GENIE DES SYSTEMES D'INFORMATION

Intégration d'une solution informatique

Automne 2006

CESI

Bertrand LIAUDET

SOMMAIRE

INTRODUCTION

1. Génie Logiciel *vs* Ingénierie des systèmes d'information

Génie logiciel - Software

Le terme de génie logiciel (*software engineering*) est né en Europe à la fin des années 60.

Le G.L. regroupe l'ensemble des

- Méthodes (organisation du travail)
- Techniques (langages de programmation, documentation des programmes)
- Outils (compilateurs, systèmes de gestion de la documentation)

de développement du logiciel.

Le G.L vise à transformer les besoins et attentes des utilisateurs en une application informatique.

Besoins et attentes ⟶ **Application informatique**

Quoi : software

Qui : les informaticiens.

Ingénierie des systèmes d'information - Brainware

Le terme d'ingénierie des systèmes d'information (*requirement engineering*) est né au début des années 90.

L' I.S.I vise à transformer les besoins et attentes des utilisateurs en spécifications formalisées d'une future application informatique.

Besoins et attentes ⟶ **Spécifications formalisées**

Quoi : brainware

Qui : les informaticiens, les gestionnaires et les autres utilisateurs du système d'information

De la même façon que pour le génie logiciel, on peut considérer que l' I.S.I. regroupe l'ensemble des méthodes, techniques et outils utilisés pour le développement des spécifications.

- Méthode d'organisation du travail de spécification : MERISE
- Techniques de modélisations : MCD, MLD, etc.
- Outils de modélisation et de spécifications : logiciel Win'Design, logiciel AMC designer…

<u>Le brainware</u>

Le concept de brainware, très peu usité, a été introduit par Tosio Kitagawa en septembre 1974 dans le n°39 des Research Report of Research Institute of Fundamental Information Science.

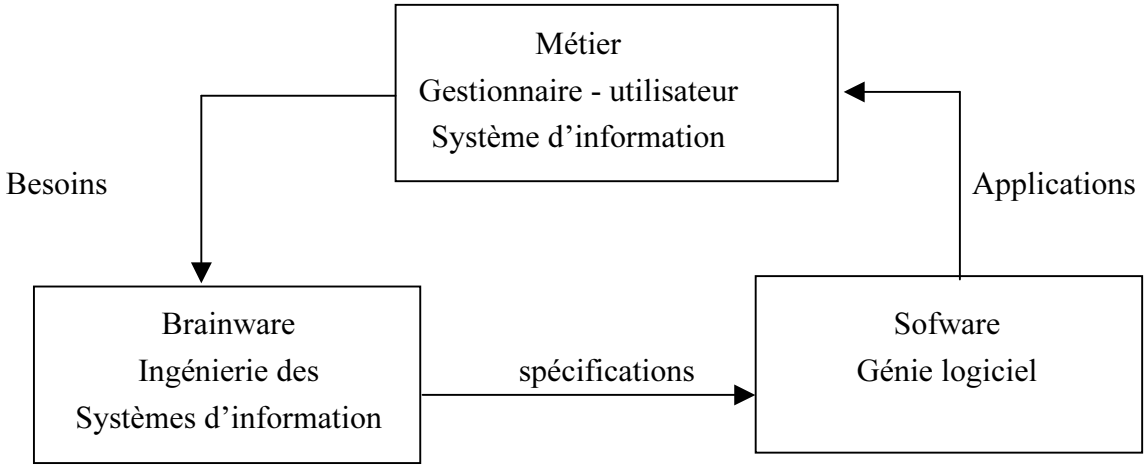
Le brainware est la fondation intellectuelle qui fonde le software.

Le brainware est un matériau (ware) en ce sens que c'est un stock objectif de connaissances et d'informations.

On peut donc distinguer entre :

software ingenering - brainware ingenering

Relations entre software engineering et brainware engineering



ISIM, p. 2

2. La méthode

Méthode vs Logique

Quelques citations

Chercher une méthode, c'est chercher un système d'opérations extériorisables qui fasse mieux que l'esprit le travail de l'esprit.

Variétés, Paul Valery (1753-1801)

Les méthodes sont les habitudes de l'esprit et les économies de la mémoire.

Rivarol (1753-1801)
surtout connu pour son Discours sur l'universalité de la langue française (1784)

Il n'y a qu'une méthode pour inventer, qui est d'imiter. Il n'y a qu'une méthode pour bien penser, qui est de continuer quelque pensée ancienne et éprouvée.

Propos sur l'éducation, 1932, Emile Auguste Chartier, dit Alain (1868-1951)
professeur, philosophe et essayiste français,
maître de Raymond Aron, Simone Weil, André Maurois…

Il vaut mieux avoir une méthode mauvaise plutôt que de n'en avoir aucune.

Le fil de l'épée, 1932, Charles de Gaulle (1890-1970)

Deux pôles de la méthode

- **Le chemin :** La méthode est le chemin par lequel on est arrivé à un certain résultat, lors même que ce chemin n'avait pas été fixé d'avance de façon voulue et réfléchie.

  La méthode, ce sont alors les procédés habituels d'un esprit, procédés qu'on peut observer et définir par induction, soit pour les pratiquer ensuite plus sûrement, soit pour les critiquer et en faire voir l'invalidité.

  Voyageur, il n'y a pas de chemin. Le chemin se fait en marchant.

  Antonio Machado (1875-1939)

- **Le programme :** La méthode est le programme réglant d'avance une suite d'opérations à accomplir et signalant certains errements à éviter, en vue d'atteindre un résultat déterminé. La méthode est alors une logique (une mécanique).

  Rappelez-vous tout simplement qu'entre les hommes il n'existe que deux relations : la logique ou la guerre. Demandez toujours des preuves, la preuve est la politesse élémentaire qu'on se doit. Si l'on refuse, souvenez-vous que vous êtes attaqué et qu'on va vous faire obéir par tous les moyens.

  La soirée avec Monsieur Teste, 1896, Paul Valery (1871-1945)

  Monsieur Teste (« la bétise n'est pas mon fort »), Léonard (Introduction à la méthode de Léonard de Vinci, 1894) et Eupalinos (Eupalinos ou l'Architecte, 1923) représentent dans les œuvres de Valéry la figure de l'esprit se réfléchissant et construisant sa méthode.

La réflexion sur le chemin conduit au programme. Cette réflexion est une compréhension supérieure du chemin parcouru.

Cependant, cette réflexion tend en final à rendre possible le parcours du chemin « les yeux fermés », c'est-à-dire sans signification, sans même comprendre ce que l'on fait. C'est le rôle de la méthode-programme.

La méthode-chemin est plus intuitive, plus implicite. Elle demande du bon sens et de l'expérience. C'est un savoir-faire.

La méthode-programme est explicite. Elle peut s'appliquer mécaniquement, sans vraiment comprendre et sans expérience.

Le concepteur de méthode, tout comme l'utilisateur de méthode, ne doit jamais perdre de vue les deux pôles de la méthode. La méthode est un programme général qui ne doit jamais faire perdre de vue que l'application concrète de ce programme est toujours un chemin particulier pendant lequel intuition, bon sens, expérience et savoir-faire doivent être mis en œuvre.

Méthode	**Logique**
Chemin	Programme
Sémantique	Syntaxe
Poursuivre un objectif	Appliquer les règles
Singulière	Universelle

- ***Critique de la raison pure, 1781, Emmanuel Kant***

> *La logique peut être abordée de deux points de vue : soit comme logique de l'usage général de l'entendement, soit comme logique de son usage particulier. La première contient les règles nécessaires de la pensée. La seconde contient les règles permettant de penser correctement une certaine sorte d'objets. On peut nommer la première « logique formelle » tandis que la seconde peut s'appeler « l'organon[1] » de telle ou telle science. C'est cette dernière qu'on place comme enseignement préparatoire d'une science, alors que, selon le parcours de la raison humaine, elle est l'ultime étape, à laquelle la raison parvient uniquement quand la science est déjà terminée depuis longtemps et n'a plus besoin que de la dernière main pour être mise en ordre et atteindre à la perfection. Car il faut déjà que l'on connaisse les objets à un assez haut degré, si l'on veut indiquer les règles de mise en œuvre d'une science.*
>
> d'après la Critique de la raison pure, Emmanuel Kant (1753-1801)

- ***La logique ou l'art de penser, Logique de Port Royal, 1662 (Arnaud et Nicole)***

Ouvrage classique qui accompagna la formation intellectuelle en France jusqu'au début du 20ème siècle.

> *La logique est l'art de bien conduire sa raison dans la connaissance des choses, tant pour s'instruire soi-même que pour en instruire les autres.*

Le livre se divise en quatre parties :

- Concept (concevoir)
- Jugement (juger)
- Raisonnement
- Méthode.

> *On peut appeler généralement méthode l'art de bien disposer une suite de plusieurs pensées, ou pour découvrir la vérité quand nous l'ignorons, ou pour la prouver aux autres quand nous l'ignorons, ou pour la prouver aux autres quand nous la connaissons déjà.*

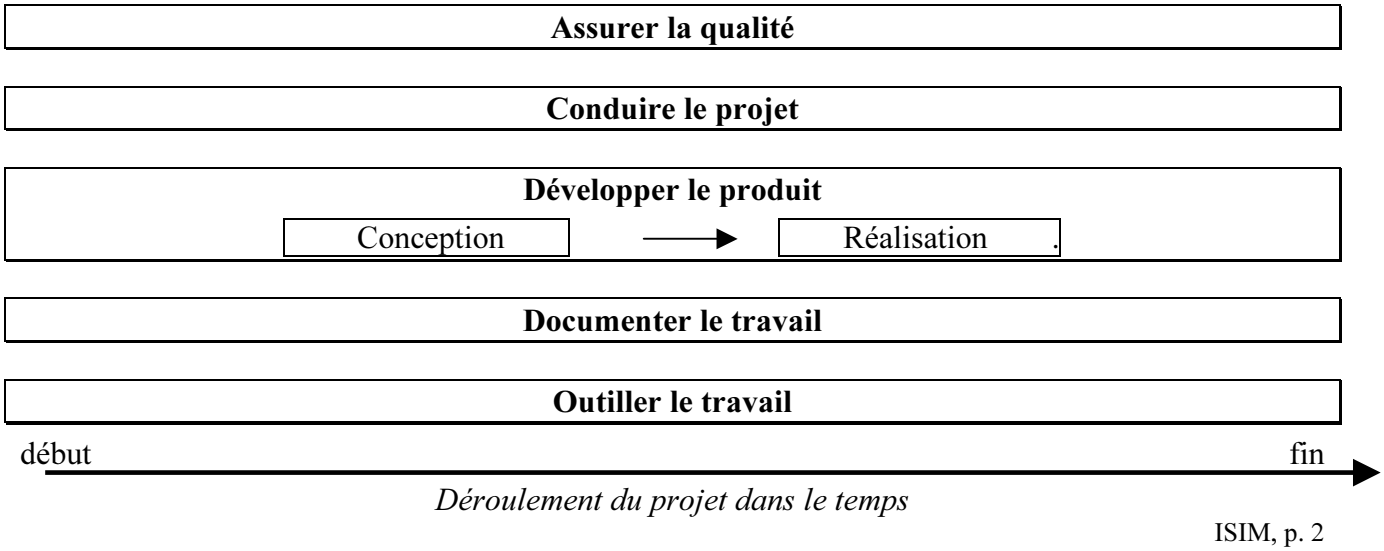
Méthode générale de l'ingénierie informatique

Méthodes de gestion de projet

La méthode va consister à organiser dans le temps les différentes activités de l'ingénierie informatique.

[1] Un « organon » est un outil. Les logiques des usages particuliers de l'entendement sont des outils logiques adaptés à telle ou telle usage, c'est-à-dire tel ou tel objet.

Très généralement, on peut présenter les choses ainsi :



Déroulement du projet dans le temps

ISIM, p. 2

Méthodes de développement

Une partie de la méthode concerne le développement du produit.

On peut alors choisir une méthode de développement parmi d'autre.

Comme méthode, citons :

- ✓ Le cycle en V
- ✓ La méthode par prototype (méthode adaptée au développement rapide)
- ✓ La méthode itérative.

Toutes ces méthodes peuvent être mixées. Nous y reviendrons.

Le cycle en V

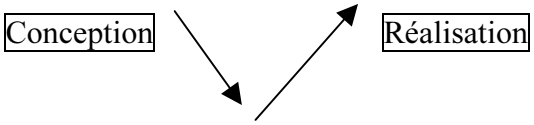
C'est la méthode de développement la plus répandue.

Le développement se divise naturellement en deux étapes :

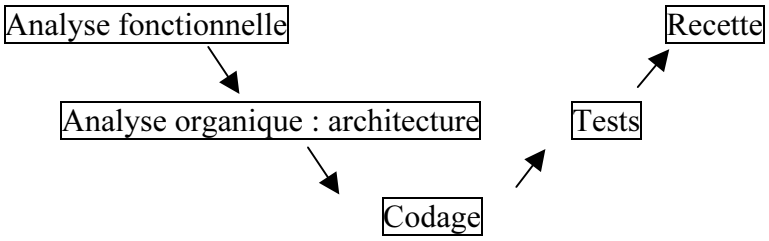
- ✓ 1 : la conception : c'est l'étape qui consiste à se représenter clairement l'objectif à atteindre.
- ✓ 2 : la réalisation : c'est l'étape qui consiste à passer de la représentation de l'objectif à atteindre à sa concrétisation (sa réalisation).

Ces deux étapes se déroulent successivement : d'abord on conçoit, ensuite on réalise.

Ces deux étapes forment les deux branches du cycle en V :

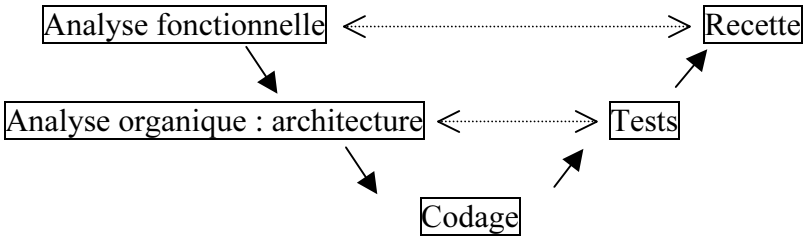


Ces deux étapes sont détaillées dans le cycle en V :



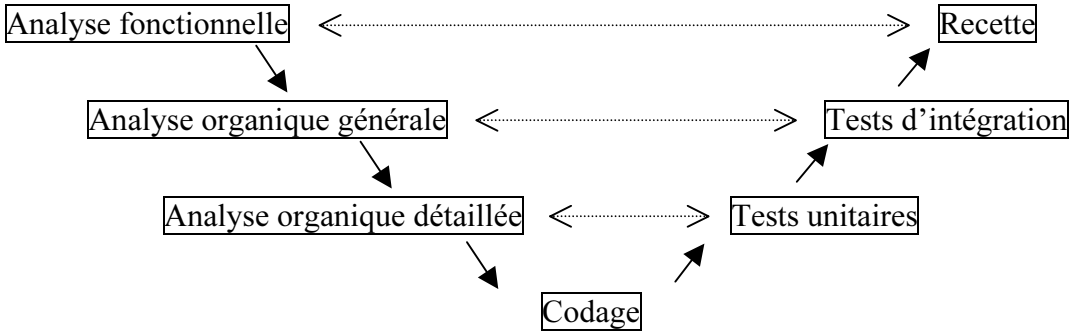
L'analyse fonctionnelle : c'est l'analyse des fonctionnalités du logiciel : ce qu'il fait.
L'analyse organique : c'est l'analyse de la façon dont on va réaliser les fonctionnalités du logiciel.
La recette : c'est la mise en service du logiciel avec les tests en situation de fonctionnement réelle.

C'est le lien entre les étapes de chaque branche qui justifie le cycle en V :



Quand on fait l'analyse fonctionnelle, on peut préparer la procédure de recette.
Quand on fait l'analyse organique, on peut préparer la procédure de test.

On peut encore détailler le cycle en V :



- ***Conception, Modélisation ou Analyse ? Un problème de représentation***

La phase de conception se divise en trois étapes d'analyse !

La phase de conception consiste à se représenter plus ou moins clairement un objectif à atteindre (par exemple fabriquer un logiciel avec certaines fonctionnalités). La représentation de cet objectif passe par une analyse la situation (le cahier des charges, les besoins et les attentes).

Cette analyse se divise en trois étapes : fonctionnelle, organique générale et organique détaillée.

De ces analyses, on va tirer les représentations tant fonctionnelles qu'organiques de l'objectif à atteindre.

A la place de conception, on parle aussi de modélisation.

➢ ***Distinction entre analyse organique et analyse fonctionnelle (architecture)***

On vient de voir que la conception se divise en deux parties :

- l'analyse fonctionnelle d'abord
- l'analyse organique (ou architecture) ensuite.

Il faut distinguer :

- le point de vue de l'utilisateur
- le point de vue de l'informaticien

Analyse - Conception - Modélisation	
Point de vue de l'utilisateur	**Point de vue de l' informaticien**
Point de vue du maître d'ouvrage	Point de vue du maître d'œuvre
Point de vue de celui qui commande le logiciel	Point de vue de celui qui réalise le logiciel
The right system	The system right
Le QUOI	Le COMMENT
Analyse fonctionnelle	Analyse organique
Externe	Interne
Build the right system	*Build the system right*
Construire le bon système	Construire bien le système

Pour l'utilisateur, ce qui compte, c'est l'usage du système : les cas d'utilisation (vocabulaire UML). L'analyse fonctionnelle permettra de modéliser l'ensemble des cas d'utilisation.

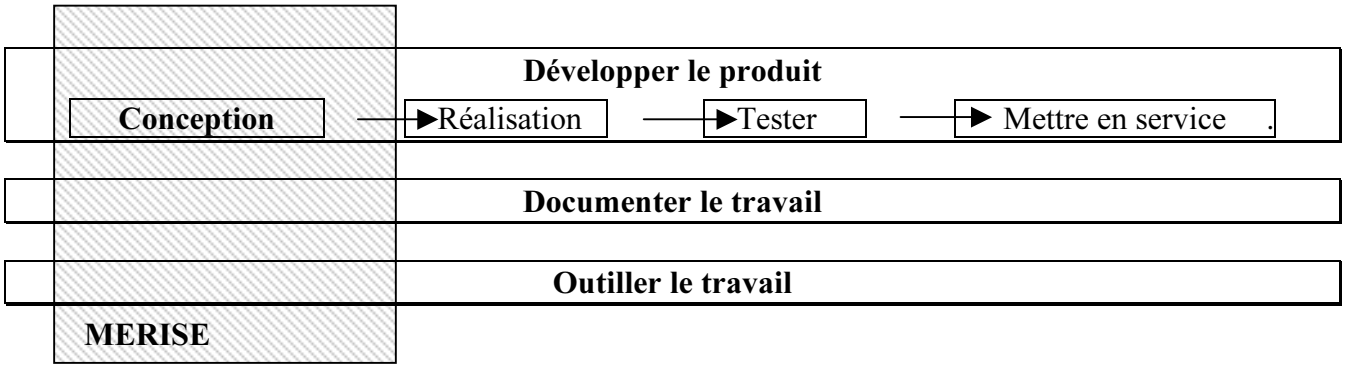
Pour l'informaticien, ce qui compte c'est l'architecture interne du système.

Méthodes de conception

La méthode peut s'appliquer plus directement au niveau de la conception, qu'elle soit fonctionnelle ou organique.

Exemples :

- ✓ La méthode MERISE
- ✓ Le méthode du RUP (qui utilise le langage de modélisation UML)



MERISE et le RUP s'utilisent pour la conception du produit à développer, ainsi que pour la documentation de la phase de conception et pour l'outillage (via les logiciels type Win'Design ou Power Designer pour MERISE, Rational ROSE ou Visio pour UML)

Objectifs de la méthode dans le développement informatique

- Faciliter le travail d'équipe
- Maîtriser les coûts et les délais
- Améliorer la qualité fonctionnelle (do the right system)
- Améliorer la qualité organique (do the system right)
- Faciliter la maintenance

3. Méthode analytique vs méthode systémique

Méthode analytique

Discours de la méthode

La méthode analytique est la méthode de décomposition classique dont on retrouve les fondements chez Descartes :

> *Certains chemins m'ont conduit à des considérations et des maximes dont j'ai formé une méthode par laquelle il me semble que j'ai moyen d'augmenter par degrés ma connaissance, et de l'élever peu à peu au plus haut point…*
>
> *Au lieu de ce grand nombre de préceptes dont la logique est composée, je crus que j'aurais assez des quatre suivants, pourvu que je prisse une ferme et constante résolution de ne manquer pas une seule fois a les observer.*
>
> ***Le premier** était de ne recevoir jamais aucune chose pour vraie que je ne la connusse évidemment être telle; c'est-à-dire, d'éviter soigneusement la précipitation et la prévention, et de ne comprendre rien de plus en mes jugements que ce qui se présenterait si clairement et si distinctement à mon esprit, que je n'eusse aucune occasion de le mettre en doute.*
>
> ***Le second,** de diviser chacune des difficultés que j'examinerais, en autant de parcelles qu'il se pourrait, et qu'il serait requis pour les mieux résoudre.*
>
> ***Le troisième,** de conduire par ordre mes pensées, en commençant par les objets les plus simples et les plus aisés à connaître, pour monter peu à peu comme par degrés jusques à la connaissance des plus composés, et supposant même de l'ordre entre ceux qui ne se précèdent point naturellement les uns les autres.*
>
> ***Et le dernier,** de faire partout des dénombrements si entiers et des revues si générales, que je fusse assuré de ne rien omettre.*
>
> Discours de la méthode, 1637, Descartes (1596-1650)

Ce discours met en avant quatre principes :

- L'évidence contre les préjugés pour décrire la réalité.
- L'analyse : La division du tout en partie.
- La synthèse : la reconstruction du tout à partir des parties.
- La totalité : ne rien omettre.

L'analyse descendante

La bonne méthode consiste à diviser le tout en parties, puis à réaliser les parties, avant de les réintégrer toutes ensembles. C'est l'analyse descendante.

Cette analyse correspond à l'analyse classique et à la méthode qu'il faut mettre en œuvre.

Les étapes de la méthode

1. Partir de la réalité : le problème à traiter.
2. S'en faire une idée claire et complète.
3. Diviser cette idée en parties.
4. Construire les parties une par une.
5. Intégrer les parties toutes ensemble.

Les erreurs de méthode : l'analyse ascendante : ne pas partir du tout

Une erreur classique consiste à ne pas partir du tout, mais à partir directement des parties. C'est **l'analyse ascendante : on part de l'étape 3.**

Le défaut de cette méthode est que l'absence d'analyse initiale de l'idée de la totalité va conduire à des grandes difficultés au moment de l'intégration des parties.

Les erreurs de méthode : se tromper sur le tout

En appliquant l'analyse descendante, on peut aussi se tromper en appliquant mal l'étape 2.

L'idée qu'on se fait du problème à traiter est incomplète ou fausse. En conséquence de quoi l'intégration des parties ne pourra pas donner un bon résultat.

Brève présentation de la méthode systémique

La science des systèmes, ou systémique, est à l'origine du développement de la notion de système d'information.

La systémique s'occupe de systèmes qui ont trois caractéristiques :

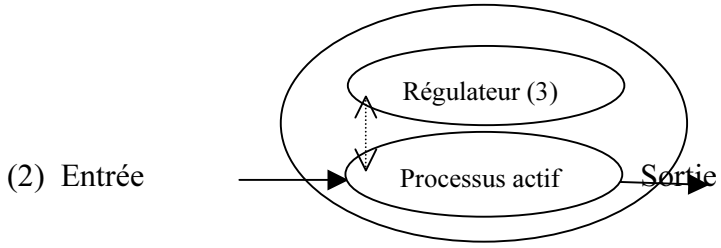
1. Ils sont dotés d'au moins un projet identifiable : c'est **l'hypothèse téléologique** (télos = finalité)
2. Ils sont ouverts sur leur environnement : c'est **l'hypothèse d'ouverture.**
3. Ils sont décrits totalement, dans l'espace et le temps : **c'est l'hypothèse structuraliste.**

De là, Boulding et Le Moigne ont proposé une classification à 9 niveaux des systèmes fondée sur leur complexité croissante.

Niveau 1 : l'objet est passif et sans activité

Niveau 2 : l'objet est actif et transforme des flux. Transformation mécanique indépendante du temps.

Niveau 3 : l'objet actif est régulé. Transformation mécanique fonction du temps : l'objet réagit en fonction de son propre état.

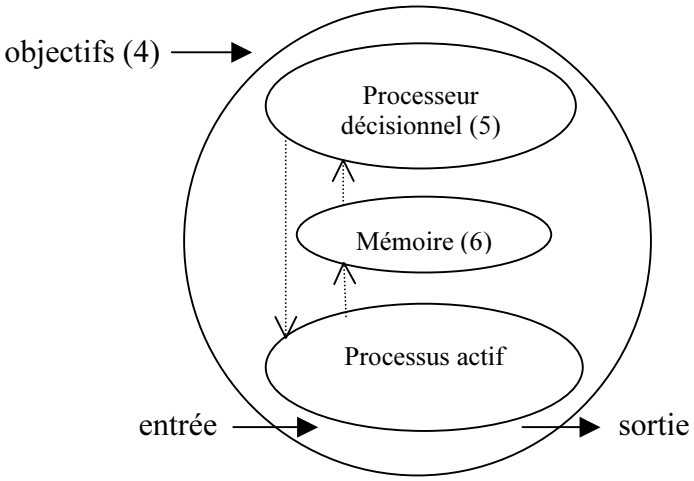


ISIM, p. 19

Niveau 4 : l'objet s'informe : l'objet peut recevoir des objectifs qui changent son comportement.

Niveau 5 : l'objet décide de son activité. Il passe d'un comportement théoriquement prévisible à un comportement « libre ».

Niveau 6 : l'objet a une mémoire. Pour prendre ses décisions, l'objet fait appel à des informations-représentations non seulement du comportement actuel, mais aussi des comportements passés.

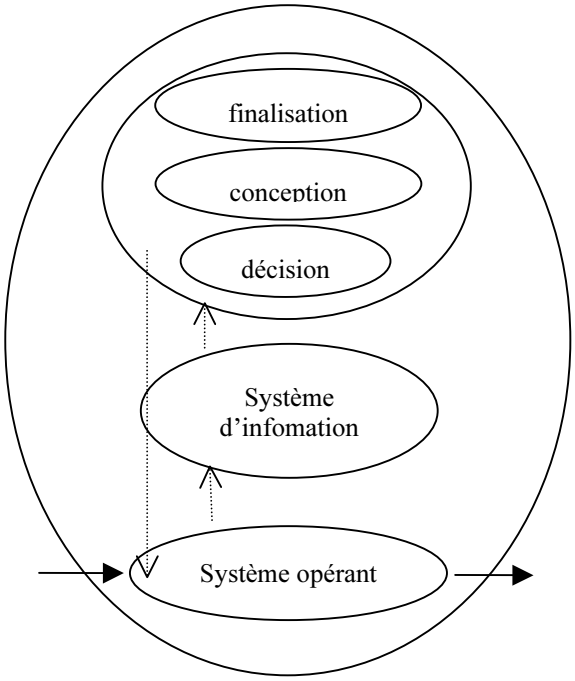


ISIM, p. 20

Niveau 7 : l'objet se coordonne. Il y a un gain de complexité et une structuration des différents centres : de décision, de mémoire et d'activité.

Niveau 8 : l'objet imagine et conçoit. Pour atteindre ses objectifs, l'objet est capable d'imaginer l'organisation la mieux adaptée pour ses sous-systèmes.

Niveau 9 : l'objet s'autofinalise. C'est le stade ultime de l'évolution. L'objet définit lui-même ses objectifs.



ISIM, p. 21

Principales caractéristiques des systèmes ouverts

La rétroaction : la rétroaction consiste à ce que les informations en sortie du système reviennent en entrée dans le système.

Equifinalité : les mêmes conséquences peuvent avoir des origines différentes.

La simulation : du fait de la rétroaction et de l'équifinalité, les systèmes ouverts s'étudient avec des simulations.

Approche analytique *vs* approche systémique ?

Approche analytique	**Approche systémique**
Indépendante du temps	Fonction du temps
Toutes les interactions sont analysées : le système est déterministe.	Les interactions sont trop nombreuses pour être analysées : le système n'est pas déterministe.
Test de fonctionnalité	Simulation de sous-système
Utilisation déterminée	Utilisation ouverte

Remarque : attention aux conclusions trop hâtives après simulation !

D'un côté, la différence entre une approche systémique et une approche analytique est réelle : le rapport au temps, au déterminisme et à la complexité est différent dans les deux cas.

Cependant, d'un autre côté, les 3ème et 4ème principes de la méthode cartésienne peuvent être compris comme des principes systémiques :

- Ne rien omettre, c'est prendre en compte la complexité.
- Partir du plus simple, c'est partir des invariants, c'est-à-dire du modèle des données et des règles de gestion.

Conclusion :

La méthode systémique consiste à analyser le système non pas par fonction, mais par structure. On recherche les invariants au niveau fonctionnel, avant de partir dans la réalisation fonctionnelle.

Ces invariants sont la structure du système : ce sont les données brutes et les règles d'organisation. Ils permettent la représentation du réel par un modèle : la modélisation.

La démarche systémique passe par la modélisation du domaine à étudier.

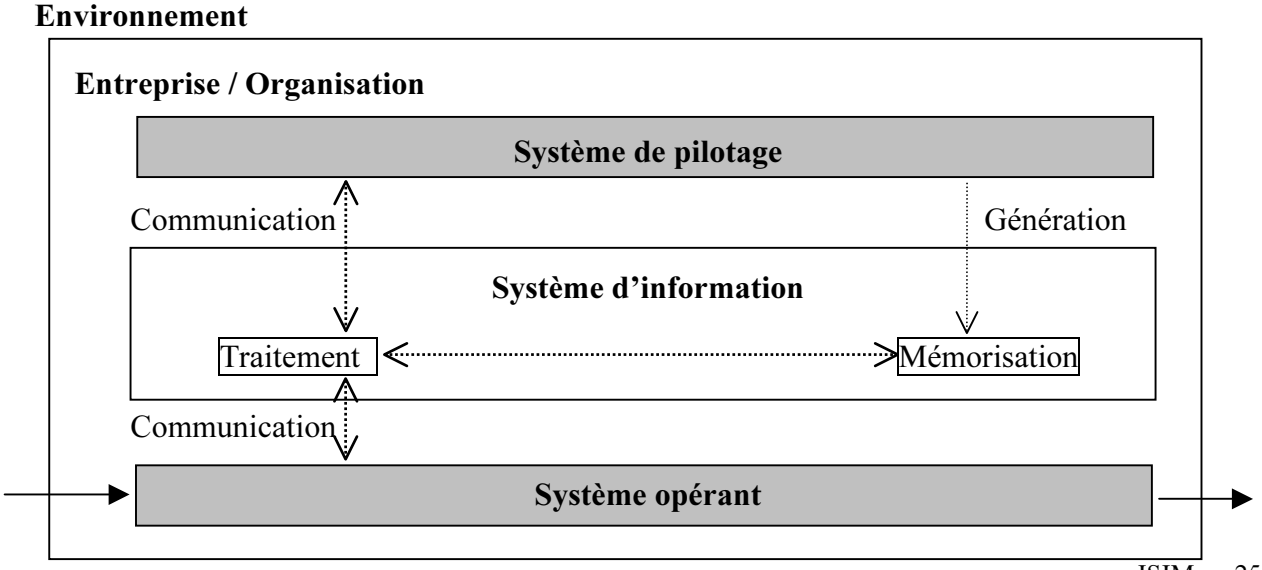
4. Système d'information

Le système d'information de l'entreprise

La systémique propose une modélisation progressive des objets qui fait apparaître la notion de système d'information.

L'analyse systémique facilite la compréhension de l'entreprise.

Elle permet d'arriver à la modélisation suivante de l'entreprise :



ISIM, p. 25

Le système opérant est le siège de l'activité productive de l'entreprise. Cette activité consiste en une transformation des flux primaires. Ces flux primaires peuvent être des flux de matière, de finance, de personnel ou d'information.

Le système de pilotage est le siège de l'activité décisionnelle de l'entreprise. Il permet le pilotage, la régulation et l'adaptation. Il conduit aux évolutions, notamment des systèmes opérant et d'information. Cette activité décisionnelle est très large et elle est assurée par tous les acteurs de l'entreprise à des niveaux divers.

Le système d'information est la représentation de l'activité du système opérant et/ou du système de pilotage, représentation conçue à l'initiative du système de pilotage en fonction des objectifs à atteindre et de l'organisation choisie.

Système d'information organisationnel, système informatisé et système informatique

Un système d'information organisationnel, SIO, est une représentation possible de n'importe quel système, notamment de tout système humain organisé (donc de toute entreprise). Une entreprise peut donc être considérée comme un SIO.

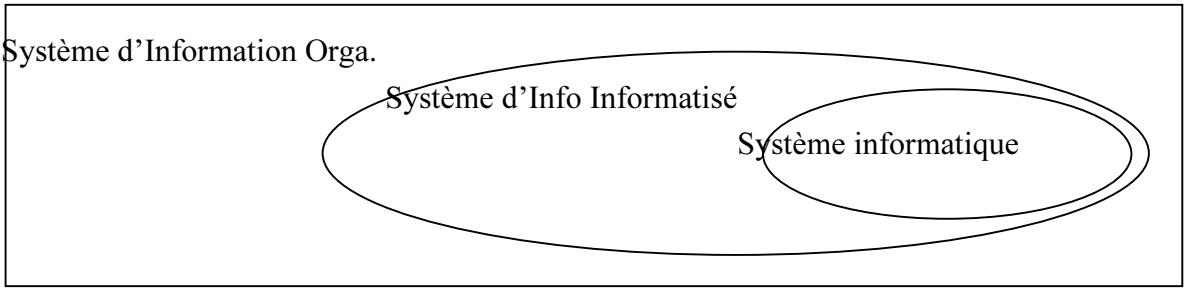
Les systèmes d'information organisationnels précèdent donc l'informatique.

Un système d'information organisationnel peut contenir un système d'information informatisé.

Le système d'information informatisé, c'est la partie informatisée du système d'information organisationnel à laquelle les utilisateurs ont accès.

Le système d'information informatisé se doit d'être au service du système d'information organisationnel mis en place par les dirigeants de l'entreprise, et non l'inverse !

Le système informatique, c'est le support matériel et logiciel du système d'information informatisé.



5. Informatique de gestion *vs* informatique industrielle

Les notions que nous venons de présenter :

- Génie logiciel
- Ingénierie des systèmes d'information
- Méthode
- Analyse systémique
- Système d'information

s'appliquent aussi bien dans le domaine de l'informatique de gestion que dans celui de l'informatique industrielle.

Informatique de gestion	**Informatique industrielle (scientifique)**
Gestion de fichier - Cobol	Langage de calcul et L3G
Modèle relationnel, base de données et L4G	L5G et programmation objet
Méthode MERISE	Cycle en V

Si la notion de système d'information peut sembler liée à la gestion de l'entreprise, elle permet aussi d'appréhender bon nombre de projets industriels importants.

Exemples où l'ensemble du système peut être considéré comme un système d'information :

- Composante Sol Utilisateur d'un satellite d'observation
- Poste de Contrôle et de Commandement d'un métro automatique
- Planification de réseau électrique

LE POINT DE VUE DU MAITRE D'OUVRAGE

1 Conduite des grands projets

Maître d'ouvrage et maître d'œuvre

L'expérience de la conduite de grands projets montre que pour maîtriser un projet tant sur le plan des coûts que des délais, il faut séparer les rôles de maître d'ouvrage et de maître d'œuvre. Cette séparation existe depuis longtemps dans les projets de travaux publics, mais n'était guère adoptée dans les projets informatiques. Cependant elle tend à se généraliser dans les projets informatiques.

En effet celui qui a besoin d'un logiciel sera préoccupé du coût, du délai et de l'adéquation à son besoin. Les préoccupations du réalisateur sont très différentes, c'est assurer son plan de charge, utiliser les technologies nouvelles…

Le maître d'ouvrage représente le futur propriétaire (ou les futurs utilisateurs) de l'ouvrage. Il gère le budget, définit la stratégie d'acquisition, prépare le cahier des charges, négocie le contrat de réalisation et accepte les fournitures.

Le maître d'œuvre est désigné par le maître d'ouvrage pour assurer la conception et le contrôle de la réalisation de l'ouvrage. Il conduit donc la réalisation du projet dans le cadre d'un contrat (explicite ou implicite) avec le maître d'ouvrage et présente les fournitures au maître d'ouvrage pour acceptation.

Le maître d'ouvrage suit un processus d'acquisition.

Le maître d'œuvre suit un processus de fourniture (qui fait appel au processus de développement et aux processus dits organisationnels : management, infrastructure, formation, amélioration).

PROCESSUS D'ACQUISITION (MAITRE D'OUVRAGE)

PROCESSUS DE FOURNITURE (MAITRE D'ŒUVRE)

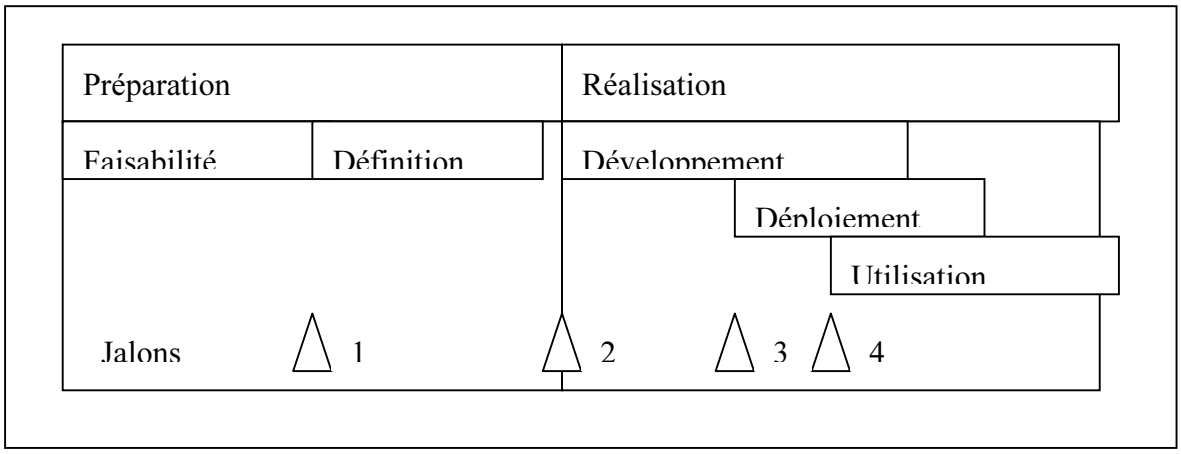
Phasage du projet

Pour maîtriser un grand projet, il est nécessaire de le découper temporellement en phases successives avec des jalons.

Le découpage en phases peut varier avec la nature des projets mais on retrouve les mêmes principes :

- des phases de préparation en commençant par une exploration large de solutions,
- des phases de réalisation et d'utilisation.

Exemple de phasage



2 Processus d'acquisition - préparation du projet

Le maître d'ouvrage représente ceux qui ont besoin du logiciel (ce peut être des utilisateurs futurs, un investisseur…) et assureront le financement. Il doit rechercher la solution qui satisfera complètement le besoin au moindre coût, ou au meilleur rapport qualité / prix.

Pour cela on distingue deux phases dans la préparation d'un projet :

- **La faisabilité** : c'est une phase exploratoire où l'on fait une exploration large des solutions.
- **La définition** : après le choix d'un type de solution, la définition est une phase d'approfondissement de la solution pour recueillir les éléments nécessaires pour démarrer une réalisation (choix d'un maître d'œuvre, budget, planning, contrat…).

Travaux de faisabilité

Pour remplir un besoin logiciel il peut exister de multiples possibilités : développement d'un logiciel spécifique, adaptation d'un logiciel existant, utilisation d'un progiciel, recours à un façonnier…

Les travaux de la phase de faisabilité consistent à examiner tous ces types de solution pour voir si elles sont faisables et si elles permettent de remplir le besoin, et évaluer grossièrement leur coût.

Les solutions peuvent avoir des coûts très différents pour l'acquisition, mais aussi pour l'utilisation. On fera donc les comparaisons de coût global de possession, somme des coûts d'acquisition et des coûts d'exploitation, maintenance et maintien à niveau sur la durée prévue d'utilisation (par exemple 5 ans ou 10 ans, il n'y a pas besoin d'être précis, on ne comparera que des ordres de grandeur).

A titre d'exemple, supposons que vous venez de fonder une start up, et que vous avez le projet d'acquérir un logiciel pour la paye de votre employé. Vous explorez les solutions « logiciel spécifique », adaptation d'un logiciel existant, achat d'un progiciel de paye, utilisation d'EXCEL, recours à un façonnier. Dans chaque solution on évalue le coût d'acquisition (licences et personnel), le coût d'exploitation annuel (journées de personnel), le coût moyen annuel de maintien à niveau (la réglementation sociale évolue perpétuellement…), et le coût global de possession sur 5 ans.

Solution	Acquisition (kF)	Exploitation /an	Maintien niveau /an	CGP 5 ans (kF)
Logiciel spécifique	600	5	50	875
Adaptation d'un logiciel	200	5	100	725
Progiciel de paye	200	10	50	750
EXCEL	10	10	10	210
Façonnier	0	20	0	100

On voit que dans ce cas, il vaut mieux ne pas acheter de logiciel, mais que deux solutions se détachent : utiliser tout bêtement un tableur ou avoir recours à un façonnier.

Le critère financier n'est pas le seul critère à prendre en compte, il faut aussi examiner d'autres critères tels que le degré de satisfaction du besoin, la pérennité de la solution…

La phase de faisabilité peut se terminer par la fourniture d'un rapport faisant des propositions de choix, ou la tenue d'une revue. Il faut choisir le type de solution avant de lancer la phase suivante (on peut cependant parfois garder deux solutions qui seront approfondies avant de choisir).

Travaux de définition

Les différents travaux à mener par le maître d'ouvrage sont :

- L'approfondissement du besoin par la rédaction d'un cahier des charges fonctionnel
- L'examen du marché, des candidats possibles pour la maîtrise d'œuvre
- La définition de la stratégie d'acquisition
- La consultation du ou des maîtres d'œuvre potentiels
- Le choix du maître d'œuvre et mise au point du contrat de réalisation
- L'établissement du calendrier et du budget du projet

Cahier des charges fonctionnel

Le cahier des charges fonctionnel présente le besoin vu de l'utilisateur, il comprend normalement :

- la présentation du problème : le produit et son marché, le contexte du projet, les objectifs
- l'énoncé fonctionnel du besoin : cycle d'utilisation du produit et identification de son environnement, présentation des services et fonctionnalités demandées, contraintes.

Plan type d'un cahier des charges

1. Objet
2. Description du produit
   - Principes
   - Services demandés
   - Présentation des informations manipulées
   - Sécurité et droits d'accès
3. Données techniques

4. Contraintes diverses

Stratégie d'acquisition et consultation

La stratégie d'acquisition dépendra du type de solution choisie. S'il s'agit d'un développement de logiciel, on peut avoir recours au service informatique de l'entreprise, négocier avec une SSII, lancer une consultation industrielle plus ou moins large…Si l'on décide d'avoir recours à une consultation, il faudra établir la liste des consultés.

Pour consulter le ou les maîtres d'œuvre potentiels, il faut préparer un dossier de consultation qui comprend le règlement de consultation, le cahier des charges fonctionnel, et éventuellement le projet de contrat.

Un règlement de consultation contient en général :

- l'objet de la consultation,
- les conditions de remise des offres,
- le contenu des offres,
- les critères d'évaluation des offres,
- les exigences de moyens (outils imposés, matériels…)
- les exigences d'assurance de management ou de qualité.

Il y a intérêt à formaliser la consultation dans tous les cas, qu'il y ait ou non concurrence, et même si on doit travailler avec le service informatique de la société. Cependant s'il n'y a pas de concurrence le règlement de consultation pourra être très simplifié.

Le ou les maîtres d'œuvre pressentis répondent à la consultation par une proposition technique et commerciale. Le maître d'ouvrage compare les proposition sur les plans techniques et financiers et choisit le meilleur.

Préparation du contrat et lancement de la réalisation

Le maître d'ouvrage met au point le contrat de réalisation (qui peut être un contrat de droit public si l'état est maître d'ouvrage, un contrat de droit privé, un contrat interne…) avec le candidat retenu.

Un contrat se compose en principe de :

- clauses administratives :
  - lotissement, prix, délais,
  - conditions d'ajustement de prix, de pénalités,
  - clauses de paiement,
  - clauses de propriété intellectuelle, de garantie, de résolution des litiges…
- clauses techniques :
  - besoin à satisfaire, contraintes techniques et conditions d'acceptation (peut faire l'objet d'un document indépendant, cahier des charges fonctionnel ou spécification technique),
  - composition détaillée des prestations,
  - utilisation imposée de moyens,
  - exigences en matière de management de projet et de qualité.

Après avoir défini le calendrier du projet, le budget du projet et le choix du maître d'œuvre, et mis au point le contrat, le maître d'ouvrage pourra proposer de franchir le jalon lancement du développement et de passer aux phases de réalisation en notifiant le contrat.

MERISE

1. Définition

MERISE est une méthode systémique de conception des systèmes d'information.

Elle est en relation avec le développement des bases de données relationnelles (SQL).

En 2001, la méthode MERISE était encore la méthode de conception de systèmes d'information la plus largement pratiquée en France.

MERISE a pris en compte les évolutions de l'informatique et continue de s'adapter aux nouvelles technologies : architectures clients/serveur, interfaces graphiques, démarche de développement rapide, approche objet, applications intra/internet.

Aujourd'hui, la méthode MERISE correspond encore globalement aux savoir-faire actuels en ingénierie des systèmes d'information de gestion.

MERISE constitue un standard de fait en conception des systèmes d'information.

2. Les trois composantes de la méthode MERISE

La démarche de développement d'un système d'information s'inscrit dans trois dimensions :

- Le cycle de vie : méthode de développement
- Le cycle d'abstraction : méthode de conception
- Le cycle de décision : méthode de gestion de projet

Le cycle de vie

Le cycle de vie MERISE est une méthode de développement au même titre que le cycle en V.

Le cycle de vie est découpé en trois périodes: la conception, la réalisation et la maintenance.

LE CYCLE DE VIE		
Etapes de la démarche		**Explications**
Conception	Schéma directeur	Définition des orientations générales du développement à moyen terme des systèmes d'information
	Étude préalable	Proposition et évaluation de solutions d'organisation et de solutions techniques pour le SI d'un domaine. Cette étape porte sur un sous-ensemble représentatif du domaine étudié.
	Étude détaillée	Spécifications complètes du futur SIO du point de **vue de l'utilisateur** (point de vue externe). Elle comporte deux phases : • la conception générale (extension de l'étude préalable à tout le domaine) • la conception détaillée (description complète de chacune des tâches à automatiser).

Réalisation	Étude technique	Spécifications complètes du futur SII du **point de vue du réalisateur** (point de vue interne).
	Production logicielle	Écriture des programmes, générations des fichiers ou des bases de données, tests.
	Mise en service	Installation de l'application informatique, vérification du bon fonctionnement, mise en place de la nouvelle organisation, formation des utilisateurs.
Maintenance	Maintenance	Rectification des anomalies, améliorations, évolutions.

ISIM, p. 32

Comparaison avec le cycle en V

Dans la terminologie MERISE, la première étape de la réalisation (l'étude technique), c'est la dernière étape de la branche de conception du cycle en V : l'analyse organique.

La production logicielle correspond à l'étape de codage. La mise en service à la phase de test et de recette.

Du point de vue du génie logiciel, on peut penser que le cycle en V est plus subtil que le cycle de vie MERISE.

Le cycle d'abstraction

Le cycle d'abstraction est découpé en quatre niveaux : conceptuel, organisationnel, logique et physique.

- **Le niveau conceptuel** exprime des choix fondamentaux de gestion (recherche d'éléments stables indépendamment des moyens à mettre en œuvre, de leurs contraintes et de leur organisation).
- **Le niveau organisationnel** exprime les choix d'organisation de ressources humaines et matérielles, au travers notamment de la définition de sites et de postes de travail.
- **Le niveau logique** exprime les choix de moyens et de ressources informatiques, en faisant abstraction de leurs caractéristiques techniques précises.
- **Le niveau physique** traduit les choix techniques et la prise en compte de leurs spécificités.

	DONNEES	**TRAITEMENTS**	
Niveau Conceptuel	**M C D** ***Modèle conceptuel des données*** Signification des informations sans contraintes techniques, organisationnelle ou économique	**M C T** ***Modèle conceptuel des traitements*** Activité du domaine sans préciser les ressources et leur organisation	**SIO** Système d'information organisationnel
Niveau organisationnel	**M O D** ***Modèle organisationnel des données*** Signification des informations avec contraintes organisationnelles et économiques (répartition et quantification des données ; droit des utilisateurs)	**M O T** ***Modèle organisationnel des traitements*** Fonctionnement du domaine avec les ressources utilisées et leur organisation (répartition des traitements sur les postes de travail)	

Niveau logique	**M L D** ***Modèle logique des données*** Description des données tenant compte de leurs conditions d'utilisation (contraintes d'intégrité, historique par les traitements et les techniques de mémorisation	**M L T** ***Modèle logique des traitements*** Fonctionnement du domaine avec les ressources et leur organisation informatique	**SII** Système d'information informatisé
Niveau physique	**M P D** ***Modèle physique des données*** Description de la (ou des) base(s) de données dans la syntaxe du SG de données (SGF ou SGBD). Optimisation des traitements (indexation et dénormalisation).	**M P T** ***Modèle physique des traitements*** Architecture technique des programmes	

D'après ISIM, p. 39

Le but de ces distinctions est de clarifier l'analyse et les niveaux de contraintes afin d'éviter de faire intervenir trop tôt dans l'analyse les conséquences de contraintes de niveaux inférieurs. Par exemple, il faut éviter de répartir l'information avant de l'avoir analyser indépendamment de toute répartition.

Le cycle d'abstraction est le principal apport durable de la méthode MERISE.

Le cycle de décision

Le cycle de décision représente l'ensemble des choix qui doivent être faits durant le déroulement du cycle de vie.

Étapes de la démarche	Résultats	Décisions
Schéma directeur	Plan de développement des SI	Approbation et mise en application
Étude préalable	Dossier des choix, n solutions	Choix d'une solution ou arrêt
Étude détaillée	Spécifications fonctionnelles	Accord des utilisateurs sur les spécifications fonctionnelles
Étude technique	Spécifications techniques pour la réalisation	Accord des réalisateurs sur les spécifications techniques
Réalisation logicielle	Système réalisé en ordre de marche	Recette provisoire : conformité du système
Mise en service	Système installé dans l'organisation	Recette définitive : système en service
Maintenance	Système maintenu	Recette simplifiée : fin de maintenance

ISIM, p. 41

3. Plans types

Les étapes de la démarche du cycle de vie donnent lieu à la production de documents.

Comme toute autre méthode, la méthode MERISE propose des plans types pour tous les documents prévus par la méthode.

L'étude préalable et de l'étude détaillée sont les deux études sont les plus spécifiques à MERISE car elles font intervenir l'essentiel du cycle d'abstraction.

Plan type de l'étude préalable

1. Recueil

Préparation et réalisation des interviews

Recherche de la documentation

Description et bilan de l'existant

2. Conception

Élaboration des divers scénarios

Élaboration des MCD et MCT

Maquette et prototype

Élaboration du cahier des charges fonctionnel

3. Qualité

Définition des exigences qualité globale

Définition des exigences qualité par fonction

4. Chiffrage

Estimation prévisionnelle des charges, coût, délais

Planning prévisionnel

Résultats obtenus :

Cahier des charges fonctionnel

Dossier de choix

Plan type de l'étude détaillée

1. Recueil complémentaire

Préparation et réalisation des interviews des utilisateurs

Recherche de la documentation

Actualisation de l'étude préalable

2. Conception

Mise à jour des MCD et MCT

Élaboration du MOT

Description des états et des écrans
Validation croisée MCD / MOT
Élaboration du MLD

3. Qualité

Définition des facteurs qualité

4. Chiffrage

Estimations globale et détaillée
Plannings global et détaillé

Résultats obtenus :

Dossier des spécifications fonctionnelles
Plan de développement logiciel

UML ET LE RUP

1 UML

- UML est un langage de modélisation.
- Ce langage permet de modéliser les systèmes à événements ou à messages. Il n'est pas fait pour modéliser des procédés continus.
- UML permet de produire une partie du code, comme le squelette des classes (attributs et signatures de méthode). Mais ce n'est pas (pas encore ?) un langage formel : ce n'est pas un langage de programmation.
- En tant que langage, UML a une sémantique et une syntaxe qu'il convient de respecter.
- UML couvre toutes les phases du cycle de développement d'un logiciel. Il permet de modéliser toutes les phases de la conception : de l'analyse fonctionnelle à l'analyse architectonique.

<table>
<tr><th></th><th>Modèle UML</th><th>Diagramme UML</th></tr>
<tr><td rowspan="3">ANALYSE FONCTIONNELLE</td><td rowspan="3"></td><td>Cas d'utilisation</td></tr>
<tr><td>Séquence</td></tr>
<tr><td>Activités</td></tr>
<tr><td rowspan="5">ANALYSE ORGANIQUE</td><td rowspan="2">Modèle objet (modèle statique)</td><td>Classes</td></tr>
<tr><td>Objets</td></tr>
<tr><td rowspan="2">Modèle dynamique objet</td><td>Séquence</td></tr>
<tr><td>Collaboration</td></tr>
<tr><td rowspan="2">Modèle dynamique non objet</td><td>Etats-transitions</td></tr>
<tr><td>Activités</td></tr>
</table>

On voit qu'on retrouve les diagrammes de séquence tant au niveau fonctionnel qu'au niveau architectonique. Les diagrammes de séquences permettent de modéliser le comment, qu'il soit externe ou interne.

- UML, comme MERISE, peut également servir à la communication avec les clients et les utilisateurs du logiciel.
- UML permet de construire plusieurs modèles d'un système, chacun mettant en valeur des aspects différents : fonctionnels, statiques, dynamiques et organisationnels.

  On comprend que si les moyens diffèrent entre UML et MERISE, les grandes catégories restent les mêmes.

- UML est un langage. Ce n'est pas une méthode. La méthode associée à UML, c'est le RUP, Rational Unified Process.
- UML est aujourd'hui le résultat d'un large consensus.

2 Le modèle RUP

Rational Unified Process : c'est le modèle de la programmation objet.

Le RUP est un processus piloté par les cas d'utilisation (fonctionnel), centré sur l'architecture (organique), itératif et incrémental.

Le RUP répète des cycles constituant la vie du système et donnant naissance à une version.

Tout cycle se conclut par la livraison d'une version du produit.

Un cycle se divise en phase, qui elles-mêmes se divisent en itérations.

L'itération désigne une étape de l'enchaînement des **activités**.

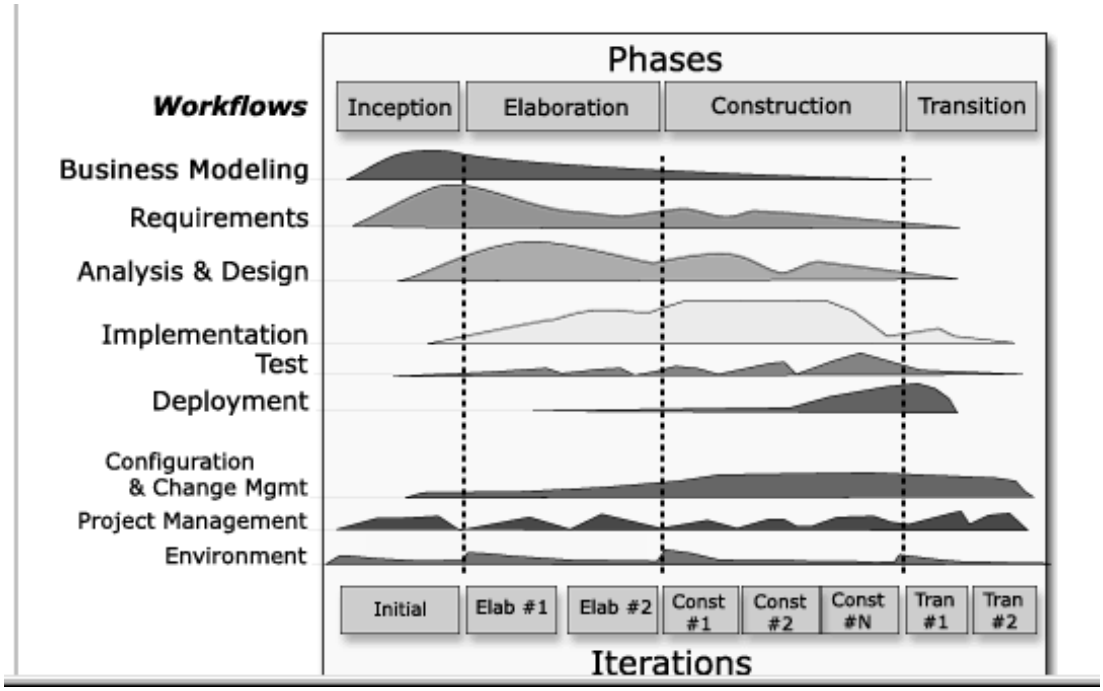
L'incrément désigne un **stade de développement** du projet.

Une itération ajoute un ou plusieurs incréments.

Le RUP définit des tâches à répartir sur les phases.

Le cycle est articulé en 4 phases

- La phase de création (étude préalable) : elle pousse l'étude de rentabilité jusqu'au stade propre à lancer le projet. Il s'agit d'obtenir l'accord de toutes les parties concernées.
- La phase d'élaboration : elle la plupart des cas d'utilisation restants. Elle met en place les fondations de l'architecture.
- La phase de construction (développement) : elle complète la description des cas d'utilisation, elle modifie l'architecture si nécessaire, implémente les sous-systèmes de l'incrément et les teste, puis les intègre au système et le teste.
- La phase de transition : elle finalise l'installation de l'incrément en intégrant les conditions du fonctionnement opérationnel et en validant la satisfaction du client.



Il existe 6 tâches réparties sur les 4 phases

- **Modélisation par métier :** elle a pour principal objectif le développement d'un modèle du métier, constitué d'un modèle de cas d'utilisation métier et d'un modèle objet métier (soit à peu près un MCT et un MCD).
- **Étude des besoins (fonctionnel) :** elle a pour principal objectif de définir une vision du système par un modèle des cas d'utilisation qui constituera le cahier des charges logiciel, et de construire un prototype de l'interface utilisateur.
- **Analyse et conception (architectonique) :** son objectif est de traduire l'ensemble des exigences en une spécification qui décrit comment réaliser le logiciel.
- Implémentation
- Tests
- Déploiement

HISTORIQUE

Date	MERISE	UML
1970	Modèle Relationnel de Codd.	
Années 70	Premiers prototypes de SQL.	
1974	Concept de brainware introduit par Tosio Kitagawa.	
1974-78	Le noyau de Merise est établi par une équipe d'ingénieurs et de chercheurs aixois.	
1976	Modèle Entité Association.	
1978	Développement de MERISE : méthode française de conception de systèmes d'information, sous l'égide du ministère de l'industrie.	
1979	Première version de SQL, proposé par ORACLE.	
1979	Conception du système d'information, construction de la base de données, H. Tardieu, D. Nanci, D. Pascot (préfacé par J.-L. Le Moigne), Editions d'Organisation.	
1983	La méthode MERISE - Tome 1 : principes et outils. H. Tardieu, A. Rochfeld, R. Colletti. Éditions d'Organisation.	
1985	La méthode MERISE - Tome 2 : démarche et pratique. H. Tardieu, A. Rochfeld, R. Colletti, G. Panet, G. Vahée. Éditions d'Organisation.	
1986	SQL ANSI (American National Standard Institute)	
1989	La méthode MERISE - Tome 3 : gamme opératoire. A. Rochfeld, J. Moréjon. Édition d'Organisation.	
Fin 80		Utilisation des langages orientés objets dans l'industrie.
1992	Ingéniérie des systèmes d'information : MERISE. 1ère édition. D. Nanci, B. Espinasse. Sybex.	
Courant 90		Plusieurs dizaines de méthodes objets.
fin années 90	PHP-MySQL	
1996		UML 0.9 de Booch, Jacobson et Rumbaugh
1997		UML 1.0
1999	SQL-3, ISO et ANSI	RUP, Le processus unifié de développement logiciel, Booch, Jacobson, Rumbaugh, Eyrolles.
2000		Modélisation UML avec Rationnal Rose 2000, Terry Quatrini, Eyrolles, 2000.
2001	Ingéniérie des systèmes d'information : MERISE. 4ème édition. D. Nanci, B. Espinasse. Vuibert.	
2004		UML 2
2004		UML 2.0 - Guide de référence, Booch, Jacobson et Rumbaugh, CampusPress.
2006	ORACLE Database XE	

ADAPTATIONS DU CYCLE EN V - TRAVAIL EN EQUIPE

1 Succession des activités

Les sept activités du cycle en V doivent en principe être exécutées dans l'ordre et successivement.

Pour minimiser les risques, il est recommandé d'attendre qu'une activité soit complètement terminée et vérifiée pour commencer l'activité suivante.

Cette démarche est souvent imposée pour les logiciels critiques, où on place en fin de chaque activité une revue qui permet de valider les résultats et d'autoriser le début de l'activité suivante.

Cependant, elle est source de perte de temps, l'équipe projet étant sous-employée pendant la période de fin d'activité. C'est pourquoi un certain chevauchement des activités est permis, mais il ne faut pas qu'il soit trop important, sinon cela pourrait conduire à reprendre une activité lorsque les résultats de l'activité précédente sont validés.

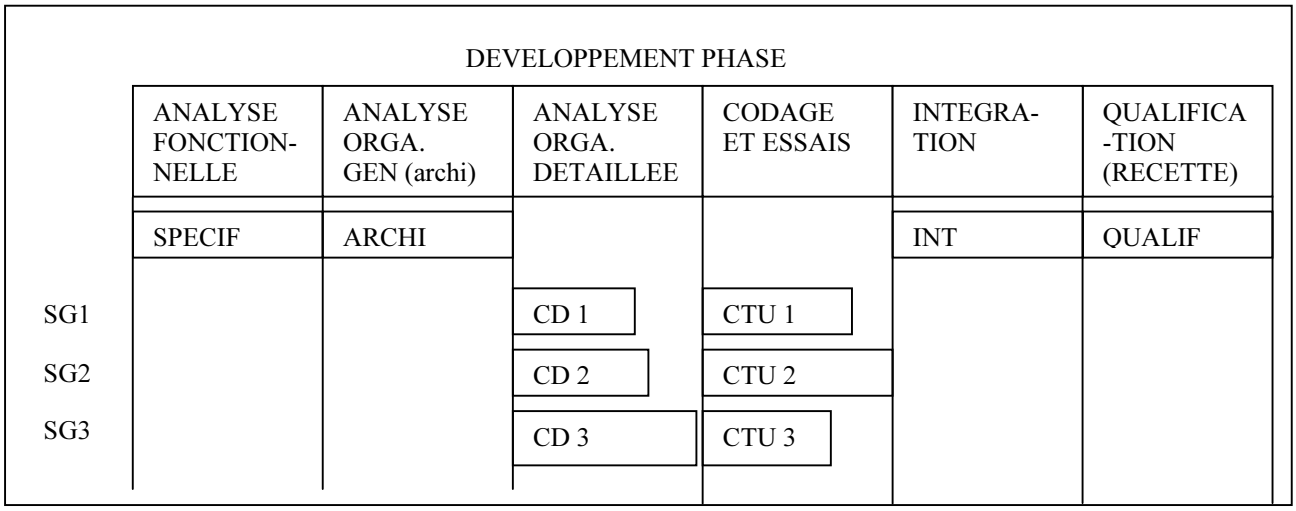
2 Conséquences du travail en équipes

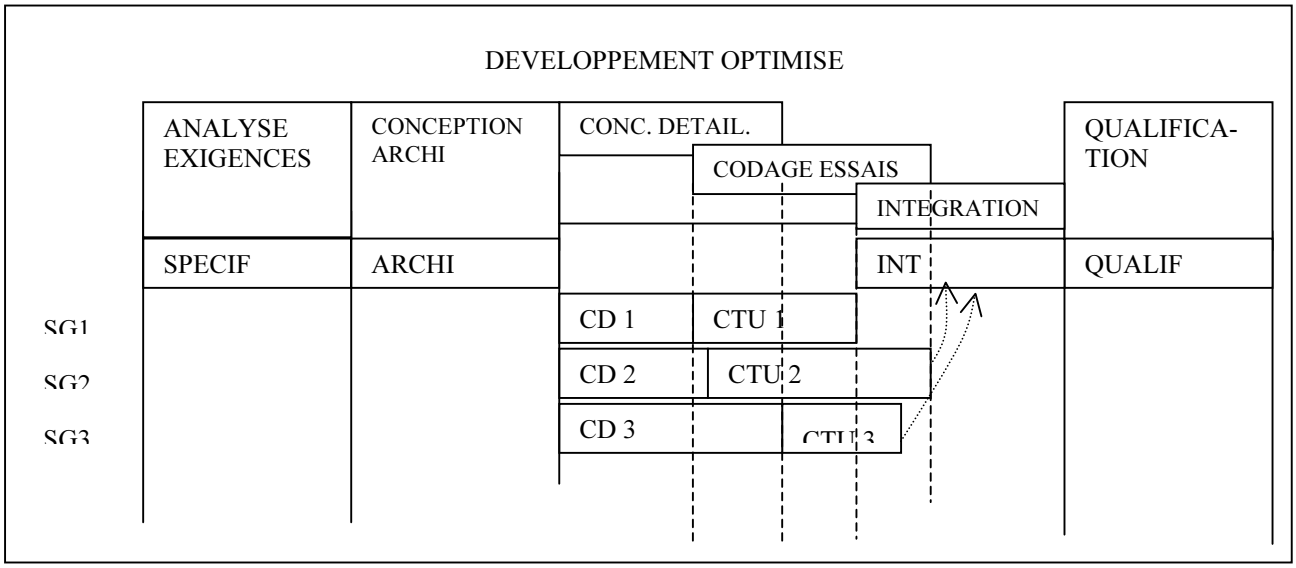
Lorsque le logiciel n'est pas réalisé par une seule personne, mais par une équipe, une première méthode consisterait à faire réaliser la première activité par toute l'équipe, et de même pour toutes les activités successives. Une telle méthode offre une grande souplesse face à l'absentéisme, mais est peu motivante pour l'équipe et conduit généralement à ce qu'une partie de l'équipe ne fasse pratiquement rien. Dans la pratique elle n'est guère utilisable que pour de toutes petites équipes (2 à 3 personnes).

C'est pourquoi il est recommandé de diviser l'équipe en sous-groupes. Lors de la conception de l'architecture, on peut identifier des sous-systèmes dont la réalisation peut être faite de façon quasi indépendante. On confie alors la responsabilité de la réalisation (conception détaillée, codage et essais) de chaque sous-système à un sous-groupe. Les activités d'analyse des exigences, de conception de l'architecture, d'intégration et de qualification ne peuvent être décomposées, et doivent être menées par l'ensemble de l'équipe (souvent dans les projets industriels l'équipe mise en place est réduite pendant ces activités).

Les activités de conception détaillée et de codage sont donc décomposées en sous-activités menées en parallèle par les sous-groupes. Deux options sont possibles :

- un développement phasé, ou on attend que toutes les sous-activités soient terminées pour passer à l'activité suivante, cela permet des contrôles complets à chaque étape, mais fait perdre du temps si les travaux des sous-groupes ne sont pas équilibrés,
- un développement optimisé, où chaque sous-groupe enchaîne ses sous-activités sans attendre les autres sous-groupes, et où l'intégration peut commencer dès qu'un sous-groupe a terminé le codage et essais d'un sous-système.





3 Adaptations du cycle en V

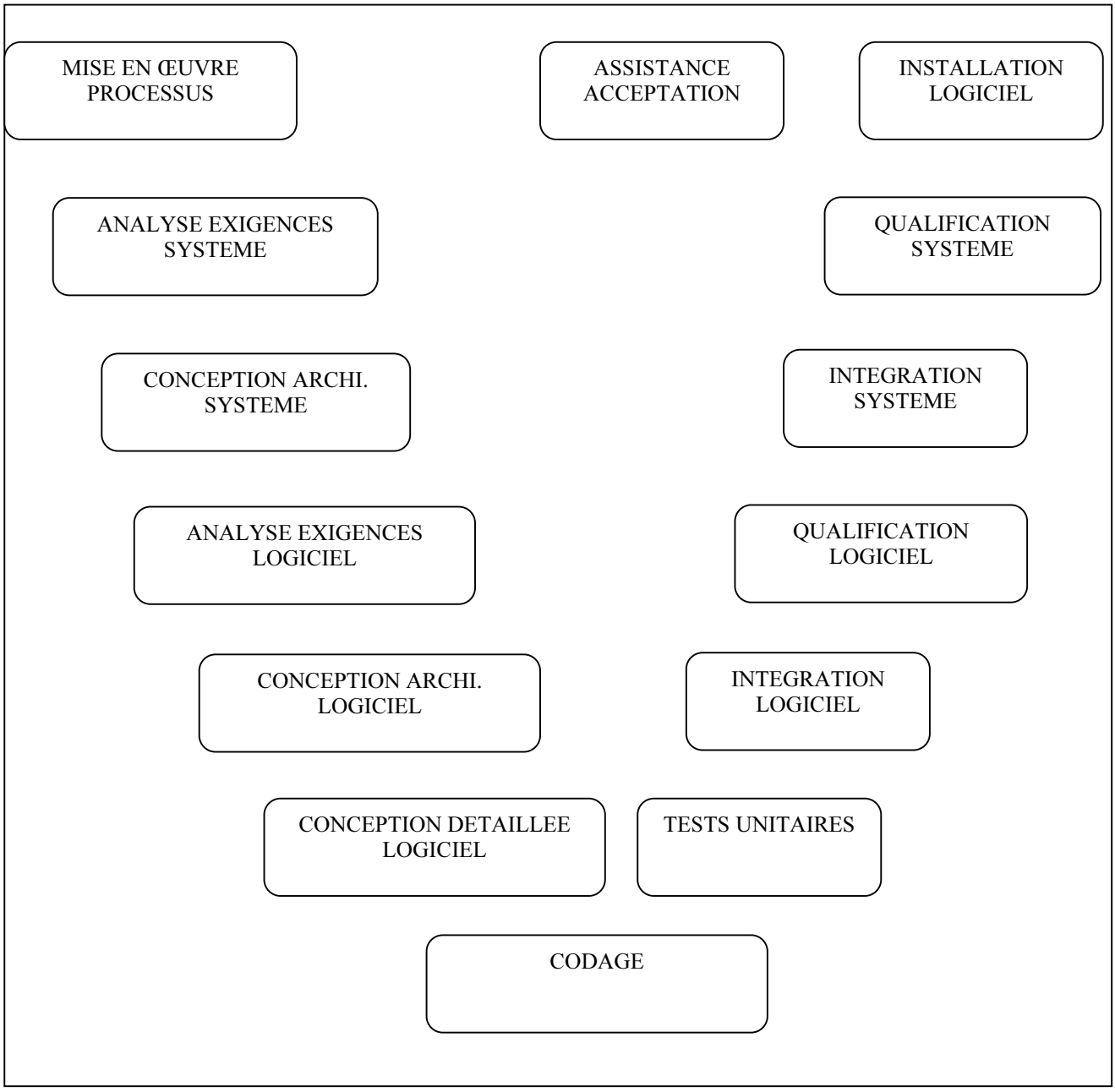
Le cycle en V standard comprend 7 activités successives. Le nombre et la définition des activités peuvent être adaptés à chaque projet en fonction du domaine d'application, de l'ampleur et la difficulté du projet, de l'expérience de l'équipe…

Par exemple :

- l'activité de qualification peut être supprimée pour le développement d'un prototype,
- la qualification peut être découpée en qualification usine et qualification sur site,
- ajout d'une activité d'évaluation d'outils,
- regroupement de la conception détaillée et du codage et essais…

A titre d'exemple d'adaptation du cycle en V, la norme ISO 12207 ajoute quelques activités :

ACTIVITES DU PROCESSUS DE DEVELOPPEMENT ISO 12207



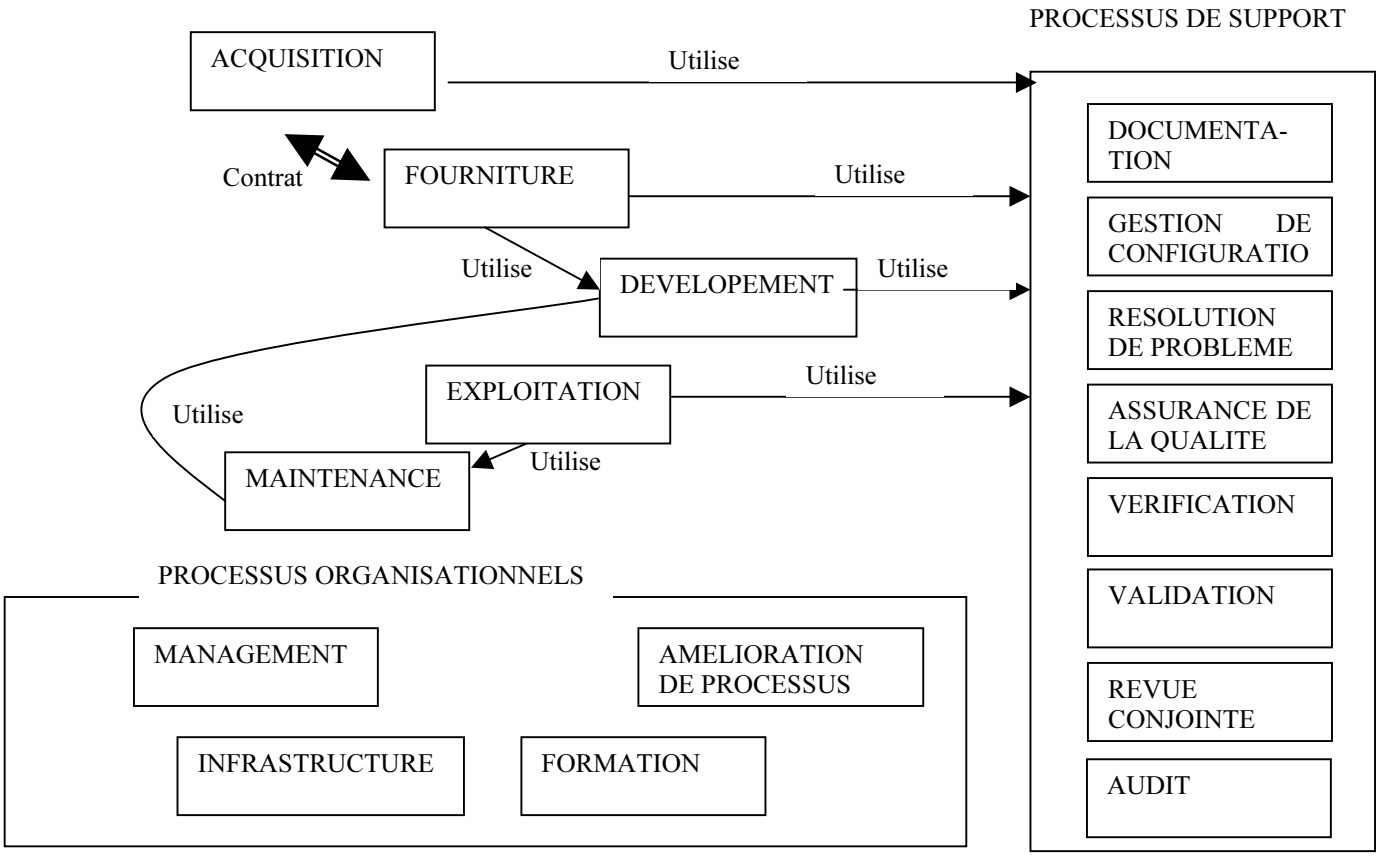
On voit que le cycle en V est adapté à tout le système : qu'il soit logiciel ou non.

EXTENSION AUX AUTRES ACTIVITES DU PROJET

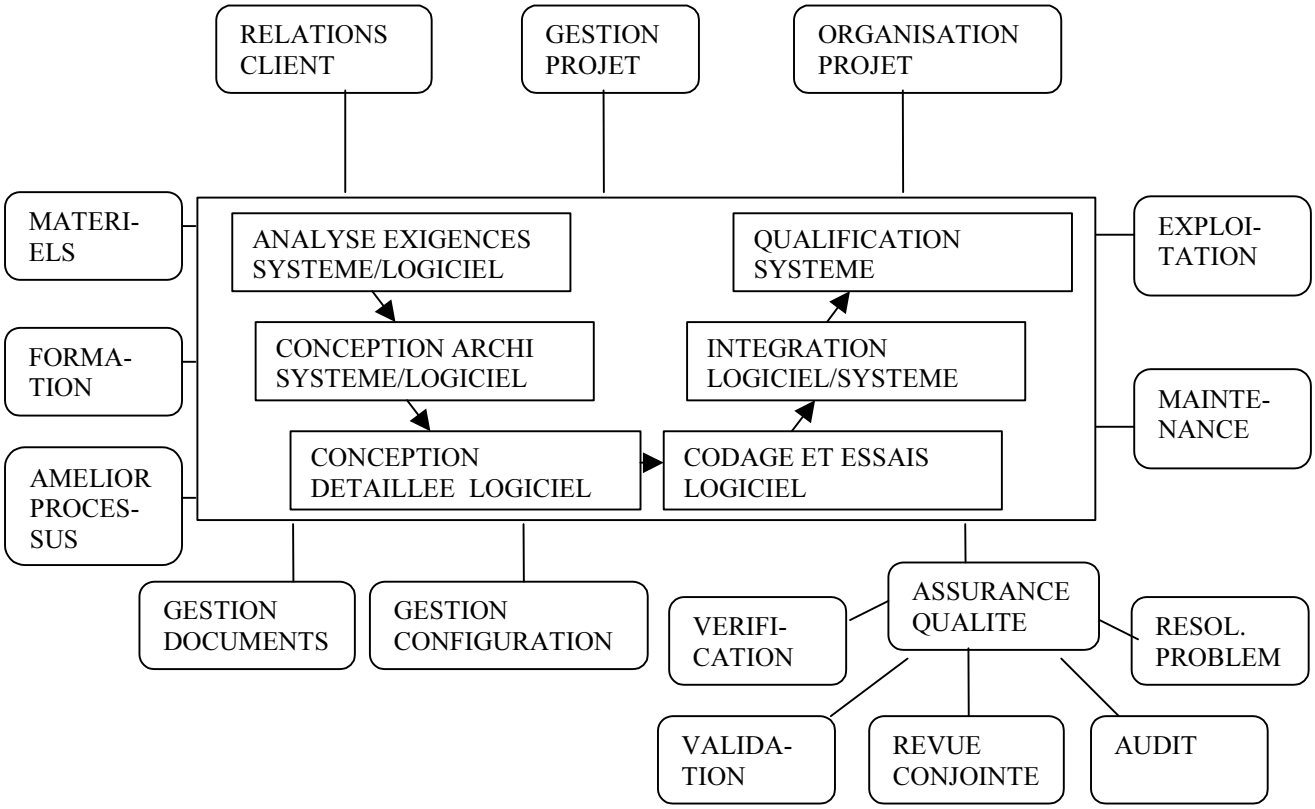
Le développement d'un logiciel est fait dans le cadre d'un projet. La réalisation du projet demande des activités supplémentaires par rapport au développement du logiciel pour organiser le projet, gérer le projet, gérer la configuration, gérer la documentation, assurer les relations avec le client, assurer la qualité.

La norme ISO 12207 a recensé 17 processus qui interviennent dans le développement et l'utilisation d'un logiciel :

- 5 processus généraux : acquisition, fourniture, développement, exploitation, maintenance,
- 4 processus organisationnels : management, infrastructure, formation, amélioration,
- 8 processus support : documentation, gestion de configuration, assurance qualité, vérification, validation, revue conjointe, audit, résolution de problème.



Ces processus induisent des activités qu'il faut prendre en compte dans le projet logiciel : relations avec le client, gestion de projet, organisation de projet…Ces activités sont représentées sur le dessin ci-dessous et devront être prises en compte dans l'identification et la planification des taches du projet.

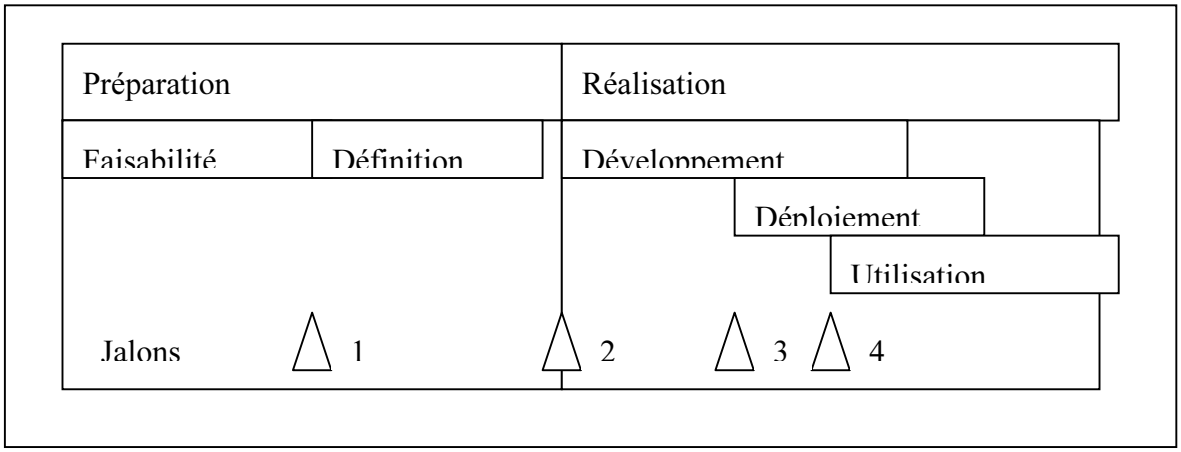

RELATIONS CLIENT
GESTION PROJET
ORGANISATION PROJET
MATERI-ELS
FORMA-TION
AMELIOR PROCES-SUS
ANALYSE EXIGENCES SYSTEME/LOGICIEL
CONCEPTION ARCHI SYSTEME/LOGICIEL
CONCEPTION DETAILLEE LOGICIEL
CODAGE ET ESSAIS LOGICIEL
INTEGRATION LOGICIEL/SYSTEME
QUALIFICATION SYSTEME
EXPLOI-TATION
MAINTE-NANCE
GESTION DOCUMENTS
GESTION CONFIGURATION
VERIFI-CATION
ASSURANCE QUALITE
RESOL. PROBLEM
VALIDA-TION
REVUE CONJOINTE
AUDIT

PHASAGE DU PROJET

La conduite d'un projet complexe demande de définir des phases et des jalons. Les activités relatives au logiciel doivent être réparties dans les phases.

On distingue les phases de préparation du projet (faisabilité et définition), et les phases de réalisation (développement, production ou déploiement, utilisation).

Les jalons sont des moments où on examine les résultats acquis, on fait des choix et on décide du lancement de nouvelles phases.



La phase de faisabilité est une phase exploratoire où on explore les solutions possibles pour satisfaire le besoin en évaluant l'adéquation au besoin, la faisabilité, le coût global (coût de développement et coût d'utilisation), les délais… A la fin de cette phase se situe le jalon choix du concept où il faut choisir le type de solution.

Dans la phase de définition, on prépare le développement : approfondissement de la solution retenue et rédaction du cahier des charges, planification du projet et établissement du budget, consultation des réalisateurs potentiels (qui devront faire une proposition technique et commerciale) et choix du réalisateur, négociation du contrat de développement. A la fin de cette phase se situe le jalon lancement du développement où on décide de lancer le développement et donc de notifier le contrat de développement (contrat interne ou contrat externe).

Dans la phase de développement, on développe le système et le logiciel. Ce développement peut comporter une ou plusieurs versions.

Lorsque la définition du système est suffisamment acquise, on place le jalon lancement de la production ou du déploiement .

Lorsqu'une première version du système a été qualifiée, on place le jalon qualification du système qui permet le début de la phase d'utilisation.

La phase de production ou déploiement consiste à produire les matériels (s'il s'agit de matériels spécifiques) ou les acheter, et installer les matériels et les logiciels.

La phase d'utilisation correspond à l'utilisation du système. Elle peut commencer lorsqu'une première version du système a été qualifiée et que le déploiement a été fait sur au moins un site.

MAITRISE DU DELAI DE DEVELOPPEMENT, PROCESSUS ITERATIF

1 Durée du développement

La productivité du programmeur étant limitée, un logiciel sera d'autant plus long à développer qu'il est plus gros. On peut certes réduire la durée du développement en augmentant la taille de l'équipe, mais la durée n'est pas réduite dans les mêmes proportions.

Des études ont été faites par B.W Boehm sur les coûts et délais de développement. Elles ont montré qu'il y avait un délai optimum de développement en fonction de la taille, ce délai est donné par la formule :

$$Dopt=C.KISL^{\alpha}$$ **(KISL kilo instructions source)**

C et α sont des coefficients à ajuster en fonction du type de logiciel. Un exemple d'application de cette formule conduit à 1,8 ans pour 100000 lignes et 3,5 ans pour 500000 lignes.

Ce temps est un temps optimal, on peut le réduire mais cela va augmenter le coût rapidement, et il y aura une limite car certaines activités (analyse des exigences, conception de l'architecture, intégration, qualification) sont peu parallélisables.

Aujourd'hui il n'est pas envisageable, si un client a besoin d'un logiciel, de lui demander un délai de 2 ou 3 ans pour lui fournir. Il faut donc avoir recours à des techniques particulières pour développer des gros logiciels et les fournir dans des délais compatibles avec les attentes des clients (un an au maximum). On pourra avoir recours au développement avec prototype, au développement itératif ou incrémental, ou à une combinaison des deux.

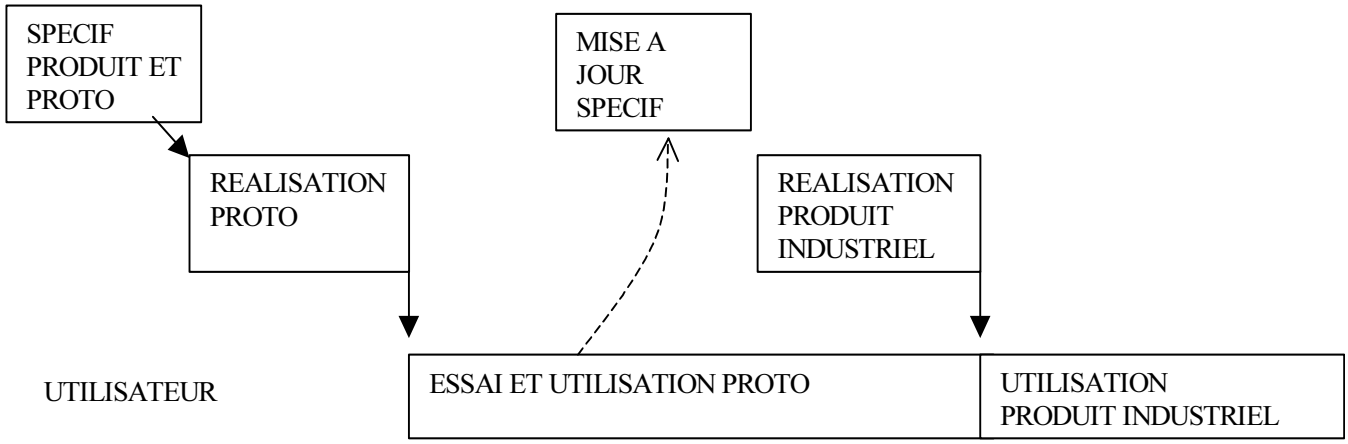
2 Développement avec prototype

Un prototype est une version du produit à réaliser dans laquelle on introduit des simplifications permettant une réalisation rapide :

- limitation aux principales fonctionnalités avec simplifications,
- pas de recherche de performances,
- allègement des validations (pas de qualification) et des dispositions qualité.

Le prototype peut ainsi être réalisé dans des délais relativement courts.

Il est livré au client qui pourra vérifier l'adéquation à son besoin, mettre au point ses procédures d'exploitation et commencer une exploitation provisoire. Pendant ce temps on peut mettre à jour l'expression du besoin et réaliser la version industrielle.



3 Développement itératif (ou incrémental)

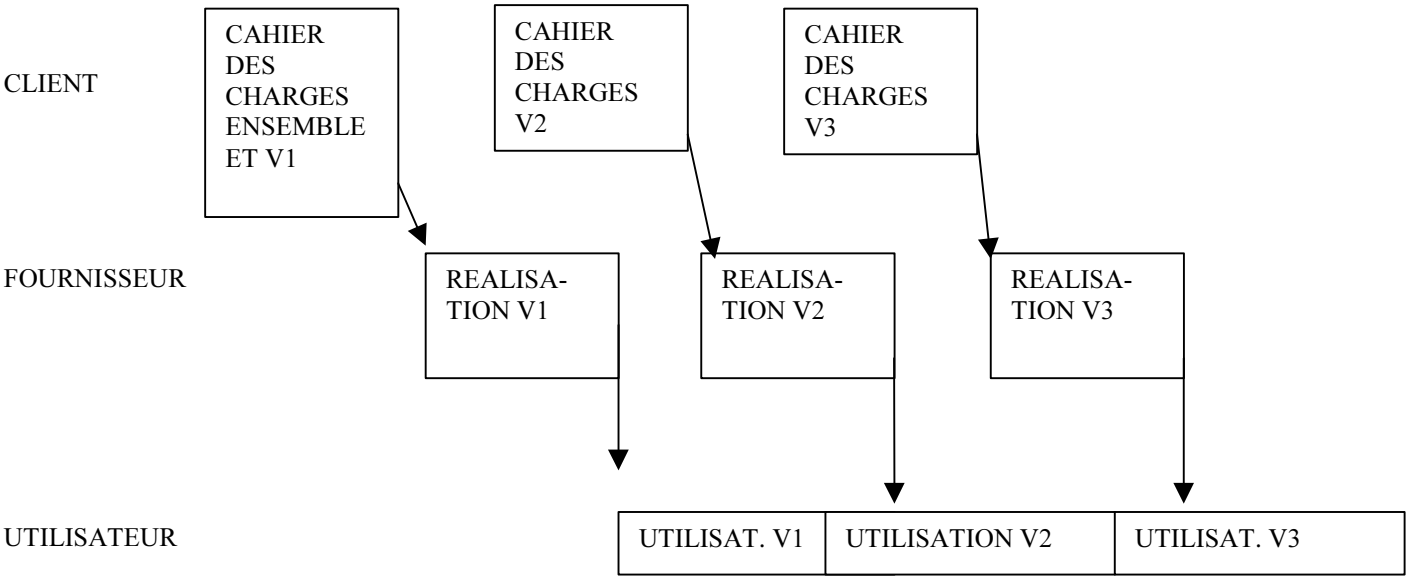
Le produit est développé par versions successives avec enrichissement des fonctionnalités dans chaque version.

Première technique

Une première technique de développement incrémental est de définir avec le client les fonctionnalités ou les nouvelles fonctionnalités de chaque version de façon à permettre un développement relativement court (12 à 18 mois au maximum).

Un premier produit peut donc être livré rapidement à l'utilisateur. Ce premier produit ne couvrira certes qu'une partie de son besoin, mais cela lui permettra de l'évaluer et de commencer l'exploitation. Il recevra par la suite des versions enrichies jusqu'à satisfaction complète du besoin. Ces nouvelles versions pourront prendre en compte le retour d'expérience de l'utilisation des premières versions.

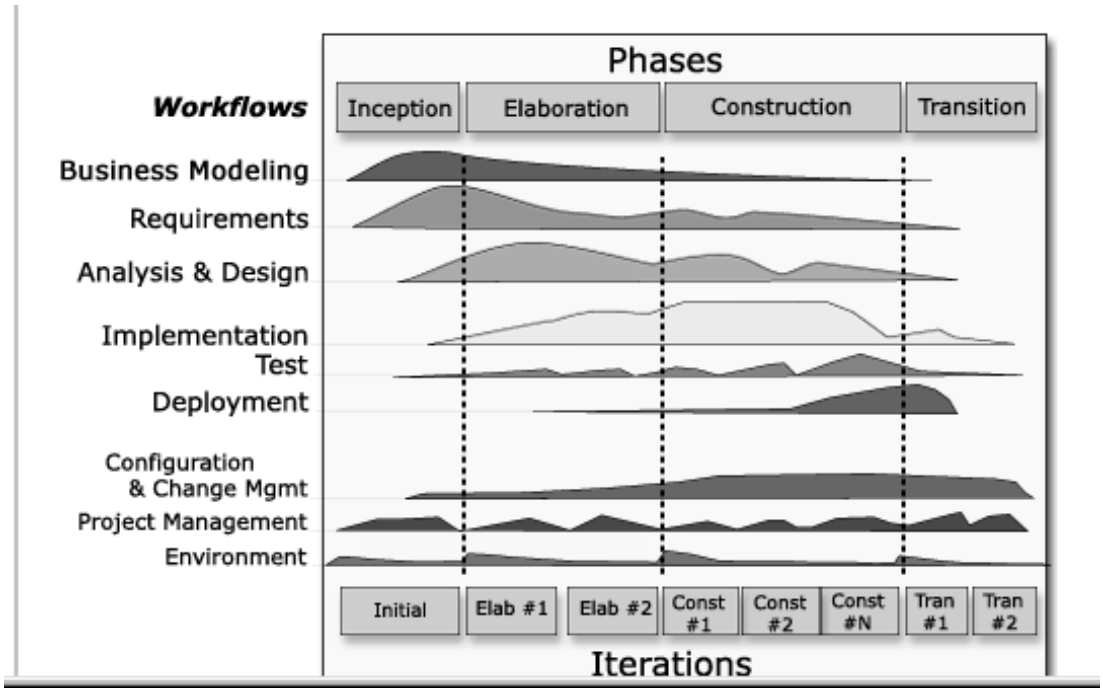
Un autre intérêt est , si certaines fonctionnalités sont peu mûres, de pouvoir les différer à des versions futures, et se donner le temps de les laisser mûrir.



Deuxième technique

Une deuxième technique est de réaliser des prototypes successifs en enrichissant à chaque fois les fonctionnalités jusqu'à ce que le produit soit complet.

Un exemple de cette technique itérative est le RUP (Rational Uniform Process) construit autour de la méthode de conception objet UML (Uniform Method Language) et commercialisé par Rational . Ce processus fait apparaître le phasage du projet, les activités du développement logiciel (appelées workflows), et le déroulement itératif du développement.



MAITRISE DES RISQUES VENANT DU CLIENT

1 Satisfaction du client

Le client a beaucoup de difficultés à exprimer son besoin réel, il faut cependant faire son possible pour lui livrer un produit qui lui donne satisfaction.

La première règle est de demander au client d'écrire son besoin (rédaction du cahier des charges). Cela le forcera à éclaircir ses idées ("ce que l'on conçoit bien s'énonce aisément") et cela fournira une référence pour le développement.

Pour maximiser les chances de satisfaction du client, un certain nombre de techniques sont recommandées :

- faire une revue des besoins (cahier des charges) avec le client pour vérifier qu'on a bien compris le besoin du client,
- réaliser des maquettes en début de développement, les présenter au client, et éventuellement lui livrer,
- fournir les documents en cours du développement au client en lui demandant son accord,
- organiser des revues avec le client aux principales étapes du développement (analyse des besoins, architecture, qualification…),
- faire un développement par étape (avec prototype ou itératif).

2 Maquettes

Les maquettes sont des logiciels provisoires destinés à montrer ou vérifier certains aspects du logiciel définitif. Elles ont en principe une utilisation très provisoire et ne sont pas réutilisées dans le logiciel définitif. Elles peuvent être réalisées avec des outils de maquettage tels que des générateurs d'interface homme machine.

Les maquettes sont un des outils privilégiés de dialogue avec le client, car elles lui permettent de constater de visu ce que sera le logiciel. Elles permettent en particulier de monter ce que seront les interfaces homme machine.

Les maquettes peuvent avoir d'autres utilisations telles que valider des aspects qui paraissent critiques par exemple les procédures d'accès au réseau, les procédures d'accès à la base de données…

Elles peuvent donc être réalisées pendant la préparation du projet pour aider à rédiger le cahier des charges ou la proposition technique et commerciale, et lors du développement pendant les activités d'analyse du besoin et de conception pour approfondir le besoin, valider les aspects critiques et dialoguer avec le client.

CONCLUSIONS

Le développement d'un logiciel d'une certaine taille n'est pas une chose facile.

Il faut avoir défini une méthode précisant

- le processus de développement (ou cycle de vie)
- l'organisation du travail en équipe
- la gestion de la configuration (corrections, modifications…)
- l'obtention de la qualité
- l'implication du client
- la documentation

Il n'y a pas de méthode universelle, la méthode doit être adaptée au type de projet et à l'expérience de l'équipe.

Si on ne dispose pas déjà d'une méthode, on pourra en construire une

- soit à partir de méthodes disponibles dans sa société, s'il y en a,
- soit à partir d'autres méthodes disponibles : MERISE, RUP, ISO 12207. Dans ce cas, on précisera les processus retenus.

Une des difficultés majeures du développement de gros logiciel réside dans le travail d'équipe.