1.2 Guide du corpus de connaissances en génie logiciel SWEBOK

1.2.1 Présentation du guide SoftWare Engineering Body Of Knowledge (SWEBOK)

SWEBOK est un guide au corpus des connaissances, c'est-a-dire, un ensemble de pointeurs vers les connaissances en génie logiciel. C'est un document de base de l'IEEE-Computer-Society pour la normalisation en ingénierie du logiciel. Il est le fruit d'une collaboration entre universités, industries et associations professionnelles soit :

- ✓ Associations professionnelles : IEEE Computer society et ACM (qui s'est retiré en 2000)
- ✓ Corporatif : Boeing, Conseil national de recherches Canada, etc.
- ✓ Académique : École de technologie supérieure, Université du Québec à Montréal

Le document a été commenté par plus de 500 professionnels et la version Trial du Guide (en anglais) est disponible depuis 2001. Depuis, le guide est resté libre au moins dans un format pour assurer une diffusion aussi large que possible. Les ingénieurs en logiciel dans le monde entier peuvent participer à l'élaboration du guide. N'importe qui peut s'inscrire comme réviseur.

Site : www.swebok.org

1.2.2 Activités du cycle de vie

Selon le SWEBOK les activités clés du cycle de vie d'un logiciel sont :

- ✓ L'analyse fonctionnelle
- ✓ L'architecture
- ✓ La programmation
- ✓ Les tests
- ✓ La validation
- ✓ La maintenance
- ✓ La gestion de projet

1.2.3 Intérêt et objectifs du SWEBOK

Le projet SWEBOK a pour but de formaliser de manière consensuelle le contenu de la discipline d'ingénierie du logiciel en plusieurs domaines distincts :

- ✓ Les exigences du logiciel
- ✓ La conception du logiciel
- ✓ La construction du logiciel
- ✓ Les tests logiciels
- ✓ La maintenance du logiciel
- ✓ La gestion de configuration du logiciel
- ✓ L'ingénierie de la gestion logicielle
- ✓ L'ingénierie des processus logiciels
- ✓ L'ingénierie des outils et méthodes logicielles
- ✓ L'assurance qualité du logiciel

Le SWEBOK s'adresse aux :

- Enseignants chargés de bâtir des programmes de l'enseignement supérieur et étudiants
- Entreprises privées et publiques : comme un guide de connaissances du domaine pour mettre en place des bonnes pratiques d'ingénierie du logiciel.

Le guide vise plusieurs objectifs, entre autres :

- Caractériser le contenu du Software Engineering Body of Knowledge;
- Promouvoir une vision cohérente du génie logiciel dans le monde entier ;
- Clarifier la place de, et régler la limite d'ingénierie logicielle par rapport à d'autres disciplines comme l'informatique et les mathématiques ;

1.3 Software Requirements (Les exigences logicielles - متطلبات البرامج)

L'espace de connaissance des exigences (ou besoins) logicielles est concernés par l'explicitation, l'analyse, la spécification, et la validation des exigences logicielles. Il est largement reconnu au sein de l'industrie du logiciel que les projets d'ingénierie logicielle sont extrêmement vulnérables lorsque ces activités sont mal réalisées.

Une exigence du logiciel est une propriété qui doit être présenté en vue de résoudre certains problèmes dans le monde réel.

A ce niveau, on étudie les besoins et la faisabilité du logiciel. Il n'y a pas de solution technique approfondie ; on ne définit que les besoins en termes de fonctions métiers et on apprécie la faisabilité informatique. Les acteurs qui interviennent sont : le client, l'analyste-ingénieur informatique.

Les éléments de l'étape expression des besoins sont :

- ✓ Document : cahier des charges
- ✓ Définition des besoins fonctionnels : modèle conceptuel
- ✓ Recensement des données
- ✓ Définition des besoins non fonctionnels
- ✓ Solution fonctionnelle
- ✓ Validation

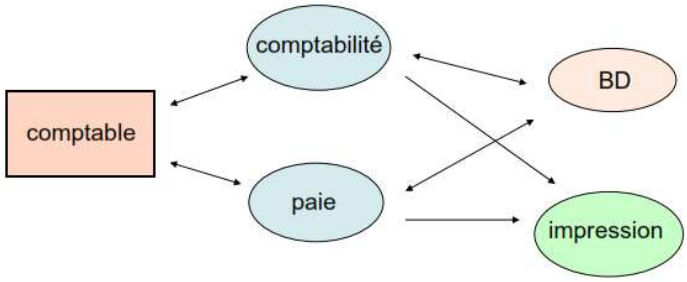
Le contenu du document en sortie comporte :

- ✓ Introduction
- ✓ Présentation du contexte, de l'existant
- ✓ Besoins fonctionnels
- ✓ Recensement des données, flux
- ✓ Besoins non fonctionnels, contraintes
- ✓ Approche de la solution
- ✓ Besoins en matériels

Le modèle conceptuel du système : approche fonctionnelle

– exemple des automates finis : commande, fonctions et données

– facilité d'une représentation graphique des fonctions que l'on affine d'étape en étape.



Définitions des besoins fonctionnels

- ✓ Définition des services
- ✓ Complétude et cohérence
- ✓ Modifications lors des étapes ultérieures

Il existe trois méthodes pour définir les besoins fonctionnels

- ✓ Langage naturel
- ✓ Langage structuré ou formaté
- ✓ Langage de spécification formelle

Langage naturel :

- ✓ très fréquent (compréhensible par tous)
- ✓ présentation par paragraphes numérotés

Exemple :

2.1. Paie

Cette fonction comporte trois fonctions :

- la saisie des éléments de paie par employé
- l'édition des bulletins
- la mise à jour de la comptabilité

2.1.1. La saisie

Pour chaque employé, un certain nombre de paramètres sont saisis...

Avantages :

- o compréhensible par l'utilisateur et l'analyste
- o facilité de rédaction

Inconvénients :

- o ambiguïté linguistique
- o manque de concision: schéma fonctionnel
- o difficulté de distinction entre besoins fonctionnels, non fonctionnels et buts
- o difficulté de distinction de complétude et cohérence

Langage structuré :

- ✓ utilisation limitée du langage naturel
- ✓ utilisation de notations graphiques

Exemple : La méthode ***SADT***

- ✓ Technique d'analyse structurée
- ✓ Graphique : structure et relations entre entités
- ✓ Symboles spéciaux faciles de compréhension
- ✓ Pas de traitement automatique des diagrammes

Langage de spécification formelle :

- ✓ exemple : langage ADA
- ✓ usage des commentaires

Exemple :

package PAIE is
procedure SAISIE_PAIE
procedure EDITION_PAIE
procedure COMPTA_PAIE
end PAIE

Validation des besoins

Elle est faites par les utilisateurs et développeurs pour vérifier :

– La cohérence

- pas de conflit entre les besoins

– La complétude:

- tous les besoins et contraintes sont recensés

–Le réalisme:

- réalisables avec la technologie matérielle et logicielle

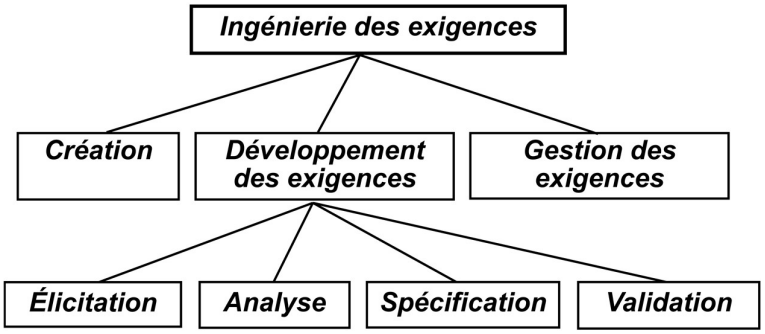
– La validité:

- réponse aux besoins
- des fonctionnalités supplémentaires si nécessaire

• ***Outils ou méthodes de validation***
– outils d'analyse et recherche d'anomalies
– générateur de simulateurs
– Prototype, Maquette

Le guide SWEBOK se réfère à des exigences sur le «logiciel» parce qu'il est préoccupé par les problèmes devant être traités par le logiciel. Par conséquent, une exigence du logiciel est une propriété qui doit être exposée par le logiciel développés ou adaptés pour résoudre un problème particulier.

1.3.1 Domaine de l'ingénierie des exigences (IE)



Les entrées sont les besoins ou exigences brutes spécifiés par le client; Les sorties sont les documents d'exigences: spécification, cahier des charges, devis, contrat, appel d'offres, norme, ...

Création: Début du processus (Apparition ou étude d'un besoin, d'un problème, d'une opportunité, etc)

Développement des exigences:

Élicitation: Recherche des exigences par des entrevues, la consultation des documents existants, l'analyse de tâches, le développement de prototypes, de maquettes, etc.

Analyse: Comparaison, Négociation, Filtrage, Structuration et Liaison des exigences, etc.

Spécification: Documentation des exigences;

Validation: Vérification et Validation des exigences;

Gestion des exigences: Implantation et Suivi de la traçabilité des exigences; Gestion des modifications / versions d'exigences (aussi appelée Gestion de la configuration).

1.3.2 Importance du domaine de l'ingénierie des exigences

Problèmes majeurs si négligée: L'IE est une activité très importante du processus de fourniture et d'acquisition. À tel point que, si elle est négligée, plusieurs besoins du client ne sont jamais compris par le fournisseur ou ne le sont qu'après ou peu avant la livraison. Il en découle les problèmes majeurs suivants:

- augmentation des coûts et délais de réalisation: la compréhension d'un besoin après ou peu avant la livraison implique souvent de recommencer la réalisation, au moins en partie;

- diminution de la qualité: l'incompréhension d'un besoin implique que le produit (ou service) ne répondra pas à ce besoin; et la compréhension d'un besoin après ou peu avant la livraison implique souvent que le produit ne répondra pas à ce besoin ou ne sera que sommairement corrigé pour y répondre le mieux possible.

Base de l'entente client-fournisseur: L'IE est une activité non seulement importante mais aussi essentielle à la fourniture et à l'acquisition. En effet, les exigences sont la base de l'entente client-fournisseur.

Base de fourniture et d'acquisition: De surcroît, elles sont la base de la fourniture et de l'acquisition: base de réalisation; base de vérification et d'acceptation par le client; base de documentation.

Selon le "Standish Group", un des plus importants cabinets d'études technologiques, les erreurs d'exigences sont:

- Les plus coûteuses;
- Un des principaux facteurs de difficulté ou d'échec des projets d'ingénierie logicielle.

Plus une erreur est introduite tôt et détectée tard, plus elle est coûteuse à corriger :

Phases d'ingénierie	Coût relatif de réparation d'une erreur
1) Ingénierie des exigences	1
2) Conception	5
3) Réalisation	10

4) Vérification et validation 20

5) Maintenance 200

1.3.3 Exigences fonctionnelles et non fonctionnelles

Les exigences fonctionnelles décrivent les fonctions que le logiciel doit exécuter. Les exigences non fonctionnelles sont celles qui agissent pour contraindre la solution.

Les exigences non fonctionnelles sont parfois connues comme des contraintes ou des exigences de qualité. Ils peuvent être classés selon qu'ils sont :

- des exigences de performance,
- des exigences de maintenabilité,
- des exigences de sécurité,
- des exigences de fiabilité,
- ou un des nombreux autres types de besoins logiciels.

1.3.4 Problèmes courants et solutions proposées

Parmi les problèmes généralement rencontré lors de l'étape IE, nous pouvons cité :

- **Exigences coûteuses**

  Budget sous-évalué: trop souvent le budget initial alloué à l'IE est largement dépassé et le final, jugé trop élevé.

  Exigences mal structurées: lorsque survient le moment de modifier une exigence, cela a des répercussions sur plusieurs autres exigences non clairement identifiées. Cela exige de revoir l'ensemble des exigences.

  Exigences incompréhensibles ou incorrectes: les exigences sont incompréhensibles ou incorrectes du point de vue du client ou du fournisseur. Cela exige de revoir les exigences à plusieurs reprises, constituant en effet une autre raison pour laquelle l'IE est une activité souvent coûteuse.

- **Exigences incompréhensibles**

  Exigences ambiguës: elles ont plusieurs interprétations possibles. Elles peuvent être claires pour le client mais ambiguës pour le fournisseur, ou inversement, le contexte du client étant différent de celui du fournisseur.

  Exigences difficilement retraçables: il est difficile voire impossible de trouver l'exigence source de laquelle elles découlent, en particulier lorsque cette exigence source est spécifiée dans un autre document.

- **Exigences incorrectes**

  Exigences inexactes: le produit n'a pas à répondre à ces exigences du point de vue du client et du fournisseur. Elles proviennent généralement d'une incompréhension du besoin ou d'un problème de gestion des modifications d'exigences.

  Exigences incohérentes: elles se contredisent ou utilisent des termes différents pour exprimer la même notion.

Exigences invérifiables: il n'existe aucune procédure acceptable permettant de les vérifier. Les exigences sont invérifiables, souvent en raison d'utilisation de termes vagues, à éviter dans tout énoncé d'exigence:

Compte tenu de l'importance de l'IE, il est souhaitable que ces problèmes soient résolus. Comme solution, on peut prévoir :

- **Allocation d'un budget suffisant** :

La solution consiste d'abord à s'assurer de l'allocation d'un budget suffisant: le budget alloué à l'IE représente généralement de 15 à 30% du budget total de développement, excluant la maintenance.

- **Adoption de principes de base** : Deux méthodes d'IE:

  - 1ère: Cas d'utilisation détaillés (Scénarios détaillés);
  - 2ème: Énoncés d'exigences ("Le système doit …");
  - Choix: selon le contexte;
  - Attention, ne pas faire le travail en double; si combinées, une méthode doit compléter l'autre;
  - Approche recommandée: démarrer avec la 1ère et poursuivre avec la 2ème.

- **Utilisation d'un outil**:

Vu que les principes sont lourds à suivre avec un traitement de texte, on préconise l'utilisation d'un outil spécialisé. Plusieurs outils commerciaux sont disponibles: DOORS, GenSpec, Analyst Pro, System Architect, etc.

Avantages: outils commerciaux puissants, offrant notamment des facilités de traçabilité des exigences, de conception et même de génération de code logiciel;

Inconvénients: en général, outils commerciaux complexes, coûteux, courbe d'apprentissage longue, outils peu flexibles quant aux formats des documents générés (anglais, …), etc.