الكلية متعددة التخصصات - ورزازات
ⵜⴰⵖⵍⵉⵜ ⵜⴰⵣⵜⵣⴰⵡⵉⵜ – ⵡⵓⵔⵣⴰⵣⴰⵜ
FACULTÉ POLYDISCIPLINAIRE DE OUARZAZATE

Génie logiciel

Sciences Mathématiques et informatique
S6

Professeur : Abdelhadi Bouain

Introduction

- Un système informatique:
  - Matériel + logiciel

- Un système d'information:

  - Le système d'information est un ensemble organisé de ressources (matériel, logiciel, humain) qui permet de collecter, stocker, traiter et distribuer de l'information au sein d'une organisation.

  - Le système d'information vise à s'aligner avec la stratégie métier (processus métiers) de l'entreprise.

Introduction

- Systèmes informatiques :
  - 80 % de logiciel
  - 20 % de matériel
- Depuis quelques années, la fabrication du matériel est assurée par quelques fabricants seulement.
  - Le matériel est relativement fiable.
  - Le marché du matériel est standardisé.

Les problèmes liés à l'informatique sont essentiellement des problèmes de logiciel.

Introduction

- La crise du logiciel (fin des années 1960)
- Exemples:

  - L'abandon du projet d'informatisation des opérations boursières « la Bourse de Londres ». Projet Taurus (lancé en 1981). Plus de 100 millions de livres (**1 252 298 027 dh).**
  - bug de l’an 2000.
  - 1ère sonde Mariner vers Vénus : perdue dans l’espace (erreur Fortran).
  - Etc.

Introduction

- Étude sur 8 380 projets (Standish Group, 1995) :
  - Succès : 16 %;
  - Problématique : 53 % (budget ou délais non respectés, défaut de fonctionnalités) ;
  - Échec : 31 % (abandonné).

Le taux de succès décroît avec la taille des projets et la taille des entreprises.

- Comment faire des logiciels de qualité ?
- Qu'attend-on d'un logiciel ? Quels sont les critères de qualité ?

Naissance du « Génie Logiciel »

- **P. Naur and B. Randell -** La conférence du NATO « OTAN » **1968.**
- Naissance du « ***Génie Logiciel*** » (**software engineering**).

« Le génie logiciel **cherche à définir et à appliquer des méthodes, des outils et des pratiques propres à assurer la production de logiciel répondant à des besoins spécifiés et respectant certains critères de qualité**, ainsi que **les délais et les coûts** »

Naissance du « Génie Logiciel »

- Les qualités d'un logiciel : (La norme **ISO/IEC 25010:2011**)

  - **La capacité fonctionnelle:** respecter les spécifications et surtout répondre aux attentes de l'utilisateur + l'interopérabilité + la sécurité.
  - **Fiabilité :** la capacité d'un logiciel de rendre des résultats corrects quelles que soient les conditions d'exploitation.
  - **Facilité d'utilisation.**
  - **Performance:** (temps de réponse, consommation des ressources,…)
  - **Maintenabilité: (**peu d'effort nécessaire pour y ajouter de nouvelles fonctions)
  - **La portabilité.**

Méthodologies de développement des logiciels

Le cycle de vie d'un logiciel

- Comme pour tout produit manufacturé complexe :
  - on **décompose** la production en « **phases** »
  - l'ensemble des phases constitue un « **cycle de vie** »

Le cycle de vie d'un logiciel

- Définition des besoins
- Analyse et spécification des besoins
- Planification
- Conception
- Programmation
- Intégration
- Vérification et validation
- Maintenance

Le cycle de vie d'un logiciel : Analyse des besoins

- Besoins fonctionnels
  - Description des services(fonctions).
  - Description des données manipulées
  - "Comment souhaite-on pouvoir utiliser le système".
- Besoins non fonctionnels
  - Description des contraintes
  - Pour chaque service et pour le système global, il est possible d'exprimer différents types de contraintes:
  - contraintes de performance
  - contraintes de sécurité
  - Contrainte de convivialité et d'apparence
  - Etc.

Le cycle de vie d'un logiciel : Analyse des besoins

- Comment identifier les besoins ?
  - Entretiens, questionnaires.
  - Observation de l'existant (documents , logiciels).
  - Etude de situations similaires.

- Définir Chaque besoin (objectif) en s'assurant qu'il soit (SMART) :
  - **S**pécifique (clairement défini),
  - **M**esurable (chiffrable),
  - **A**tteignable (en tenant compte des ressources nécessaires disponibles par exemple),
  - **R**éaliste (pertinent),
  - **T**emporellement défini (en se fixant une deadline).

Le cycle de vie d'un logiciel : Analyse des besoins

- **A l 'issue de cette phase : cahier des charges.**

- Un cahier des charges est normalement ***établi par le client*** en interaction avec utilisateurs et encadrement:
  - Besoins fonctionnels.
  - Besoins non fonctionnels .

  → ***Possibilité d'utilisation des Use Case « Cas d'utilisation UML »***

Le cycle de vie d'un logiciel : Cahier de charges

Cahier des charges fonctionnel

(Normes: AFNOR NF X50-151 et NF EN 16271)

- **1- Présentation générale du problème**
  - **1.1 Projet**

    Présentation du projet, de ses finalités, de son possible retour sur investissement.
  - **1.2 Contexte**

    Situation générale de l'organisation, autres projets en cours, études éventuelles menées sur le sujet (ou des sujets similaires), prestations attendues, parties concernées par le projet, confidentialité…
  - **1.3 Énoncé du besoin**

    Services rendus par le produit pour l'utilisateur final.
  - **1.4 Environnement du produit**

    Personnes, équipements, matériaux, contraintes de l'environnement, caractéristiques de chaque élément.

Le cycle de vie d'un logiciel : Cahier de charges

Cahier des charges fonctionnel

(Normes: AFNOR NF X50-151 et NF EN 16271)

- **2- Expression fonctionnelle du besoin**
  - **2.1 Fonctions de service et de contrainte**

    Fonctions de services principales, fonctions complémentaires (améliorant ou simplifiant le service rendu), contraintes pouvant entraver la liberté du réalisateur du produit.
  - **2.2 Critères d'appréciation**

    Souligner les critères déterminants pour l'évaluation des propositions.
  - **2.3 Niveaux des critères d'appréciation et ce qui les caractérise**

    Bien définir, d'une part, les niveaux indispensables et, d'autre part, les niveaux voulus mais révisables.

Le cycle de vie d'un logiciel : Cahier de charges

Cahier des charges fonctionnel

(Normes: AFNOR NF X50-151 et NF EN 16271)

- **3- Cadre de réponse**

  Cette partie encadre la manière dont les réponses au besoin devront être formulées.

  - **3.1 Pour chaque fonction**

  Solution proposée par le fournisseur, niveau atteint pour chaque critère d'appréciation, modalités de contrôle., part du prix attribué à chaque fonction.

  - **3.2 Pour l'ensemble du produit**

  Définir le prix de la réalisation de base, lister les options et variantes éventuellement proposées, les mesures prises pour se soumettre aux contraintes et leur impact économique, les outils liées à la mise en place et à la maintenance, décomposition en sous-ensembles, prévisions et perspectives de fiabilité et d'évolution.

* Exemple de Plan-type (d'après la norme AFNOR X50-151)

Cahier des charges fonctionnel

1. Présentation générale du problème

1.1 Projet
1.1.1 Finalités
1.1.2 Espérance de retour sur investissement

1.2 Contexte
1.2.1 Situation du projet par rapport aux autres projets de l'entreprise
1.2.2 Études déjà effectuées
1.2.3 Études menées sur des sujets voisins
1.2.4 Suites prévues
1.2.5 Nature des prestations demandées
1.2.6 Parties concernées par le déroulement du projet et ses résultats (demandeurs, utilisateurs)
1.2.7 Caractère confidentiel s'il y a lieu

1.3 Enoncé du besoin (finalités du produit pour le futur utilisateur tel que prévu par le demandeur)

1.4 Environnement du produit recherché
1.4.1 Listes exhaustives des éléments (personnes, équipements, matières...) et contraintes (environnement)
1.4.2 Caractéristiques pour chaque élément de l'environnement

2. Expression fonctionnelle du besoin

2.1 Fonctions de service et de contrainte
2.1.1 Fonctions de service principales (qui sont la raison d'être du produit)
2.1.2 Fonctions de service complémentaires (qui améliorent, facilitent ou complètent le service rendu)
2.1.3 Contraintes (limitations à la liberté du concepteur-réalisateur)

2.2 Critères d'appréciation (en soulignant ceux qui sont déterminants pour l'évaluation des réponses)

2.3 Niveaux des critères d'appréciation et ce qui les caractérise
2.3.1 Niveaux dont l'obtention est imposée
2.3.2 Niveaux souhaités mais révisables

3. Cadre de réponse

3.1 Pour chaque fonction
3.1.1 Solution proposée
3.1.2 Niveau atteint pour chaque critère d'appréciation de cette fonction et modalités de contrôle
3.1.3 Part du prix attribué à chaque fonction

3.2 Pour l'ensemble du produit
3.2.1 Prix de la réalisation de la version de base
3.2.2 Options et variantes proposées non retenues au cahier des charges
3.2.3 Mesures prises pour respecter les contraintes et leurs conséquences économiques
3.2.4 Outils d'installation, de maintenance ... à prévoir
3.2.5 Décomposition en modules, sous-ensembles
3.2.6 Prévisions de fiabilité
3.2.7 Perspectives d'évolution technologique

Le cycle de vie d'un logiciel : Cahier de charges

Cahier des charges technique:

Les éléments techniques nécessaires à mettre en œuvre pour réaliser le logiciel.

- **Quelques exemples :**
  - les moyens de paiement en ligne,
  - la solution d'hébergement,
  - l'architecture des serveurs,
  - le choix de la plateforme ou du CMS,
  - les outils d'administration,
  - les contraintes d'intégration,
  - le langage informatique,
  - la gestion de la sécurité des données,
  - la maintenance,
  - la migration,
  - la compatibilité avec les navigateurs, etc.

Le cycle de vie d'un logiciel : Cahier de charges

La charte graphique

- La charte graphique est un document de travail qui contient l'ensemble des règles fondamentales d'utilisation des signes graphiques qui constituent l'identité graphique d'une organisation, d'un projet, d'une entreprise.

- Le but de la charte graphique est de conserver une cohérence graphique dans les réalisations graphiques d'une même organisation, projet ou entreprise quels que soient les différents intervenants de la production (graphiste, directeur artistique…).

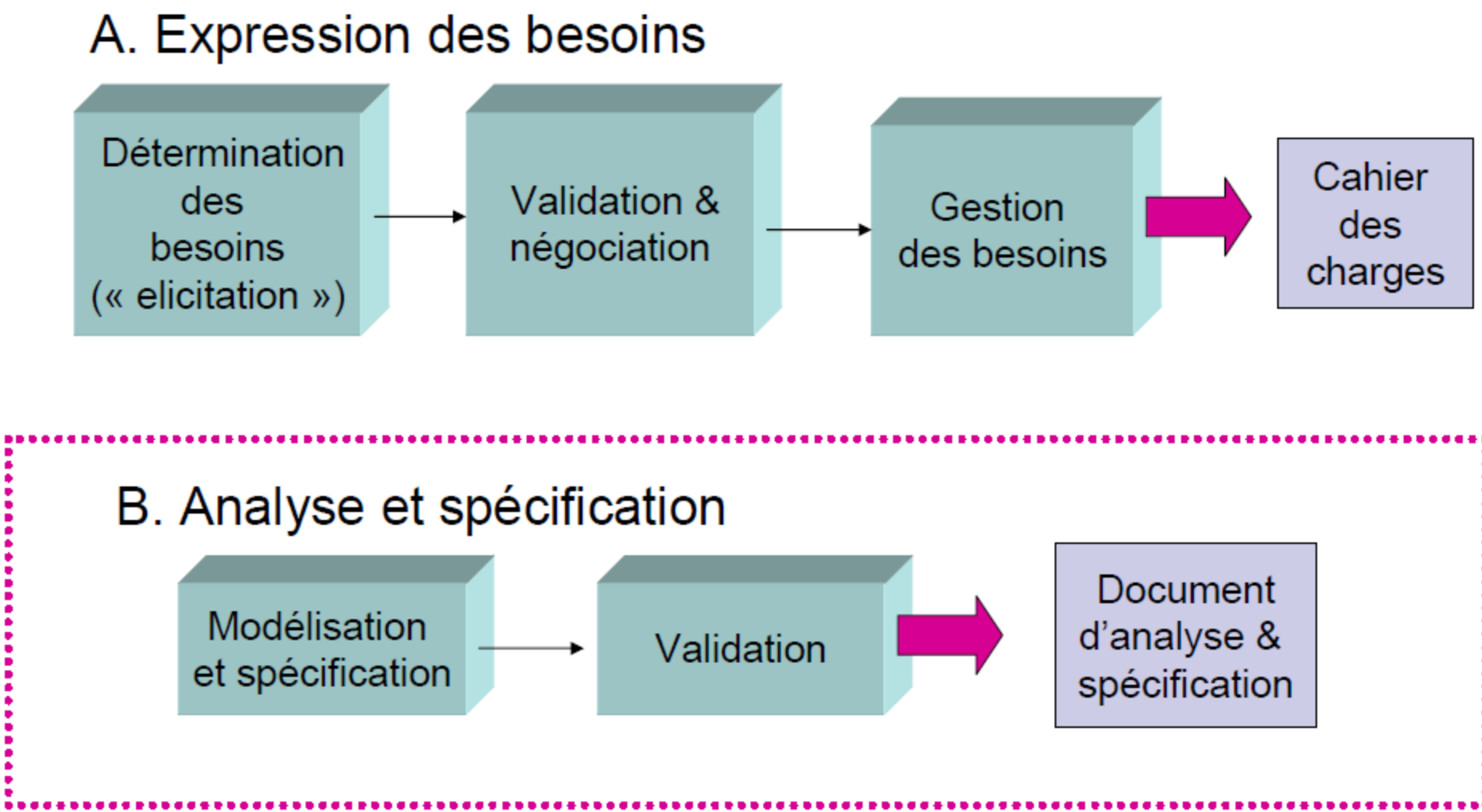
- Exemples d'éléments à spécifier :
  - le logo,
  - la typographie (police, taille, etc.),
  - les couleurs,
  - les illustrations, etc.

Le cycle de vie d’un logiciel : Cahier de charges

Autres informations :

- **Planning préliminaire**
- **Budget préliminaire**
- **Glossaire**
- **Documents et formulaires d'entreprise**
- **Références bibliographiques**

Le cycle de vie d'un logiciel : Analyse des besoins



Le cycle de vie d'un logiciel : La spécification

- **La spécification des besoins:**

- La spécification aura pour but de décrire avec rigueur:
  - Les données du systèmes (**vue statique**).
  - Les fonctions du système (**vue fonctionnelle**).
  - Les changements d'états et le contrôle du système (**vue comportementale**).

Le cycle de vie d'un logiciel : La spécification

- **La spécification:**

  - Schémas.
  - Langages formels.

- **La Spécifications souvent incompréhensibles pour les non initiés.**

Le cycle de vie d'un logiciel : La spécification

- **La spécification des besoins:**

- **Participants**: analyste

- **Document (résultat)** : *dossier d'analyse et de spécification*
  - Notation graphique ou textuelle rigoureuse.
  - Rédigé par: l'analyste.
  - Découpage: modèles statique, fonctionnel et comportemental

Le cycle de vie d'un logiciel : La spécification

- **La spécification des besoins:** Degré de formalisme.

- Spécifications **informelles**:
  - Ex. langue naturelle, croquis, etc.

- Spécifications **semi-formelles:**
  - Notation graphique dont la sémantique n'est pas sémantique pas absolument précise. Ex. UML, Merise, etc.

- Spécifications **formelles:**
  - Ex.: Spéc. algébriques, spéc. logiques, réseaux de Petri, automates, méthode Z, Méthode B, etc.

Le cycle de vie d'un logiciel : La spécification

- **La spécification des besoins:** Spécification semi-formelle.
  - SADT,
  - SA-RT
  - SSADM,
  - JSD
  - JSP
  - ***Merise***
  - Axial
  - OOA, OMT
  - ***UML***

Le cycle de vie d'un logiciel : La spécification

- **La spécification des besoins:** Spécification semi-formelle.

- Il permet de couvrir les aspects du système (statique, dynamique, comportemental)
- Il est facile à utiliser.
- **MAIS** Il est impossible de raisonner/analyser formellement sur le système en vue.

Le cycle de vie d'un logiciel : La spécification

- **La spécification des besoins: Spécification formelle.**

- « **La rédaction des spécifications de l'application doit être réalisée dans un langage rigoureux ne donnant pas prise aux interprétations** ».

- Expression dans un langage formel du quoi d'un système à développer.

- Plusieurs formes possibles selon la nature du système.

- Langages ou formalismes de spécification formelle : Logique, Z, B, Langages de spécification algébriques, algèbres de processus, etc.

Le cycle de vie d'un logiciel : La spécification

- **La spécification des besoins: Spécification formelle.**

- On reconnaît trois axes mathématiques principaux qui servent de supports aux méthodes formelles :

  - La logique du premier ordre.
  - La théorie des ensembles.
  - La théorie des types.

Le cycle de vie d'un logiciel : La spécification

- **La spécification des besoins: Spécification formelle**. Base mathématiques.
- **Logique du premier ordre (logique des prédicats)**

- La logique du premier est relativement simple et permet d'exprimer les propriétés de nombreux objets informatiques.

- Afin de désigner les objets de la spécification, la logique utilise des symboles d'individus ***tels*** que ***téléphone*** ou bien ***Caroline***.

- La logique utilise également des symboles de fonctions comme numéro : ***numéro (x)*** servirait à désigner le numéro de téléphone d'une personne x.

- Enfin, elle utilise des symboles de prédicats servant à exprimer des propriétés des objets. Ainsi, on peut choisir que le prédicat ***est_abonné (x)*** signifie que la personne x est un abonné du téléphone.

Le cycle de vie d'un logiciel : La spécification

- **La spécification des besoins: Spécification formelle**. Base mathématiques.
  - **Logique du premier ordre (logique des prédicats)**

- L'utilisateur a donc toute latitude concernant le vocabulaire qui servira à exprimer les formules élémentaires.

- Celles-ci peuvent ensuite être combinées au moyen de connecteurs qui eux sont bien définis :

  $\wedge$ pour ET, $\vee$ pour OU, $\neg$ pour NON, $\Rightarrow$ pour «implique » ;

  $\forall$ pour « quel que soit », $\exists$ pour « il existe ».

Le cycle de vie d'un logiciel : La spécification

- **La spécification des besoins: Spécification formelle**. Base mathématiques.

Exemple :

Supposons que nous avons une exigence logicielle qui indique : "***Le système ne doit autoriser que les utilisateurs âgés de plus de 18 ans à créer un compte.***"

- Nous pouvons exprimer cette exigence en utilisant la logique prédicative comme suit :
- Soit U l'ensemble de tous les utilisateurs et age(x) un prédicat qui retourne l'âge de l'utilisateur x.
- Nous pouvons définir un autre prédicat, createAccount(x), qui renvoie vrai si l'utilisateur x peut créer un compte, et faux sinon.
- Maintenant, nous pouvons exprimer l'exigence sous forme d'une formule logique : $\forall x \in U, age(x) \geq 18 \rightarrow createAccount(x)$
- Cette formule se lit comme suit : "Pour tous les utilisateurs x dans U, si l'âge de x est supérieur ou égal à 18, alors x doit pouvoir créer un compte."
- Cette formule logique peut être utilisée pour vérifier que l'implémentation logicielle respecte correctement l'exigence, et peut également aider à concevoir des cas de test pour s'assurer que l'exigence est satisfaite.

Le cycle de vie d'un logiciel : La spécification

- **La spécification des besoins: Spécification formelle**. Base mathématiques.

- **Exemple:** Traduisez les énoncés suivants en formules de la logique des prédicats (par exemple Aime (x,y) = x aime y).
  - Jean est plus grand que Marie
  - b. Paul a vu Léa et elle ne l'a pas vu
  - c. Si Jean est un homme, alors il est mortel
  - d. Un chat est entré
  - e. Certains enfants ne sont pas malades
  - f. Tous les éléphants ont une trompe
  - g. Tous les hommes n'aiment pas Marie
  - h. Il y a une chanson qu'aucun enfant ne chante
  - i. Si tous les hommes aiment Marie, alors elle est contente
  - j. Tous les étudiants apprécient un professeur

Le cycle de vie d'un logiciel : La spécification

- **La spécification des besoins: Spécification formelle**. Base mathématiques.

a. Jean est plus grand que Marie — $G(j, m)$

b. Paul a vu Léa et elle ne l'a pas vu — $V(p, l) \wedge \neg V(l, p)$

c. Si Jean est un homme, alors il est mortel — $H(j) \rightarrow M(j)$

d. Un chat est entré — $\exists x(C(x) \wedge E(x))$

e. Certains enfants ne sont pas malades — $\exists x(E(x) \wedge \neg M(x))$

f. Tous les éléphants ont une trompe — $\forall x(E(x) \rightarrow T(x))$

g. Tous les hommes n'aiment pas Marie — $\forall x(H(x) \rightarrow \neg A(x, m))$

$\neq \neg \forall x(H(x) \rightarrow A(x, m))$

h. Il y a une chanson qu'aucun enfant ne chante — $\exists x \forall y((C(x) \wedge E(y)) \rightarrow \neg C(y, x))$

OU

$= \exists x \neg \exists y(C(x) \wedge E(y) \wedge C(y, x))$

i. Si tous les hommes aiment Marie, alors elle est contente

$(\forall x(H(x) \rightarrow A(x, m)) \rightarrow C(m))$

j. Tous les étudiants apprécient un professeur — $\forall x \exists y((E(x) \wedge P(y)) \rightarrow A(x, y))$

Le cycle de vie d'un logiciel : La spécification

- **La spécification des besoins: Spécification formelle**. Base mathématiques.

Exercice : Traduire dans le langage des prédicats du premier ordre les phrases suivantes :

- Tous les hommes sont méchants.
- Seulement les hommes sont méchants.
- Il existe des hommes méchants.
- Il existe un homme qui n'est pas méchant.
- Il n'existe pas d'homme méchant.
- Il existe un homme qui aime toutes les femmes.
- Chaque chat connaît un chien qui le déteste.
- Tous les poissons, sauf les requins, sont gentils avec les enfants.
- Tous les oiseaux ne peuvent pas voler.
- Chaque personne aime quelqu'un et personne n'aime tout le monde, ou bien quelqu'un aime tout le monde et quelqu'un n'aime personne.
- Il y a des gens que l'on peut rouler tout le temps et quelquefois on peut rouler tout le monde, mais on ne peut pas rouler tout le monde à chaque fois.

Le cycle de vie d'un logiciel : La spécification

- **La spécification des besoins: Spécification formelle**. Base mathématiques.

Solution : Traduire dans le langage des prédicats du premier ordre les phrases suivantes :

- ***Prédicats unaires*** : Homme, Méchant, Femme, Chat, Chien, Poisson, Requin, Enfant, Oiseau, Volant (quelque chose qui peut voler)
- **Prédicats binaires** : aime, connait, déteste, estGentilAvec, peutÊtreRoulé (vrai lorsque le premier argument peut être roulé, dans le sens de trompé/dupé, à l'instant du second argument)
  - $\forall x$ (Homme(x) → Méchant(x))
  - $\forall x$ (Méchant(x) → Homme(x))
  - $\exists x$ (Homme(x) ∧ Méchant(x))
  - $\exists x$ (Homme(x) ∧ ¬Méchant(x))
  - ¬$\exists x$ (Homme(x) ∧ Méchant(x))
  - $\exists x$ (Homme(x) ∧ $\forall y$ (Femme(y) → aime(x,y)))
  - $\forall x$ (Chat(x) → >$\exists y$ (Chien(y) ∧ connait(x,y) ∧ déteste(y,x)))
  - $\forall x$ ((Poisson(x) ∧ ¬Requin(x)) → $\forall y$ (Enfant(y) → estGentilAvec(x,y)))[2]
  - ¬$\forall x$ (Oiseau(x) → Volant(x))
  - (($\forall x \exists y$ aime(x,y)) ∧ ¬$\exists x \forall y$ aime(x,y)) ∨ ($\exists x \forall y$ aime(x,y) ∧ $\exists x \forall y$ ¬aime(x,y)))
  - ($\exists x \forall t$ peutÊtreRoulé(x,t) ∧ $\exists t \forall x$ peutÊtreRoulé(x,t) ∧ $\forall x \forall t$ ¬peutÊtreRoulé(x,t))

Le cycle de vie d'un logiciel : Analyse des besoins

Planification et estimation des charges