Développement d’application graphique

# Rôle dans le développement de projets

* Chef de projet
* Analystes spécialisés (graphiste, hardware, sécurité, bases de données, réseau, etc.)
* Concepteur
  + Architecte
  + Expert dans le domaine
  + Ergonomiste
  + Programme d’interface utilisateur
  + Programmeur senior et junior

# Modèle Classique

* On ne peut construire la toiture sans fondation

Expression du besoin

Analyse du besoin

Spécification fonctionnelles

Conception globale

Conception détaillée

Codage

Test

## EN V

Besoin Évolution

Analyse Ecploitation

Spec

Conc. Globale

Conc. Dét.

Code

## Critique

* Trop lourd
* Ne prend pas en considération
  + Les changements très fréquents
  + Le feedback du client

# Méthode itératives

* On reprend les étapes de la cascade mais en les multipliant plusieurs fois.

## Principe Agile

* Satisfaction du client en livrant continuellement des logiciels utiles
* Release fréquents
* Le progrès se mesure au logiciel « qui marche »
* Les exigences tardives sont bienvenues
* Coop. Intime et quotidienne entre les gens d’affaires et les dév.
* La discussion en face à face est la meilleure communication
* Les projets se construisent autour d’un individu motivés qui doivent être considérés dignes de confiance.
* Simplicité
* Équipe autonomes qui s’auto-organise
* Adaptation régulière aux circonstances

## Résumé en 4 oppositions

* Individus et interaction contre processus et outils
* Logiciel qui fonctionne contre documentation exhaustive
* Collaboration du client contre négociation de contrat
* Réponse au changement contre suivi d’un plan définit

# Xtrem Programming

* La communication
* La simplicité
* Le feed-back
* Le courage

1. Planification des exigences
   1. Évaluation de la valeur commerciale de chacune des options du logiciel
   2. Chaque option majeure est réécrite sous la forme d’une histoire
2. Petit releases incrémentaux
3. Métaphore système
   1. Bureau Windows
   2. Panier (Site web)
   3. Bouton
4. Conception simple
   1. Le changement viendra; ne pas investir trop de temps à concevoir
5. Tester continuellement
   1. Tests unitaires avant d’écrire le code
6. Programmer à quatre mains/ deux cerveau / un clavier
7. Propriété collective du code
8. Intégration continuelle
9. Semaine de 40h
10. Client est le bienvenu

## User stories

*Faire une histoire pour chaque fonctionnalité à Dev.*

* En tant que nouveau membre je veux m’inscrire pour accéder aux possibilités du logiciel
* En tant que membre inscrit, je veux me loger pour voir mes statistiques
* En tant qu’admin, je veux éliminer un membre ne respectant pas les règles

## Backlog

* Liste des fonctionnalités, tâches, user stories nécessaire et suffisantes
* Pas de forme imposée
* Atomicité des éléments

## Élection du contenu d’une itération

* Les développeurs estiment la valeur de chaque item du back log

## Méthode d’estimation

* Poker planning

## Vélocité

* A la fin d’une itération, on additionne les estimations associées.

## Réunion fréquente

* Qu’as-tu terminé depuis la dernière réunion ?
* Qu’auras-tu terminé avant la prochaine réunion

## Conclusion