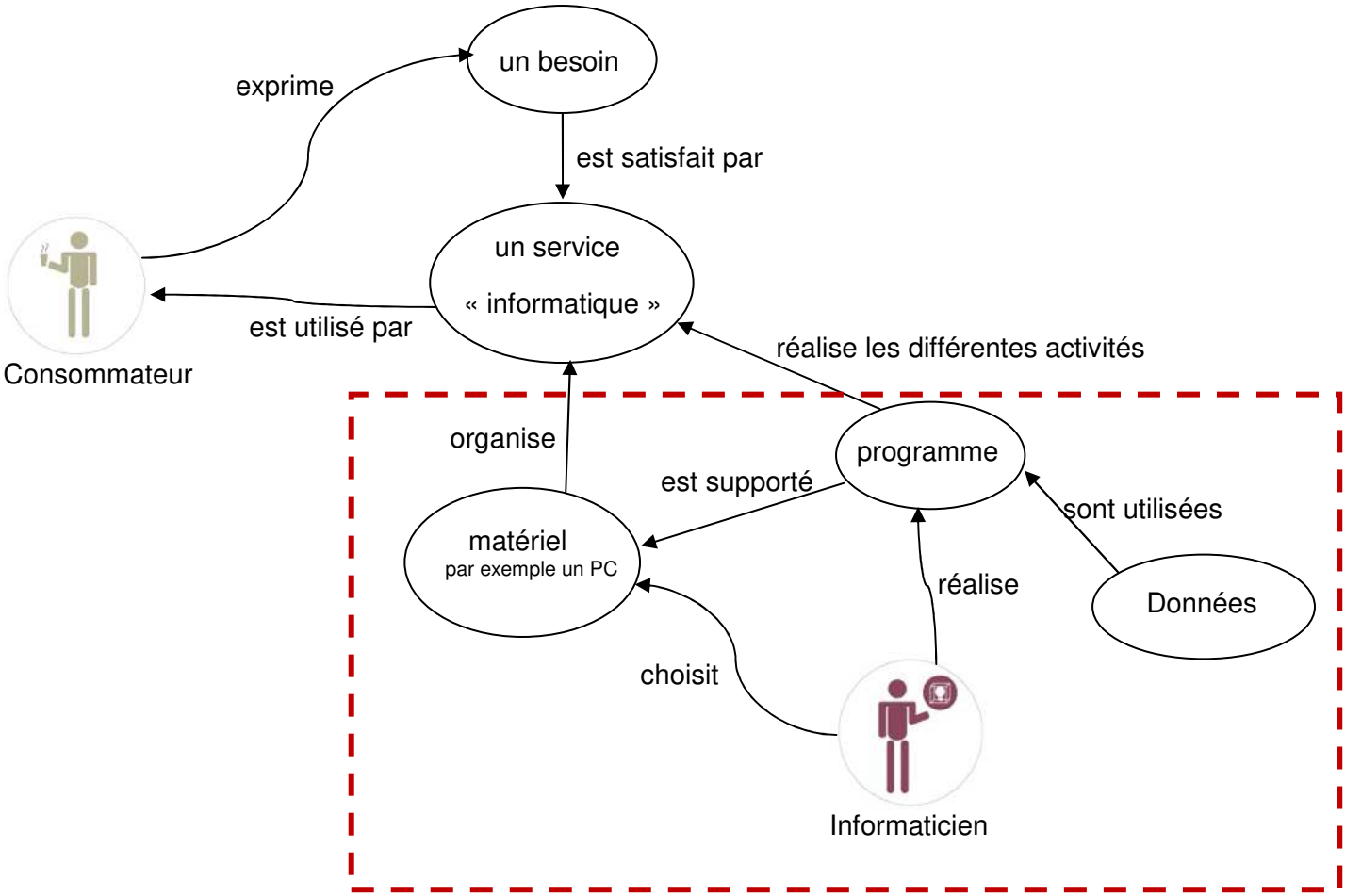
Programmation
&
Généralités

1/ Définition de l'"informatique :

Deux personnages interviennent : l'utilisateur et le concepteur qui ont en commun le service informatique.

Nous pouvons représenter leurs relations par le schéma ci-dessous :



2/ Du besoin au programme

Nous allons nous placer en phase de conception d'un programme, donc du point de vue de l'informaticien.

Exemple de programme à concevoir : Un besoin de la vie courante : téléphoner à un(e) copain(e) depuis le répertoire de notre téléphone.

On peut « découper » l'appel en plusieurs activités, ici successives :

1/ Lancement de l'application « répertoire »
2/ recherche de notre correspondant dans le répertoire
3/ appel de notre correspondant
4/ communication avec notre correspondant
5/ fin de l'appel

On peut en déduire 2 actions principales, l'une étant la gestion du répertoire de nos contacts et l'autre la gestion de la communication téléphonique proprement dit.

En informatique, il existe plusieurs phases dans la conception :

2.1 Première phase : Analyse du besoin

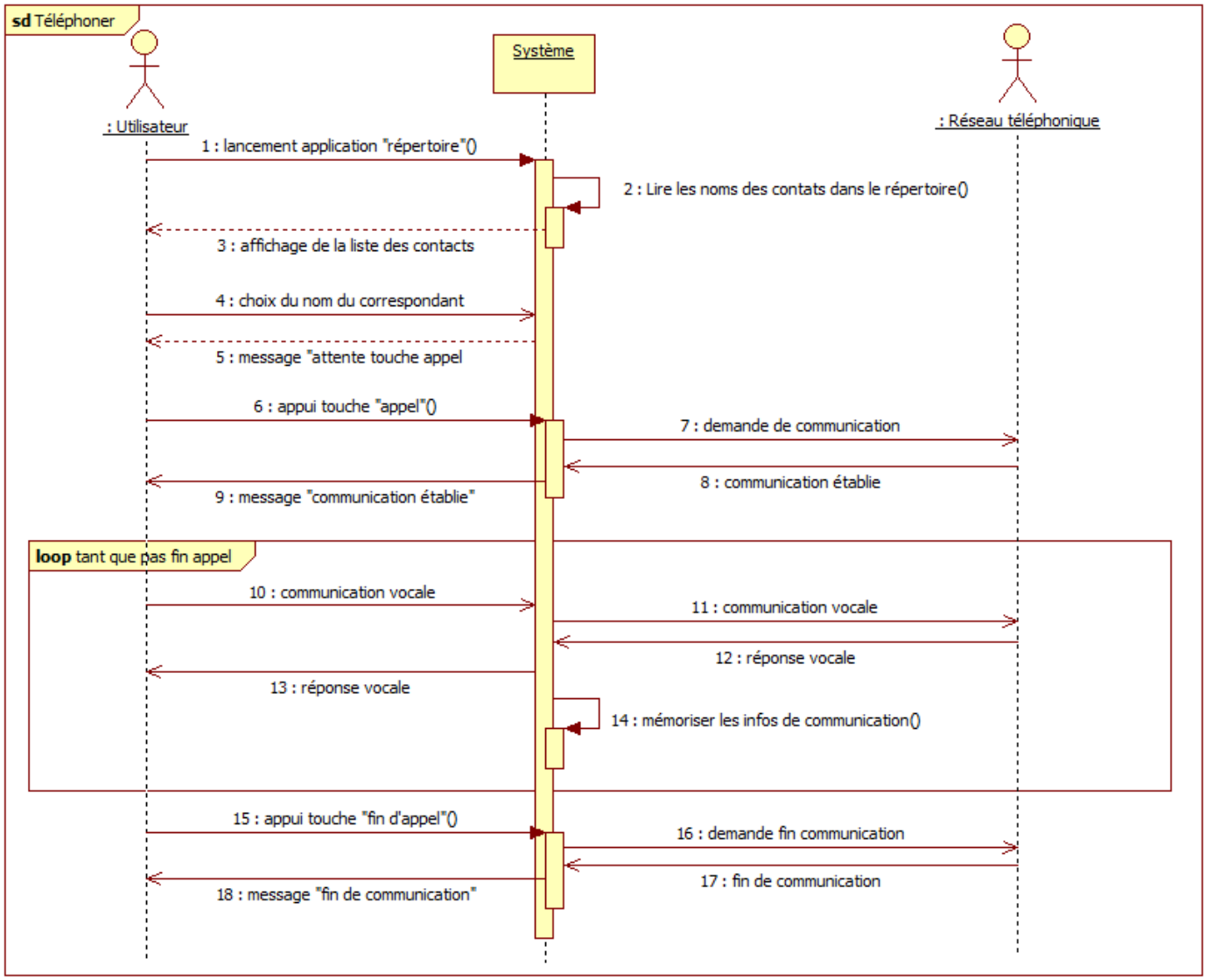
Le besoin est formalisé par l'approche fonctionnelle : utilisation du formalisme UML avec le diagramme des cas d'utilisation et le diagramme de séquence **« c'est le quoi faire ».**

Exemple :
Avec qui est en relation notre application ?

Ce diagramme est un diagramme des cas d'utilisation : ce diagramme montre les interactions entre le système supportant le programme et l'(es)utilisateur(s) et l'environnement.

On décrit ici que notre application est utilisée (ou a des relations) avec l'utilisateur et le réseau téléphonique.

Nous allons détailler les relations de l'utilisateur avec notre application :



Ce diagramme est un diagramme de séquence système (UML): ce diagramme montre d'une part les échanges chronologiques entre l'utilisateur et le système (programme) et d'autre part une chronologie des différentes actions que doit réaliser le programme.

On peut ici proposer les IHM (Interface Homme Machine).

Nous allons proposer des tests permettant de valider le bon fonctionnement de notre programme.

Puis nous nous occupons du « comment faire ».

2.2 Deuxième phase : Conception préliminaire :

Réflexion sur les types de données, les structures de données à mettre en place, et les différents programmes à réaliser.

Exemple : il faut que l'on choisisse comment mémoriser tous les noms et les numéros de tel (voir plus d'informations) dans notre téléphone. Une application logicielle suffit.

2.3 Troisième phase : Conception détaillée :

Elaboration de l'algorithme : enchaînement des actions présentes sur la ligne de vie du diagramme de séquence (symbolisée par des pointillés sur le système)

Nous allons écrire le programme en un pseudo-langage, en français compréhensible par nous tous avec notre formalisme.

Cet algorithme montre les données mises en jeu, et les enchainements des actions.

2.4 Quatrième phase : codage de l'application.

Codage de l'application : c'est la transcription de l'algorithme dans un langage de programmation.

Ici nous, humain, pouvons utiliser des langages de différents « niveaux », mais seul le langage machine sera compris par l'ordinateur.

Nous écrivons nos programmes en un langage dit de « haut niveau » : il nous faut un interprétateur (ou un compilateur), qui va changer notre langage machine.

Remarques sur les langages :

_ le langage machine est l'association d'informations binaires (suite de « 0 » et de « 1 ») assemblés en octets. Ce « langage machine » est évidemment presque incompréhensible pour nous. Pour « parler » à un ordinateur, il nous faudra utiliser des systèmes de traduction automatiques, capables de convertir en nombres binaires des suites de caractères formant des mots-clés (anglais en général) qui seront plus significatifs pour nous.

_nos langages : il en existe beaucoup, nous pouvons les classer en 3 niveaux.

- Langage de bas niveau : exemple « assembleur » est constitué d'instructions très élémentaires, très « proches de la machine ».
- Langage de haut niveau : exemples le C/C++, le Pascal, le JAVA et le PYTHON. Un langage de haut niveau comporte des instructions plus abstraites, plus « puissantes » (et donc plus « magiques »). Cela signifie que chacune de ces instructions pourra être traduite par l'interpréteur ou le compilateur en un grand nombre d'instructions machine élémentaires.
- Langage de très niveau ou langage graphique : exemple LABVIEW : ce type de langage est souvent l'association graphique de « boîtes » ou d'icones représentant un certain travail.

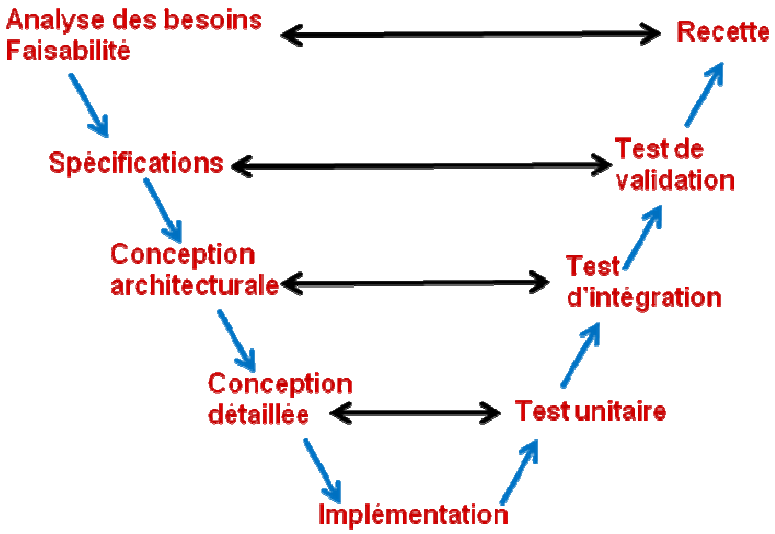
2.5 Cinquième phase : Test de l'application.

Validation du programme par des tests permettant de prouver que notre programme répond bien au besoin.

Après cette phase réussie, le programme est utilisable donc commercialisable.

2.6 Synthèse :

Dans le monde des entreprises, l'élaboration d'un programme devait respecter le cycle en Vé.



Modèle du cycle de vie en Vé

A l'heure actuelle, la méthode AGILE est utilisée

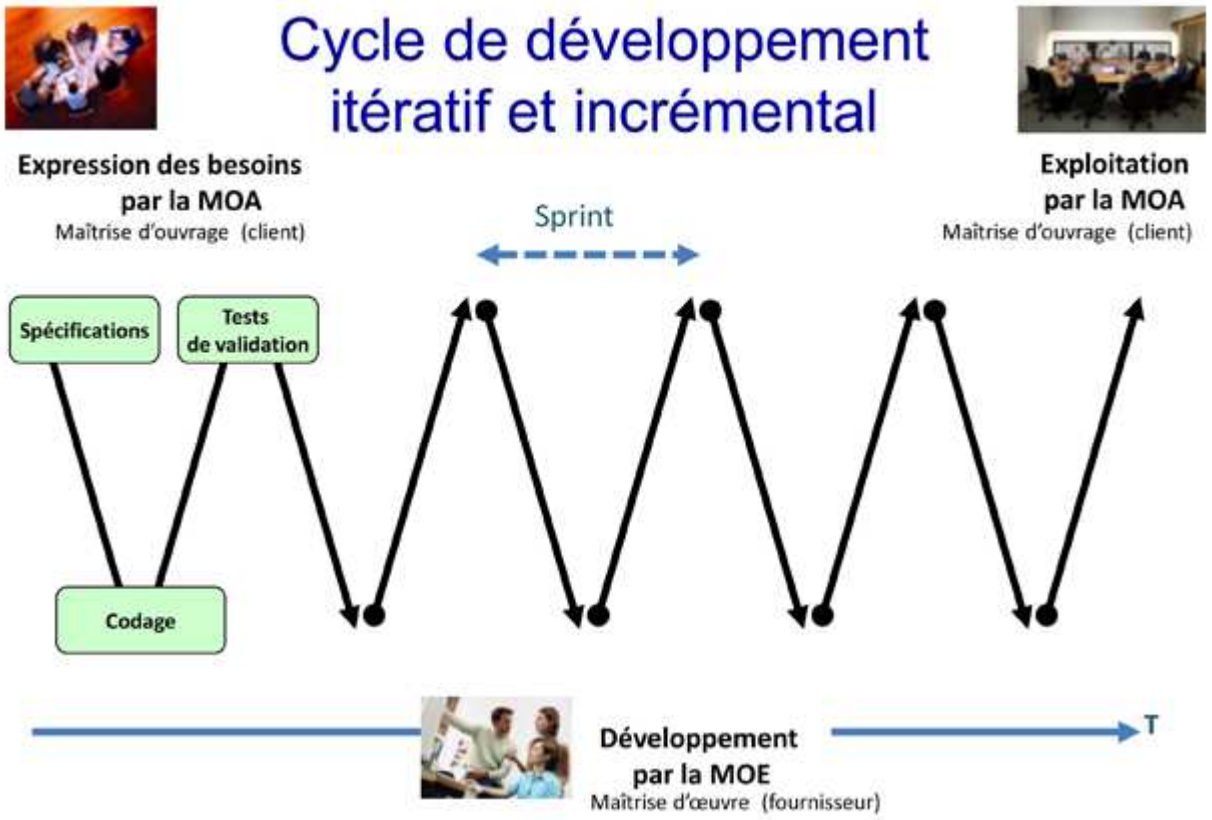
http://ingenierie-creations.fr/WP/methodes-agiles/

Cette méthode se base sur 4 valeurs :

- **Les individus et leurs interactions** plus que les processus et les outils.
- **Un logiciel qui fonctionne** plus qu'une documentation exhaustive.
- **La collaboration avec les clients** plus que la négociation contractuelle.
- **L'adaptation au changement** plus que le suivi d'un plan.

Source : https://fr.wikipedia.org/wiki/Manifeste_agile

Voici donc le nouveau cycle de développement dit en W:



source : https://interstices.info/jcms/int_71726/l-informatique-agile-c-est-quoi