle et non des modèles concrets, que l'on pourrait stocker dans une bibliothèque de composants conceptuels.
Par ailleurs, on a peu de *spécifications implicites*. Par exemple, avant de dessiner les plans d'un immeuble, on définit sa destination. Il en découle un certain nombre de spécifications standard. Différents critères (standing, segment de marché visé…) donnent des normes orientant la conception (matériaux, taille des pièces, garage…). Si l'on transposait à ce type de projet la démarche système d'information, l'immeuble serait conçu avec la plupart des futurs occupants !
Pour élaborer des spécifications, on est donc conduit à chercher une approche permettant de produire un résultat de qualité, en maîtrisant les coûts et les délais.
Dans les projets de système d'information, le problème de faisabilité se pose aussi de façon particulière. Dans un sens, on peut toujours faire quelque chose, dans la mesure où les contraintes ne sont en général pas d'ordres physiques, mais techniques, financières ou organisationnelles compte tenu des infrastructures de télécommunication à l'époque.
Par exemple, dans la mise en œuvre en 1991 du projet Relit (Règlement-livraison de titres), permettant de relier les acteurs bancaires de la place de Paris, les problèmes de temps de réponse provenant de la capacité des machines et du réseau, ont conduit à fonctionner pendant une période intermédiaire avec un nombre limité de banques et sociétés de bourse.
En conclusion, on peut dire que dans le domaine qui nous intéresse, le management de projet couvre l'ensemble du cycle et s'appuie sur des découpages temporels pouvant s'éloigner du découpage standard. Nous allons d'abord décrire le découpage classiquement utilisé, que l'on retrouve dans beaucoup de méthodes de conception. Puis nous présenterons un panorama des découpages temporels génériques utilisés dans notre domaine, souvent appelés modèles de développement. Nous terminerons sur certains découpages spécifiques.

2.5 LE DÉCOUPAGE CLASSIQUE

Durant les deux dernières décennies, les méthodes de développement de système d'information ont proposé un découpage temporel de référence, parfois appelé « cycle de vie classique ».
La figure 2.7 présente ce découpage avec les correspondances entre le vocabulaire de la méthode Merise [Nanci, 2001], celui de la méthode SDMS, et celui issu de la méthode MCP qui a servi de référence à la norme AFNOR Z67-101.

NORME AFNOR Z67-101	MERISE	SDMS
	Schéma directeur	
Étude préalable	Étude préalable	
Exploration	Observation	DBS (Définition des besoins du système)
Conception	Conception/Organisation	CAS (Conception de l'architecture du système)
Appréciation	Appréciation	
Conception détaillée	Étude détaillée	SES (Spécifications externes du système)
Réalisation	Étude technique	SIS (Spécifications internes du système)
	Réalisation	Programmation
		Test
Mise en œuvre	Mise en œuvre	Conversion
		Installation
Évaluation	Qualification	Bilan

Figure 2.7 — Découpage temporel classique des méthodes de développement S.I.

2.5.1 Le schéma directeur

SD	EP	ED	ET	REAL	MEO	QUALIF

L'*objectif* d'un schéma directeur est de définir le scénario d'évolution du patrimoine informatique, sous l'un ou l'autre de ces trois angles :

• évolution de l'architecture technique (matériels, réseaux) ;
• évolution de l'architecture applicative (données communes, identification des domaines, évaluation des applications) ;
• évolution de la fonction informatique (méthodes, normes, outils).

Le champ d'un schéma directeur est l'entreprise tout entière ou bien un grand secteur de l'entreprise ; on l'appelle alors schéma directeur sectoriel. Ce type de projet est mené par une petite équipe (2 à 3 personnes), en un temps limité (quelques semaines). Le directeur informatique est directement impliqué, mais aussi les responsables des autres directions.

Le *résultat* d'un schéma directeur dépend de l'objectif majeur. En général on a une photographie de la situation existante, un diagnostic et des orientations d'évolution choisies à partir de deux ou trois scénarios.
Le travail sur l'architecture applicative débouche sur une cartographie des domaines et une modélisation des principaux concepts. Cela conduit à définir des objectifs et des priorités par domaine et par application.

2.5.2 L'étude préalable

SD	EP	ED	ET	REAL	MEO	QUALIF

C'est souvent à partir de ce point que l'on fait partir le cycle de vie d'un système d'information spécifique à un domaine. On aborde une étude préalable soit à l'issue d'un schéma directeur, soit hors de toute opération schéma directeur, pour repenser une application vieillissante sur un domaine bien identifié ou pour répondre
à un besoin nouveau.
Par exemple, le domaine du crédit a connu ces dernières années de profondes évolutions réglementaires et commerciales. L'application gestion des prêts est à la limite de ses possibilités d'évolution et doit être refaite. Dans le domaine commercial, on peut mener une étude préalable pour repenser le système d'information dans une optique de *e-commerce*.
choix structurants pour la future application : évaluer l'adéquation de la solution aux objectifs, choisir éventuellement entre plusieurs solutions, évaluer l'investissement (budget, temps), ajuster la solution à l'enveloppe si cela est nécessaire.
D'autre part, il s'agit de fournir une base de référence pour la suite du projet : le rapport d'étude préalable peut donc être considéré comme un cahier des charges pour l'étude détaillée.
Une étude préalable comporte trois étapes : observation, conception-organisation et appréciation.
L'*objectif* de l'étape observation est de donner une représentation pertinente du domaine étudié, suffisante pour porter un diagnostic et mettre en évidence des besoins. Le *résultat* comprend :
• une structuration du domaine en processus, qui va ensuite guider un éventuel découpage structurel, pour établir un WBS par exemple ;
• le choix d'un sous-ensemble représentatif (SER) : en effet, si le domaine est important, on va se limiter à une partie du domaine, en utilisant la notion de variante de procédure ;
• une description du fonctionnement du SER ;
• une description modélisée des données ;
• un diagnostic.
L'*objectif* de l'étape conception-organisation est de proposer une ou plusieurs solutions, aux niveaux conceptuel et organisationnel, sur tout ou partie du domaine. Le *résultat* comprend un modèle de données consolidé ou enrichi, ainsi qu'une description d'au moins une variante de chaque processus, avec les traitements
et les règles de gestion.
L'*objectif* de l'étape appréciation est d'une part de dresser un bilan des avantages attendus et des coûts prévisibles (étude de rentabilité), d'autre part d'élaborer un plan pour la poursuite du projet. On peut ainsi identifier différents sous projets qu'il faut ordonnancer. Le découpage en sous-projets repose sur un découpage structurel ; par exemple, on peut définir un sous-projet par processus.
L'ordonnancement se fait selon :
• une éventuelle priorité stratégique de certains processus ;
• la périodicité (traitements quotidiens, puis mensuels, puis annuels) ;
• les contraintes logistiques (arrivée d'un matériel, mise en place d'un réseau).

2.5.3 L'étude détaillée

SD	EP	ED	ET	REAL	MEO	QUALIF

L'*objectif* d'une étude détaillée est de concevoir et décrire de façon exhaustive la solution sur tout le champ de l'étude. Elle sera ensuite complétée par l'étude technique. Les spécifications ainsi obtenues doivent faire l'objet d'un consensus entre futurs utilisateurs et informaticiens.
Elles représentent le cahier des charges pour la réalisation.
Le *résultat* comprend toute la vision externe du système : l'interface homme-machine (maquettes d'écran, cinématique), la description des traitements à une maille suffisamment fine pour qu'il n'y ait plus d'ambiguïté fonctionnelle, ainsi que les sorties (maquettes d'état). On y ajoute souvent l'organisation à mettre en place et le planning détaillé.

2.5.4 L'étude technique

SD	EP	ED	**ET**	REAL	MEO	QUALIF

L'*objectif* de cette phase, qui ne concerne que les informaticiens, est d'optimiser les structures physiques de données et de construire les traitements (dossier programmes) en essayant de préparer la réutilisation du code.
Le *résultat* comprend des normes techniques, des dossiers de programme et la structure physique des données.
Il complète le cahier des charges de réalisation.

2.5.5 La réalisation

SD	EP	ED	ET	**REAL**	MEO	QUALIF

Cette phase est parfois appelée « développement ».
L'*objectif* est de produire un logiciel testé. Elle comprend donc des tâches d'élaboration de jeu d'essai, de programmation et de test.
Elle se termine par une procédure d'acceptation officielle est appelée *recette* :
Le client fournit un jeu d'essai et vérifie avec le fournisseur la conformité du logiciel à ce qu'il avait demandé. Dans la pratique, la recette fait souvent une étape séparée. On effectue parfois une recette provisoire après la réalisation et une recette définitive après la mise en œuvre. Dans une relation contractuelle donnant lieu à des flux financiers, la recette conditionne le paiement.

2.5.6 La mise en œuvre

SD	EP	ED	ET	REAL	**MEO**	QUALIF

L'*objectif* est de préparer le démarrage effectif de la nouvelle application. Cette phase comprend notamment le paramétrage, la reprise ou l'alimentation des données, le développement d'interfaces, la formation des utilisateurs, l'installation d'environnement d'exploitation.

2.5.7 La qualification

SD	EP	ED	ET	REAL	MEO	**QUALIF**

L'*objectif* est de réaliser des tests dans l'environnement opérationnel et de tirer un bilan du système d'information installé, selon différents critères qualité.

2.6 LES MODÈLES DE DÉVELOPPEMENT

On considère aujourd'hui qu'on ne peut plus avoir une démarche unique, mais qu'il faut construire le découpage temporel en fonction des caractéristiques de l'entreprise et du projet. On s'appuie pour cela sur des découpages temporels génériques, appelés *modèles de développement* (*process models*) ou modèles de cycle de vie.
Les principaux modèles sont :

- **le modèle du *code-and-fix* ;**
- **le modèle de la transformation automatique ;**
- **le modèle de la cascade ;**
- **le modèle en V ;**
- **le modèle en W ;**
- **le modèle de développement évolutif ;**
- **le modèle de la spirale.**

2.6.1 Le modèle du code-and-fix

Ce modèle repose sur la possibilité d'une détermination facile des besoins (figure 2.8). Après une étape brève de compréhension de l'objectif, l'application est développée. Ensuite, plusieurs cycles de mise au point, parfois en collaboration avec l'utilisateur du futur système, permettent d'atteindre le résultat visé.

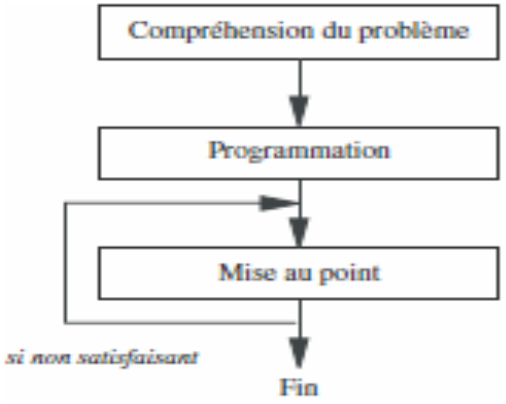

Figure 2.8 — Le modèle du *code-and-fix*.

2.6.2 Le modèle de la transformation automatique

Le modèle de la transformation automatique (*transform model*) est basé sur la possibilité de transformer automatiquement des spécifications en programmes (figure 2.9).
L'essentiel de l'effort va donc porter sur une description exhaustive des spécifications. Celles-ci devront être complètement validées. Une succession de cycles de spécification/validation s'achève par la génération de code.

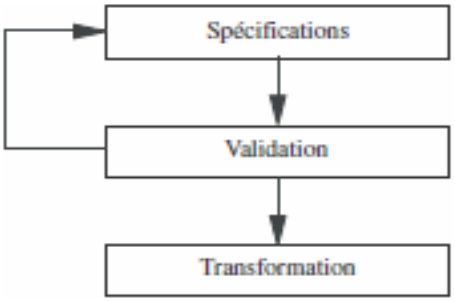

Figure 2.9 — Le modèle de la transformation automatique.

2.6.3 Le modèle de la cascade

Le modèle de la cascade (*waterfall model*) est totalement opposé au modèle du *code-and-fix*. Il a comme objectif majeur de jalonner rigoureusement le processus de développement et de définir de façon précise les rôles respectifs du fournisseur — qui produit un livrable — et du client — qui accepte ou refuse le résultat.
Le découpage temporel se présente donc comme une succession de phases correspondant à une approche descendante (figure 2.10).
Chacune donne lieu à une validation officielle : on ne passe à la suivante que si le résultat du contrôle est satisfaisant. Sinon, on modifie le livrable pour qu'il devienne acceptable. En revanche, il n'y a pas de retour possible sur les options validées à l'issue des phases antérieures.

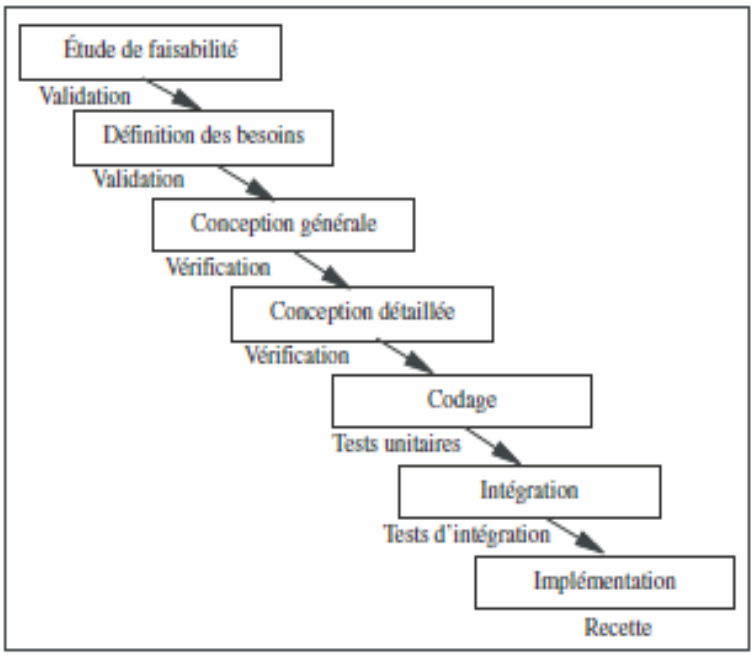

Figure 2.10 — Le modèle de la cascade.

2.6.4 Le modèle en V

Ce modèle (figure 2.11) est une amélioration du modèle de la cascade. Il vise, d'une certaine façon, à réduire ce que l'on a appelé l'« effet tunnel » : A partir d'un moment donné, les clients perdent la visibilité sur le projet. Quand ce dernier ressort du tunnel, on découvre des livrables qui ne sont pas toujours ceux que l'on attendait, non pas qu'ils ne soient pas conformes aux spécifications, mais parce les spécifications sont parfois impuissantes à décrire les attentes. La seule validation de documents est donc insuffisante.

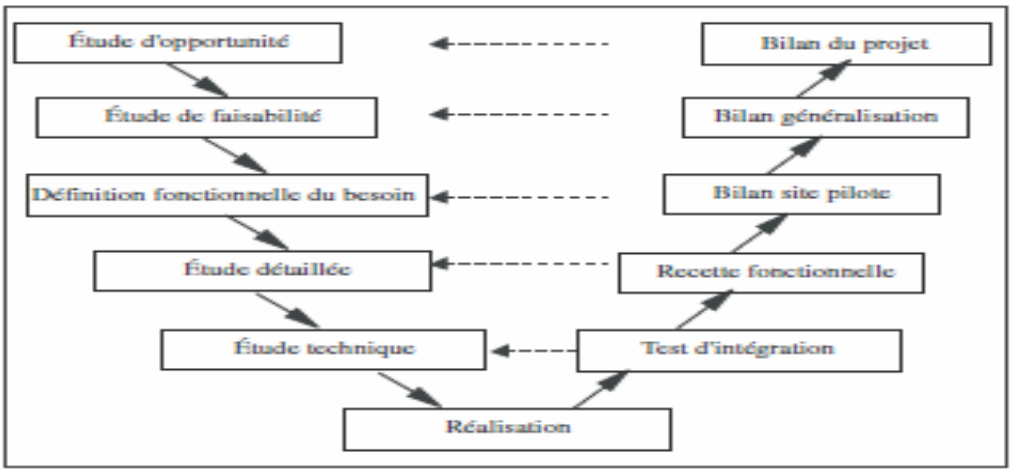

Figure 2.11 — Le modèle en V.

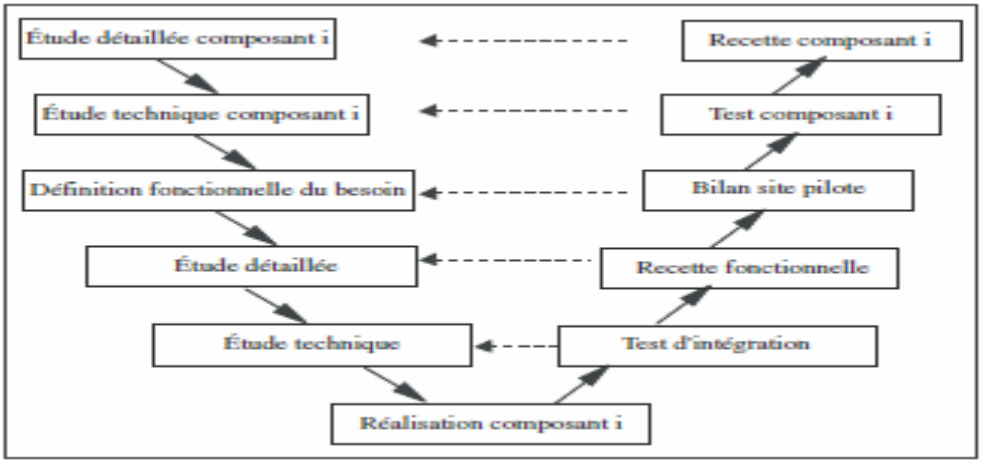

Figure 2.12 — Le modèle en V avec des composants.

2.6.5 Le modèle en W

Ce modèle enrichit le modèle en V (figure 2.13) dans le même esprit d'anticipation sur le livrable final. La première partie du W vise à dégager avec les clients des orientations solides pour la conception ou bien à explorer les possibilités d'une nouvelle technique.
Le développement de maquettes ou prototypes permet une validation plus concrète, voire une expérimentation.

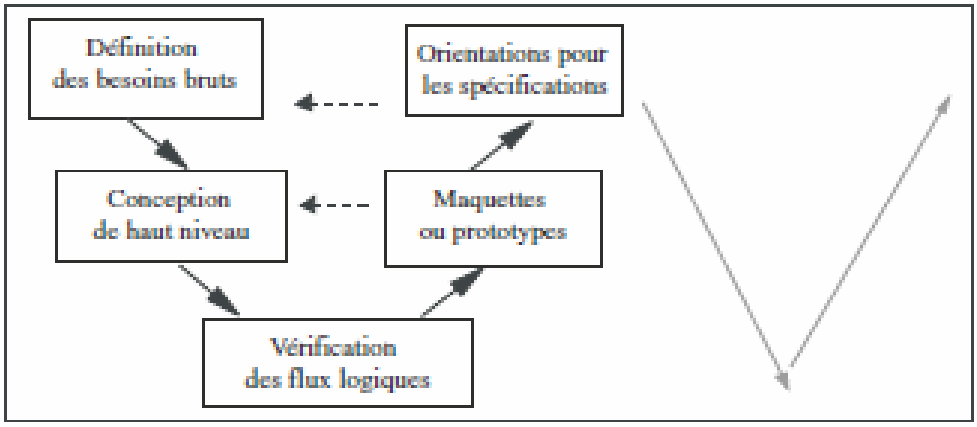

Figure 2.13 — Le modèle en W.

2.6.6 Le modèle de développement itératif

L'objectif du modèle du développement itératif, qui fut initialement appelé évolutif *(evolutionary design model),* est de construire progressivement le système de façon participative (figure 2.14).
Il repose sur l'idée que les besoins ne peuvent s'exprimer qu'après une expérimentation, même sur un système rudimentaire ou incomplet. Chaque cycle aboutit à une nouvelle version du système : on s'arrête lorsque le client juge le système satisfaisant.
Lorsque l'on veut indiquer que non seulement chaque itération fournira au commanditaire un ensemble de fonctionnalités qui forment une version exécutable du système cible, mais que chaque version apportera de nouvelles fonctionnalités, on qualifie parfois le modèle d'*incrémental*.
Cependant, tout cycle itératif n'est pas obligatoirement incrémental : la première itération peut livrer un système complet qui est ensuite affiné et ajusté dans les itérations suivantes.

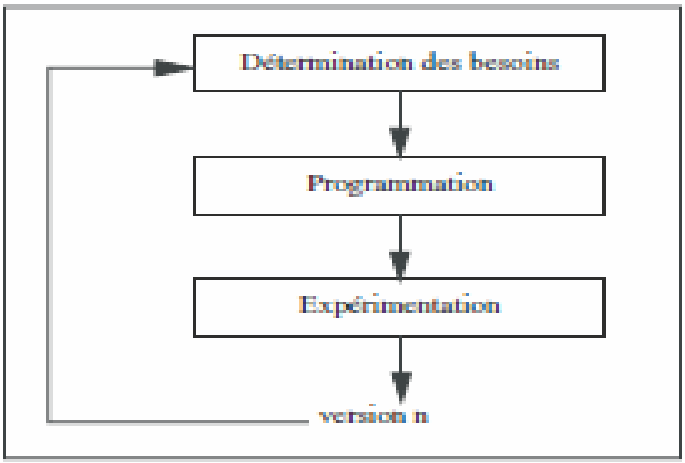

Figure 2.14 — Le modèle du développement itératif.

2.6.7 Le modèle de la spirale

Le modèle de la spirale (*spiral model*) repose sur le même principe que le modèle évolutif (figure 2.15), mais il s'inscrit dans une relation contractuelle entre le client et le fournisseur. De ce fait les engagements et validations présentent un caractère formalisé.
Chaque cycle donne lieu à une contractualisation préalable, s'appuyant sur les besoins exprimés lors du cycle précédent. Un cycle peut être considéré comme une phase, qui comporte les six étapes suivantes :
• analyse du risque ;

• développement d'un prototype ;
• simulation et essais du prototype ;
• détermination des besoins (à des mailles différentes selon le cycle), à partir des résultats des essais ;
• validation des besoins par un comité de pilotage ;
• planification du cycle suivant.
Le dernier cycle permet de développer la version finale et d'implémenter le logiciel.

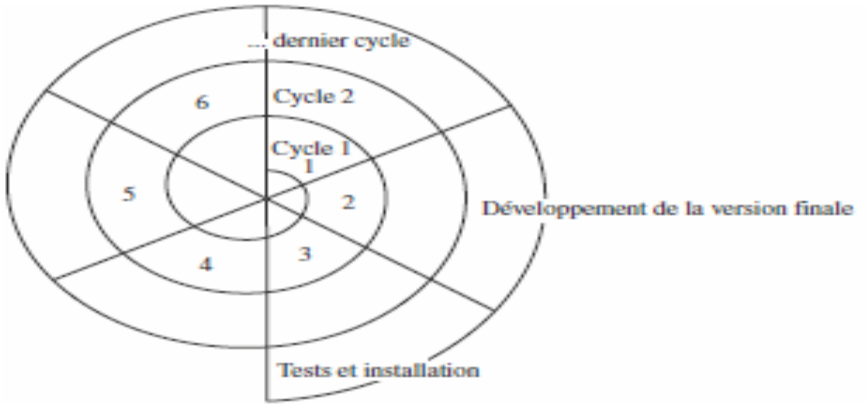

Figure 2.15 — Le modèle de la spirale.

2.7 LES MODÈLES DE CYCLE DE VIE SPÉCIFIQUES

Certains découpages temporels sont liés soit à une méthode, soit à un type de projet bien particulier. Nous en proposons deux exemples : le découpage préconisé pour mettre en place un progiciel intégré et le modèle RUP proposé par la société Rational Software.
Nous présentons dans le paragraphe suivant les modèles de cycle de vie que l'on trouve dans les méthodes agiles, introduites brièvement à la fin du chapitre 1.

2.7.1 Le cycle ERP

La mise en place d'un progiciel de gestion intégré, souvent appelé du terme anglo-saxon ERP (*Enterprise Resource Planning*)[1] s'appuie sur un découpage spécifique (figure 2.16).
En effet, il s'agit de construire, en tirant le meilleur parti du progiciel, un système améliorant la performance de l'entreprise. Deux étapes doivent donc être menées en parallèle : Description des processus et Formation au progiciel.
Ensuite, il y a autant de cycles d'analyse — paramétrage — prototypage qu'il y a de processus. La validation par le Comité de pilotage permet une simulation en grandeur réelle. Il faut alors prendre en compte ce qui est resté en dehors du champ couvert par le progiciel.

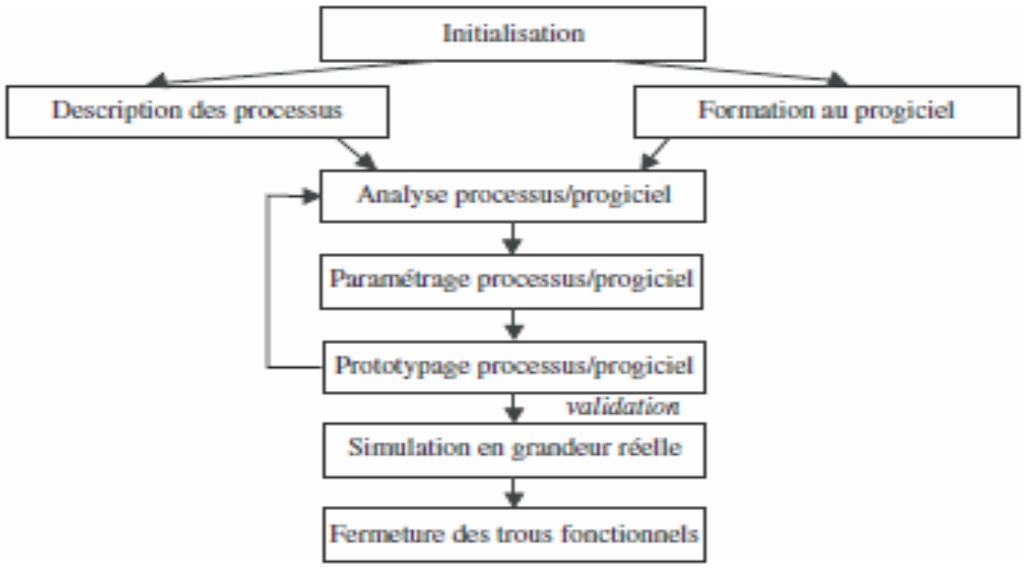

Figure 2.16 — Le cycle ERP.

2.7.2 Le modèle RUP

Le modèle RUP (*Rational Unified Process*) est représentatif d'une approche combinant plusieurs modèles. Sa structure fait l'objet d'un assez large accord, notamment parmi les praticiens (figure 2.17).
Il peut être lu de la façon suivante :

1. Le cycle est constitué de quatre phases principales, que l'on retrouve globalement dans toutes les approches descendantes : étude préalable (opportunité), conception de la solution détaillée (élaboration), développement de
la solution (construction) et mise en œuvre (transition),

2. Il existe six types de tâches qui, au lieu d'être affectées exclusivement à une phase, se retrouvent à des degrés divers dans chacune des phases. Par exemple, l'étude des besoins peut apparaître jusqu'à la fin du projet, mais la plus grande partie est effectuée dans les deux premières phases. L'implémentation (développement) a principalement lieu dans la phase de construction, mais on peut réaliser un prototype dès la première phase.
Certaines tâches, comme la direction de projet, s'effectuent sur toute la durée du projet,

3. Certaines phases peuvent être menées de façon cyclique. Ainsi, l'élaboration se fait en deux cycles, conduisant par exemple à la production de spécifications externes (vision utilisateur) et spécifications techniques (vision développeur). La construction est itérative et incrémentale.

De plus, l'ensemble du modèle représente un tour de spirale, dans le cas d'une approche globale en spirale.

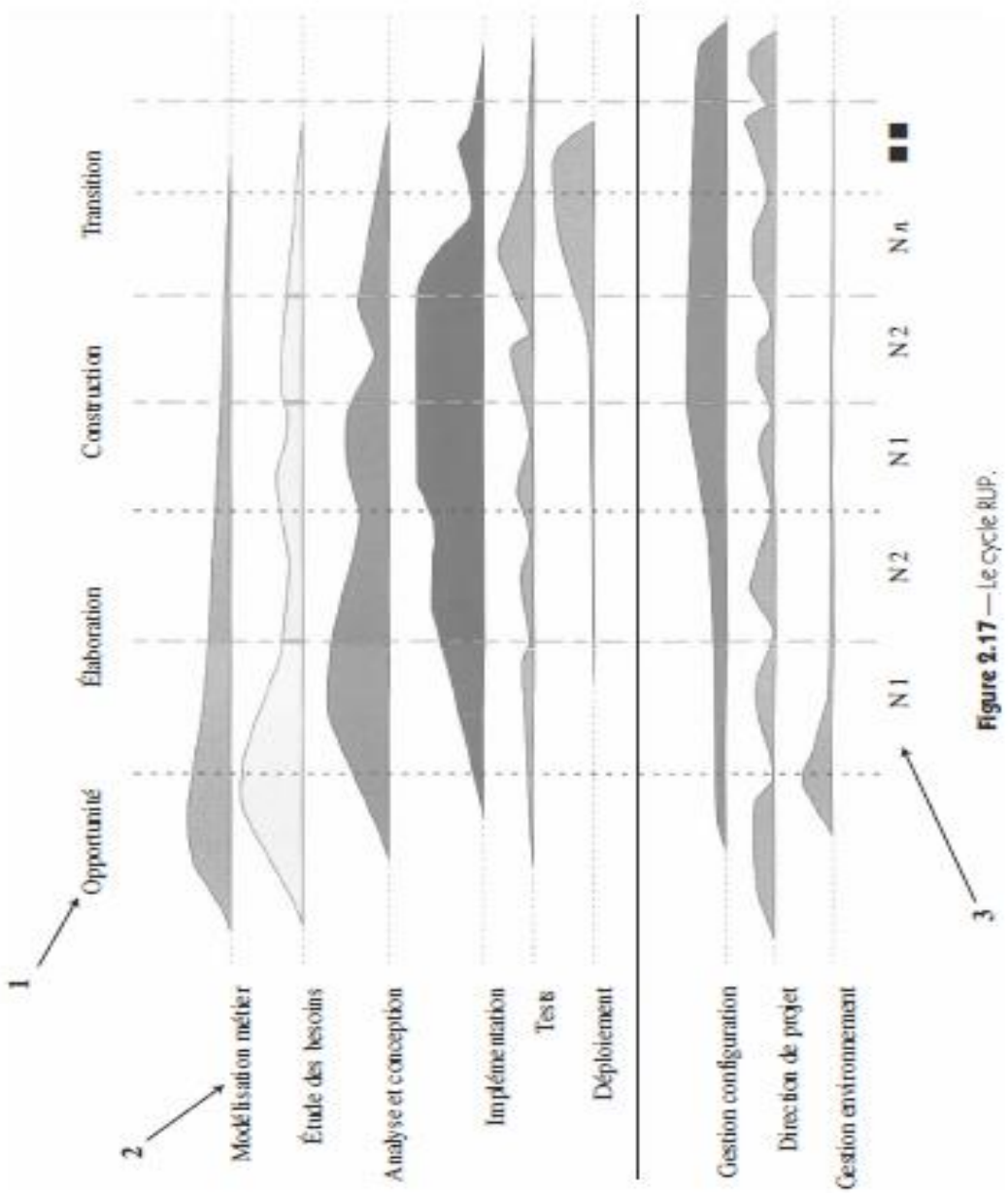

Figure 2.17 — Le cycle RUP.

2.8 LES MODÈLES DE CYCLE DE VIE DES MÉTHODES AGILES

Toutes les méthodes agiles prennent en compte dans leur modèle de cycle de vie trois exigences : une forte participation entre développeurs et utilisateurs, des livraisons fréquentes de logiciel et une prise en compte de possibles changements dans les besoins des utilisateurs au cours du projet.
C'est pourquoi toutes font appel, d'une façon ou d'une autre, à un modèle itératif et incrémental.
De plus, elles préconisent en général des durées de cycle de vie des projets ne dépassant pas un an.

2.8.1 Le modèle RAD

Le cycle de vie RAD conjugue modèle linéaire, structuré en cinq phases, et modèle itératif pour la phase Construction du logiciel en plusieurs modules successivement livrés1 (figure 2.18).

La participation des utilisateurs est placée au cœur du cycle. En effet, le déroulement d'une phase comprend une ou plusieurs sous-phases, et chaque sous-phase présente une structure à trois temps, dans laquelle la tenue d'une session participative joue un rôle central (figure 2.19). Des travaux préparatoires rassemblent et construisent le matériau (modèle ou prototype) qui sera ensuite discuté par les différents acteurs et ajusté.

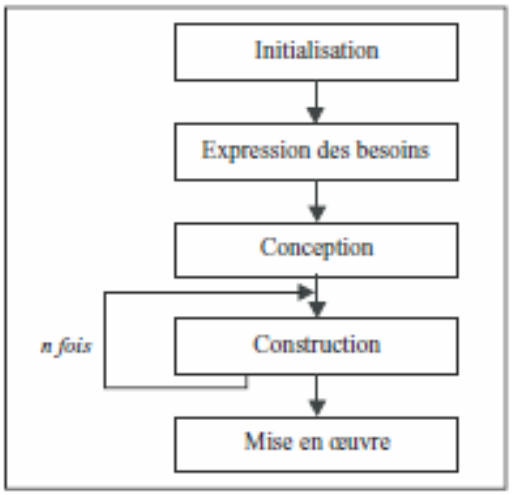

Figure 2.18 — Modèle de cycle de vie RAD.

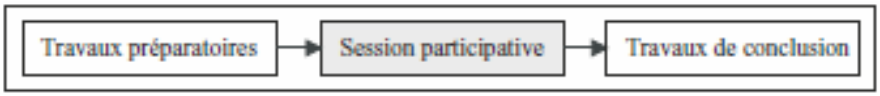

Figure 2.19 — Structure ternaire des sous-phases RAD.

2.8.2 Le modèle DSDM

La méthode DSDM a évolué au cours des années. L'actuelle version1 distingue le cycle de vie du système et le cycle de vie du projet. Le premier comprend, outre les phases du projet lui-même, une phase de pré-projet qui doit conduire au lancement du projet et une phase post-projet qui recouvre l'exploitation et la maintenance de l'application.

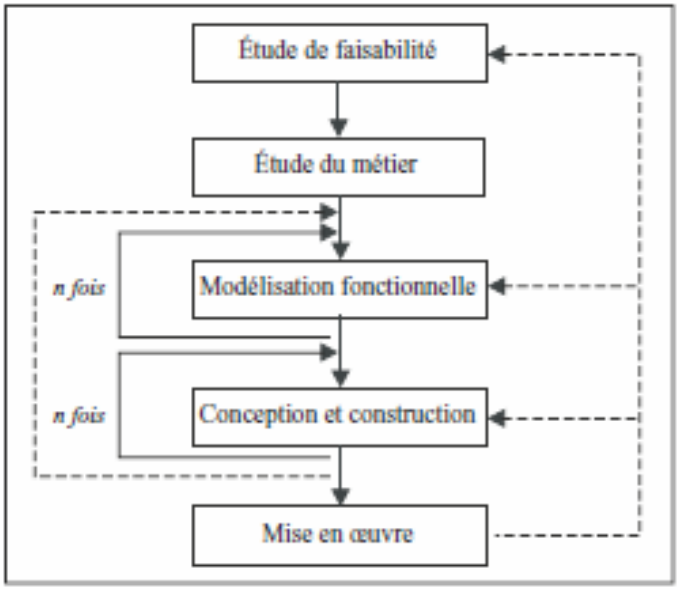

Figure 2.20 — Modèle du cycle de vie du projet DSDM.

Le cycle de vie du projet comprend cinq phases, dont deux sont cycliques (figure 2.20). Les flèches pleines indiquent un déroulement normal. Les flèches en pointillé montrent des retours possibles à une phase antérieure, soit après la phase Conception et construction, soit après celle de Mise en œuvre Après une Étude de faisabilité,

- la phase Étude du métier permet, à travers des ateliers (*workshops*) entre équipe de projet et manageurs, de définir le périmètre du projet, avec une liste d'exigences prioritaires et une architecture fonctionnelle et technique du futur système.

- La phase Modélisation fonctionnelle est une suite de cycles. Chacun permet de définir précisément les fonctionnalités souhaitées et leur priorité. L'acceptation par toutes les parties prenantes d'un prototype fonctionnel, sur tout ou partie du périmètre, permet de passer à la phase Conception et construction.

L'objectif de cette phase est de développer un logiciel testé, par des cycles successifs de développement/acceptation par les utilisateurs.

2.8.3 Le modèle XP

La méthode XP, focalisée sur la partie programmation du projet, propose un modèle itératif avec une structure à deux niveaux : d'abord des itérations de livraison (*release*), puis des itérations de développement (figure 2.21). Les premières conduisent à livrer des fonctionnalités complètes pour le client, les secondes portent sur des éléments plus fins appelés scénarios qui contribuent à la définition d'une fonctionnalité.
Après une phase initiale d'Exploration des besoins, un plan de livraison est défini avec le client. Chaque livraison, d'une durée de quelques mois, se termine par la fourniture d'une version opérationnelle du logiciel. Une itération de livraison est découpée en plusieurs itérations de développement de courte durée (deux semaines à un mois), chacune donnant lieu à la livraison d'une ou plusieurs fonctionnalités pouvant être testées, voire intégrées dans une version en cours.
De façon plus détaillée, chaque itération de développement (figure 2.22) commence par l'écriture de cas d'utilisation ou scénarios (*user stories*), c'est-à dire des fonctions simples, concrètement décrites, avec les exigences associées, qui participent à la définition d'une fonctionnalité plus globale. Utilisateurs et

développeurs déterminent ensemble ce qui doit être développé dans la prochaine itération. Une fonctionnalité est ainsi découpée en plusieurs tâches.
Les plans de test sont écrits, les développeurs sont répartis en binôme (comme nous le développons au chapitre 5), ils codent les tâches qui leur sont affectées, puis effectuent avec les utilisateurs des tests d'acceptation. En cas d'échec, on revoit les scénarios et on reprend la boucle. Sinon, on continue jusqu'à avoir développé tous les scénarios retenus. Une version livrable est alors arrêtée et mise à disposition, ainsi que la documentation.

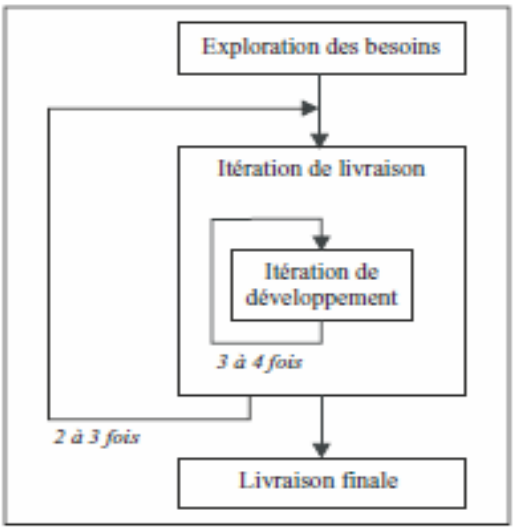

Figure 2.21 — Cycle de vie XP.

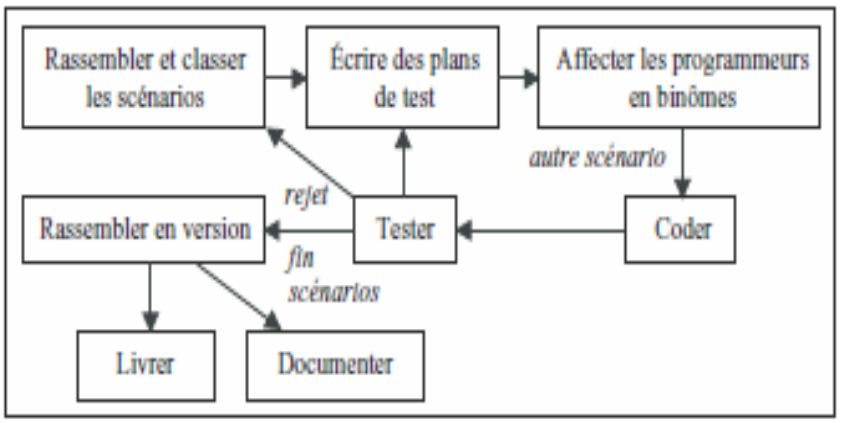

Figure 2.22 — Modèle XP d'une itération de développement.

2.8.4 Le modèle SCRUM

SCRUM emprunte au vocabulaire du jeu le qualificatif des trois phases du cycle préconisé (figure 2.23).

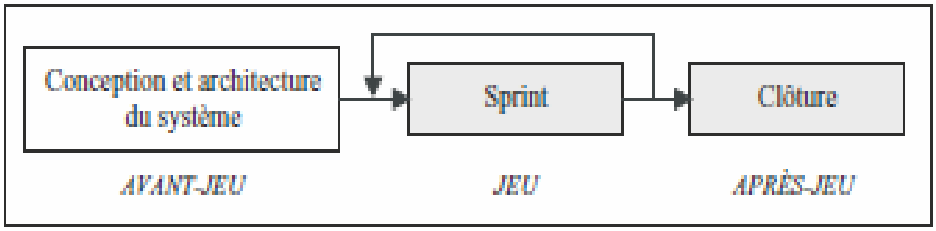

Figure 2.23 — Modèle SCRUM de cycle de vie du projet.

- La phase d'Avant-jeu (***pre-game***), Conception et architecture du système, se déroule de façon structurée, en général linéaire, et permet de déterminer le périmètre, la base du contenu du produit à développer et une analyse de haut niveau.

- La phase de Jeu (***game***) est itérative et qualifiée d'empirique, dans la mesure où le travail effectué ne fait pas l'objet d'une planification. Une itération, dont la durée oscille entre une et quatre semaines, est appelée un Sprint, en référence à ces poussées rapides et fortes que les joueurs de rugby peuvent effectuer sur le terrain.

  Un **Sprint** est découpé en trois sous-phases :
  - Développement (***develop***) : il s'agit de déterminer l'objectif visé au terme de l'itération, de le répartir en « paquets » de fonctions élémentaires, de développer et tester chaque paquet.
  - Emballage (***wrap***) : on referme les « paquets » et on les assemble pour faire une version exécutable.
  - Revue (***review***) : une revue élargie permet de faire le point sur les problèmes et l'avancement.
  - Ajustement (***adjust***) : ajuster le travail restant.

- La phase d'Après-Jeu (***postgame***), Clôture, vise à livrer un produit complet et documenté. Comme dans la première phase, on peut en planifier les tâches et les dérouler de façon linéaire.

2.9 LE DÉCOUPAGE STRUCTUREL DANS LE CAS DES MÉTHODES AGILES

Lorsque l'on utilise un modèle itératif et incrémental, notamment dans le cadre d'une méthode agile, le découpage en composants affectés à des itérations différentes est particulièrement important.
En effet, d'une part, chaque itération doit pouvoir fournir un logiciel utilisable, d'autre part il est souhaitable que toute livraison produise sur le client un effet démonstratif.
Ainsi, la première itération doit convaincre de la pertinence de la conception, même si le système est rudimentaire, et inciter à poursuivre.
Les itérations suivantes doivent montrer une progression effective vers un produit stabilisé.
Par exemple, le cycle de vie d'un projet peut être structuré en trois livraisons :

- la première permet de livrer les fonctionnalités de base,
- la deuxième l'ensemble des fonctionnalités souhaitées,
- la troisième d'améliorer les fonctionnalités.

La détermination de l'ordre de livraison des fonctionnalités est parfois une question délicate nécessitant de mettre en balance différents arguments.

Ainsi, le plus confortable pour les développeurs est de commencer par les référentiels et de s'attaquer ensuite au développement des processus utilisant ces structures de données.
Cependant, ce choix peut s'avérer peu mobilisateur pour les clients qui s'attendent à voir rapidement des fonctions qu'ils espèrent créatrices de valeur ou pouvant répondre à un besoin urgent.
De façon analogue, il est préférable en dehors de toute contrainte, de sélectionner d'abord les fonctionnalités les plus simples pour aborder ensuite les plus complexes, ou les cas standard avant les cas particuliers.
Mais ces choix, qui peuvent d'ailleurs être incompatibles, risquent de ne pas emporter l'adhésion des utilisateurs, peu convaincus par l'apport annoncé du nouveau système.
De plus, si les utilisateurs ne partagent pas les mêmes objectifs, il est souvent conseillé de commencer par ce qui est le plus consensuel pour bâtir des relations positives de travail en commun, avant de s'attaquer à des fonctions susceptibles de générer des positions conflictuelles.
On peut parfois utiliser la notion de « strate ».
Par exemple[1], dans un projet de développement d'un nouveau téléphone mobile chez Motorola, les strates correspondaient aux différents aspects du système (liaison radio, contrôle d'appel, liaison avec la base…) et chaque version du produit implémentait à des degrés divers des éléments de chacune des strates, ce qui a permis d'avoir des retours des commanditaires beaucoup plus tôt dans le déroulement du projet.

Dans le cadre plus général du contrôle des risques. On peut cependant souligner un mouvement de rapprochement depuis quelques années, visant à conjuguer un cycle classique global avec un cycle issu d'une méthode agile pour une partie du projet.
Ainsi, DSDM expose des rapprochements avec XP, certains auteurs font de même entre XP et RUP.

Pour conclure, il convient de poser la question : quel découpage choisir pour un projet donné ? En fait, le choix d'un modèle de développement s'appuie sur l'analyse des caractéristiques du projet, et notamment l'analyse des risques.

De plus, il ne suffit pas de sélectionner un modèle : il faut également l'instancier, c'est-à-dire en construire une version concrète adaptée au projet à mener, et le conjuguer avec le découpage structurel et l'organisation du projet. Cela constitue ce que l'on appelle l'établissement d'une stratégie.