Introduction

**I. PRÉSENTATION GÉNÉRALE**

**1.1. PRÉSENTATION DE L’ESTM**

**1.1.1. DOMAINE D’ACTIVITÉ**

L’ESTM est une école d’Ingénieurs et de Spécialistes du Management à la pointe des dernières technologies, s’appuyant sur un corps professoral très expérimenté.  
 Ils cultivent la passion, l’excellence, le savoir pratique, le professionnalisme, grâce à leurs excellentes relations internationales concrétisées par des partenariats privilégiés avec des institutions leaders dans leurs domaines respectifs.  
 L’objectif de l’ESTM n’est pas seulement de délivrer des formations reconnues, innovantes et de qualité à nos étudiants, mais c’est aussi de les accompagner dans leur insertion professionnelle et dans la construction d’une belle carrière.

**1.1.2. STRUCTURE ORGANIQUE ET FONCTIONNELLE**

**1.1.2.1. STRUCTURE ORGANIQUE**

ESTM a une structure organique qui est représenté comme suit :

* Un directeur général (D.G) ;
* Un directeur des études (D.E) ;
* Un service financier ;
* Un service chargé de communication et développement ;
* Un corps professoral ;

**1.1.2.2. STRUCTURE FONCTIONNELLE**

Elle explique la manière dont la structure organique fonctionne de manière détaillée.

**II. CHOIX DE LA MÉTHODE D’ANALYSE ET DE** **CONCEPTION DE LA SOLUTION**

**2.1. CHOIX DE LA MÉTHODE D’ANALYSE**

En ingénierie, une **méthode d'analyse et de conception** est un procédé qui a pour objectif de permettre de formaliser les étapes préliminaires du développement d'un système afin de rendre ce développement plus fidèle aux besoins du client.

Pour ce faire, on part d'un énoncé informel (le besoin tel qu'il est exprimé par le client, complété par des recherches d'informations auprès des experts du domaine fonctionnel, comme les futurs utilisateurs d'un logiciel), ainsi que de l'analyse de l'existant éventuel (c'est-à-dire la manière dont les processus à traiter par le système se déroulent actuellement chez le client).

**2.1.1. PRÉSENTATION DE LA MÉTHODE MERISE ET DU LANGAGE UML**

**2.1.1.1. PRESENTATION DE LA METHODE MERISE**

MERISE est une méthode de conception, de développement et de réalisation de projets informatiques. Le but de cette méthode est d'arriver à concevoir un système d'information. La méthode MERISE est basée sur la séparation des données et des traitements à effectuer en plusieurs modèles conceptuels et physiques.   
 La séparation des données et des traitements assure une longévité au modèle. En effet, l'agencement des données n'a pas à être souvent remanié, tandis que les traitements le sont plus fréquemment.

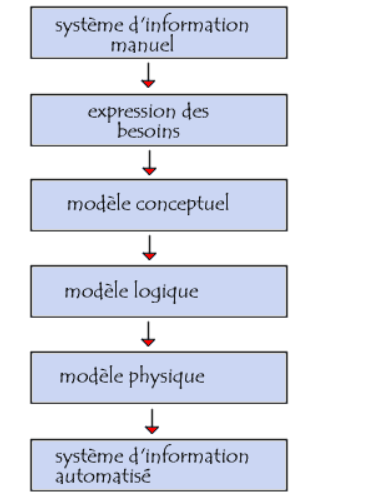
La méthode MERISE date de 1978-1979, et fait suite à une consultation nationale lancée en 1977 par le ministère de l'Industrie dans le but de choisir des sociétés de conseil en informatique afin de définir une méthode de conception de systèmes d'information.

Les deux principales sociétés ayant mis au point cette méthode sont le CTI (Centre Technique d'Informatique) chargé de gérer le projet, et le CETE (Centre d'Etudes Techniques de l'Equipement) implanté à Aix-en-Provence.

La conception du système d'information se fait par étapes, afin d'aboutir à un système d'information fonctionnel reflétant une réalité physique. Il s'agit donc de valider une à une chacune des étapes en prenant en compte les résultats de la phase précédente.

D'autre part, les données étant séparées des traitements, il faut vérifier la concordance entre données et traitements afin de vérifier que toutes les données nécessaires aux traitements sont présentes et qu'il n'y a pas de données superflues.

Cette succession d'étapes est appelée cycle d'abstraction pour la conception des systèmes d’information :



L'expression des besoins est une étape consistant à définir ce que l'on attend du système d'information automatisé, il faut pour cela :

* Faire l'inventaire des éléments nécessaires au système d'information
* Délimiter le système en s'informant auprès des futurs utilisateurs

Cela va permettre de créer le **MCC** (Modèle conceptuel de la communication) qui définit les flux d'informations à prendre en compte.

L'étape suivante consiste à mettre au point le **MCD** (Modèle conceptuel des données) et le **MCT** (Modèle conceptuel des traitements) décrivant les règles et les contraintes à prendre en compte.

Le modèle organisationnel consiste à définir le **MOT** (Modèle organisationnel des traitements) décrivant les contraintes dues à l'environnement (organisationnel, spatial et temporel).

Le modèle logique représente un choix logiciel pour le système d'information.

Le modèle physique reflète un choix matériel pour le système d'information.

**2.1.1.2. PRESENTATION DU LANGAGE UML**

Le **Langage de Modélisation Unifié**, de l'anglais *Unified Modeling Language* (**UML**), est un [langage](https://fr.wikipedia.org/wiki/Langage) de modélisation graphique à base de [pictogrammes](https://fr.wikipedia.org/wiki/Pictogramme) conçu pour fournir une méthode normalisée pour visualiser la conception d'un système. Il est couramment utilisé en [développement logiciel](https://fr.wikipedia.org/wiki/D%C3%A9veloppement_logiciel) et en [conception orientée objet](https://fr.wikipedia.org/wiki/Programmation_orient%C3%A9e_objet).

L'UML est le résultat de la fusion de précédents langages de modélisation objet : [Booch](https://fr.wikipedia.org/wiki/M%C3%A9thode_Booch" \o "Méthode Booch), [OMT](https://fr.wikipedia.org/wiki/Object_Modeling_Technique), [OOSE](https://fr.wikipedia.org/wiki/OOSE). Principalement issu des travaux de [Grady Booch](https://fr.wikipedia.org/wiki/Grady_Booch), [James Rumbaugh](https://fr.wikipedia.org/wiki/James_Rumbaugh) et [Ivar Jacobson](https://fr.wikipedia.org/wiki/Ivar_Jacobson), UML est à présent un standard adopté par l'[Object Management Group](https://fr.wikipedia.org/wiki/Object_Management_Group) (OMG). UML 1.0 a été normalisé en janvier 1997 ; UML 2.0 a été adopté par l'OMG en juillet 2005.La dernière version de la spécification validée par l'OMG est UML 2.5.1 (2017).

UML est utilisé pour spécifier, visualiser, modifier et construire les documents nécessaires au bon développement d'un logiciel orienté objet. UML offre un standard de modélisation, pour représenter l'architecture logicielle. Les différents éléments représentables sont :

* Activité d'un objet/logiciel
* Acteurs
* Processus
* Schéma de base de données
* Composants logiciels
* Réutilisation de composants

Grâce aux outils de modélisation UML, il est également possible de générer automatiquement tout ou partie du code d'une application logicielle, par exemple en langage Java, à partir des divers documents réalisés. La version actuelle, UML 2.5, propose 14 types de diagrammes dont 7 structurels et 7 comportementaux. A titre de comparaison, UML 1.3 comportait 25 types de diagrammes.

UML n'étant pas une méthode, l'utilisation des diagrammes est laissée à l'appréciation de chacun. Le diagramme de classes est généralement considéré comme l'élément central d'UML. Des méthodes, telles que le processus unifié proposé par les créateurs originels de UML, utilisent plus systématiquement l'ensemble des diagrammes et axant l'analyse sur les cas d'utilisation (« use case ») pour développer par itérations successives un modèle d'analyse, un modèle de conception, et d'autres modèles. D'autres approches se contentent de modéliser seulement partiellement un système, par exemple certaines parties critiques qui sont difficiles à déduire du code.

UML se décompose en plusieurs parties :

* Les vues : ce sont les observables du système. Elles décrivent le système d'un point de vue donné, qui peut être organisationnel, dynamique, temporel, architectural, géographique, logique, etc. En combinant toutes ces vues, il est possible de définir (ou retrouver) le système complet.
* Les diagrammes : ce sont des ensembles d'éléments graphiques. Ils décrivent le contenu des vues, qui sont des notions abstraites. Ils peuvent faire partie de plusieurs vues.
* Les modèles d'élément : ce sont les éléments graphiques des diagrammes.

**2.1.2. COMPARAISON ET JUSTIFICATION DU CHOIX DES APPROCHES MERISE ET UML**

**2.1.2.1. COMPARAISON DES APPROCHES MERISE ET UML**

Merise propose une approche descendante où le système réel est décomposé en activités, elles-mêmes déclinées en fonctions. Les fonctions sont composées de règles de gestion, elles-mêmes regroupées en opérations. Ces règles de gestion au niveau conceptuel génèrent des modules décomposés en modules plus simples et ainsi de suite jusqu'à obtenir des modules élémentaires... Les limites d'une telle approche résident dans le fait que les modules sont difficilement extensibles et exploitables pour de nouveaux systèmes.

Dans UML les fonctions cèdent la place aux cas d'utilisation qui permettent de situer les besoins de l'utilisateur dans le contexte réel. A chaque scénario correspond des diagrammes d'interaction entre les objets du système et non pas un diagramme de fonction... La méthode Merise ressemble à la méthode UML pour la phase de modélisation de la base de données. La différence principale est que Merise est une méthode d'analyse, et UML un langage de modélisation de données. UML ne propose pas de cycle précis : les organisations sont libres de choisir le cycle qui leur convient.

UML fonctionne sur un principe d’itérations qui ne s’oppose pas aux phases définies dans MERISE.

MERISE découpe plus au travers de ses phases l’analyse métier et l’architecture logicielle. Dans UML, l’architecture logicielle a une place prépondérante et est intégrée très en amont dans l’élaboration du système d’information

Dans UML, l’avancement du projet est mesuré par le nombre de cas d’utilisation, de classes... réellement implantées et non par la documentation produite ce qui est le cas dans Merise. Les itérations servent en outre à répartir l’intégration et les tests tout au long du processus d’élaboration du système d’information.

**2.1.2.2. JUSTIFCATION DU CHOIX DE LA MÉTHODE**

Merise et UML ont des caractéristiques voisines au niveau de la modélisation des bases de données mais également des points de divergence. En effet :

* + La méthode MERISE nécessite une démarche par étape qui favorise la qualité de chaque modèle avec ses différents niveaux de validations. Alors que le langage UML n’impose pas de méthode de travail particulière.
  + MERISE présente l'intérêt d'avoir des modèles logiques moins détaillés facilement compréhensibles. En revanche UML conçu pour s'adapter à n'importe quel langage de programmation orientée objet (POO), présente plusieurs modèles (diagrammes) dont leurs compréhensions nécessitent une grande attention.
  + MERISE est moins préférable. Malgré sa clarté, il manque une précision du fait qu'elle est éloignée du langage donc difficile à implémenter alors mais UML intègre les éléments communs des différents langages, sa volonté est d'être fidèle à la réalisation finale. Elle est beaucoup plus complète avec ses différents diagrammes.
  + Pour en finir avec l'exploitabilité, MERISE est une méthode plus généraliste. Elle donne une vue globale de la solution sans autant entrer dans les petits détails. Contrairement à UML qui est conçu pour l'implémentation objet avec ses différents détails et sa portabilité (s'adapte à n'importe quelle plateforme) elle est donc plus exploitable.

L'une ou l'autre présente des avantages et des inconvénients. Il est réservé au concepteur de choisir la méthode la mieux adaptée pour son cas. Si on cherche la précision et l'exploitabilité UML devance MERISE. En revanche si c'est la clarté et l'accessibilité qui sont en question MERISE est préférable. Dans notre cas, on va gérer des données plus complexes, donc il est préférable d’utiliser UML, vu qu’elle rend la modélisation plus simple à implémenter.

**2.1. ANALYSE ET CONCEPTION DE LA SOLUTION**