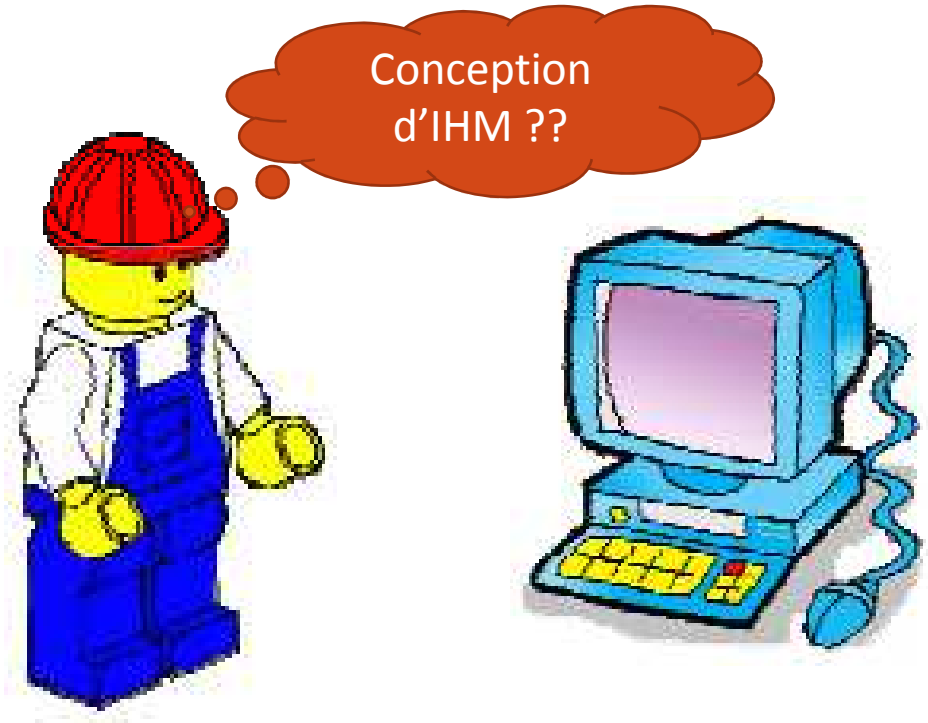
Chapter VI

Methodologies for Designing Interactive Systems



Outline

I. Introduction
II. Software Engineering Models
III. Seeheim model
IV. Steps in Designing a User Interface
V. User Interface Design Methods
VI. Norme ISO 13407 99
VII. Information Gathering Techniques
VIII. Conclusion

Une réflexion pour commencer …

J'ai toujours rêvé d'un ordinateur qui soit aussi facile à utiliser qu'un téléphone. Mon rêve s'est réalisé : je ne sais plus comment utiliser mon téléphone.

Bjarne Stroustrup
(Concepteur du C++)

I. Introduction

Beautiful buttons, menus and animations are not enough to make a system usable.

The objective of this course is to introduce you to the different methods and techniques for designing interactive systems and HMIs.

An useful and easy-to-learn interface amplifies positive feelings of success and control. It helps reduce training and time costs while also lowering maintenance costs.

Why a design method ?

- 97% of applications have a user interface (Brad A. Myers, 1995),
- Approximately 70% of the costs of interactive software are spent on user interface design (Bill Buxton, 1991).
- The risks of a poor interface are:

1. Clear reject by users,
2. Learning (training) costs,
3. Maintenance costs,
4. Loss of productivity,
5. Loss of credibility (fiabilité),
6. Incomplete use.

Why a design method ?

- 1979: US Government Accounting Office survey on software spending:
  - ✓ 2% for software delivered and used,
  - ✓ 25% for software never delivered,
  - ✓ 50% for software delivered but never used.
- 1995 : Software Engineering Institute Survey:
  - ✓ More than 1/3 of large-scale software development projects are canceled.
  - ✓ On average, a project takes twice as long as planned.
  - ✓ More than ¾ (three-quarters) of large-scale exhibit operational failures and do not function as originally intended.

➢ One solution proposed at the time:

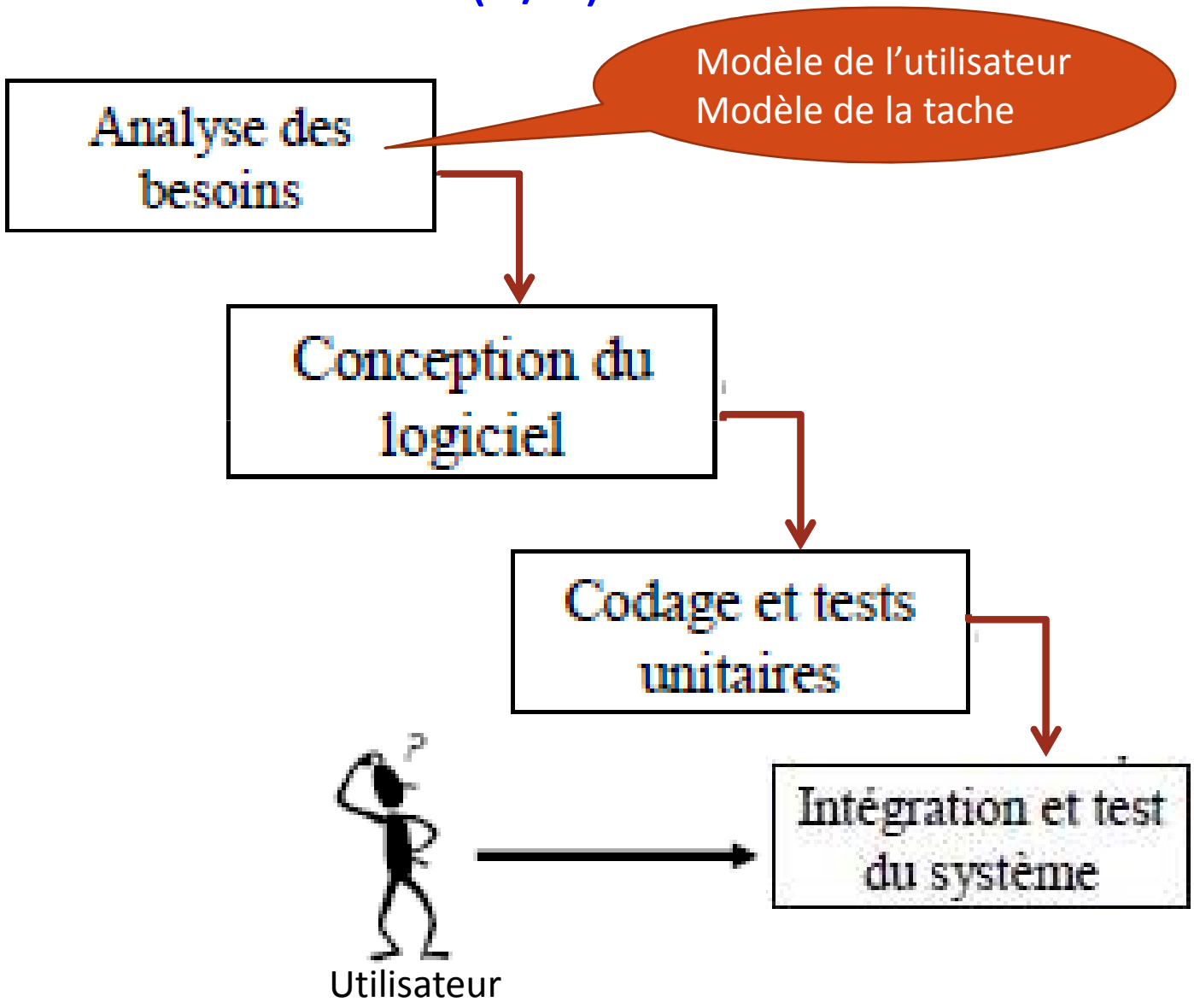
Software Engineering

II. Software Engineering Models

- The cascade model
- Waterfall model with iterations
- V-shaped model
- Spiral model
- Incremental model
- Iterative model
- Etc.

1. Modèle en Cascade (Royce 70)

(1/2)



1.Modèle en Cascade (2/2)

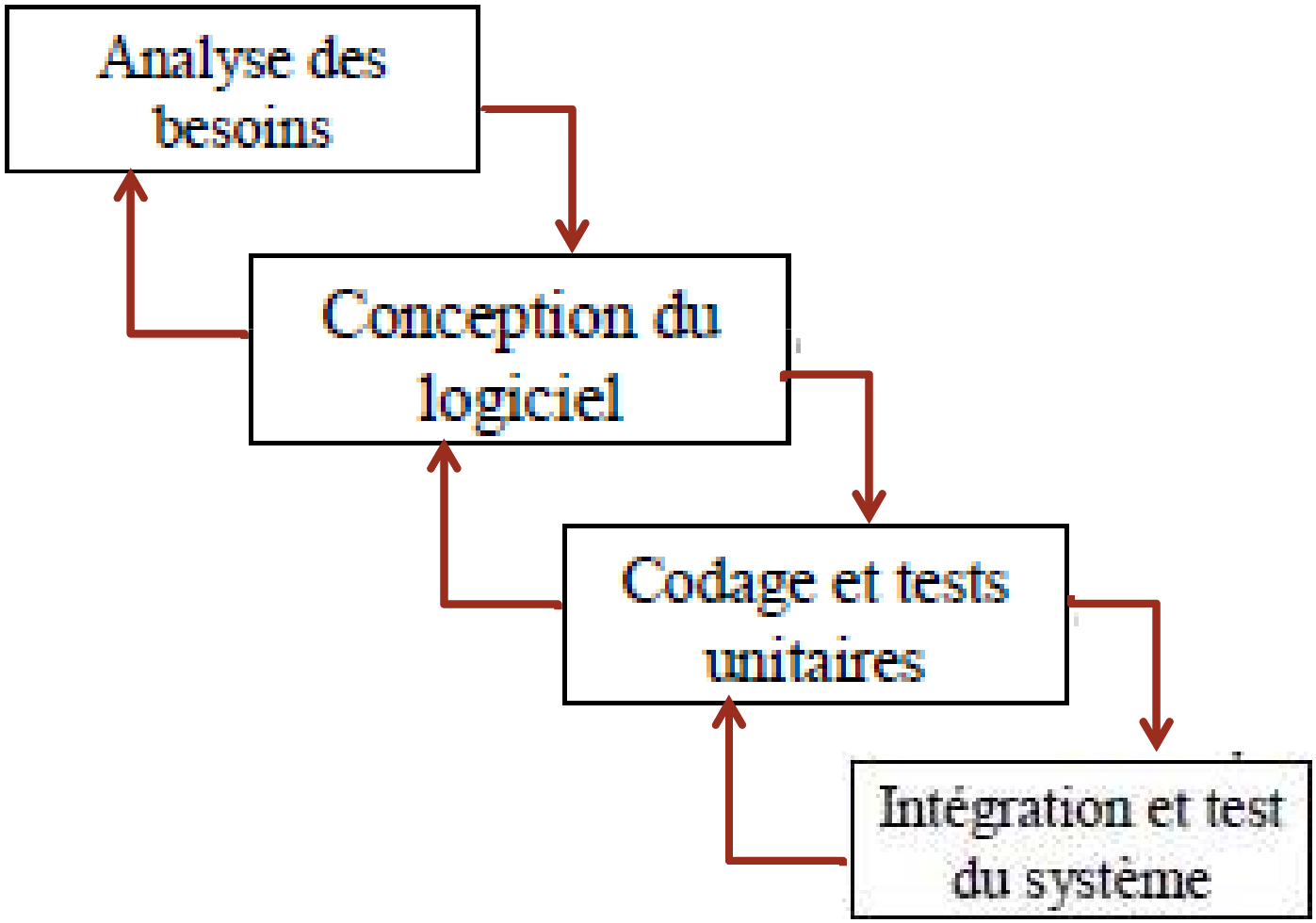
Principe

- Découpage en différentes activités prédéfinies.
- Documents livrés à chaque fin d'étape.
- Chacune de ces phases doit être terminée avant d'entamer la suivante.
- Un avancement séquentiel basé sur le principe de non-retour.

Avantages: Simple, "facile" à mettre en œuvre; idéal pour les projets stables.

Inconvénients: évaluation tardive, pas de retour en arrière, risque de rejet ou d'ajustements bricolés.

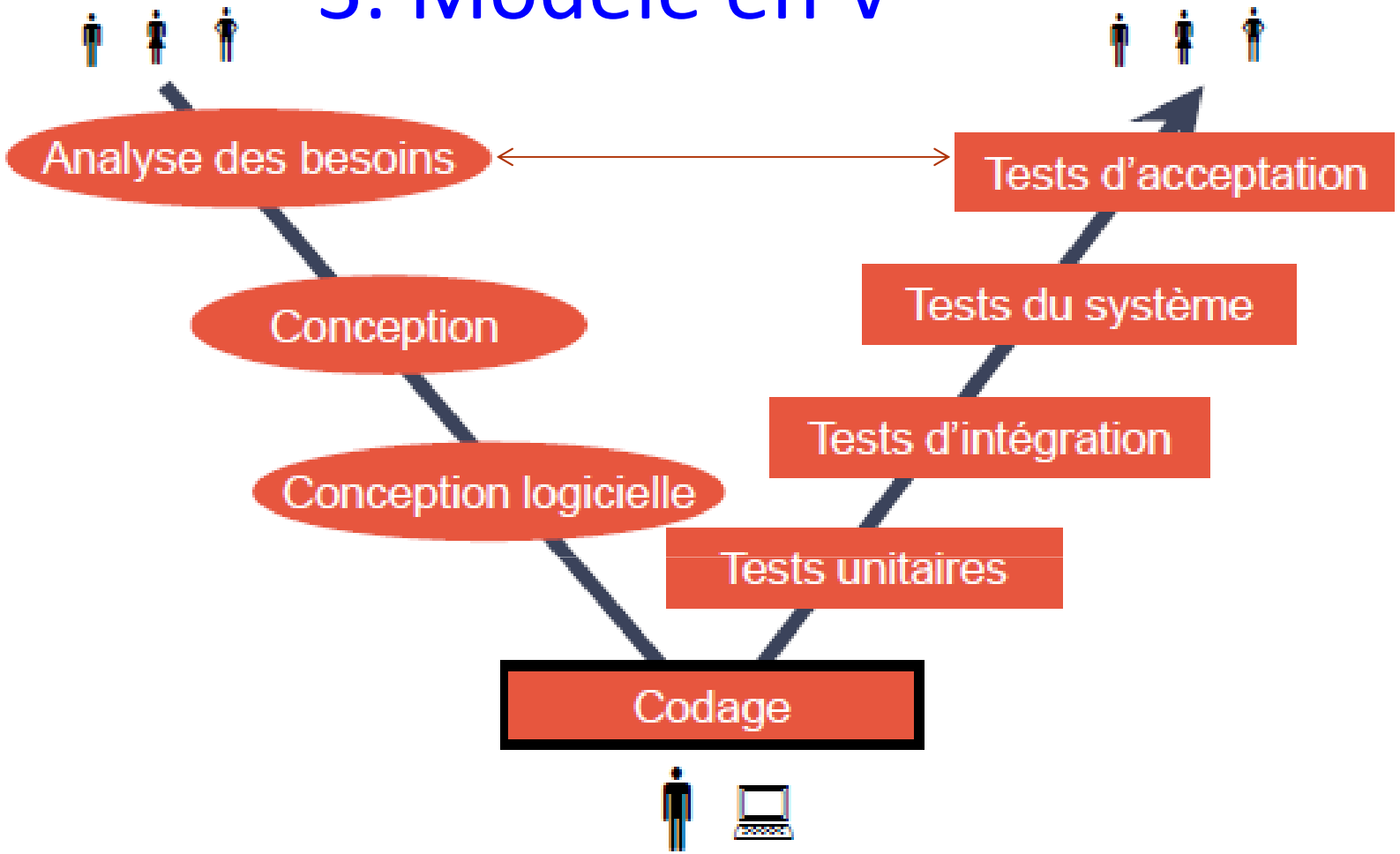
2.Modèle en Cascade avec itérations (1/2)



2.Modèle en Cascade avec itérations (2/2)

- Cycle de vie en cascade itératif
  - Etape suivante uniquement quand une étape est satisfaisante
  - Conception orientée vers l'implantation
  - Evaluation en dernier
  - Prise en compte de l'utilisateur trop implicite
- Modèle créé pour les grands projets
  - Importance des documents (cahier des charges, spécifications)
  - Signés par les clients

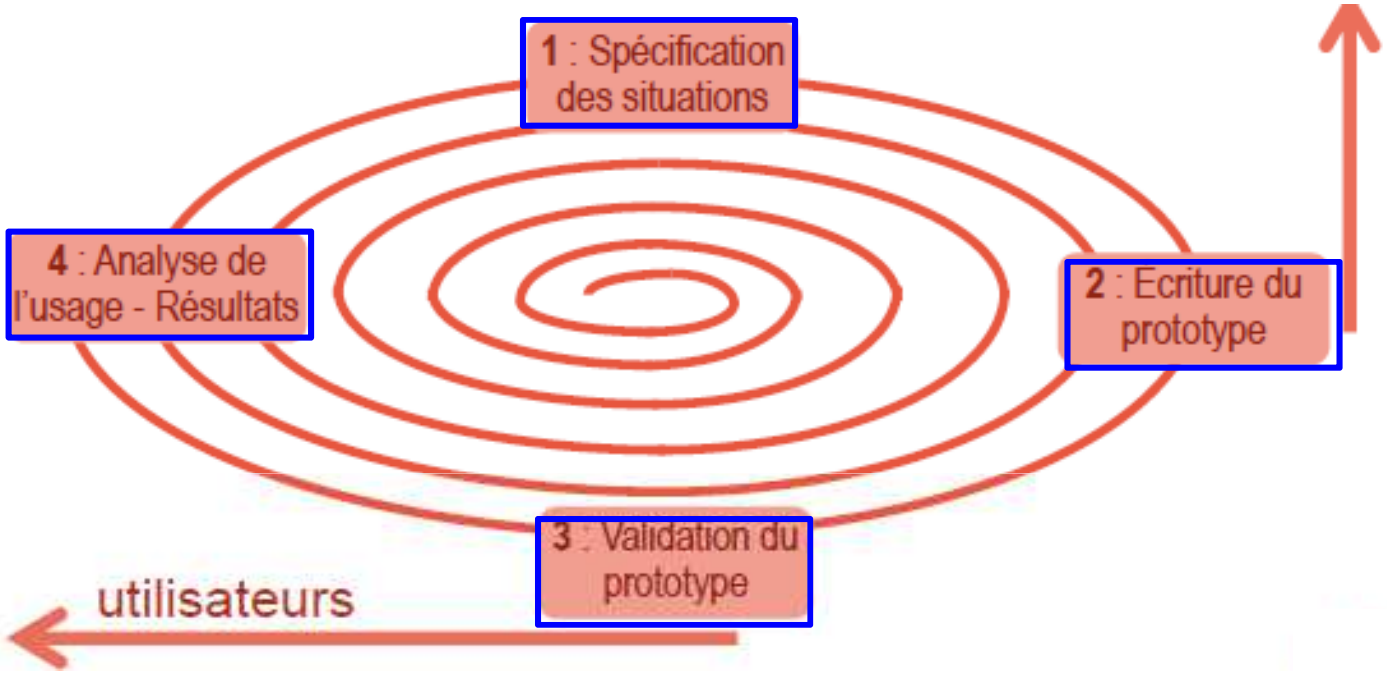
3. Modèle en V



- Devenu un standard de l'industrie logicielle depuis les années 1980.
- Adapté aux projets de taille et de complexité moyenne
- L'évaluation se fait seulement après le codage
- Le modèle ne précise pas la portée des retours arrière
- Implication utilisateur limitée au début et à la fin

4. Modèle en spirale

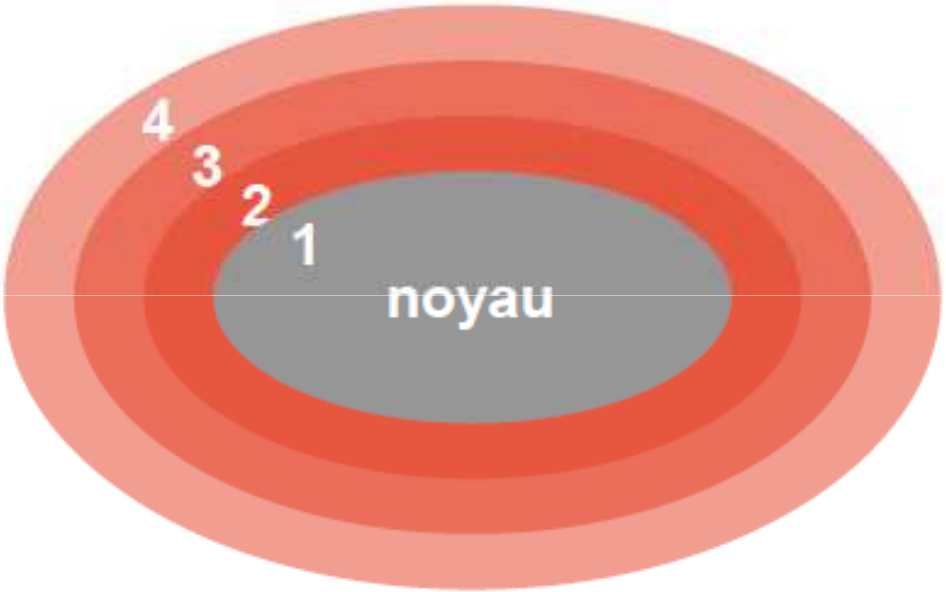
(Barry Boehm en 1988)



- Il propose des prototypes successifs
- Pour chaque cycle le modèle explicite
  - Les objectifs, alternative retenue et contraintes
  - L'analyse et résolution des problèmes
  - Le développement, validation et vérification de la phase
  - La planification de la phase suivante

5. Modèle par incréments

- On développe tout d'abord le noyau
- On ajoute petit à petit des fonctions



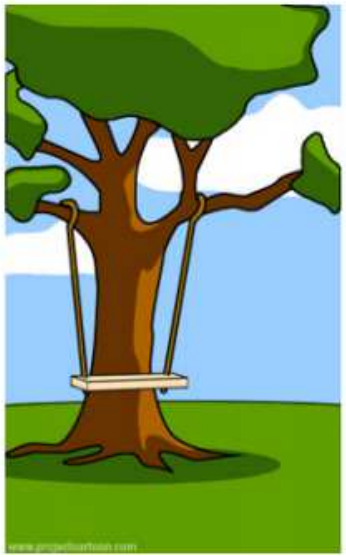
- Risques:
  - Rencontrer un problème pour l'ajout d'un élément
  - Remettre en question les éléments précédents
  - Voire même le noyau

Software Engineering Models: Review

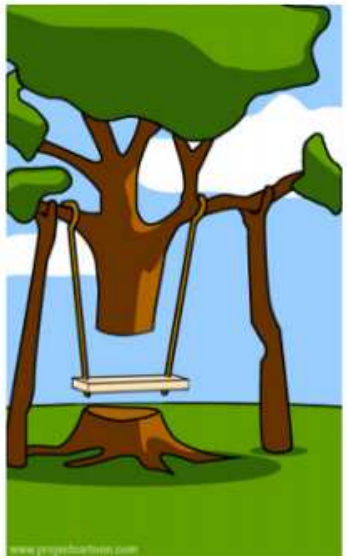
- Limited user involvement (primarily during analysis and evaluation),
- Principle of independence between the core functionality and the user interface:
  - ✓ The interface and interaction are defined only afterward. (In interactive software, this separation is not so clear).
  - ✓ It is essential to plan for usage at the same time as the functionalities.
- Late evaluation.

⇒ Méthode de conception spécifique pour les IHM

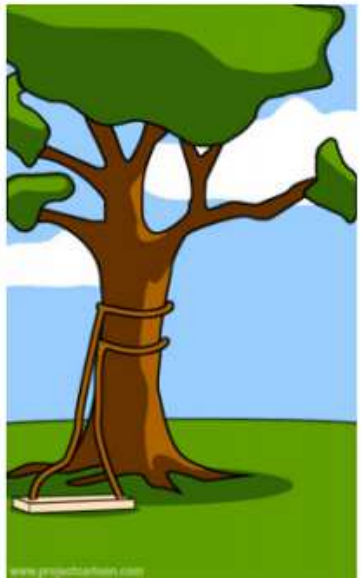
What the customer explained



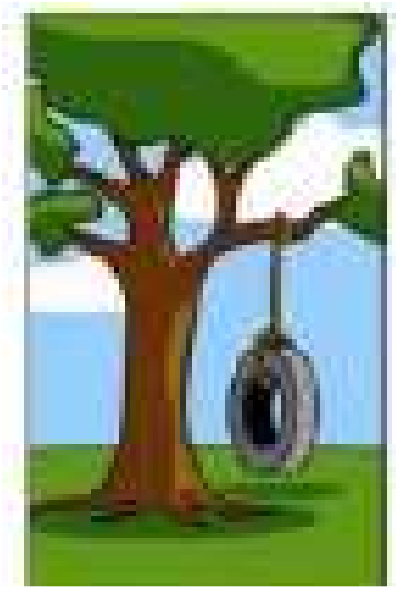
What the project manager understood



What the designer has designed



What the programmer wrote



What the client really needed

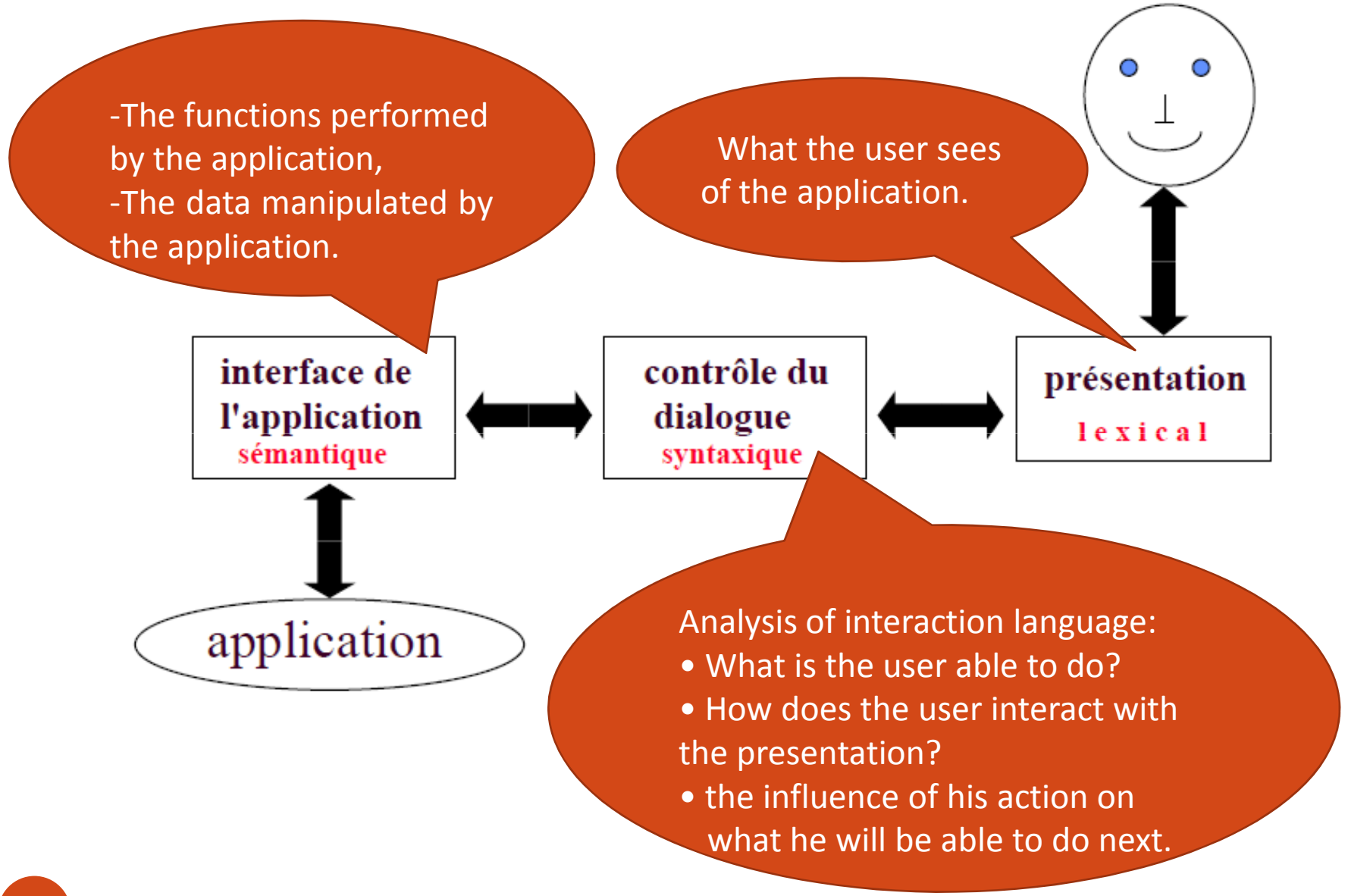
III. Seeheim Model(1/2)

Seeheim is a town in Germany where a workshop on "User Interface Management Systems" was held.

The Seeheim model is a software architecture pattern introduced in 1983 to structure the HCI in interactive software.

Given a user and a set of functions (called the functional core), the Seeheim model separates an application's user interface into three interconnected components :

III. Seeheim Model (2/2)



IV. Steps in Designing a User Interface

A user interface design methodology is (generally) divided into three phases:

1. Analysis: clarify user expectations and needs, understand their tasks and the context;

2. Development: create all or part of an interface (in a more or less complete form);

3. Evaluation: measure the usability of the completed interface (performance, user satisfaction, ease of use), identify areas for improvement in the next version, etc.

User: Multiple profiles, varied characteristics

Task: User objective (e.g., searching for a book)
Repetitive, regular, occasional, sensitive to environmental changes, time-constrained, risky, etc.

Context: Environment and usage constraints

- General public (offer immediate usability),
- Leisure (make the product appealing),
- Industry (increase productivity),
- Critical systems (ensure zero risk), etc.
- Technical (e.g., platform, memory size, screen, sensors, reuse of old code)

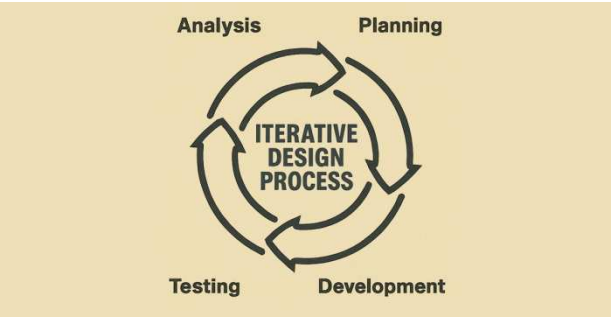
V. User Interface Design Methods

Different existing methods (inspired by software engeneering design methods):

1. Iterative design,
2. Incremental design,
3. Design by prototyping,
4. User-centered design (UCD),
5. Participatory design,
6. Design by personas and scenarios.

1. Iterative Design

Is a process where designers continuously improve a product by repeating cycles of testing and refining.

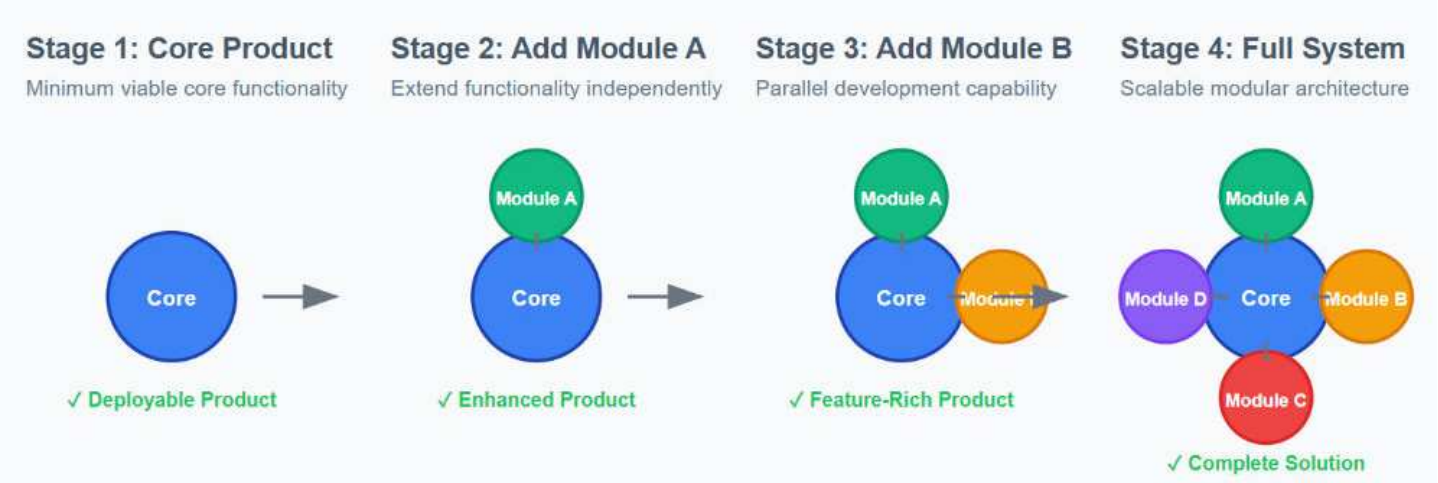


The typical steps of iterative design are as follows:

1. Complete an initial interface design
2. Present the design to several test users
3. Note any problems had by the test user
4. Refine interface to account for/fix the problems
5. Repeat steps 2-4 until user interface problems are resolved

2. Incremental design

- It's an iterative approach that focuses on building and improving systems through small, manageable segments or increments,
- Allows for easier maintenance and scalability because every module is developed and tested on its own before integration.
- It helps in reducing risks, improving user feedback, and enhancing flexibility by allowing modifications at each stage of the development process.



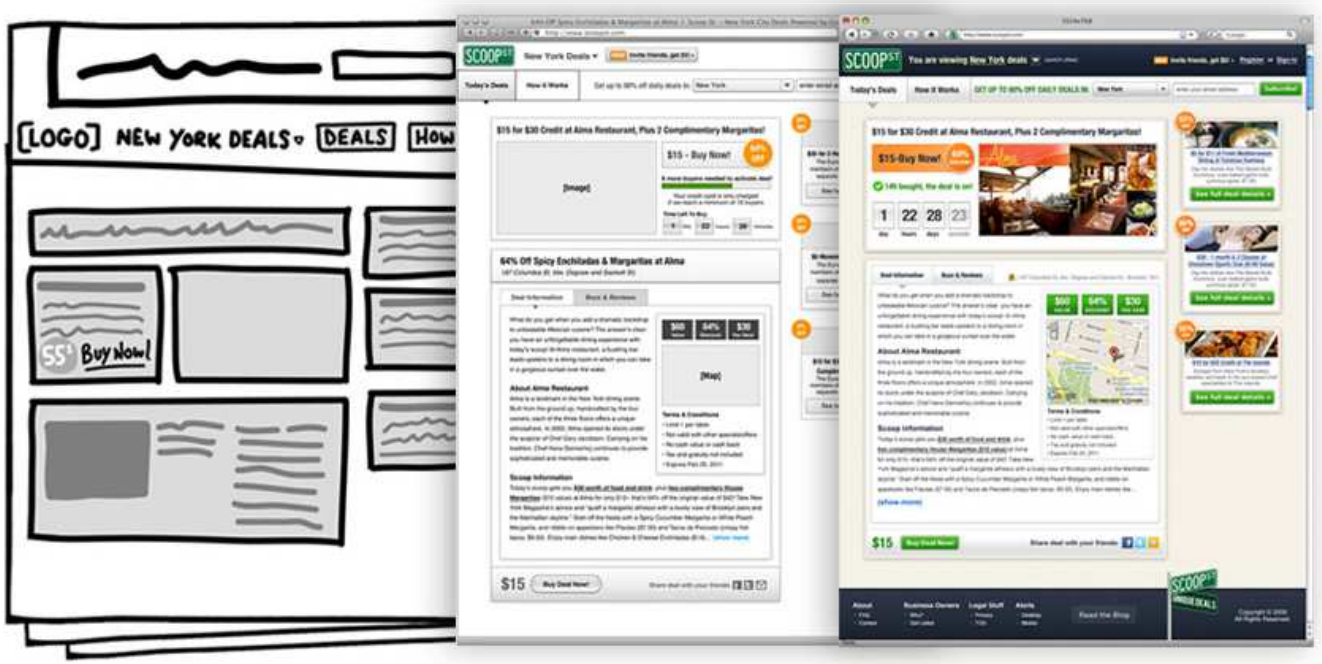
3. Design by prototyping

de la maquette

au prototype

à jeter ou réutilisable

vers le produit final



3. Design by prototyping

- Involves building simplified versions of a product to quickly gather feedback, validate concepts, and identify usability issues early on.
- Ranging from low-fidelity paper mock-ups to high-fidelity digital simulation,
- Helps to reduce development costs by allowing designers to make changes to prototypes, which is faster than altering code.

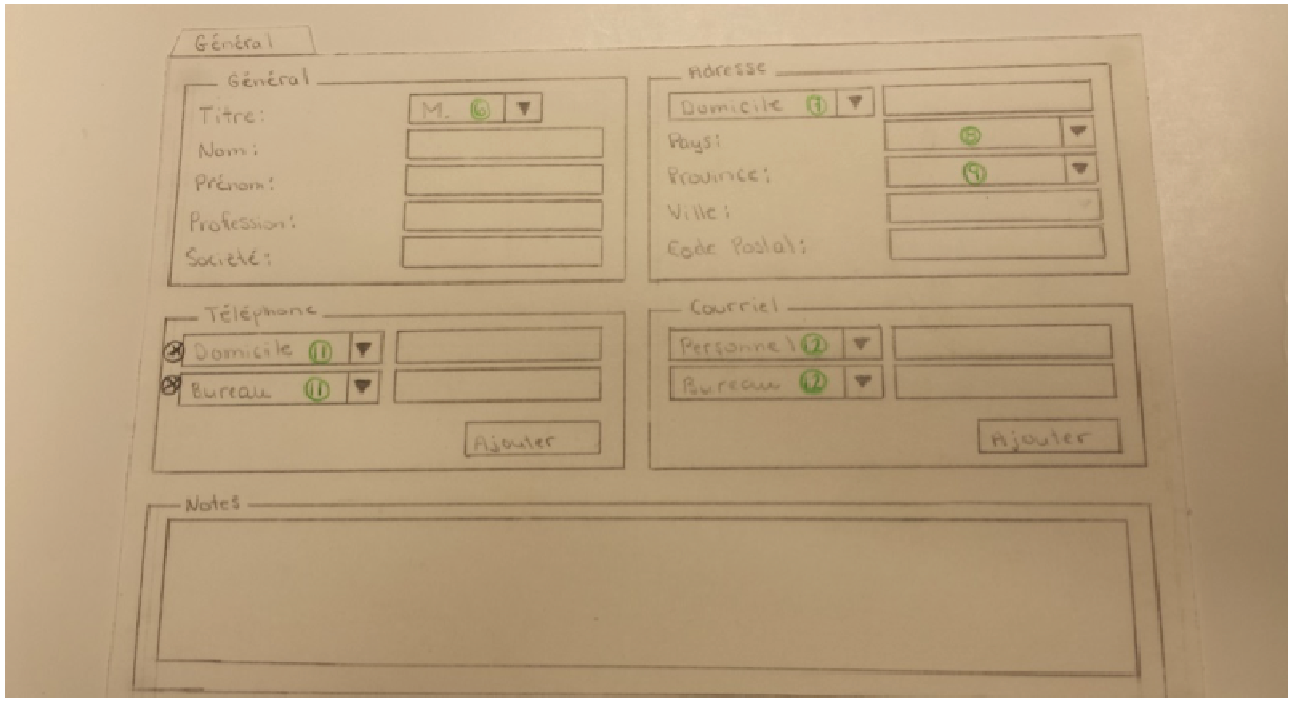
1. Mock-up

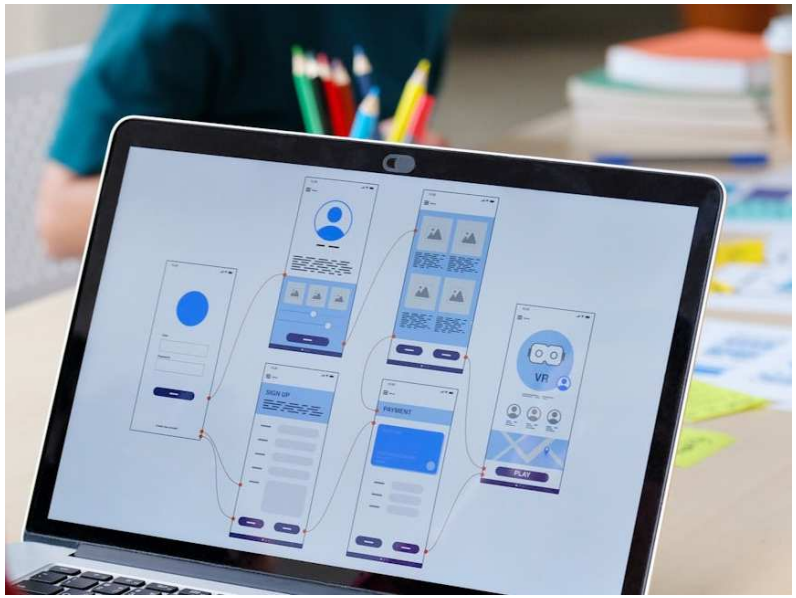
Definition

- A set of graphical objects providing an image of the user screen,
- Communication support between designers (initial phase),
- Contains no data access, no calculations.

Advantages

- Accessible representation for non-IT professionals,
- Avoids detecting anomalies too late.


Général
Général
Titre:
M.
Nom:
Prénom:
Profession:
Société:
Adresse
Domicile
Pays:
Province:
Ville:
Code Postal:
Téléphone
Domicile
Bureau
Ajouter
Courriel
Personnel
Bureau
Ajouter
Notes


VR
SIGN UP
PAYMENT
PLAY

2. Prototype

Definition

- Reduced representation of a part of the system for demonstration purposes,
- Full development of the interface for certain tasks,
- Full functionalities (calculations, data access).

Benefits

- Communicating with clients,
- Verifying technical feasibility,
- Validating a technical solution,
- Measuring response time.

Outils d'aide au prototypage :

- Papier, post-its
- Transparents, vidéo (e.g., Libre Office Impress)
- Logiciels de maquettage
  - haute fidélité, i.e., avec interactions (e.g., Invision, Maqetta)
  - basse fidélité, i.e., seulement des liens entre écrans (e.g., Mocking Bird, Pencil, Balsamiq)
- Logiciels de développement (e.g., frameworks web, Netbeans, Visual Studio)

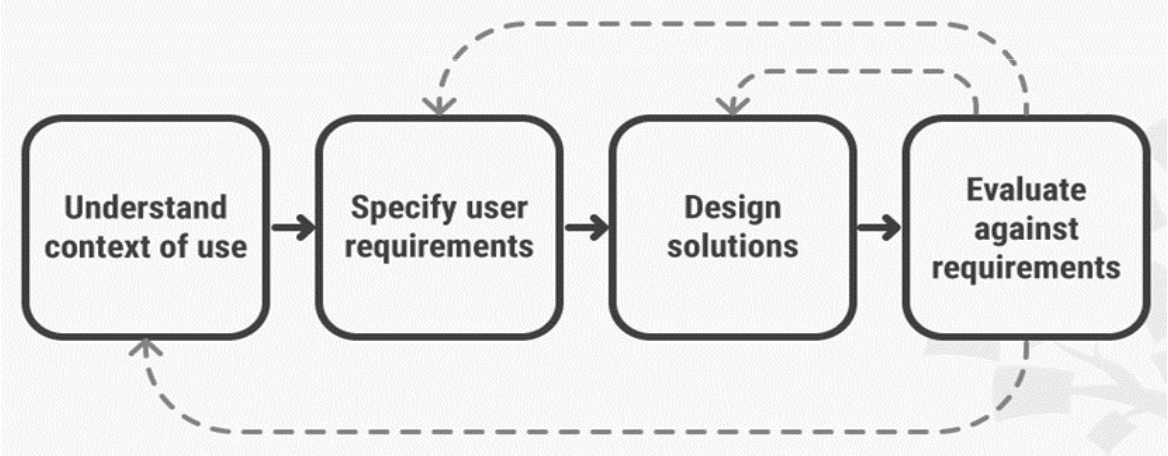
Lien qui regroupe plusieurs outils de prototypage:
https://dzone.com/articles/12-kick-ass-software-prototyping-and-mockup-tools

4. User-Centered Design

- The term *user-centered design* was coined by Rob Kling in 1977,
- Popularized by Donald A. Norman in 1988

Principle

- UCD is an iterative design process that prioritizes the needs and preferences of the end-user at every stage of product development.



- UCD involves users through research, testing, and feedback loops to ensure the final product aligns with user requirements, rather than forcing users to adapt to the product.

4. User-Centered Design

Challenges

- Choosing representative and available users,
- Real-world usage context,
- Clearing the behaviors, knowledge involved, etc.

Techniques for collecting information from users

1. interviews and surveys,
2. focus groups,
3. observation for direct user input, as well as document analysis,
4. brainstorming, and
5. benchmarking for gathering information about existing systems and user needs.

Examples of the contribution of UCD

- **Site web IBM:** 1 semaine après son « redesign »

  aide: -84%, et vente: +400%
- **Vente en-ligne** : 39% d'échecs à cause du design

- **Production de logiciel** : 84% d'échecs (temps, fonctions, budgets)

- **63% d'excès du budget**
  - Demande de changements par utilisateurs, taches trop grandes, manque de compréhension des propres requis des utilisateurs, mauvaise communication entre utilisateurs et concepteurs.

- **Maintenance**:
  - 33% de bugging, 67% changements par utilisateurs,
  - 40 à 100 fois plus cher de changer l'interface plutôt que de le prévoir pendant la conception.

4.1. User Model

Identify relevant user characteristics:

1.General data

- ✓height, age, disabilities, etc.
- ✓education level, cultural habits (date format, writing direction), etc.

2. Application-related data

- ✓ skills in the field,
- ✓ computer skills and system knowledge (beginner, expert, occasional, daily use)

4.2. Task Model

Definitions

- Task
  - goal: what needs to be done
  - procedure: is a set of subtasks
    - linked by composition relationships
    - linked by temporal relationships

- Elementary task

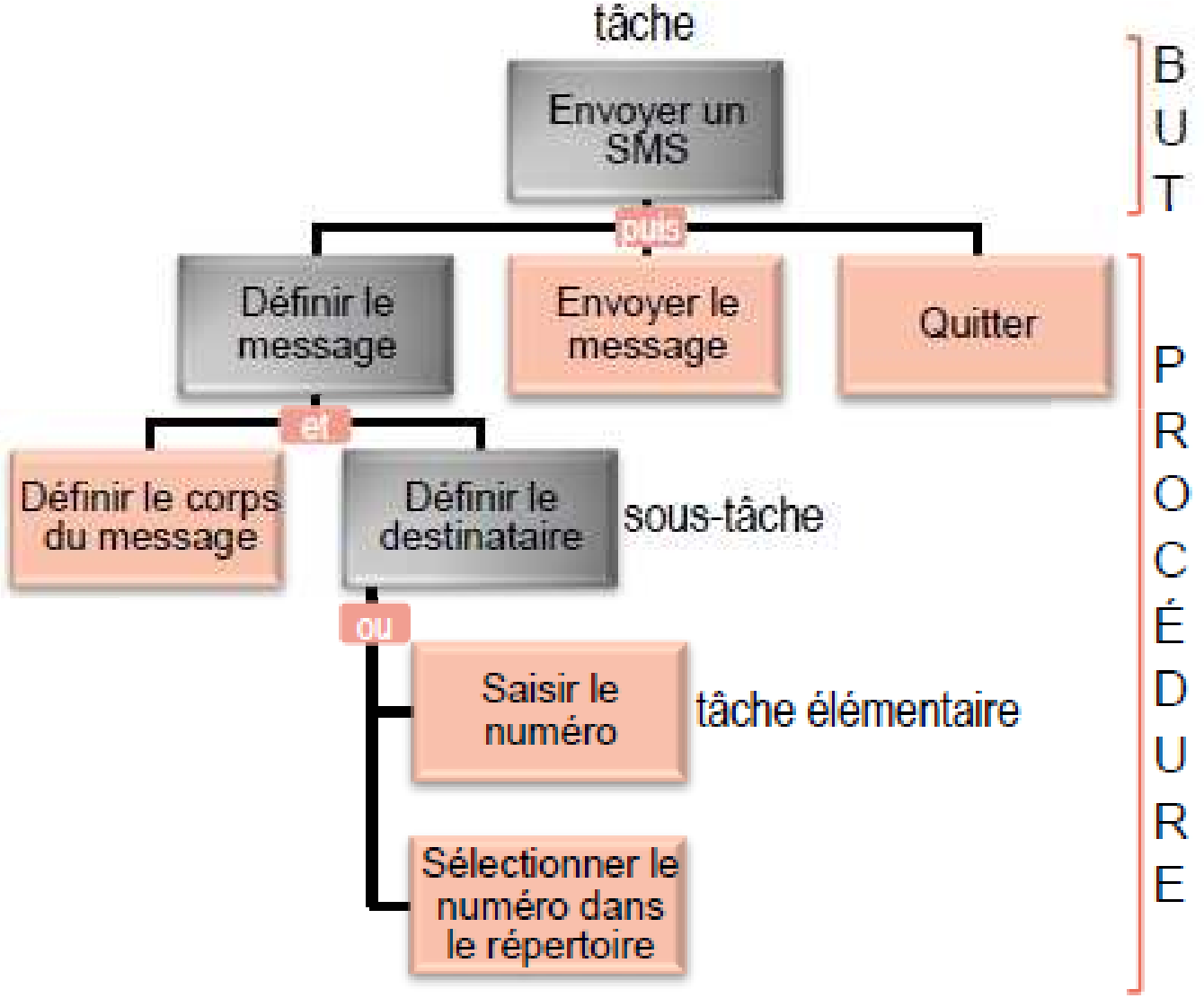
  Task decomposable into physical actions: I/O operations

Methode

- Build the system's task hierarchy,
- specify each task, consider exceptions,
- evaluate the decomposition with the user.

4.2. Task Model

Example: send a message



4.3. Interaction Model

- Establish a direct correspondence between:
  1. computer objects ex : a file
  2. interaction and presentation objects
     ex : file representation in the screen:
     - closed: iconic representation
     - open: content representation

     file operations:
     - modification
     - deleting, etc.

- This correspondence must
  - appear as "natural"
  - be part of an overall coherence: metaphor

5. Design by Personas and Scenarios

Definitions

Is a user-centered design method that uses fictional user representations (personas) and goal-oriented stories (scenarios) to guide product development.

Personas make complex user groups tangible by creating profiles with demographics, goals, and pain points,

Scenarios describe how these personas would use a product in specific situations to achieve a goal.

Personas

- are fictional user to represent the different user types
- help to understand users' needs, experiences, behaviors, motivations and goals.

Scenario

- a descriptive narrative that outlines a specific instance of interaction between a user and a system to achieve a goal.
  Specify the user's purpose, context, and a sequence of actions.

Vous pouvez créer des personas via ce lien: https://personagenerator.com/

Elodie

Persona primaire

Ses attentes

- Travailler au calme.
- Pouvoir par moment travailler avec ses camarades.
- Utiliser son PC portable.
- Accéder au Wifi.
- Brancher son ordinateur.
- Accéder à des revues ou livres sur place.

Ses frustrations

- Pas assez de prises de courant disponibles.
- Mauvaise connexion Wifi à certains endroits de la BU.
- Pas de pause café possible en dehors d'un espace enfumé.
- Peu d'espaces de travail en groupe.

Personnalité

Introvertie — Extravertie

Conservatrice — Créatrice

Passive — Active

Non fumeuse — Fumeuse

Isolée — Sociale

Technologies utilisées

Smartphone

Ordinateur portable (MacBook)

Tablette

Internet

Bio

Elodie est en L3 de Lettres modernes. Elle a d'abord réalisé ses deux premières années universitaires à Nancy, puis s'est inscrite à l'Université de Metz. Certains jours, en fonction de son emploi du temps, elle dispose de 2 à 5 heures de "creux" pendant lesquelles elle aime aller à la BU pour travailler. Il lui arrive même de venir le samedi matin, afin d'éviter de déranger (et d'être dérangée par) les colocataires de son appartement qu'elle loue en centre ville.

Âge : 22 ans
Travail : Job étudiant au Mc Do
Situation : Célibataire
Domicile : Metz
Filière : L3 Lettres modernes

Lectrice

Allia : la prof

65 ans, Divorcée 3 enfants
Retraitée, ancienne professeur de Français, Boulogne

« La littérature française aurait bien besoin d'un petit remontant»

Biographie

Allia a été professeur de Français au lycée de Thiers pendant 40 ans. Plutôt découragée par le niveau d'orthographe et le faible vocabulaire de ses élèves, elle a tout de même continué à transmettre sa passion.. Avec un certain succès puisque plusieurs de ses élèves sont devenus des écrivains à succès. Aujourd'hui à la retraite, elle dévore des livres à longueur de journée. Elle est souvent déçue mais parfois un auteur ravive sa flamme.

Sites clefs

- LeMonde.fr
- Picasa
- Projet Gutenberg (soutien pas utilisation)

N'aime pas

- Les fautes d'orthographes
- Les gens malpolis
- San Antonio

Pratique informatique

- Dialogue avec Skype depuis que son fils lui a installé
- Evite les réseaux sociaux, trop souvent bourrés de fautes et de stupidité

Attente

- Aider de jeunes auteurs à s'améliorer
- Satisfaire sa boulimie livresque
- Dialoguer avec des gens civilisés et cultivés

En conclusion

Allia sera exigeante sur le contenu et la forme du site mais pourra beaucoup s'investir et faire avancer les auteurs.

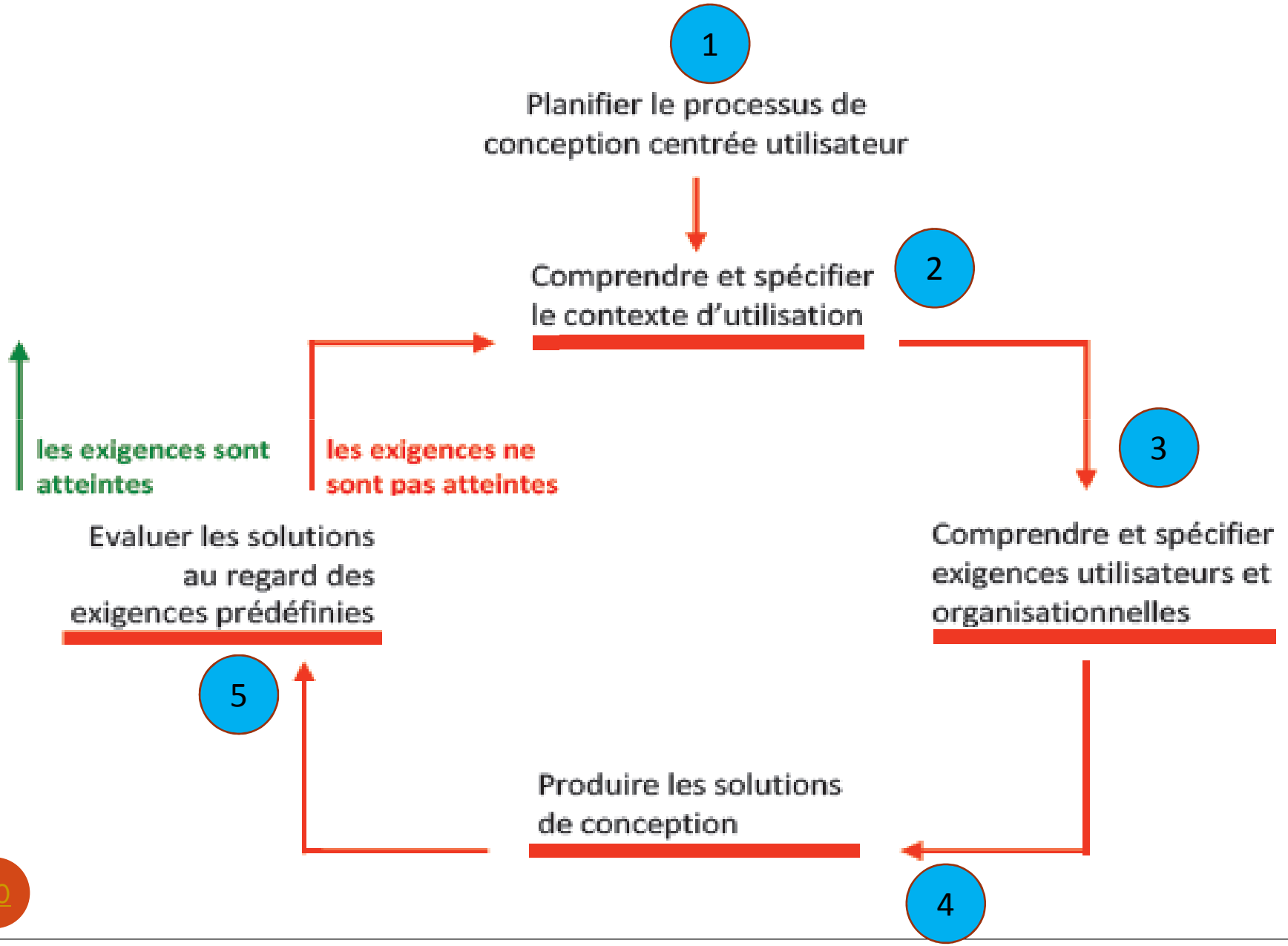
Advantages of persona-based design

- ✓Cognitive empathy (understanding another person's states or beliefs),
- ✓Particularly applicable to the Web / large scale.

Disadvantages of persona-based design:

- ✓Poorly defined personas lead to failure,
- ✓Distance from real users,
- ✓Need to modify personas in response to new findings or a different environment.

VII. ISO 13407 standard for UCD



VII. ISO 13407 standard for UCD

1. Planning the UCD process

   Adapting tools and methods based on consulting documentation and discussions around design practices and constraints.

2. Specification of the used context

- Understand the characteristics, goals, and tasks of the users, as well as their usage environment.
- Describe the environment from a technical, material, social, organizational, and legal perspective.

VII. ISO 13407 standard for UCD

3. User requirements specification

Consider the needs, skills and working environment of all stakeholders in the system.

4. Production of solutions

Use the knowledge from the previous steps to perform prototypes.

5. Evaluation of solutions

- Use the prototypes created to evaluate the solutions designed based on the requirements.
- The evaluation helps to detect defects.

VIII. Information Gathering Techniques

A user interface design methodology requires gathering information about users, their tasks, and interface evaluations.
Several techniques exist, distinguished by their goals and procedures:

1. **Design scenario**
2. **Cognitive inspection**
3. **Surveys / Interviews**
4. **Observations**
5. **Focus group**
6. **Quiz**
7. **Brainstorming**
8. **Audit ergonomique**
9. **Etc.**

VIII. Conclusion

- ✓ User-centered design,
- ✓ Consider potential users from the outset,
- ✓ Create a multidisciplinary team,
- ✓ Plan evaluations in each phase of the design.

Questions …