CHAPITRE 5

PHASE 3 – ANALYSE SELON LES CRITÈRES D'ENSEMBLE ET LIENS AVEC LA NORME ISO/IEC12207

5.1 Introduction

La phase 2 a permis de diminuer le nombre de propositions de 308 à 35. À celles-ci, quatre propositions ont été ajoutées, suite à une reformulation mineure, pour un total de 39 propositions. Le tableau CXXII présente les propositions retenues qui serviront d'entrée à la phase 3.

Tableau CXXII

Liste des propositions retenues de la phase 2

1. Align incentives for developer and customer	24. Maintain clear accountability for results
2. Apply and use quantitative measurements in decision making	25. Produce software in a stepwise fashion
3. Build software so that it needs a short user manual	26. Quality is the top priority; long term productivity is a natural consequence of high quality
4. Build with and for reuse	27. Rotate people through product assurance
5. Communicate with customers/users	28. Since change is inherent to software, plan for it and manage it
6. Define software artifacts rigorously	29. Since tradeoffs are inherent to software engineering, make them explicit and document it
7. Design for change	30. Strive to have a peer, rather than a customer, find a defect
8. Design for errors	31. Tailor cost estimation methods
9. Design for maintenance	32. To improve design, study previous solutions to similar problems
10. Determine requirements now	33. Use better and fewer people
11. Don't overstrain your hardware	34. Use documentation standards
12. Don't test your own software	35. Write programs for people first
13. Don't try to retrofit quality	
14. Don't write your own test plans	
15. Establish a software process that provides flexibility	
16. Fix requirements specification error now	
17. Give product to customers early	

Tableau CXXII (suite)

18. Give software tools to good engineers 19. Grow systems incrementally 20. Implement a disciplined approach and improve it continuously 21. Invest in the understanding of the problem 22. Involve the customer 23. Keep design under intellectual control	36. Know software engineering's techniques before using development tools 37. Select tests based on the likelihood that they will find faults 38. Choose a programming language according to maintainability 39. In face of unstructured code, rethink the module and redesign it from scratch.

5.2 Objectifs de la phase 3

Lors de la deuxième phase, nous avons appliqué les cinq critères individuels pour chacune des 308 propositions. La troisième phase a pour objectif premier de classer les propositions selon les catégories : processus, produit et individu. Le deuxième objectif est d'appliquer les deux critères d'ensemble aux 39 propositions retenues. Par la suite, les propositions de catégories processus seront liés aux processus de la norme ISO/IEC12207.

5.3 Méthode utilisée

Les 39 propositions issues de la phase 2 sont le principal intrant de la phase 3. À celles-ci s'ajoutent les deux critères d'ensemble. Les critères d'ensemble ne peuvent s'appliquer aux propositions prises individuellement. Ces critères doivent être appliqués sur l'ensemble des propositions. Afin d'obtenir une vision structurée de l'ensemble des propositions à analyser, les propositions seront catégorisées en fonction des trois catégories identifiées : processus, produit et individu. Ces catégories seront, au préalable, définies pour éviter les ambiguïtés. Une proposition peut être rattachée à plus d'une catégorie, le cas échéant. Ces regroupements faciliteront par la suite l'application des critères d'ensemble. Compte tenu que plusieurs propositions sont de catégorie

processus , l'utilisation de la classification des processus offerte par la norme ISO/IEC 12207 permettra de raffiner le classement des propositions de cette catégorie.

La figure 27 présente les principaux éléments de la méthode de recherche suivie pour la phase 3 de l'évaluation des propositions.

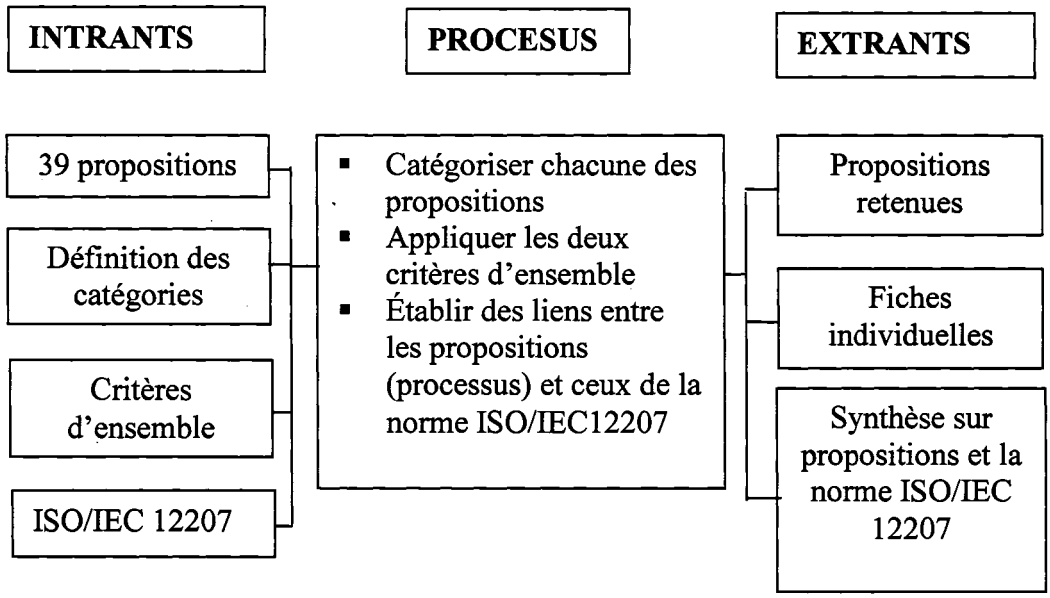

Figure 27 Méthode de recherche pour l'évaluation des propositions - phase 3

5.4 Définitions des catégories

Les catégories servant de base à l'analyse sont identifiées à la figure 28.

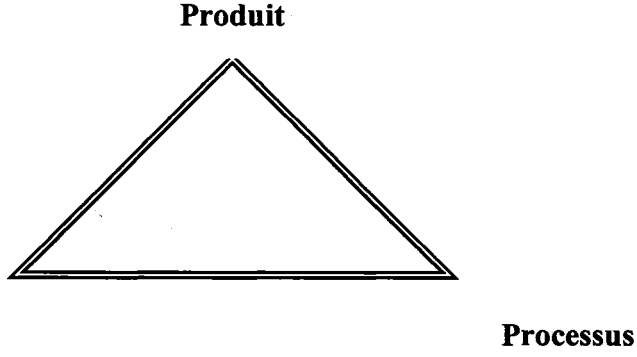

Figure 28 Catégories de principes

Ces catégories ont déjà été identifiées dans le chapitre sur les concepts à la base du génie. Chacune des catégories est maintenant définie.

5.4.1 Catégorie Produit

La norme ISO/IEC 12207 définit le terme produit (logiciel) comme suit : « *Set of computer programs, procedures, and possibly associated documentation and data* » (p.4).

Ghezzi et al. (2003) soulignent que le terme produit représente généralement ce qui est livré au client (« *end product* »), ce qui inclut le code exécutable et le guide de l'utilisateur. Cependant, ils précisent qu'il y a aussi des produits intermédiaires nommés « *work products* » conçus tout au long du processus de développement. Les produits intermédiaires comprennent, entre autres, le document des exigences logicielles, de conception, de tests, d'assurance qualité et de projet. Selon ces auteurs, les produits intermédiaires sont assujettis aux mêmes standards de qualité que les produits livrables au client. La gestion de configuration permet de maintenir une cohésion entre les différents produits intermédiaires et une version du produit final.

Nous considérons donc que le produit englobe tant le produit final (le logiciel) que les produits intermédiaires issus du processus du cycle de vie du logiciel. Ainsi, tout ce qui est sujet à la gestion de configuration sera considéré comme étant de catégorie produit.

5.4.2 Catégorie processus

Le Project Management Institute définit le terme processus comme suit : « *A series of actions bringing about a result* » (IEEE1490-2003). La norme ISO/IEC 12207 définit le terme processus comme étant : « *A set of interrelated activities which transform inputs into outputs* ». De plus, cette norme précise que les activités incluent l'utilisation de ressources pour transformer les intrants en extrants.

Comme les processus du génie logiciel sont nombreux et variés, il n'est pas suffisant, pour notre analyse, d'identifier qu'une proposition est de catégorie processus. Un niveau de granularité plus fin est requis. À cet effet, la structure de la norme ISO/IEC 12207 est intéressante. D'une part, elle offre un niveau de granularité permettant de préciser à quelle catégorie de processus se rattache la proposition. D'autre part, la norme est reconnue internationalement ; ainsi, on peut s'appuyer sur ses bases pour effectuer le raffinement des propositions de catégorie processus. Ce raffinement permettra de constater si plusieurs propositions couvrent, à titre d'exemple, les mêmes thèmes. Ainsi, il sera possible d'identifier si des propositions sont déduites ou si elles se contredisent. À la suite de la phase 3, il nous sera possible de constater le degré de couverture des propositions retenues selon la structure de la norme ISO/IEC 12207 et d'en tirer des conclusions.

La norme ISO/IEC 12207 présente les processus du cycle de vie utilisés pour acquérir un logiciel, le développer, le fournir à un client, l'exploiter et le maintenir. La norme identifie 17 processus principaux couvrant l'ensemble du cycle de vie du logiciel. Ces

processus sont regroupés sous trois catégories : processus primaires, de soutien et organisationnels.

Le tableau CXXIII présente les différentes catégories de processus du cycle de vie (ISO/IEC 12207).

Tableau CXXIII

Catégories de processus ISO/IEC 12207 (traduction Séguin)

Processus du cycle de vie ISO/IEC 12207		
Primaires	**De soutien**	**Organisationnels**
1. Acquisition 2. Approvisionnement 3. Développement 4. Exploitation 5. Maintenance	1. Documentation 2. Gestion des configurations 3. Assurance qualité 4. Vérification 5. Validation 6. Revue 7. Audit 8. Résolution des problèmes	1. Management 2. Infrastructure 3. Amélioration 4. Formation

Les processus primaires s'appliquent à l'entité (personne ou une organisation) qui procède au développement, à l'exploitation ou à la maintenance de produits logiciels. L'entité peut représenter un fournisseur, un acquéreur, un développeur, un opérateur ou un mainteneur (ISO/IEC12207). Le tableau CXXIV présente les cinq processus primaires.

Tableau CXXIV

Description des processus primaires ISO/IEC 12207 (adaptation Seguin)

Processus primaires	**Description et activités**	
1. Acquisition	Processus d'acquisition de produits logiciels (système, produit, service)	
	▪ Appel d'offre ▪ Préparation du contrat	▪ Suivi des fournisseurs ▪ Acceptation
2. Approvisionnement	Processus visant à fournir un produit ou un service logiciels	
	▪ Préparation d'appel d'offre ▪ Contrat ▪ Planification	▪ Exécution et suivi ▪ Revue et évaluation ▪ Livraison
3. Développement	Processus pour la définition et le développement de produits logiciels	
	▪ Analyse des exigences du système ▪ Architecture du système ▪ Analyse des exigences logicielles ▪ Conception architecturale du logiciel ▪ Conception détaillée ▪ Programmation et tests	▪ Intégration logicielle ▪ Test de qualification de logiciel ▪ Intégration système ▪ Test de qualification du système ▪ Installation du logiciel ▪ Acceptation du logiciel
4. Exploitation	Processus pour l'exploitation d'un système logiciel dans un environnement réel	
	▪ Test opérationnel ▪ Opération du système	▪ Soutien aux utilisateurs
5. Maintenance	Processus pour l'entretien du logiciel incluant aussi la migration et le retrait du logiciel	
	▪ Analyse des problèmes et des modifications ▪ Mise en place des modifications	▪ Revue et acceptation ▪ Migration ▪ Retrait du logiciel

Les processus de soutien viennent appuyer les processus primaires dans le but de favoriser le succès et la qualité du projet logiciel. Un processus de soutien est utilisé et exécuté par un autre processus. Le tableau CXXV présente les huit processus de soutien identifiés par la norme ISO/IEC12207.

Tableau CXXV

Description des processus de soutien ISO/IEC12207 (adaptation Seguin)

Processus de soutien	**Description et activités**	
1. Documentation	Processus régissant la documentation de l'information générée par les processus du cycle de vie	
	▪ Conception et développement	▪ Rédaction ▪ Mise à jour
2. Gestion de configuration	Processus de gestion de configuration des produits intermédiaires et finaux	
	▪ Identification des éléments ▪ Contrôle de la configuration	▪ Suivi de la configuration ▪ Évaluation de la configuration ▪ Gestion des versions et des livraisons
3. Assurance qualité	Processus qui assure que les produits et les processus sont conformes aux exigences et aux plans	
	▪ Conformité du produit ▪ Conformité du processus	▪ Conformité du système de qualité
4. Vérification	Processus déterminant si les produits issus d'une activité sont conformes aux exigences ou aux conditions imposées dans les activités antérieures	
	▪ Contrats ▪ Processus	▪ Exigences, Design ▪ Code, intégration et documentation
5. Validation	Processus permettant de valider si le produit final correspond aux attentes et aux exigences initiales	
	▪ Cas de tests ▪ Conformité des tests avec les exigences ▪ Effectuer les tests	▪ Valider la conformité du logiciel ▪ Tester le logiciel dans son environnement
6. Revue	Processus permettant d'évaluer l'état d'une activité et des produits qui en découlent	
	▪ Revues de gestion de projet	▪ Revues techniques

Tableau CXXV (suite)

Processus de soutien	**Description et activités**	
7. Audit	Processus permettant de déterminer la conformité des exigences, des plans et des contrats	
	▪ Revue de suivi de projet	▪ Revues techniques
8. Résolution de problème	Processus d'analyse et de résolution des problèmes découverts en cours de projet	
	▪ Description du problème ▪ Analyse du problème	▪ Solution au problème ▪ Mise en place

Les processus organisationnels permettent d'organiser et d'améliorer les autres processus. Les processus organisationnels sont au nombre de quatre, tels que présentés au tableau CXXVI.

Tableau CXXVI

Description des processus organisationnels ISO/IEC12207 (Adaption Séguin)

Processus organisationnels	**Description et activités**	
1. Management	Processus regroupant les activités de gestion du produit, du projet et des tâches	
	▪ Planification ▪ Déroulement et suivi	▪ Revue et évaluation ▪ Fin du projet
2. Infrastructure	Processus permettant d'établir et de maintenir l'infrastructure nécessaire au projet. L'infrastructure peut comprendre le matériel, le logiciel, les outils, les techniques, les normes, les aménagements pour le développement, l'opération et la maintenance.	
	▪ Mise en place de l'infrastructure	▪ Mise à jour de l'infrastructure
3. Amélioration	Processus qui permet d'établir, d'évaluer, de mesurer, de contrôler et d'améliorer les processus du cycle de vie	
	▪ Évaluation	▪ Amélioration

Tableau CXXVI (suite)

Processus organisationnels	**Description et activités**	
4. Formation	Processus qui permet de former le personnel et de maintenir leurs connaissances à jour	
	▪ Conception du matériel de formation	▪ Mise en place du plan de formation

5.4.3 Catégorie Individus

Wiegers (1996) affirme que les individus impliqués dans un projet logiciel ont un impact certain sur le succès du projet et sur la qualité du produit. Même si les processus sont clairement définis, ce sont les individus qui vont les interpréter et les mettre en œuvre.

Le PMBOK (2004) identifie des catégories d'intervenants à un projet. L'utilisation des catégories du PMBOK est pertinente puisque que SWEBOK se réfère au PMBOK pour tous les sujets concernant la gestion de projet qui ne sont pas traités au niveau de SWEBOK. La figure 29 présente les catégories d'intervenants tirées du PMBOK.

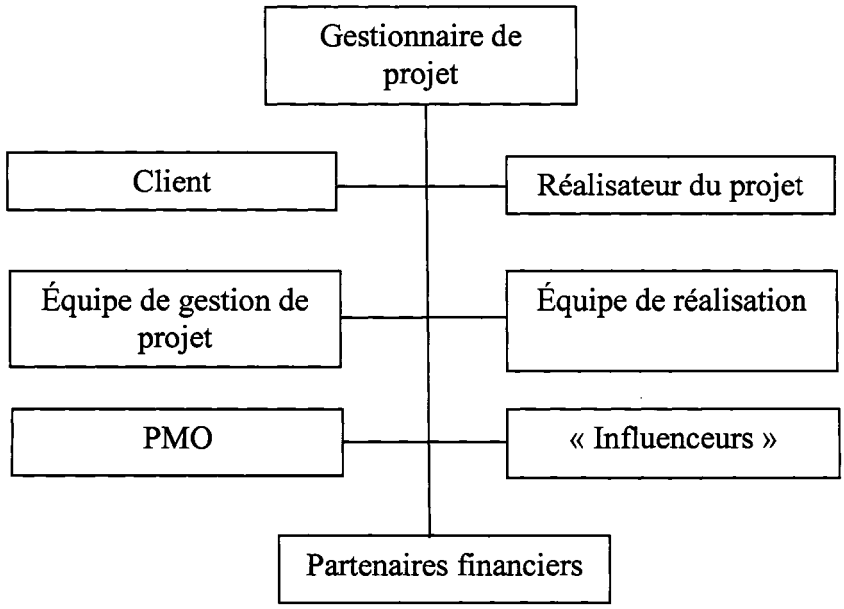

Figure 29 Catégories d'intervenants dans un projet PMBOK 2004 (adaptation Séguin)

Le gestionnaire de projet est l'individu responsable de la gestion du projet. Cette catégorie est explicitement identifiée puisque le responsable du projet n'est pas nécessairement un individu appartenant à la catégorie client ou à l'organisation qui réalise le projet. La catégorie « Client » représente les individus ou l'organisation qui utilisent le produit développé. Pour notre étude, les individus tels les utilisateurs directs du logiciel sont principalement ciblés. La catégorie « réalisateur » représente l'organisation dont les employés vont réaliser le projet. Comme il s'agit d'une entité organisationnelle, cette catégorie ne sera pas retenue, au profit de la catégorie « équipe de réalisation ». La catégorie « équipe de réalisation » comprend les individus qui feront le développement du produit. L'équipe de réalisation peut aussi être un groupe interne du client qui fera le développement du produit, tel un service de l'informatique. Cette catégorie comprend divers types d'individus, tels, entre autres, les analystes ou ingénieurs, les techniciens, les administrateurs de base de données. La catégorie « équipe de gestion de projet » représente les individus qui sont directement impliqués dans la gestion du projet. La catégorie « partenaires financiers » (Sponsor) représente les individus ou les organisations qui contribuent au soutien financier du projet. Pour notre étude, seul le volet individu sera considéré. La catégorie des « influenceurs » représente les individus ou les groupes appartenant soit au groupe client ou au groupe réalisateur qui ne sont pas directement liés au produit, mais qui peuvent influencer positivement ou négativement le déroulement du projet. La catégorie Project Management Office (PMO) représente une entité organisationnelle responsable de la gestion des différents projets. Le PMBOK souligne que le PMO peut être un intervenant direct dans le projet s'il a une influence sur le résultat du projet. Comme le PMO est une entité organisationnelle et non un individu, cette catégorie ne sera pas retenue.

Les catégories retenues du PMBOK serviront à identifier le rattachement d'un principe de catégorie individu.

5.5 Critères d'ensemble

Au chapitre 3, nous avons établi sept critères d'identification des principes: cinq critères d'identification individuels et deux critères d'identification d'ensemble. La phase 2 a appliqué les critères individuels sur les 308 propositions. Les deux critères d'ensemble seront maintenant appliqués sur l'ensemble des 39 propositions retenues suite à la phase 2. Les deux critères d'ensemble sont les suivants :

1. Les principes devraient être indépendants (non déduits) (Boehm 1983)
La définition du terme principe souligne qu'un principe est une proposition non-déduite. Ainsi, ce critère permettra de mettre en force ce volet de la définition retenue pour l'étude. Si une proposition est déduite d'une autre proposition, alors la proposition ne peut pas être considérée comme fondamentale.

2. Un principe ne doit pas contredire un autre principe connu. (Bourque et al. 2002)
Ce critère permettra d'écarter des propositions qui seraient en contradiction. En cas de contradiction entre deux propositions, celles-ci seront isolées et un arbitrage devra être fait.

Pour chacune des trois catégories (individus, produit et processus), un tableau sera produit présentant la concordance ou non de chacune des propositions selon les deux critères. Le tableau CXXVII présente le format type qui sera utilisé.

Tableau CXXVII

Format type de présentation de concordance entre les propositions et les critères

No.	Proposition	Critère 1	Critère 2
1	Align incentives for developer and customer	√	√
5	Communicate with customers/users		√
18	Give software tools to good engineers		√

Le premier élément est le numéro de la proposition. Un numéro a été assigné à chacune des 39 propositions retenues à la phase 2. Ce numéro fait référence au tableau 1 présenté au début de ce chapitre. La proposition est par la suite retranscrite pour faciliter la lecture du tableau. Il est à noter que si la proposition originale d'un auteur a été reformulée, c'est la reformulation qui sera utilisée à la phase 3. Les deux colonnes suivantes indiquent si les critères sont satisfaits ou non. Le critère no. 1 représente l'indépendance du principe et le critère no. 2 représente le fait que le principe n'en contredit pas un autre. Le « √ » indique que la proposition satisfait le critère. La case de couleur rouge indique que le critère n'est pas satisfait. Dans le cas du critère no.1, en plus de la couleur rouge, le numéro de la proposition « parent » est indiqué. Ainsi, la proposition évaluée est alors déduite de la proposition « parent ».

5.6 Résultat de la phase 3

Dans un premier temps les 39 propositions ont été classées selon les catégories: produit, individu et processus. Ainsi, la catégorie individus recueille six propositions, la catégorie produit sept et 36 propositions pour la catégorie processus, tel que présenté au tableau CXXVIII.

Tableau CXXVIII

Ventilation des propositions selon les catégories

Catégories	**Nombre de propositions**
Individu	6
Produit	7
Processus	36

Le nombre total de propositions est supérieur à 39 compte tenu que certaines se retrouvent dans plus d'une catégorie. Dans les prochaines sections, nous présenterons pour chacune des catégories les résultats de l'analyse.

5.6.1 Catégorie individu

La catégorie individus regroupe six propositions tel que présenté au tableau CXXIX.

Tableau CXXIX

Liste des propositions rattachées à la catégorie Individu

No.	Proposition	Critère 1	Critère 2
1	Align incentives for developer and customer	√	√
5	Communicate with customers/users		√
18	Give software tools to good engineers		√
26	Quality is the top priority; long term productivity is a natural consequence of high quality	√	√
33	Use better and fewer people	√	√
36	Know software engineering techniques before using development tools	√	√

Proposition 1: *Align incentives for developer and customer*

La proposition no. 1, selon le sens donné par Davis 1995, suggère que les développeurs soient gratifiés ou pénalisés selon l'atteinte ou non des objectifs. L'explication donnée par l'auteur vise essentiellement les développeurs, malgré la présence du terme client dans la proposition. Cette proposition implique les individus du groupe "équipe de réalisation". La proposition satisfait les deux critères, elle est donc retenue.

Proposition 5: *Communicate with customers/users*

La proposition no. 5 suggère d'assurer une bonne communication avec le client. Cette proposition peut être déduite de la proposition no. 22 ("Involve the customer") qui est plus générale. Cette dernière étant plus générale peut être considérée comme plus fondamentale que la proposition no. 5. La proposition no. 5 implique les individus des groupes client, gestionnaire de projet et équipe de réalisation. La proposition no. 5 n'ajoute pas d'éléments nouveaux ou de précisions par rapport à la proposition père. La proposition n'est donc pas retenue.

Proposition 18: *Give software tools to good engineers*

La proposition no. 18 souligne que les outils de développement (ex : CASE) devraient êtres fournies aux développeurs compétents. Davis (1995) souligne qu'un individu ne maitrisant pas les techniques de base du génie logiciel ne peut tirer profit des avantages de ces outils, ni même les utiliser adéquatement. Cette proposition peut être déduite de la proposition plus générale no. 36 ("Technique before tools"). Cette dernière a été reformulée comme suit à la phase 1 : "Know software engineering techniques before using development tools". La proposition no. 18 n'ajoute pas de précision supplémentaire par rapport à la proposition no.36. La proposition no. 18 n'est donc pas retenue.

Proposition 26: *Quality is the top priority; long term productivity is a natural consequence of high quality*

Wiegers (1996) souligne que l'attitude des développeurs et des gestionnaires envers la qualité a un impact certain sur la qualité des processus et du produit final. L'attitude des individus envers la qualité est un élément déterminant que même les processus de qualité ne peuvent substituer. La proposition n'est pas déduite, ni en contradiction avec d'autres. Elle est donc retenue.

Proposition 33: *Use better and fewer people*

La proposition no. 33 se retrouve dans deux catégories : individu et processus. Au niveau de la catégorie individu, la proposition est un guide pour le gestionnaire du projet lorsqu'il fait l'allocation des ressources humaines d'un projet. La proposition satisfait les deux critères et elle est donc retenue.

Proposition 36: *Know software engineering techniques before using development tools*

La proposition no. 36 vise principalement les individus de l'équipe de réalisation qui doivent maitriser les techniques de base du génie logiciel avant d'utiliser les outils de développement. Ces individus pourront ainsi mieux tirer profit des possibilités des outils de développement. La proposition satisfait les deux critères et elle est donc retenue.

5.6.1.1 Sommaire de la catégorie individu

Nous constatons qu'aucune des propositions de la catégorie individu n'est en contradiction avec une autre proposition. Les propositions no.5 et no.18 sont déduites de propositions plus générales. Des six propositions de la catégorie individu, quatre sont retenues. Le tableau CXXX présente celles-ci.

Tableau CXXX

Propositions retenues de la catégorie Individu

No.	**Propositions**
1.	Align incentives for developer and customer
26.	Quality is the top priority; long term productivity is a natural consequence of high quality
33.	Use better and fewer people
36.	Know software engineering's techniques before using development tools

Le tableau CXXXI présente une ventilation des propositions de la catégorie individu en fonction des groupes du PMBOK.

Tableau CXXXI

Ventilation des propositions selon les groupes d'intervenants du PMBOK

Groupes d'intervenants	**Propositions associées**
Client	Aucune
Gestionnaire de projet	#26 et #33
Équipe de gestion de projet	Aucune
Équipe de réalisation	#1, #26, et #36
PMO	Aucune
Influenceurs	Aucune
Partenaires	Aucune

Nous constatons que seulement deux groupes sur les sept retenus sont associés à des propositions de la catégorie individus. Les groupes client, équipe de projet, PMO, influenceurs et partenaires sont vacants. Seuls les groupes gestionnaire de projet et équipe de réalisation sont associés à des propositions. Nous constatons que les propositions visent en premier les développeurs et par la suite le gestionnaire du projet. Tous les autres groupes d'intervenants ne sont pas associés aux propositions retenues de la catégorie individu. Nous soulignons que le groupe client reste vacant, groupe ayant pourtant une importance certaine dans un projet logiciel.

5.6.2 Catégorie produit

La catégorie produit regroupe sept propositions, tel que présenté au tableau CXXXII. Ces propositions sont associées aux produits intermédiaires du processus de développement ou au produit final.

Tableau CXXXII

Résultat de l'application des critères aux propositions de la catégorie produit

No.	Propositions	Critère 1	Critère 2	Type produit
3.	Build software so that it needs a short user manual	√	√	Final
4.	Build with and for reuse	√	√	Final
6.	Define software artifacts rigorously	√	√	Intermédiaire
11.	Don't overstrain your hardware	√	√	Final
34.	Use documentation standards		√	Intermédiaire
35.	Write programs for people first	√	√	Final
38.	Choose a programming language to assure maintainability	√	√	Final

Proposition 3: *Build software so that it needs a short user manual*

La proposition no. 3 concerne le produit final et plus précisément la facilité d'utilisation du logiciel. Davis (1995) affirme que plus le manuel d'utilisation est petit, meilleur est le logiciel. Nous considérons que cette affirmation est nettement exagérée puisqu'il y a de nombreuses caractéristiques de qualité du logiciel (ISO9126) qui ne sont pas visibles de l'extérieur, ni à l'utilisation. Néanmoins, la proposition satisfait les deux critères et elle est retenue.

Proposition 4: *Build with and for reuse*

La proposition no. 4 (Bourque et al. 2002) n'est pas commentée par les auteurs. Nous interprétons donc cette proposition à l'effet de construire le logiciel en réutilisant des composants existants. Ces composants doivent être conçus pour favoriser la réutilisation. La proposition est directement liée au logiciel, donc au produit final. Cette proposition satisfait les deux critères et elle est donc retenue.

Proposition 6: *Define software artifacts rigorously*

La proposition no. 6 (Bourque et al. 2002) n'est également pas commentée par les auteurs. Le terme central de la proposition est « artifact ». Nous considérons que les artéfacts représentent tous les produits intermédiaires générés par le processus de développement. Les produits intermédiaires doivent être définis rigoureusement. Cette proposition satisfait les deux critères et elle est donc retenue.

Proposition 11: *Don't overstrain your hardware*

La proposition no. 11 souligne que si le matériel est surchargé par le logiciel, il y aura beaucoup d'efforts de consentis pour ajuster le logiciel aux contraintes du matériel. Ces efforts auront un impact sur les coûts et les délais de développement. La proposition concerne le produit final, le logiciel. Cette proposition satisfait les deux critères et elle est donc retenue.

Proposition 34: *Use documentation standards*

La proposition no. 34 souligne l'utilisation de normes de documentation. Ces normes peuvent s'appliquer au produit final (manuel d'utilisation) et aux différents produits intermédiaires. La proposition est déduite de la proposition no. 20 plus générale qui stipule qu'il faut implémenter un processus discipliné et de l'améliorer constamment. L'utilisation de normes est des éléments d'une approche disciplinée. Malgré que la proposition puisse être déduite de la proposition no. 20, elle ajoute une précision que n'offre pas la proposition parent, à l'effet d'utiliser concrètement des normes de documentation. Également, l'application de la proposition permet de mieux soutenir la proposition no. 6, à l'effet de définir rigoureusement les artéfacts du logiciel. Cependant, on ne peut conclure que la proposition no. 6 est déduite de la no. 34, car celle-ci se

concentre spécifiquement sur les normes de documentation. D'autres normes peuvent être utilisées pour soutenir la proposition no. 6. La proposition no. 34 est donc retenue.

Proposition 35: *Write programs for people first*

La proposition no. 35 souligne l'importance de la lisibilité du code afin que d'autres personnes puissent, par la suite, facilement le lire et le modifier. Les programmes sont des produits et la lisibilité de ceux-ci est une caractéristique du produit final. Cette proposition satisfait les deux critères et elle est donc retenue.

Proposition 38: *Choose a programming language according to maintainability*

La proposition no. 38, « Language affect maintability » a été reformulée à la phase 2 de l'analyse comme suit : « Choose a programming language to assure maintainability». Selon Davis (1995), les langages ne sont pas tous égaux au niveau de la facilité de maintenance des programmes. Ainsi, lors du choix d'un langage de programmation, les considérations de maintenance devraient être prises en compte. Cette proposition se classe sous deux catégories. Le choix comme tel du langage est une étape dans le processus de développement tandis que la facilité de maintenance d'un logiciel est une des caractéristiques du produit final. Cette proposition satisfait les deux critères et elle est donc retenue.

5.6.2.1 Sommaire de la catégorie produit

Suite à l'application des critères d'ensemble sur les propositions de la catégorie produit, toutes les propositions sont retenues tel que présenté au tableau CXXXIII.

Tableau CXXXIII

Propositions retenues de la catégorie produit

No.	Propositions
3	Build software so that it needs a short user manual
4	Build with and for reuse
6	Define software artifacts rigorously
11	Don't overstrain your hardware
34	Use documentation standards
35	Write programs for people first
38	Choose a programming language to assure maintainability

Une seule proposition (no. 34) peut être déduite d'une autre. Cependant, la formulation de la proposition no. 34 ajoute une précision permettant de mieux l'appliquer en guidant une action précise. La proposition no. 34 a donc été conservée, malgré une déduction possible.

Le tableau CXXXIV présente la répartition des propositions en fonction du type de produit.

Tableau CXXXIV

Ventilation des propositions de catégorie produit

Type de produit	Propositions
Intermédiaire	No.6 et 34
Final	No.3, 4, 11, 35 et 38

Nous constatons que les propositions retenues visent principalement le produit final (cinq sur sept) et que seulement deux propositions sont associées aux produits intermédiaires. Nous soulignons qu'il y a possiblement une carence de propositions concernant les produits intermédiaires qui ont une grande importance dans le déroulement du processus de développement.

5.6.3 Catégorie processus

La catégorie processus regroupe la majorité des propositions de cette étude, soit 36 sur 39 propositions. Le tableau CXXXV présente la liste des propositions classées dans cette catégorie.

Tableau CXXXV

Résultat de l'application des critères aux propositions de la catégorie processus

No.	Proposition	Critère 1	Critère 2
1	Align incentives for developer and customer	√	√
2	Apply and use quantitative measurements in decision making	√	√
3	Build software so that it needs a short user manual	√	√
4	Build with and for reuse	√	√
5	Communicate with customers/users		√
6	Define software artifacts rigorously	√	√
7	Design for change		√
8	Design for errors	√	√
9	Design for maintenance		√
10	Determine requirements now		√
12	Don't test your own software		√
13	Don't try to retrofit quality	√	√
14	Don't write your own test plans		√
15	Establish a software process that provides flexibility	√	√
16	Fix requirements specification error now		√
17	Give product to customers early		√
18	Give software tools to good engineers		√
19	Grow systems incrementally	√	√
20	Implement a disciplined approach and improve it continuously	√	√
21	Invest in the understanding of the problem	√	√
22	Involve the customer	√	√
23	Keep design under intellectual control		√
24	Maintain clear accountability for results		√
25	Produce software in a stepwise fashion		√

Tableau CXXXV (suite)

No.	Proposition	Critère 1	Critère 2
26	Quality is the top priority; long term productivity is a natural consequence of high quality	√	√
27	Rotate people through product assurance	√	√
28	Since change is inherent to software, plan for it and manage it	√	√
29	Since tradeoffs are inherent to software engineering, make them explicit and document it		√
30	Strive to have a peer, rather than a customer, find a defect	√	√
31	Tailor cost estimation methods	√	√
32	To improve design, study previous solutions to similar problems	√	√
33	Use better and fewer people	√	√
35	Write programs for people first	√	√
37	Select tests based on the likelihood that they will find faults	√	√
38	Choose a programming language according to maintainability	√	√
39	In face of unstructured code, rethink the module and redesign it from scratch.		√

En plus de vérifier si les propositions satisfont aux deux critères d'ensemble, des liens seront aussi faits avec les processus identifiés par la norme ISO/IEC 12207.

Nous observons que sur les 36 propositions, 14 propositions sont déduites d'autres propositions plus générales. Ces 14 propositions ne satisfont pas le critère no. 1. Par contre, toutes les propositions satisfont le critère no. 2. Comme nous le verrons, certaines propositions ont été conservées malgré qu'elles soient déduites d'autres propositions puisqu'elles ajoutent ou précisent d'avantage un élément par rapport à la proposition parent.

Nous allons commenter pour chacune des 36 propositions le résultat de notre analyse.

Proposition no.1: *Align incentives for developer and customer*

Nous avons déjà discuté de cette proposition qui se retrouve également dans la catégorie individu. Davis (1995) souligne que les exigences doivent être priorisées avec le client afin que les développeurs puissent par la suite percevoir leur importance relative. Nous interprétons l'explication donnée par Davis à l'effet qu'il cible essentiellement le développement du code et non les étapes précédentes ou subséquentes à la construction du logiciel. La proposition satisfait aux deux critères, elle est donc retenue.

Cette proposition est liée aux processus primaires de développement, mais il serait possible d'étendre sa portée aux activités précédant le code jusqu'aux activités d'installation du logiciel. Cependant, Davis (1995) a strictement limité la portée de ce principe à la phase de programmation.

La mise en place des incitatifs et de leur vérification relèvent des processus de type organisationnel, particulièrement du management. La gestion de projet est responsable de la mise en place des incitatifs et du suivi de ceux-ci. Le tableau CXXXVI présente les relations entre la proposition et les processus de la norme ISO/IEC12207.

Tableau CXXXVI

Proposition no.1 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire 5.3 Développement	 5.3.7 Software coding and testing
Organisationnel 7.1 Management	 7.1.2 Planning 7.1.3 Execution and control 7.1.4 Review and evaluation

Proposition no.2: *Apply and use quantitative measurements in decision making*

Bourque et al. (2002) ne commentent pas cette proposition. Une interprétation possible est à l'effet que le contrôle (le management) devrait procéder à la prise de mesure au

niveau des processus et d'utiliser les résultats obtenus pour la prise de décision. Le contrôle peut difficilement être efficace sans l'utilisation de mesures. Cette proposition est principalement associée au processus organisationnel de management. Cependant, nous soulignons que cette proposition peut aussi englober des processus de développement et que certaines décisions, nécessitant des données quantitatives, peuvent être prises par les développeurs.

Tableau CXXXVII

Proposition no.2 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Organisationnel 7.1 Management	 7.1.3 Execution and control 7.1.4 Review and evaluation

Proposition no.3: *Build software so that it needs a short user manual*

La proposition no. 3 a été analysée au niveau de la section des propositions de catégorie produit. Au niveau de la catégorie processus, c'est le terme « Build » implique explicitement le processus de développement du logiciel. Le processus doit tenir compte de l'objectif que le logiciel développé devrait être facile à utiliser sans nécessité un volumineux guide d'utilisation. L'explication donnée par l'auteur vise exclusivement la construction du logiciel, malgré le fait que d'un point de vue général, elle pourrait aussi toucher à la maintenance et à l'exploitation du logiciel. La proposition satisfait les deux critères et elle est retenue. La proposition est liée aux processus primaires de développement, tel que présenté au tableau CXXXVIII.

Tableau CXXXVIII

Proposition no.3 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire 5.3 Développement	5.3.4 Software requirement analysis 5.3.5 Software architectural design 5.3.6 Software detailed design 5.3.7 Software coding and testing

Proposition no.4: *Build with and for reuse*

Cette proposition est aussi membre de la catégorie produit. En effet, c'est une qualité de certains composants logiciels d'être réutilisables. Cet aspect est couvert par la partie "for reuse" de la proposition. Au niveau de la catégorie processus, c'est la portion "Build with" qui est ciblée. Construire un logiciel en réutilisant des composants déjà faits est un aspect du processus qui doit être pris en compte au niveau de la conception et de la construction du logiciel. La proposition no. 4 satisfait les deux critères, elle est donc retenue.

La proposition est liée aux processus primaires de développement, tel que présenté au tableau CXXXIX.

Tableau CXXXIX

Proposition no.4 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire 5.3 Développement	5.3.5 Software architectural design 5.3.6 Software detailed design 5.3.7 Software coding and testing

Proposition no.5: *Communicate with customers/users*

Cette proposition peut être déduite de la proposition plus générale no. 22 ("Involve the customer"). La proposition no. 5 n'apporte gère plus de précision à la proposition père no. 22. Si le client s'implique dans le projet, la communication avec le client est implicite. De ce fait, la proposition no. 5 n'est donc pas retenue.

Proposition no.6: *Define software artifacts rigorously*

Cette proposition a été commentée au niveau de la catégorie produit. Concernant la catégorie processus, c'est le terme « *define* » qui a une implication directe sur le processus. De plus, le terme « *rigorously* » ajoute une contrainte sur le processus. La proposition satisfait les deux critères, elle est donc retenue.

La proposition est liée aux processus primaires de développement, tel que présenté au tableau CXL.

Tableau CXL

Proposition no.6 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire 5.3 Développement	5.3.4 Software requirement analysis 5.3.5 Software architectural design 5.3.6 Software detailed design 5.3.7 Software coding and testing

Proposition no.7: *Design for change*

Davis (1995) souligne que la facilité de modifier le logiciel pour l'adapter au changement dépend en bonne partie des choix faits au niveau du design. L'auteur cite

des caractéristiques de qualité du design qui favoriseraient la facilité d'adapter par la suite le logiciel aux changements. Entre autres, le design doit être modulaire afin qu'un changement apporté à un module ait un impact mineur sur les autres modules du logiciel. Le design devrait être portable afin qu'il soit adaptable à des changements de matériel et de système d'exploitation. Le design doit être aussi malléable afin de permettre l'ajout de nouvelles exigences.

La proposition no. 7 est déduite de la proposition no. 9 (Design for maintenance). La portée de la proposition no. 9 englobe la proposition no. 7 et en plus tous les correctifs faits au logiciel tel les défauts et l'amélioration des performances. Comme le processus de maintenance inclut les changements apportés au logiciel, ainsi que les correctifs, nous ne considérons pas que la proposition no. 7 ajoute d'élément nouveau à la proposition no. 9. La proposition no. 7 n'est donc pas retenue.

Proposition no.8: *Design for errors*

Davis (1995) souligne qu'à l'étape du design, il serait requis de s'assurer que des erreurs ne soient pas introduites dans le design. Le cas échéant, celles-ci devraient être détectées et corrigées. Davis propose des actions à prendre qui sont reliés plutôt à la construction du logiciel et à son code. Nous considérons, d'une part que l'auteur utilise le terme design dans la formulation de la proposition, mais que les actions à prendre sont plutôt de nature de codage du logiciel. D'autre part, la formulation même de la proposition est boiteuse, car dans les faits on ne fait pas de design pour des erreurs. Ainsi, l'action suggérée est différente de la formulation de la proposition. Pour ces raisons, la proposition no. 8 sera écartée.

Proposition no.9 : *Design for maintenance*

Cette proposition ressemble à la proposition no. 7 qui est déduite de celle-ci. Adapter le logiciel au changement est un des éléments couverts par le processus de maintenance. La proposition no. 9 est déduite à son tour (figure 30) de la proposition no. 28 (Since change is inherent to software, plan for it and manage it). Cependant, la proposition no. 9 précise que le design doit prévoir les activités de la maintenance (évolutive et corrective). La proposition est donc retenue.

Figure 30 Proposition déduite de la proposition no.9

La proposition no. 9 est liée au processus du type primaire de la norme ISO/IEC12207, en particulier aux processus de développement et de maintenance tel que présenté au tableau CXLI.

Tableau CXLI

Proposition no.9: relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire	
5.3 Développement	5.3.5 Software architectural design 5.3.6 Software detailed design
5.5 Maintenance	5.5.2 Problem and modification analysis 5.5.3 Modification implementation 5.5.4 Maintenance review/acceptance

Proposition no.10: *Determine requirements now*

Davis (1995) souligne que les exigences devraient être définies avant d'entreprendre les étapes subséquentes. L'auteur mentionne également des techniques pour élucider les exigences telles le prototypage. La proposition no. 10 peut se déduire de la proposition no. 21 qui est plus générale (Invest in the understanding of the problem). Malgré cette déduction possible, nous considérons que la proposition no. 10 précise l'action à faire afin de mieux comprendre le problème à résoudre. La définition des exigences logicielles représente un moyen de comprendre le problème. La proposition no. 21 ne précise pas les moyens ou l'action à prendre pour comprendre le problème. La proposition no. 10 est donc retenue.

Au niveau des processus de la norme ISO/IEC12207, la proposition no. 10 est liée aux processus de type primaire de développement tel que présenté au tableau CXLII.

Tableau CXLII

Proposition no.10 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire 5.3 Développement	5.3.2 System requirement analysis 5.3.4 Software requirement analysis

Proposition no.12: *Don't test your own software*

Par cette proposition, Davis (1995) mentionne que les développeurs ne devraient pas tester le logiciel qu'ils ont développé, sauf pour les tests unitaires. Cette proposition peut être déduite de la proposition no. 30 (*Strive to have a peer, rather than a customer, find a defect*). La proposition no. 30 souligne qu'il est préférable que les pairs détectent les défauts plutôt que le client. Il est donc sous-entendu que les pairs effectuent les tests. C'est précisément cet aspect qui est en lien avec la proposition no. 12. La proposition

no. 12 n'ajoute pas d'éléments nouveaux par rapport à la proposition no. 30. Elle n'est donc pas retenue.

Proposition no.13 : *Don't try to retrofit quality*

Davis (1995) souligne que la qualité ne peut s'ajouter au logiciel en fin du processus. La qualité s'introduit à chacune des étapes du processus de développement par des activités spécifiques. La proposition satisfait aux deux critères. Elle est donc retenue.

La proposition est liée aux processus de soutien de la norme ISO/IEC 12207, particulièrement aux processus d'assurance qualité tel que présenté au tableau CXLIII.

Tableau CXLIII

Proposition no.13 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Soutien	
6.3 Quality assurance	6.3.2 Product assurance 6.3.3 Process assurance
6.4 Verification	6.4.2 Verification 6.4.2.2 Process verification 6.4.2.3 Requirements verification 6.4.2.4 Design verification 6.4.2.5 Code verification 6.4.2.6 Integration verification 6.4.2.7 Documentation verification
6.5 Validation	6.5.2 Validation

Proposition no.14: *Don't write your own test plans*

Davis (1995) souligne que les développeurs ne devraient pas concevoir les plans de tests pour le logiciel qu'ils ont développé. L'auteur affirme que les développeurs pourraient faire les même erreurs ou omissions qu'ils ont introduites au niveau du logiciel. La proposition no. 14 s'apparente à la proposition no. 12 que nous avons écartée antérieurement. La proposition no. 12 stipulait de confier les tests à des pairs, comme la proposition no. 30 le souligne. Par contre au niveau de la proposition no. 14, il est souligné que les développeurs ne doivent pas faire non plus la conception des plans de tests de leur logiciel. Nous considérons que la proposition no. 14 ajoute une précision intéressante (plan de tests) par rapport à la proposition no. 30. Elle est donc retenue

La proposition no. 14 est liée aux processus de type primaire de développement et de maintenance de la norme ISO/IEC12207, tel que présenté au tableau CXLIV.

Tableau CXLIV

Proposition no.14 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire	
5.3 Développement	5.3.7 Software coding and testing 5.3.9 Software qualification testing 5.3.11 System qualification testing
5.5 Maintenance	5.5.3 Modification implementation 5.5.3.2 a)

Proposition no.15: *Establish a software process that provides flexibility*

Bourque et al. (2002) ne fournissent pas d'explication pour cette proposition. Nous l'interprétons donc comme suit : le processus mis en place doit pouvoir s'adapter (flexibilité) au contexte spécifique des projets et de l'organisation.

Nous devons vérifier si la flexibilité mentionnée dans la proposition entre en contradiction avec l'approche disciplinée de la proposition no.20. Nous considérons qu'un processus peut être discipliné tout en étant adaptable au contexte spécifique de l'entreprise. Il n'y a donc pas de contradiction entre la flexibilité et la discipline. La proposition satisfait aux deux critères et elle est donc retenue.

Au niveau de la norme ISO/IEC 12207, la proposition ne s'adresse pas à un processus en particulier identifié par la norme. Cependant, la norme souligne en introduction: « *The international standard is[...] designed to be tailored for an individual organization, project, or application. » (p.x).* La proposition a donc une portée globale sur la norme, mais pas pour un processus en particulier.

Proposition no.16 : *Fix requirements specification error now*

Davis (1995) souligne que les erreurs au niveau des exigences pour le logiciel doivent être détectées et corrigées dès cette étape. Une erreur provenant des exigences qui serait découverte plus tard dans le processus peut entraîner des coûts de correction importants.

Cette proposition peut être déduite de la proposition de la proposition no. 10 (Determine requirement now). La déduction provient du fait que l'étape de définition des exigences devrait inclure des activités de vérification et de validation. Cependant, la proposition no. 16 mentionne explicitement de corriger les erreurs dans les exigences. Même si elle est déduite, la proposition no. 16 sera retenue car elle ajoute une précision à la proposition no. 10.

La proposition no. 16 est liée aux processus de type primaire de développement et aux processus de soutien tel que présenté au tableau CXLV.

Tableau CXLV

Proposition no.16 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire	
5.3 Développement	5.3.2 System requirement analysis 5.3.4 Software requirement analysis
Soutien	
6.4 Verification	6.4.2.3 Requirement verification

Proposition no.17 : *Give product to customers early*

Davis (1995) précise que le client peut s'impliquer dans le processus en utilisant des versions prototypes du logiciel. Ces prototypes permettent, entre autres, de valider les besoins exprimés par le client et d'obtenir ses commentaires. Par la suite, les exigences du logiciel seront complétées et le produit final pourra être développé.

Cette proposition est déduite de la proposition no. 22 (Involve the customer) qui plus générale (figure 31). Par contre, la proposition no. 17 précise de quelle façon le client peut s'impliquer dans le projet. La proposition est donc retenue car elle ajoute une précision par rapport à la proposition père.

22 : Involve the customer
17 : Give product to customers early

Figure 31 Proposition déduite de la proposition no.22

La proposition est liée aux processus primaires et de soutien de la norme ISO/IEC12207 tel que présenté au tableau CXLVI.

Tableau CXLVI

Proposition no.17 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire 5.3 Développement Soutien 6.5 Validation	5.3.4 Software requirement analysis

Proposition no.18: *Give software tools to good engineers*

Davis (1995) souligne que l'utilisation d'un outil de développement ne transforme pas un développeur médiocre en un excellent développeur. Tout comme l'utilisation d'un traitement de texte ne fait pas d'une personne un auteur de renom. L'auteur souligne que les outils (CASE) permettent à un développeur qui connaît bien les techniques du génie logiciel d'être plus performant. Par contre, les mêmes outils donnés à des développeurs ayant des carences au niveau de leurs connaissances ne rendront pas ceux-ci plus performants.

La proposition peut se déduire (figure 32) de la proposition no. 36 (Know software engineering's techniques before using development tools).

36 : Know software engineering's techniques before using development tools
18 : Give software tools to good engineers

Figure 32 Proposition déduite de la proposition no.36

Cette dernière souligne que le développeur devrait connaître les techniques du génie logiciel avant d'utiliser des outils de développement tel les outils CASE. Tout comme mentionné antérieurement au niveau de la catégorie individu, la proposition no. 18 n'ajoute pas d'éléments nouveaux par rapport à la proposition no. 36. La proposition no. 18 n'est donc pas retenue.

Proposition no.19: *Grow systems incrementally*

Davis (1995) souligne que le développement d'un système devrait se faire par l'ajout successif de fonction, soit de façon incrémentale. Davis favorise la mise en place d'un système d'envergure réduite, mais fonctionnel plutôt qu'attendre que toutes les fonctionnalités soient développées et procéder à une implantation globale de l'ensemble du système. Cette approche permet aussi de réduire le risque du projet. La proposition no. 19 n'est pas déduite et pas en contradiction avec une autre proposition. Elle est donc retenue.

Au niveau des processus de la norme ISO/IEC12207, la proposition est liée aux processus primaires de développement et particulièrement à effectuer des micro-cycles des activités 5.3.4 à 5.3.12 tel que présenté au tableau CXLVII.

Tableau CXLVII

Proposition no.19 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire 5.3 Développement	5.3.4 Software requirement analysis 5.3.5 Software architectural design 5.3.6 Software detailed design 5.3.8 Software integration 5.3.9 Software qualification testing 5.3.10 System integration 5.3.11 System qualification testing 5.3.12 Software installation

Proposition no.20: *Implement a disciplined approach and improve it continuously*

Bourque et al. (2002) ne fournissent pas d'explication sur cette proposition. Nous interprétons donc celle-ci à l'effet de mettre en place un processus discipliné de développement et de procéder à son amélioration d'une façon continue. L'utilisation du terme discipliné souligne une forme de rigueur dans le déroulement et dans la séquence des activités de développement. La proposition no. 20 est de nature générale et cinq autres propositions peuvent en être déduites tel que présenté à la figure 33. La proposition satisfait aux deux critères et elle est donc retenue.

20 : Implement a disciplined approach and improve it continuously
6 : Define software artifacts rigorously
23 : Keep design under intellectual control
24 : Maintain clear accountability for results
25 : Produce software in a stepwise fashion
34 : Use documentation standards

Figure 33 Propositions déduites de la proposition no.20

Toutes les propositions déduites comportent une forme de rigueur, donc de discipline à suivre.

La proposition no. 20 n'est pas liée à un processus en particulier de la norme ISO/IEC12207. La portée de cette proposition est plutôt de niveau méta. Tout comme la proposition no. 15, la proposition no. 20 est de niveau général. Par contre, l'utilisation de la norme permet de mettre en pratique la proposition no. 20.

Proposition no.21: *Invest in the understanding of the problem*

Bourque et al. (2002) ne décrivent pas le sens de cette proposition. Nous l'interprétons comme suit : avant de développer une solution logicielle, il est préférable de bien comprendre le problème et de prendre les actions nécessaires pour l'élucider. La

proposition n'est pas déduite, mais d'autres propositions sont déduites de celle-ci tel que présenté à la figure 34.

21 : Invest in the understanding of the problem
    10 : Determine requirements now
        16 : Fix requirements specification error now

Figure 34 Propositions déduites de la proposition no.21

La proposition 21 n'est pas déduite et ni en contradiction avec d'autres. Elle est donc retenue.

La proposition est liée à plusieurs processus de la norme ISO/IEC 12207, tel que présenté au tableau CXLVIII. La proposition n'implique pas seulement les processus reliés au développement, mais aussi ceux de l'acquisition de produits et de services, ainsi qu'à l'approvisionnement.

Tableau CXLVIII

Proposition no.21 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire	
5.1 Acquistion	5.1.1 Initiation
5.2 Approvisionnement	5.2.1 Initiation 5.2.2 Preparation of response
5.3 Développement	5.3.2 System requirement analysis 5.3.3 System architectural design 5.3.4 Software requirement analysis
5.5 Maintenance	5.5.2 Problem and modification analysis

Tableau CXLVIII (suite)

Processus	**Activités**
Soutien 6.8 Problem resolution	6.8.2 Problem resolution
Organisationnel 7.1 Management	7.1.3 Execution and control

Proposition no.22: *Involve the customer*

Royce (1970) souligne que le client devrait être impliqué tout au long du projet. L'implication du client permet, entre autres, d'obtenir son appréciation sur les produits intermédiaires et son soutien tout au long du processus de développement. L'auteur vise dans son explication essentiellement le processus de développement, mais l'implication du client peut se manifester également dans l'approvisionnement, l'acquisition et la maintenance. La proposition no. 22 n'est pas déduite ni en contradiction avec d'autres, elle est donc retenue Par contre, la proposition no.5 est déduite de celle-ci tel que présenté à la figure 35.

22 : Involve the customer
5 : Communicate with customers/users

Figure 35 Proposition déduite de la proposition no.22

L'implication du client est aussi liée aux processus de validation et de vérification de la norme ISO/IEC12207 tel que présenté au tableau CXLIX.

Tableau CXLIX

Proposition no.22 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire	
5.1 Acquistion	5.1.1 Initiation 5.1.2 Request for proposal preparation 5.1.3 Contract preparation 5.1.4 Supplier monitoring 5.1.5 Acceptance and completion
5.2 Approvisionnement	5.2.4 Planning 5.2.4.5 j)
5.3 Développement	5.3.2 System requirement analysis 5.3.3 System architectural design 5.3.4 Software requirement analysis 5.3.5 Software architectural design 5.3.10 System integration 5.3.11 System qualification testing 5.3.12 Software installation 5.3.13 Software acceptance support
5.5 Maintenance	5.5.2 Problem and modification analysis 5.5.4 Maintenance review/acceptance
Soutien	
6.4 Verification	6.4.2.1 Contract verification 6.4.2.2 Process verification 6.4.2.3 Requirements verification 6.4.2.4 Design verification 6.4.2.6 Integration verification 6.4.2.7 Documentation verification
6.5 Validation	6.5.2 Validation
6.6 Joint review process	6.6.2 Project management review

Proposition no.23: *Keep design under intellectual control*

Davis (1995) souligne que le design doit être fait et documenté de façon à ce que les développeurs puissent le comprendre dans son ensemble. Il souligne également que le design devrait être construit d'une façon hiérarchique et à l'aide de vues multiples.

La proposition peut être déduite de la proposition no. 20 qui est plus générale : « *Implement a disciplined approach and improve it continuously* ». Cependant, la proposition no. 23 ajoute une précision portant sur l'activité du design. La proposition est donc retenue.

La proposition no. 23 est liée aux processus primaires de développement de la norme ISO/IEC12207, ainsi qu'aux processus de soutien de la gestion de configuration tel que présenté au tableau CL.

Tableau CL

Proposition no.23 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire 5.3 Développement	5.3.3 System architectural design 5.3.5 Software architectural design 5.3.6 Software detailed design
Soutien 6.2 Configuration management	

Proposition no.24: *Maintain clear accountability for results*

Boehm (1983) souligne que les individus qui composent les équipes de développement doivent savoir avec précision quels sont les résultats attendus dont ils sont responsables. Cette proposition touche directement l'organisation du travail et le découpage des tâches et des responsabilités. La proposition peut être déduite de la proposition no. 20 qui prône une approche disciplinée. De plus, la proposition no. 1 (*Align incentives for developer*

and customer) a un lien avec la proposition no. 24. Les incitatifs ne peuvent être mis en place, ni vérifiés que si les résultats attendus sont bien définis. Malgré que la proposition no. 24 puisse se déduire de la proposition no. 20, nous considérons qu'elle précise un aspect supplémentaire permettant de mettre en application une approche disciplinée. Elle est donc retenue.

La proposition no. 24 est liée aux processus de type organisationnel de la norme ISO/IEC12207, en particulier aux processus de management tel que présenté au tableau CLI.

Tableau CLI

Proposition no.24 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Organisationnel 7.1 Management	7.1.2 Planning 7.1.2.1 a), d) et e)

Proposition no.25: *Produce software in a stepwise fashion*

Bourque et al. (2002) n'ont pas commenté cette proposition, nous l'interprétons donc comme suit: le logiciel doit être développé à l'aide d'étapes bien définies. Cette proposition peut être déduite de la proposition no. 20 qui est plus générale. Réaliser le logiciel en suivant des étapes s'inscrit dans la portée de la proposition no. 20 à l'effet qu'il faut implémenter une approche disciplinée pour le développement du logiciel. Malgré la déduction possible, la proposition no. 25 ajoute une précision (les étapes) par rapport à la proposition no. 20. La proposition est donc retenue.

La proposition no. 25 est liée aux processus de type primaire de développement de la norme ISO/IEC12207 tel que présenté au tableau CLII.

Tableau CLII

Proposition no.25 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire 5.3 Développement	5.3.2 System requirement analysis 5.3.3 System architectural design 5.3.4 Software requirement analysis 5.3.5 Software architectural design 5.3.6 Software detailed design 5.3.7 Software coding and testing 5.3.8 Software integration 5.3.9 Software qualification testing 5.3.10 System integration 5.3.11 System qualification testing 5.3.12 Software installation

Proposition no.26: *Quality is the top priority; long term productivity is a natural consequence of high quality*

Wiegers (1996) affirme que la qualité serait la plus haute priorité dans les activités du génie logiciel. La portée de l'explication de l'auteur se divise en deux principaux points : l'attitude des individus envers la qualité et la qualité des processus et des pratiques. Au niveau des processus, Wiegers souligne que la qualité dépend des processus utilisés pour détecter les erreurs en cours de développement et des pratiques appliquées pour les détecter.

La proposition no. 26 n'est pas déduite, ni en contradiction avec d'autres propositions. Elle est donc retenue. La proposition est liée aux trois groupes principaux de processus de la norme ISO/IEC12207, tel que présenté au tableau CLIII.

Tableau CLIII

Proposition no.26 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire 5.3 Développement	 5.3.2 System requirement analysis 5.3.3 System architectural design 5.3.4 Software requirement analysis 5.3.5 Software architectural design 5.3.6 Software detailed design 5.3.7 Software coding and testing 5.3.8 Software integration 5.3.9 Software qualification testing 5.3.10 System integration 5.3.11 System qualification testing
5.5 Maintenance	5.5.3 Modification implementation
Soutien 6.3 Quality Assusrance 6.4 Verification 6.5 Validation 6.6 Joint review 6.7 Audit	
Organisationnel 7.1 Management 7.3 Processus improvement	 7.1.2.1 g)

Proposition no.27: *Rotate people through product assurance*

Davis (1995) déplore que certaines organisations déplacent des développeurs peu performants vers le groupe d'assurance qualité. Il souligne que ce groupe requiert le même niveau de qualité et de discipline que l'on doit retrouver dans l'équipe de développement. Davis propose plutôt que les *meilleurs* développeurs fassent une visite de quelques mois au sein du groupe d'assurance qualité. La venue des meilleurs

développeurs aura un effet positif sur les activités de ce groupe en permettant de partager les expériences.

Nous constatons que l'idée principale de l'auteur ne se manifeste pas dans la formulation de la proposition. La proposition ne fait pas mention de faire une rotation des meilleures personnes. La proposition satisfait aux deux critères, mais elle comporte une omission importante au niveau de sa formulation.

La proposition est liée aux processus de type organisationnel de la norme ISO/IEC12207 tel que présenté au tableau CLIV.

Tableau CLIV

Proposition no.27 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Organisationnel 7.1 Management	7.1.2 Planning 7.1.2.1 c), d) et e)

Proposition no.28: *Since change is inherent to software, plan for it and manage it*

Bourque et al (2002) ne commentent cette proposition. Nous l'interprétons donc de la façon suivante : le changement fait partie de la réalité du développement du logiciel, ainsi, il faut le planifier et le gérer correctement. Nous considérons que le concept du changement souligné dans la formulation est en lien spécifique au logiciel. Les changements apportés aux ressources humaines, aux délais de livraison ou au budget ne sont pas considérés dans la portée de cette proposition. Nous avons déjà établi que la proposition no. 9 est déduite de la proposition no. 28 tel que présenté à la figure 36.

28 : Since change is inherent to software, plan for it and manage it
9 : Design for maintenance
7 : Design for change

Figure 36 Propositions déduites de la proposition no.22

La proposition satisfait aux deux critères et elle est donc retenue.

La proposition no. 28 est liée à plusieurs processus de la norme ISO/IEC12207 tel que présenté au tableau CLV.

Tableau CLV

Proposition no.28 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire	
5.3 Developpement	5.3.1.2 Configuration management (Change control)
5.5 Maintenance	
Soutien	
6.2 Configuration management	6.2.3.1 Change control
Organisationnel	
7.1 Management	7.1.3 Execution and Control

Proposition no.29: *Since tradeoffs are inherent to software engineering, make them explicit and document it*

Bourque et al. (2002) ne fournissent pas d'explication sur la portée de cette proposition. Nous l'interprétons donc comme suit: tous les compromis faits au cours du processus de

développement devraient être explicitement documentés. La proposition comporte un élément implicite de discipline. La proposition peut se déduire de la proposition no. 20. Cependant, la proposition no. 29 précise l'action de documenter les compromis. La proposition satisfait aux deux critères et elle est donc retenue.

La proposition est liée aux processus de type primaire de développement et de maintenance de la norme ISO/IEC12207 tel que présenté au tableau CLVI.

Tableau CLVI

Proposition no.29 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire	
5.3 Developpement	5.3.2 System requirement analysis 5.3.3 System architectural design 5.3.4 Software requirement analysis 5.3.5 Software architectural design 5.3.6 Software detailed design 5.3.7 Software coding and testing
5.5 Maintenance	5.5.2 Problem and modification analysis 5.5.3 Modification implementation

Proposition no.30: *Strive to have a peer, rather than a customer, find a defect*

Wiegers (1996) souligne qu'il est impératif que les défauts du logiciel soient détectés à l'interne plutôt que par le client. L'auteur favorise fortement les activités de revue et d'inspection par les pairs. Il souligne également que le développeur peut trouver lui même quelques erreurs, mais les pairs peuvent en trouver un plus grand nombre. La figure 37 présente les propositions déduites de la proposition no. 30.

30 : Strive to have a peer, rather than a customer, find a defect

12 : Don't test your own software

14 : Don't write your own test plans

Figure 37 Propositions déduites de la proposition no.30

Deux propositions (12 et 14) peuvent être déduites de la proposition no. 30. La proposition no. 30 n'est pas déduite ni en contradiction avec d'autres. La proposition est retenue.

La proposition no. 30 est liée aux processus de type primaire et de soutien de la norme ISO/IEC12207 tel que présenté au tableau CLVII.

Tableau CLVII

Proposition no.30 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire	
5.3 Developpement	5.3.7 Software coding and testing 5.3.8 Software integration 5.3.9 Software qualification testing 5.3.10 System integration 5.3.11 System qualification testing
5.5 Maintenance	5.5.3 Modification implementation
Soutien	
6.6 Joint review	6.6.3 Technical review

Proposition no.31: *Tailor cost estimation methods*

Davis (1995) souligne essentiellement par cette proposition que les méthodes d'estimation (ex : COCOMO) doivent être adaptées au contexte de l'organisation et des projets afin qu'elles puissent générer de meilleurs résultats. Davis classe cette proposition dans la catégorie management.

La proposition de Davis pourrait être reformulée afin de retirer l'aspect « *cost estimation* » afin d'obtenir une formulation plus générale pouvant s'appliquer à toutes les méthodes. Dans sa formulation actuelle, la proposition a une portée limitée. La proposition n'est pas déduite et ni en contradiction avec d'autres. Elle est donc retenue.

La proposition no. 31 est liée aux processus organisationnel de management, en particulier à l'activité de planification dans laquelle les sous-activités b et h sont spécifiques à l'estimation des coûts.

Tableau CLVIII

Proposition no.31 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Organisationnel 7.1 Management	7.1.2 Planning

Proposition no.32: *To improve design, study previous solutions to similar problems*

Bourque et al. (2002) ne commentent pas la proposition. Nous l'interprétons à l'effet que l'étude de solutions logicielles existantes peut améliorer la conception d'une nouvelle solution au même type de problème. La proposition n'est pas déduite ni en contradiction avec d'autres propositions. Elle est donc retenue.

La proposition no. 32 a une portée essentiellement sur la conception du logiciel. Elle est liée aux processus de type primaire de développement tel que présenté au tableau CLIX.

Tableau CLIX

Proposition no.32 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire 5.3 Développement	5.3.3 System architectural design 5.3.5 Software architectural design 5.3.6 Software detailed design

Proposition no.33: *Use better and fewer people*

Boehm (1983) souligne qu'il y a une grande variance de productivité entre les développeurs. Il suggère donc par cette proposition d'utiliser les meilleurs développeurs plutôt que plusieurs développeurs moyens. Il ajoute qu'un grand nombre de personnes dans un projet augmente la lourdeur des interactions. Boehm suggère donc de retirer de l'équipe de développement les individus les moins performants. La proposition n'est pas déduite ni en contradiction avec d'autres. Elle est donc retenue.

La proposition no. 33 est liée aux processus de type organisationnel de la norme ISO/IEC 12207 tel que présenté au tableau CLX.

Tableau CLX

Proposition no.33 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Organisationnel 7.1 Management	7.1.2 Planning 7.1.2.1 c), d) et e)

Proposition no.35: *Write programs for people first*

La proposition no. 35 a déjà été commentée au niveau de la catégorie produit. Concernant la catégorie processus, la proposition contient le terme « *write* » qui a un lien direct avec le processus puisque c'est celui-ci qui réalise l'écriture des programmes Cette proposition satisfait les deux critères et elle est donc retenue. La proposition no. 35 pourrait être déduite à la proposition no. 9 : « Design for maintenance ». Cependant, la proposition no. 35 se concentre sur l'écriture même des programmes afin qu'ils soient lisibles facilement par d'autres programmeurs.

La proposition no. 35 est lié aux processus de type primaire de développement et de maintenance de la norme ISO/IEC 12207 tel que présenté au tableau CLXI.

Tableau CLXI

Proposition no.35 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire	
5.3 Developpement	5.3.7 Software coding and testing
5.5 Maintenance	5.5.3 Modification implementation

Proposition no.37: *Select tests based on the likelihood that they will find faults*

Cette proposition de Davis (1995) a été reformulée à la phase 2 afin de refléter une formulation prescriptive guidant l'action. L'auteur souligne que les cas de tests doivent être sélectionnés de façon à provoquer des erreurs du logiciel. La proposition n'est pas déduite, ni en contradiction avec d'autres. Elle est donc retenue.

La proposition no.37 est lié aux processus de type primaire de développement et de maintenance de la norme ISO/IEC 12207 tel que présenté au tableau CLXII.

Tableau CLXII

Proposition no.37 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire	
5.3 Developpement	5.3.7 Software coding and testing 5.3.8 Software integration 5.3.10 System integration
5.5 Maintenance	5.5.3 Modification implementation (5.5.3.2 a)

Proposition no.38: *Choose a programming language to assure maintainability*

Cette proposition de Davis (1995) a été reformulée à la phase 2 afin de lui donner une forme prescriptive. Davis affirme que le choix du langage de programmation a un impact sur l'effort de maintenance. La proposition n'est pas déduite, ni en contradiction avec d'autres. Elle est donc retenue.

La norme ISO/IEC 12207 fait abstraction du choix du langage de programmation L'annexe E, paragraphe 11, souligne que "*the standard is flexible and usable with[..] any programming languages*". Le langage de programmation peut être aussi considéré comme un outil de développement. À ce titre, la proposition peut être liée aux processus de type organisationnel d'infrastructure qui comprend les outils. Le tableau CLXIII présente la relation entre la proposition no. 38 et la norme ISO/IEC12207.

Tableau CLXIII

Proposition no.38 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Organisationnel	
7.2 Infrastructure	7.2.2 Establishment of the infrastructure

Proposition no.39: *In face of unstructured code, rethink the module and redesign it from scratch*

Cette proposition a également été reformulée à la phase 2. Davis (1995) affirme qu'à l'étape de maintenance, il ne serait pas utile de restructurer le code d'un programme. L'auteur suggère plutôt de repenser le programme et de le réécrire.

La proposition peut se déduire de la proposition no. 6 : *design for maintenance.* Également, la proposition a un lien avec la proposition no. 9 à l'effet de bien définir les artéfacts du logiciel, dont le programme fait partie. Malgré la déduction possible, nous considérons que la proposition no. 39 ajoute une précision guidant l'action du mainteneur. Elle est donc retenue. La figure 38 présente les déductions possibles.

6 : Design for maintenance

9 : In face of unstructured code, rethink the module and redesign it from scratch

Figure 38 Proposition déduite de la proposition no.6

La proposition no. 39 est liée aux processus de type primaire de maintenance tel que présenté au tableau CLXIV.

Tableau CLXIV

Proposition no.39 : relation avec les processus de la norme ISO/IEC12207

Processus	**Activités**
Primaire 5.5 Maintenance	5.5.3 Modification implementation

5.6.3.1 Sommaire de la catégorie processus

Nous avions 36 propositions au départ dans la catégorie processus et 31 ont été conservées. Nous constatons que sur les 36 propositions de départ, quatorze peuvent être déduites de propositions plus générales. Sur ces quatorze propositions, dix apportent une précision supplémentaire par rapport à la proposition parent qui permet de mieux guider l'action. Ces dix propositions ont été conservées et celles-ci sont identifiées par un « * » dans le tableau CLXV qui présente toute les propositions retenues pour la catégorie processus.

Tableau CLXV

Propositions retenues de la catégorie processus

No.	**Proposition**
1	Align incentives for developer and customer
2	Apply and use quantitative measurements in decision making
3	Build software so that it needs a short user manual
4	Build with and for reuse
6	Define software artifacts rigorously
9	Design for maintenance*
10	Determine requirements now*
13	Don't try to retrofit quality
14	Don't write your own test plans*
15	Establish a software process that provides flexibility
16	Fix requirements specification error now*
17	Give product to customers early*
19	Grow systems incrementally
20	Implement a disciplined approach and improve it continuously
21	Invest in the understanding of the problem
22	Involve the customer
23	Keep design under intellectual control*
24	Maintain clear accountability for results*
25	Produce software in a stepwise fashion*
26	Quality is the top priority; long term productivity is a natural consequence of high quality
27	Rotate (high performer) people through product assurance
28	Since change is inherent to software, plan for it and manage it

Tableau CLXV (suite)

No.	Proposition
29	Since tradeoffs are inherent to software engineering, make them explicit and document it*
30	Strive to have a peer, rather than a customer, find a defect
31	Tailor cost estimation methods
32	To improve design, study previous solutions to similar problems
33	Use better and fewer people
35	Write programs for people first
37	Select tests based on the likelihood that they will find faults
38	Choose a programming language according to maintainability
39	In face of unstructured code, rethink the module and redesign it from scratch.*

* : proposition déduite, mais conservée

5.7 Vérification du degré de couverture des propositions par rapport à la norme ISO/IEC12207

Lors de l'analyse des propositions de la catégorie processus, nous avons fait des liens entre chacune des propositions et les groupes de processus identifiés par la norme ISO/IEC12207. Tel que présenté au début de ce chapitre, la classification des processus offerte par la norme permet de vérifier le degré de couverture des propositions retenues. Sur les 31 propositions retenues de la catégorie processus, 29 ont été associés aux processus de la norme ISO/IEC12207. Deux propositions (15 et 20) n'ont pas été associées car elles ne s'appliquent pas à un processus en particulier. Ces deux proposition ont une portée plus générale et possiblement plus fondamentale. Le tableau CLXVI présente la répartition des 29 propositions en fonction des groupes de processus de la norme ISO/IEC12207.

Tableau CLXVI

Répartition des propositions retenues en fonction des groupes de processus de la norme ISO/IEC12207

Processus ISO/IEC12207	**Numéros des propositions associées**
Primaires	
5.1 Acquisition	21, 22
5.2 Approvisionnement	21, 22
5.3 Développement	1, 3, 4, 6, 9, 10, 14, 16, 17, 19, 21, 22, 23, 25, 28, 29, 30, 32, 35, 37
5.4 Exploitation	Aucune
5.5 Maintenance	9, 14, 21, 22, 28, 29, 30, 35, 37, 39
Soutien	
6.1 Documentation	Aucune
6.2 Gestion configuration	23, 28
6.3 Assurance qualité	13, 26
6.4 Vérification	13, 16, 22, 26
6.5 Validation	13, 17, 22, 26
6.6 Revue	22, 26, 30
6.7 Audit	26
6.8 Résolution de problème	21
Organisationnel	
7.1 Management	1, 2, 21, 24, 26, 27, 28, 31, 33
7.2 Infrastructure	38
7.3 Amélioration	26
7.4 Formation	Aucune

À la lumière des résultats présentés par le tableau CLXV, nous constatons que les processus de type primaire recueillent la majorité des propositions, suivis par le groupe de processus de soutien et finalement au troisième rang, le groupe de processus organisationnels. Le tableau CLXVII présente le dénombrement des propositions associées à chacun des trois groupes de processus de la norme ISO/IEC12207. Le nombre total dépasse 29 du fait que certaines propositions se retrouvent à plus d'un endroit.

Tableau CLXVII

Dénombrement des propositions par groupes de processus

Groupes de processus ISO/IEC12207	Nombre de propositions
Primaire (34)	
Développement	20
Maintenance	10
Acquisition	2
Approvisionnement	2
Exploitation	0
Soutien (17)	
Vérification	4
Validation	4
Revue	3
Gestion configuration	2
Assurance qualité	2
Audit	1
Résolution de problème	1
Documentation	0
Organisationnel (11)	
Management	9
Amélioration processus	1
Infrastructure	1
Formation	0

5.7.1 Groupe des processus primaires

Nous constatons que 20 des 29 propositions (71%) se retrouvent associées au processus primaire de développement. En deuxième rang, se retrouve le processus primaire de maintenance qui recueille 10 propositions (36%). Nous soulignons que huit des neuf propositions du groupe maintenance sont aussi associées au groupe développement. Seule la proposition no. 39 du groupe maintenance n'est pas partagée.

Les groupes de processus d'acquisition et d'approvisionnement ne recueillent que deux propositions, les deux mêmes pour les deux groupes. Enfin, nous constatons qu'aucune proposition n'est associée au groupe de processus d'exploitation.

À la lumière de ces résultats, nous constatons une certaine tendance à l'effet que les propositions retenues visent principalement les activités de développement et de maintenance. Cependant, cette constatation ne signifie pas qu'il n'existe pas de proposition (éventuellement des principes) associée au groupe exploitation.

5.7.2 Groupe des processus de soutien

Le groupe des processus de soutien se subdivise en huit processus. Les processus de vérification, de validation et de revue recueillent la majorité des propositions. Nous constatons que le processus de documentation ne comporte aucune proposition. Cependant, la proposition no. 3 (Use documentation standards) pourrait y être associée à première vue. En premier lieu, cette proposition a été classée dans la catégorie des produits. L'utilisation de normes améliore la qualité des produits. En deuxième lieu, la proposition ne comporte pas le sens de documenter les activités comme le processus de documentation (ISO/IEC 12207 - 6.1) l'indique. C'est la raison qui fait en sorte que la proposition no. 3 n'est pas liée au processus de documentation

5.7.3 Groupe des processus organisationnels

Ce groupe se place au troisième rang dans la répartition des propositions en recueillant que onze propositions. Le processus de management en recueille à lui seul huit des onze propositions. Les processus d'infrastructure et d'amélioration ne recueillent qu'une proposition, ce qui est bien peu comparativement à l'importance de ces processus. Le processus de formation n'est associé à aucune proposition.

De ces résultats, nous constatons que le processus primaire de développement vient au premier rang avec 20 propositions, suivi du processus primaire de maintenance avec 10 propositions et du processus organisationnel de management avec huit propositions. Globalement, nous constatons qu'après le retrait des doublons, ces trois processus regroupent 25 des 29 propositions retenues. Si nous renons compte que huit des neuf

propositions du groupe maintenance se retrouvent aussi au niveau du groupe développement, nous constatons qu'en fait il y a deux principaux pôles de processus soit: le groupe des processus de développement et le groupe des processus de management. Il y a donc une nette tendance à l'effet que les propositions du groupe processus retenues soutiennent majoritairement les activités de développement, de maintenance et de management. Nous constatons qu'aucun processus de soutien ne se classe parmi les trois premiers rangs.

5.8 Conclusion de la phase 3

Dans ce chapitre, les 39 propositions issues de la phase 2 ont été analysées. Les propositions ont été classées selon les trois catégories identifiées soit : individu, produit et processus. Par la suite, nous avons appliqué les deux critères d'ensemble suivants sur les 39 propositions:

1. Les principes devraient être indépendants (non déduits) (Boehm 1983)
2. Un principe ne doit pas contredire un autre principe connu. (Bourque et al. 2002)

Aucune des 39 propositions ne s'est révélée en contradiction avec une autre proposition. Ce critère n'a donc pas permis d'éliminer des propositions. Au niveau du premier critère, son application a évolué au cours de l'analyse. Nous avons évalué que certaines propositions, malgré qu'elles puissent être déduites, comportaient une précision intéressante qui méritait une attention. Puisqu'un principe guide l'action, certaines propositions précisent plus explicitement une action concrète à faire que la formulation de la proposition parent, souvent plus générale. À la lecture de certaines propositions parents, il peut y avoir interprétation différente des actions à faire. Les propositions déduites qui précisent explicitement l'action à faire sont plus facilement applicables que les propositions plus générales. C'est dans cette optique que nous avons évalué les

propositions déduites. Est-que la proposition ajoute une précision guidant l'action par rapport à la proposition parent? Dans l'affirmative, la proposition a été conservée.

Au niveau de la catégorie individu, nous avions six propositions au départ, mais quatre ont été conservées. Au niveau de la catégorie produit, nous avions sept propositions au départ et toutes ont été conservées même si une de celles-ci est déduite d'une autre, mais comme elle ajoute une précision, elle a été conservée. La catégorie processus regroupe 36 des 39 propositions. Nous avons écarté quatre propositions qui sont déduites, mais sans ajouter de précisions supplémentaires. Nous avons conservé dix propositions qui peuvent être déduites, mais qui ajoutent une précision guidant l'action à entreprendre par rapport à la proposition parent. La proposition no.8 a été écartée compte tenu de sa formulation déficiente et l'explication contradictoire de l'auteur. Sur les 39 propositions au départ, 34 propositions ont été retenues, tel que présenté au tableau CLXVIII.

Tableau CLXVIII

Les 34 propositions retenues suite à la phase 3

No.	**Proposition**
1	Align incentives for developer and customer
2	Apply and use quantitative measurements in decision making
3	Build software so that it needs a short user manual
4	Build with and for reuse
6	Define software artifacts rigorously
9	Design for maintenance*
10	Determine requirements now*
11	Don't overstrain your hardware
13	Don't try to retrofit quality
14	Don't write your own test plans*
15	Establish a software process that provides flexibility
16	Fix requirements specification error now*
17	Give product to customers early*
19	Grow systems incrementally
20	Implement a disciplined approach and improve it continuously
21	Invest in the understanding of the problem

Tableau CLXVIII (suite)

No.	Proposition
22	Involve the customer
23	Keep design under intellectual control*
24	Maintain clear accountability for results*
25	Produce software in a stepwise fashion*
26	Quality is the top priority; long term productivity is a natural consequence of high quality
27	Rotate (high performer) people through product assurance
28	Since change is inherent to software, plan for it and manage it
29	Since tradeoffs are inherent to software engineering, make them explicit and document it*
30	Strive to have a peer, rather than a customer, find a defect
31	Tailor cost estimation methods
32	To improve design, study previous solutions to similar problems
33	Use better and fewer people
34	Use documentation standards
35	Write programs for people first
36	Know software engineering's techniques before using development tools
37	Select tests based on the likelihood that they will find faults
38	Choose a programming language to assure maintainability
39	In face of unstructured code, rethink the module and redesign it from scratch.*

* : propositions déduites, mais conservées

Nous avons effectué une association des propositions de la catégorie processus avec les groupes de processus identifiés par la norme ISO/IEC12207. Nous observons que les propositions s'associent principalement à trois processus: développement, maintenance et management. Certains processus, telle la formation, la documentation et l'exploitation ne sont pas associés à aucune proposition.

CHAPITRE 6

PHASE 4 – ÉVALUATION DU DEGRÉ DE COUVERTURE DES PRINCIPES

La phase 3 a retenu 34 propositions. Les sept critères d'identification ont été appliqués, cinq critères à la deuxième phase et deux critères à la troisième phase. Le processus d'élimination des propositions est maintenant complété.

6.1 Objectif de la phase 4

L'objectif principal de la phase 4 est de procéder à une vérification de la couverture des propositions candidates retenues, en deux volets. Nous désirons, maintenant, vérifier si les propositions retenues soutiennent le génie logiciel. Le premier volet consiste à analyser les propositions en fonction des éléments du modèle d'ingénierie présenté par Moore (2006). Le deuxième volet consiste à vérifier la couverture des propositions en fonction du corpus de normes du génie logiciel de l'IEEE.

6.2 Couverture des propositions en fonction du modèle d'ingénierie.

James W. Moore (2006) présente un modèle type de l'ingénierie représentant les concepts des activités de génie (figure 39). Dans ce volet, nous analysons chacune des propositions retenues en fonction des éléments de ce modèle. L'objectif de cette étape est de vérifier si tous les éléments du modèle sont couverts par les propositions. Nous faisons l'hypothèse qu'ayant retenu des propositions pertinentes au génie logiciel, nous devrions obtenir une couverture de l'ensemble des éléments du modèle.

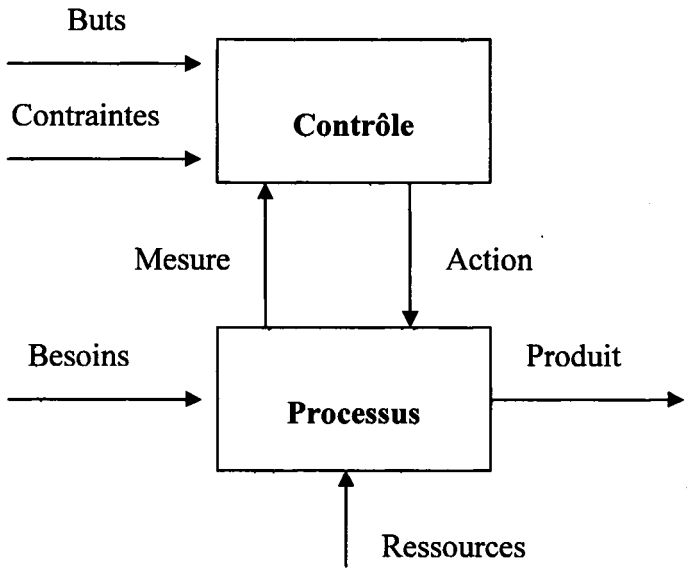

Figure 39 Modèle de l'ingénierie (Traduit de Moore 2006)

6.2.1 Méthode de recherche pour la phase 4

Dans un premier temps, les éléments du modèle sont décrits afin de bien en saisir la portée. Par la suite, pour chacune des propositions, un bref rappel est fait concernant sa signification lorsque donnée par l'auteur. Les éléments ou le sens explicite de la proposition est identifié et une association est faite avec les éléments du modèle.

Une proposition peut aussi contenir des éléments implicites, par exemple, en termes de conséquences, sans qu'ils soient explicitement formulés dans la proposition. Ces éléments implicites contenus dans la proposition sont aussi identifiés.

Une vérification est faite avec le classement des propositions selon les trois catégories (produit, processus, individu) réalisé antérieurement afin d'assurer une certaine cohérence. Une synthèse sera faite sur les observations constatées.

6.2.2 Description des éléments du modèle

Le modèle est composé de deux principaux pôles : le processus et le contrôle. Le processus représente l'ensemble des activités à réaliser pour créer le produit, le logiciel. Le contrôle représente le volet management et ingénierie qui s'assure que les activités du processus se déroulent dans le respect des buts et des contraintes du projet et de l'entreprise. Le modèle peut s'appliquer globalement au processus d'ingénierie ou à chacun de ses sous-processus. Décrivons maintenant chacun des éléments du modèle.

6.2.2.1 Processus

Le processus reçoit en entrée les besoins exprimés par le client. Une suite d'activités s'en suit pour concevoir et réaliser un produit répondant aux besoins exprimés. Le processus requiert des ressources pour réaliser les activités. Concernant le génie logiciel, le processus intègre les activités principales suivantes :

- Les exigences logicielles
- La conception
- La construction
- Les tests
- La maintenance

Ces activités correspondent aux cinq premiers domaines de connaissance de SWEBOK (2004). Il est à noter que SWEBOK intègre aussi certains aspects de mesure et de contrôle à ces domaines de connaissance.

6.2.2.2 Besoins

Le processus de réalisation du produit requiert les besoins du client. Les besoins représentent, entre autres, certaines caractéristiques que le produit final doit avoir pour satisfaire les attentes du client. Pour le génie logiciel, les besoins peuvent être les exigences fonctionnelles et non-fonctionnelles exprimées par le client. Le processus aura

comme activité première de faire la synthèse des besoins exprimés et par la suite, de les transformer en spécifications.

6.2.2.3 Ressources

Le déroulement du processus requiert des ressources. Ces ressources sont, entre autres :

- La main d'œuvre
- Les matériaux (moins applicable au génie logiciel)
- Les outils (logiciel et matériel)
- Les normes
- Les méthodes
- Les livres de référence
- Les produits intermédiaires

Il est à noter qu'un produit intermédiaire issu d'un sous-processus est considéré, lui-même, comme une ressource lorsqu'il est utilisé par un autre sous-processus.

6.2.2.4 Produit

Globalement, le produit final est l'objectif ou le résultat attendu du processus du génie. Dans le cas du génie logiciel, le logiciel et le guide d'utilisation sont les produits finaux issus du processus de développement. Le processus de développement se découpe en sous-processus. Ceux-ci génèrent un ou plusieurs produits intermédiaires qui seront utilisés comme des ressources par les sous-processus suivants.

6.2.2.5 Contrôle

Le contrôle est le pôle orienté management et ingénierie où des décisions sont prises pour gérer le processus, le corriger et l'améliorer. Le contrôle s'assure que le processus se déroule normalement afin que les objectifs (les buts) soient atteints dans le respect des