2021-2022

TD1 : Conduite de projets
Besoins et tâches

Compétences

- Transformer un besoin en exigence
- Transformer une exigence en une ou plusieurs tâches
- Implémenter les tâches

Propriétaire — Développeur

Introduction à la conduite de projet

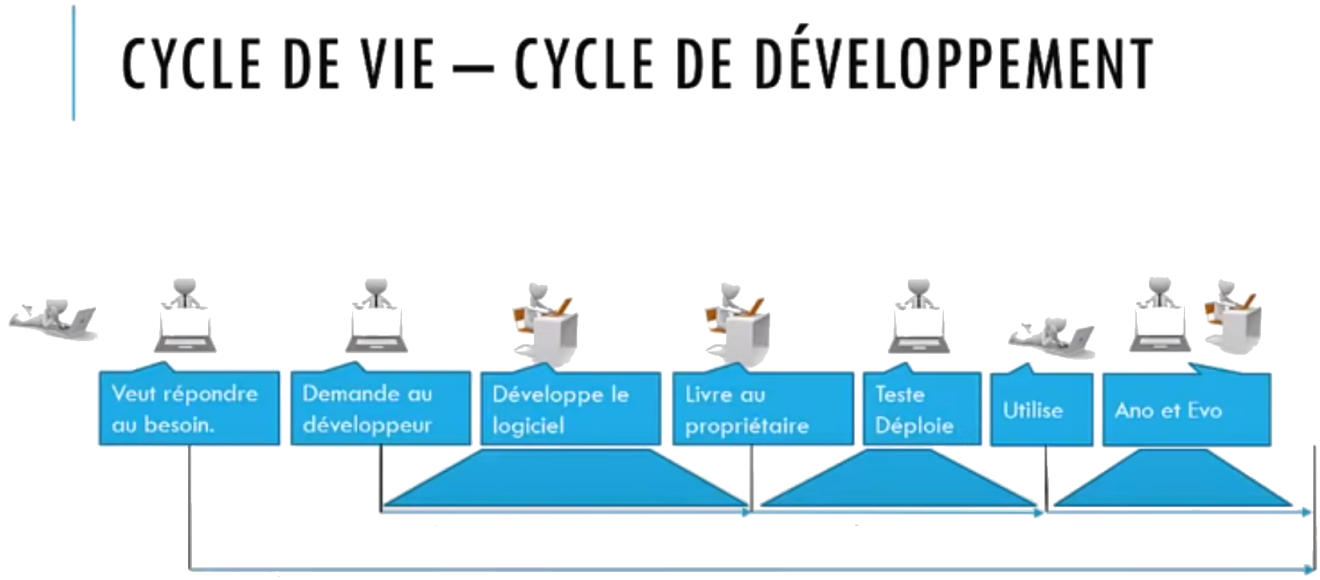
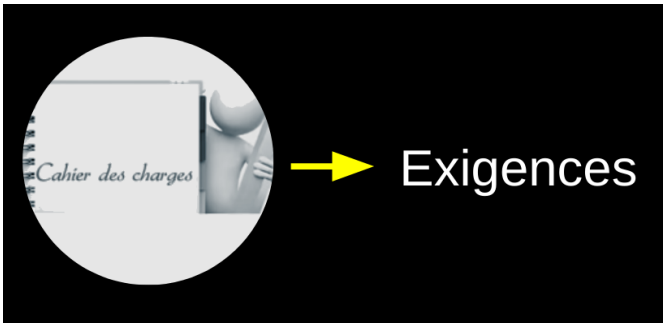


FIGURE 1 – Image extraite de la vidéo : *conduite de projet* de Xavier Blanc

1. À partir des documents fournis définir sur l'image ci-dessus les grandes étapes d'un projet logiciel
   - Cycle de vie
   - Cycle de développement
   - Pré-production
   - Production et maintenance
2. Une première étape à franchir pour entrer dans le cycle de développement est celle de la transormations des **besoins**, exprimés par le propriétaire dans cahier des charges, en **exigences** par le développeur. C'est une étape importante car elle engage le développeur sur ce qu'il est capable de faire.



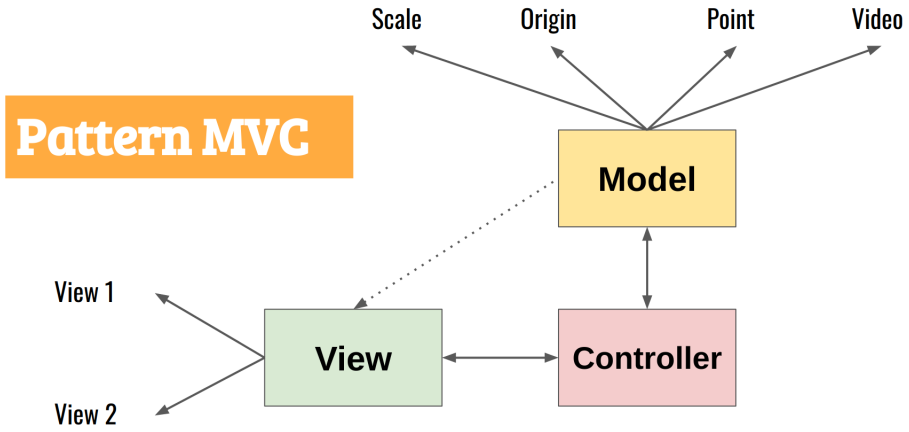
À partir du cahier des charges et en s'inspirant de l'exemple fourni dans *besoin.md*, rédiger les exigences correspondantes aux besoins suivants :

- enregister les données acquises lors du pointage dans un fichier au format csv.
- mise en place d'une échelle par l'utilisateur

Cycle de développement

Architecture

- Il existe différentes manières de structurer le code des applications qui comportent une interface utilisateur.
- Une des architectures, communément utilisée, et qui comporte de nombreuses variantes, est connue sous l'acronyme **MVC** qui signifie **Model - View - Controller.**
- Dans cette architecture on divise le code des applications en entités distinctes (modèles, vues et contrôleurs) qui communiquent entre-elles au moyen de divers mécanismes (invocation de méthodes, génération et réception d'événements, etc.).
- Cette architecture (ou modèle de conception, design pattern) a été introduite avec le langage Smalltalk-80 dans le but de simplifier le développement ainsi que la maintenance des applications, en répartissant et en découplant les fonctionnalités dans différents sous-systèmes (plus ou moins) indépendants.



1. Créer un répertoire *VideoTracker* pour le projet.
2. Puis créer l'arborescence suivante qui permettra d'organiser les fichiers pour le développement du logiciel *VideoTracker*

```
VideoTracker
   |
   |--- src
   |     |--- controllers
   |     |--- models
   |     |--- views
   |--- resources
   |     |--- images
   |     |--- videos
   |--- tests
```

Conception —> Tâches

Objectif : Décomposer le logiciel en gros modules, préciser les tâches permettant la construction de ces modules. L'identification des tâches permet :

- de mesurer le travail à réaliser
- d'organiser le travil dans une équipe
- de maîtriser les délais et les risques

1. À partir du document *tache.md*, écrire les tâches correspondantes aux exigences de la première partie.
2. Faire un tableau dans lequel vous indiquez
   - Le numéro de l'exigence
   - un numéro pour chacune des tâches ainsi que sa description
   - Le nom du responsable de cette tâche
   - Le temps estimé pour réaliser cette tâche
   - son état d'avancement

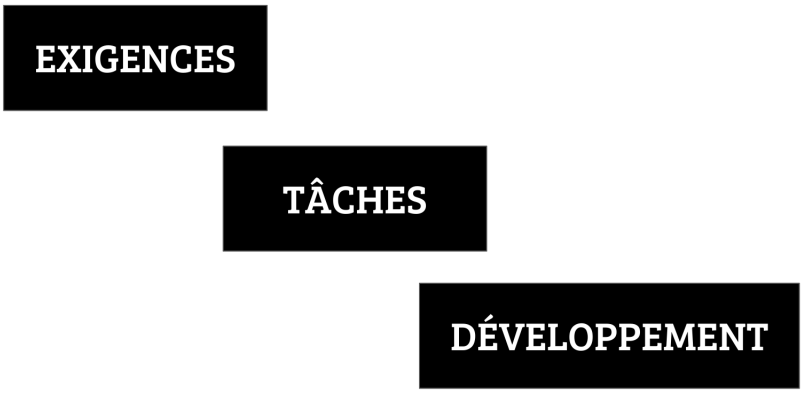


FIGURE 2 – Cycle de développement en cascade

Développement

Dans un premier temps nous ne nous occuperons pas de la partie IHM, c'est à dire de l'interface graphique qui permet à l'utilisateur d'intéragir avec le logiciel. Nous allons commencer le code de notre application en définissant les classes :

- Point
- Scale
- Origin

1. Associer chacune des classes à un fichier python rangé dans le bon dossier.
2. Implémenter les classes à partir des définitions des tâches.
3. Écrire un fichier Test.py qui permettra de tester votre travail.