# Génie logiciel

Génie civil

Art de construction

Progs informatiques

Documentation

**Discipline** 🡪 méthodes, techniques et outils

**Basée sur le savoir** 🡪 théorique

**Le savoir-faire** 🡪 compétence, ressource humaine

**Le faire savoir** 🡪 communication

**Pour produire** 🡪 développement, réalisation

**De façon industrielle** 🡪 travail d’équipe, méthodologie

**Des logiciels** 🡪 les produits

**De qualité au meilleur prix** 🡪 qualité/prix

## **facteur de qualité de logiciel**

Critères internes

Critères externes

EXTERNES :

**Fiabilité** 🡪 conformité de ses spécifications

**Robustesse, sureté** 🡪 stable et disponible

**Efficacité** 🡪 bon usage en termes de mémoire et temps d’exécution

**Convivialité, utilisabilité** 🡪 facile et agréable à utiliser

**Documentable** 🡪 tuto ou un manuel utilisateur

**Ergonomique** 🡪 adaptable aux conditions de travail et d’utilisateur

**Sécurité** 🡪 absence du danger et la garantie offerte

**Adéquation et validité** 🡪 conformité au maquettage et but

**Intégrité** 🡪 aptitude à protéger le code et les données

INETRNES :

**Documentable** 🡪 précédé par un doc de conception

**Lisibilité et clarté** 🡪 écrit proprement

**Portabilité** 🡪 fonctionnel sur plusieurs machines et indépendant de son environnement d’exécution

**Extensibilité** 🡪 ajout de nouvelles fonctionnalités(flexible)

**Réutilisabilité** 🡪 des parties pour d’autres logiciels =

**Interopérabilité et coulabilité** 🡪 interagir en synergie

**Traçabilité** 🡪 le suivi d’un produit aux diff stades

**Testabilité et vérifiabilité** 🡪 le soumettre à une épreuve de confirmation de la tâche à exécuter

# **SCRUM**

* Ce cadre **méthodologique agile** révolutionne la manière dont on gère un projet
* Il permet de délivrer et de modifier un projet, un produit ou une fonctionnalité très rapidement
* Basée sur des rôles, composée de 6 à 10 personnes :
* **Développeurs**
* **Testeurs**
* **Architectes**
* **Product owner :** le chef de produit représentent le client, il définit les spécifications fonctionnelles et établit la liste des priorités de ce qu'il faut développer.c'est aussi le product owner qui valide les fonctionnalités
* **Scrum master :** est garant du respect des processus scrum il s'assure d'une bonne communication entre les membres de l'équipe

User Story 🡪 Product Backlog 🡪 Sprint Backlog

* **User Story :** capturer les besoins et les fonctionnalités souhaitées du point de vue de l'utilisateur
* **Product Backlog** : une liste dynamique et priorisée de toutes les fonctionnalités, les exigences, les améliorations et les corrections qui doivent être réalisées sur un produit
* **Sprint Backlog :** contient les tâches spécifiques qui doivent être accomplies pour atteindre les objectifs du sprint
* **Sprint Planning Meeting :** permet également de clarifier les exigences, de discuter des dépendances et des contraintes, et de définir les critères d'acceptation pour chaque élément du product backlog sélectionné
* **Sprint Meeting Review :** l'équipe de développement présente les éléments terminés du sprint backlog, en mettant l'accent sur les fonctionnalités développées et les objectifs atteints. Les participants peuvent interagir avec le produit, poser des questions et fournir des commentaires

## **gestion des exigences**

Recueil des données 🡪 Analyse des besoins 🡪 Spécification des exigences

* Collecte des besoins
* Questionnaires
* Sondages
* Interviews
* Brainstorming
* Filtrage des besoins
* Elimination d’ambiguïté, incohérences et incomplétude

**Cahier de spécification des besoins ou cahier de charge**

# **cycle de vie logiciel**

* Ensemble des étapes qui composent le processus de développement et d’utilisation du logiciel

Modèle de cycle de vie : servent de guide pour structurer et gérer efficacement les projets, en s'assurant que toutes les activités nécessaires sont prises en compte et exécutées de manière appropriée pour atteindre les objectifs fixés

* On a 2 familles de modèles

Linéaire

Itérative et Incrémentale

Prototypage

Spirale

Incrémental

Cascade

En V

Cascade :

* Se divise en étapes
* Entre 2 étapes se trouvent une validation
* On commence une activité qu’après avoir terminé celle qui précède
* En cas d'erreur découverte lors des tests, il est nécessaire de retourner à la phase de programmation

**Avantages :**

* Structure claire

**Inconvénients :**

* **Une image contenant texte, capture d’écran, diagramme, Police

  Description générée automatiquement**Détection tardive des erreurs (cout élevé)

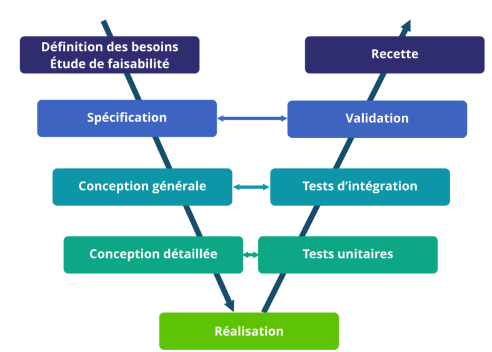
en V :

* Composé de 3 phases
* Conception
* Réalisation
* Validation
* Enchaînement séquentiel (modèle en cascade) de gauche à droite, les résultats des étapes de départ sont utilisés par les étapes d’arrivée

Avantages :

* Détection précoce des erreurs

Inconvénients :

* Faible flexibilité aux changements

par prototypage :

* Enchaîner les développements de prototypes et à les soumettre pour validation/remarque au client

Avantages :

* Le client participe activement dans le développement du produit

Inconvénients :

* Temps et couts supplémentaires

Une image contenant texte, diagramme, Police, capture d’écran

Description générée automatiquement

incremental :

* Développer des applications en étendant PROGRESSIVEMENT ses fonctionnalités.
* Permet d’éviter de TOUT CONCEVOIR, de TOUT TESTER
* Extension successive à partir d’un produit « Noyau » du logiciel

Avantages :

* Les intégrations sont progressives

Inconvénients :

* Risque de fonctionnalités inachevées

Une image contenant texte, capture d’écran, Police, ligne

Description générée automatiquement

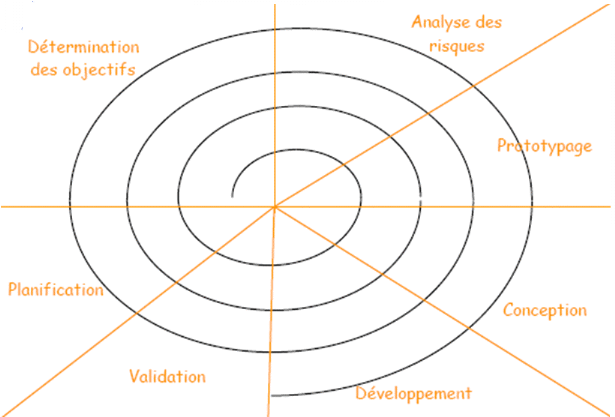
en spiral :

* Développement itératif (prototypes)

Avantages :

* Modèle complet, complexe et général

Inconvénients :

* Difficile à appliquer

## **les méthodes**

Lourde

Agile

UP

RUP

2TUP

Scrum

**Processus Unifié**

**Caractéristiques :**

* Piloté par les cas d’utilisation
* Itératif et incrémental
* Centré sur l’architecture
* Orienté risques

**Composants :**

* 4 phases :
* Etude d’opportunité
* Elaboration
* Construction
* Transition
* 5 activités :
* Expression des besoins
* Analyse
* Conception
* Implémentation
* Test

**RUP (Rational Unified Process)**

* Est une instance/implémentation de UP

**Artéfact :** élément d’information utilisé lors d’une activité de développement logiciel

**Activité :** opération exécutée au sein d’un état

**Rôle :** comportement et responsabilité d’un ensemble de personnes

**2TUP (2 Track Unified Process)**

* **Branche fonctionnelle :** Capture les besoins fonctionnels
* **Branche technique :** Capitalise un savoir-faire technique et/ou des contraintes techniques
* **Phase de réalisation :** Réunir les deux branches, permettant de mener une conception applicative

**Agile**

**4 valeurs :**

* Priorité aux **personnes** et aux **interactions** sur les procédures et les outils
* Priorité aux **applications fonctionnelles** sur une documentation pléthorique
* Priorité à la **collaboration avec le client** sur la négociation de contrat
* Priorité à **l'acceptation du changement** sur la planification

**12 principes :**

1. Priorité à la satisfaction du client
2. Accueillir favorablement les changements de besoins
3. Livrer fréquemment des logiciels opérationnels
4. Collaboration étroite entre les parties prenantes
5. Construire des projets autour de personnes motivés.
6. Conversation face à face
7. Logiciel opérationnel comme principale mesure de progrès
8. Maintenir un rythme de développement soutenable
9. Excellence technique et conception
10. La simplicité
11. Les équipes autoorganisées
12. Réflexion et adaptation régulières

# architecture et conception de logiciels

Logique

Physique

CONCEPTION :

Niveaux conceptuels

* architecture globale
* architecture détaillée

## patrons d’architecture

* **Modèle en couches :**
* 3 couches :
* Présentation (IHM)
* Applicative
* Infrastructure
* 5 couches :
* IHM
* Logique applicative
* Métier
* Accès donnés
* Infrastructure
* **Modèle MVC :**
* Model
* Controller
* View

**ARCHITECTURE PHYSIQUE**

1-niveau : couches situées sur la même machine

2-niveaux : séparées sur 2 sites (serveur et postes)

## qualite de la conception architecturale

* **Critères de qualité :**
* Faible couplage
* Forte cohésion

## patrons de conception

|  |  |  |
| --- | --- | --- |
| De création | De structure | De comportement |
| Factory  Builder  Prototype  Singleton  Factory method | Adapter  Bridge  Composite  Facade  Proxy | Command  Iterator  Observer  State  Template method |

* Problèmes liés au non-respect des critères de qualités :
* Difficulté de maintenabilité
* Difficulté de réutilisabilité
* Problème de performance

# gestion de configuration logicielle

* Discipline de gestion qui vise à contrôler et gérer les éléments constitutifs d'un logiciel tout au long de son cycle de vie. Elle comprend la gestion des versions, la gestion des modifications, la traçabilité des composants, la gestion des baselines (lignes de base), et la gestion de la documentation associée
* **Les principes de base :**
* La journalisation :
* Controller les modifications apportées à chaque fichier ou composant
* La gestion des versions :
* Gérer les changements
* Num de version : modification majeure
* Num de révision : ajout de fonctionnalités
* Num de correction : correction des fautes
* La gestion des conflits
* Optimiste : chaque développeur travaille à son rythme (en cas de conflit intervention manuelle)
* Pessimiste : utilisation de lock
* **Architecture centralisée :**
* Tout passe par le serveur (nœuds)
* **Architecture décentralisée :**
* Chacun travaille à son rythme de manière indépendante

# tests et validation

* Tester c’est exécuter le programme dans l’intention d’y trouver des anomalies ou des défauts
* Le test logiciel est le maillon principal dans la chaîne d’assurance qualité produit

## anomalies logicielles

**Bug :**

* Dysfonctionnement causé par un défaut de conception ou de réalisation

**Crash applicatif ou Deny of Service:**

* Logiciel cesse de fonctionner d’une manière inattendue et abrupte à cause des violations d’accès mémoire ou erreurs de gestion d’exception

**Fuite de mémoire :**

* Dysfonctionnement dans les opérations d'allocation de mémoire. La quantité de mémoire utilisée par le logiciel défaillant va en augmentant continuellement et gêne le déroulement des autres logiciels et les entraîne à des dysfonctionnements

**Vulnérabilité :**

* Faiblesse dans un système informatique permettant à un attaquant de porter atteinte à l'intégrité de ce système, c'est-à-dire à son fonctionnement normal, à la confidentialité et l'intégrité des données qu'il contient. On parle aussi de faille de sécurité informatique.

**Faute de segmentation :**

* Dysfonctionnement dans des opérations de manipulations de pointeurs ou d'adresses mémoire.

**Buffer Overflow :**

* Le comportement de l'ordinateur devient imprévisible à cause du Dépassement de tampon ou débordement est un bug par lequel un processus, lors de l'écriture dans un tampon, écrit à l'extérieur de l'espace alloué au tampon, écrasant ainsi des informations nécessaires au processus.

**Deadlock ou inter blocage :**

* Un mécanisme de prévention provoque l'annulation de l'opération lorsque la durée d'attente dépasse le délai admissible
* Dysfonctionnement durant lequel plusieurs processus s'attendent mutuellement, c'est à dire qu'ils attendent chacun que l'autre libère les ressources qu'il utilise pour poursuivre.

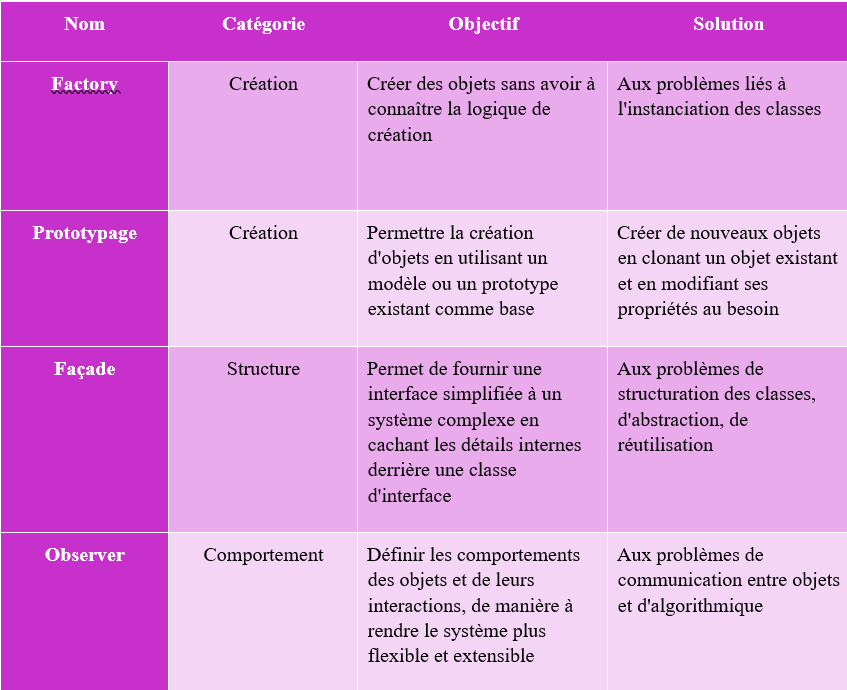
## methodes de tests

**Méthode boite noire :** fonctionnement externe (fonctionnalités)

**Méthode boite blanche :** fonctionnement interne (code)

**Méthode boite grise :** combinaison des deux approches (externe et interne)

## types de tests

* **Test nominal :** données volontairement valides
* **Test de robustesse :** données volontairement invalides
* **Test unitaire :** tester les fonctions ou les modules du code par les programmeurs
* **Test d’intégration :** valider le bon fonctionnement d’une ou plusieurs parties
* **Test de non-régression :** tests liés au changement
* **Test d’administration :** se focaliser sur les aspects d’administration
* **Test de sécurité :** détecter les instructions et les failles de sécurité