

Ingénierie des besoins

Jeanine Souquières

LORIA – Université Nancy 2



Quelques références

- Etat de l'art paru dans ICSE 2000
Requirements Engineering in the Year 00: A Research Perspective, A.van Lamsweerde
- De nombreux travaux sur le sujets depuis 1970. Quelques auteurs
Zave, Jackson, Boem, Parnas, ...
- Conférence: RE
- Groupe de travail:AFIS, Association Française d'Ingénierie Système



Problèmes de l'IB

– Succès ou échec d'un développement se joue lors des phases amont du développement logiciel

« *requirements do not arise naturally ... need to be engineered and have continuing review and revision* »

(Bell and Thayer, empirical study, 1976)

– *coût d'une correction tardive = 200 fois coût d'une correction initiale*

(B. Boehm, 1981)



Problèmes de l'IB

- IB = phase difficile et critique
 - « *hardest, most important function of SE is deciding precisely what to build ...* »
- (Brooks, IEEE Computer, 1987)



Enquête américaine

- Succès = 16%
 - projets achevés dans les temps et les budgets, remplissant complètement le cahier des charges
- Partiel = 53%
 - surcoût (189%), retards, réalisations partielles
- Echecs = 31%
 - projet abandonné

***Standish group USA, 1995 (actualisée en 2001)
Echantillon de 365 entreprises et 8380 projets
<http://www.standishgroup.com.chaos.html>***



Les 10 premières causes d'échec

- *Pas assez d'informations de l'utilisateur* 13%
- *Exigences incomplètes* 12%
- *Changement exigences* 11%
- Pas assez de soutien de la direction 8%
- Incompétence technique 7%
- Manque de ressources 6%
- Attentes irréalistes 6%
- Objectifs confus 5%
- Phasage irréaliste 4%
- Nouvelles technologies 3%



Les erreurs d'analyse sont (1)

- les plus coûteuses

<u>Erreur détectée ...</u>	<u>Coût de réparation</u>
<i>Analyse</i>	<i>1</i>
<i>Conception</i>	<i>5</i>
<i>Codage</i>	<i>10</i>
<i>Tests</i>	<i>20</i>
<i>En exploitation</i>	<i>200</i>

- les plus nombreuses

- 67% : erreurs de spécification et conception
- 33% : erreurs de codage



Les erreurs d'analyse sont (2)

- les plus dangereuses

- Ariane 5
- Boeing 737 (Cali-Colombie)

- les plus tenaces

- 45% erreur de spécification (après livraison)



Enquête européenne

- Enquête auprès de 3800 entreprises européennes dans 17 pays différents

Problèmes principaux de développement dans

- L'expression des besoins (50 % des réponses)
- La gestion des besoins (50 % des réponses)

European Software Institute, 1996



IB couvre de multiples activités

- Analyse du domaine
 - compréhension
 - priorité
- Elicitation
 - capture
 - identification
- Négociation et agrément
- Spécification
 - formulation précise



IB couvre de multiples activités

- Analyse de la spécification
 - faisabilité
- Documentation des décisions
- Evolution
 - modifications (corrections, changement de l'environnement, nouveaux objectifs, ...)

Ingénierie des besoins: qui, quoi, où?

Clients ou maîtres d'ouvrage

Ingénierie des besoins

Développeurs ou maîtres d'oeuvre



Problèmes : pourquoi

- Activité conceptuelle cruciale, complexe, efforts importants de mise en œuvre
- Difficultés culturelles : de nombreuses personnes impliquées → **conflits de points de vue**
 - communication entre clients et développeurs → incompréhensions
 - clients pas toujours conscients de leurs besoins
 - développeurs n'ont pas la connaissance du domaine d'application



Problèmes : pourquoi

- Difficultés d'expression
 - Quoi exprimer
 - problème/solution
 - choix
 - détails techniques
 - Comment l'exprimer
 - sous quelle forme
 - dans quel langage: naturel, méthodes formelles, uses cases, ...



Questions posées par la modélisation

- Quels aspects modéliser?
 - Unités conceptuelles
- Comment les modéliser?
 - Composition des unités
- Comment définir précisément le modèle?
 - Informel, semi-formel, formel
- Comment raisonner sur le modèle?



En pratique

- Peu d'organisations utilisent des méthodes et des outils systématiques
- Problèmes de traçabilité entre documents
- Les modifications dans les besoins sont rarement contrôlées et mises à jour dans les documents initiaux



Quelques approches: systèmes d'information

- 1977: Ross and Shoman
- SADT
- Bubenko 1980
- Jackson 1978
- Entité-Association, Chen 1976
- RML
- ...



Introduction des agents

- Idée: composants actifs
 - affectation de responsabilité aux buts et contraintes
 - agents ont des obligations
 - *travaux de Feather 1987*



Raisonnement basé sur les buts

- But = un objectif que le système doit atteindre par la coopération d'agents
 - raffinement de buts (structures de graphes ET/OU): mécanismes de structuration
 - notion de conflits entre buts
 - liens d'opérationnalisation
 - *Kaos (Louvain-La-Neuve)*
 - *Outil: Objectiver (CEDITI)*



Points de vues et conflits

- Décrire différentes perspectives d'un système
- Problème d'adéquation et de complétude
 - détection des inconsistences
 - *travaux de Nuseibeh, Easterbrook, 1994*
 - *expression multiparadigmes: OMT, UML*



Elicitation basée sur les scénarios

■ Buts difficile à éliciter

- Scénario = suite temporelle d'événements entre logiciel et son environnement (exprimé par ex. à l'aide de PN)
- Scénarios partiels (problème de couverture), procéduraux (risque de sur-spécification), propriétés requises restent implicites
- *travaux de Jarque 98*



Jackson, Parnas, Zave

- Travail plus fondamental depuis 95
 - distinction entre propriétés du domaine et besoins (*ex: lois physiques de l'environnement ne sont pas des besoins*)
 - distinction entre besoins et spécifications
 - besoins exprimés en termes d'objets sur le monde réel, vocabulaire accessible par client. Notion de phénomènes de l'environnement, surveillés et contrôlés par le logiciel.



Jackson, Parnas, Zave

- spécifications en termes d'objets manipulés par le logiciel, vocabulaire accessible par programmeur (relation entre entrées et sorties du logiciel)
- Distinction entre besoins et hypothèses
 - besoins respectés par le système
 - hypothèses par les agents de l'environnement seulement (notion développée en Kaos)



Réutilisation : problem frame

- Idée : classer et caractériser des patterns de problèmes
 - représentés par un frame diagramme avec machine à construire, domaines d'application et relations
 - objectif: trouver le problème frame adapté, construire une solution basée sur les requirements
 - *bouquin de Jackson, 1995 « Software requirements & specification »*



Documentation de l'IB

Spécification du domaine et modèles des besoins sont des composants essentiels pour communication, négociation, évolution

- exprimer le processus de construction (arbre des tâches, pourquoi des décisions: travaux dans *projet Esprit2 Icarus, Nancy*)
- responsabilité des décisions → traçabilité (*Finkelstein 95*)



Vers une approche intégrée dans les phases amonts du cycle de vie

Idée de méthode (Heisel-Souquières)

- Les besoins font référence au système et à son environnement
- La spécification fait référence à la partie informatique
- Documents informels et formels
 - mis à jour durant le processus → traçabilité
- Indépendant d'un langage



Objectifs

- Comprendre le problème
- Fixer le vocabulaire (un nom par concept)
- Désambiguïser les besoins
- Détecter les incohérences
- Détecter les besoins manquants
- Etablir un document de départ adéquat pour la spécification du système informatique

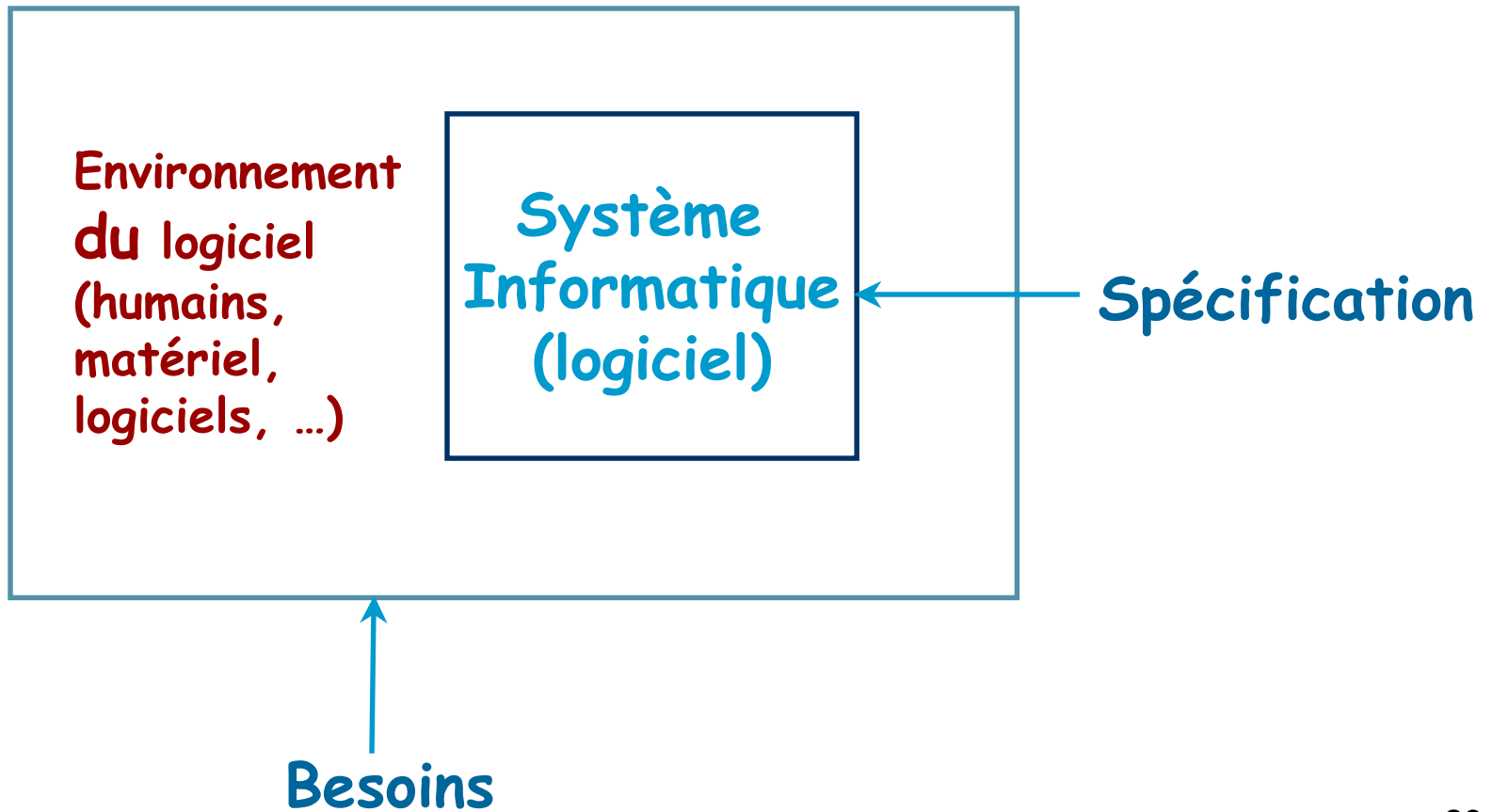


Eliciter les besoins

- Par fragments
- Les formaliser
 - pour les valider
 - valider la spécification/besoins
- Détecter les interactions
 - lorsqu'on rajoute de nouveaux besoins
- Supprimer les incohérences

Vocabulaire

Systeme





Quelques règles

- Les besoins devraient exprimer le **Quoi** et non le **Comment**
- Seuls les clients ont des besoins, ces besoins devant guider la spécification
- Tous les besoins exprimés par le client doivent être implantés



Idée: faire une distinction entre

- **Hypothèses** : certains besoins ne peuvent être imposés. Ils décrivent un comportement correspondant à un utilisateur parfait.

Le système peut ne pas être à même de les satisfaire.

Ex: le client prend ses billets (ATM)



Idée: faire une distinction entre

- **Faits**: propriétés du domaine d'application
Ex: le lecteur lit une seule carte à la fois
- **Besoins** du client: contraignent la spécification et l'implantation du système



Besoins/spécification-implantation

Dans la phase de spécification-implantation, montrer que la modélisation du système informatique :

- satisfait les **besoins**,
- en prenant en compte les **faits**,
- sous réserve que les **hypothèses** soient satisfaites.



Approche proposée (1)

1. **Fixer** le vocabulaire du domaine
2. Enoncer les **faits**, les **hypothèses** et les **besoins** du système:
 - fragment les plus petits possibles (portion de scénarios)
 - les nommer (traçabilité)
 - lien avec le document initial



Approche proposée (2)

3. Identifier les **opérations** pertinentes avec leurs paramètres
– systèmes transformationnels

4. Identifier les événements pertinents avec leurs paramètres
– systèmes réactifs

Rem: le vocabulaire introduit en 1. doit contenir exactement les notions introduites en 2, 3 et 4.



Approche proposée (3)

5. **Classer** les événements introduits

- Qui les commande?
- Qui les observe?

Rem: il n'y a pas d'événements commandés par le SI et non partagés par l'environnement

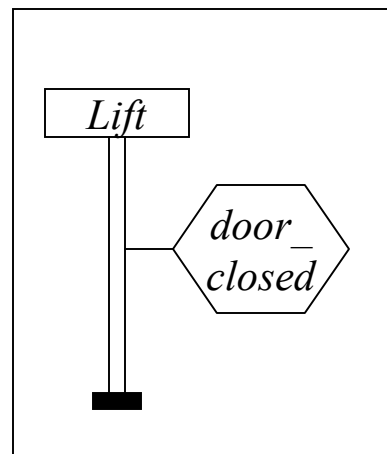
6. **Formaliser** faits, hypothèses et besoins comme des contraintes sur les traces possibles des événements du système ou par des LSCs.

Ex de formalisation: « la porte est fermée pendant le mouvement de l'ascenseur »

Ce besoin décrit un invariant du système:

$$\forall tr: Trace \bullet (\forall i: dom\ tr \bullet \neg halted(tr(i).s) \Rightarrow door_closed(tr(i).s))$$

AC: $\neg halted$





IB: que reste-t-il à faire?

- Passage entre IB et architecture de logiciels
- Passage entre IB et spécifications formelles
- Travail important sur les langages et notations
 - nécessité de disposer de techniques
 - pour guider l'élaboration incrémentale



IB: que reste-t-il à faire?

- quand et où passer de l'informel au formel
- quand et où passer de scénarios à modèles
- quand passer de différents points de vue à une documentation consistante
- ...

■ Ré-ingénierie des besoins

- documents trop pauvres et peu structurés
d'où difficultés de travailler avec →
abstraction, techniques de restructuration



IB: que reste-t-il à faire?

- Intégration de langages pour capturer multiples aspects des systèmes
- Outils
 - assister à la génération du document final de l'IB en
 - préservant la structure du modèle
 - extrayant des parties importantes
 - maintenant des liens de traçabilité