UNIVERSITE ASSANE SECK DE ZIGUINCHOR
UFR Sciences et Technologies
**********
DEPARTEMENT D'INFORMATIQUE

# Modélisation Orientée Objet
# Unified Modeling Language (UML)

## L2 Informatique Ingénierie
## 2021-2022

© Bassirou DIENE



Unified Modeling Language

## OBJECTIFS DU COURS

- Introduction à l'ingénierie du logiciel
  - Principes et objectifs
- Modélisation avec UML(Unified Modeling Language)
  - Comprendre les concepts clés de l'approche orientée objet
  - Savoir aborder un problème en se basant sur une approche orientée objet
  - Maîtriser les principaux diagrammes UML qui permettent de concevoir un système orienté objet

Unified Modeling Language

# MOTIVATIONS (1/3)

- Systèmes d'information des entreprises sont composé de matériels et de logiciels
  - 80 % de logiciel (software)
  - 20 % de matériel (hardware)
- Depuis quelques années, la fabrication du matériel est assurée par quelques fabricants seulement
  - Le matériel est relativement fiable
  - Le marché est standardisé
- Les problèmes liés à l'informatique sont essentiellement des problèmes de **Logiciel (applications)**

2



Unified Modeling Language

# MOTIVATIONS (2/3)

- Un logiciel ou une application est un ensemble de programmes, qui permet à un ordinateur ou à un système informatique d'assurer une tâche ou une fonction particulière
  - **Exemple**: logiciel de comptabilite, logiciel de gestion des prêts, etc.
- Suivant leur taille, ils peuvent être développés par une personne seule, une petite équipe, ou un ensemble d'équipes coordonnées
- Le développement de grands logiciels par de grandes équipes pose d'importants problèmes de conception et de coordination

3

Unified Modeling Language

# MOTIVATIONS (3/3)

- La plus part des applications dans les entreprises de développement
  - *dépassements en coût (budget ) et délai*
  - *abandonnés durant leur développement.*
- Pour ces raisons, le développement de logiciels dans un contexte professionnel suit souvent des règles strictes encadrant la conception et permettant le travail en groupes et la maintenance du code
- Naissance d'une nouvelle discipline: le Génie Logiciel





Unified Modeling Language

# QUELQUES PRINCIPES DU GL

- **Généralisation**
  - regroupement d'un ensemble de fonctionnalités semblables en une fonctionnalité paramétrable (généricité, héritage)
- **Structuration**
  - façon de décomposer un logiciel (utilisation d'une méthode bottom-up ou top-down)
- **Abstraction**
  - mécanisme qui permet de présenter un contexte en exprimant les éléments pertinents et en omettant ceux qui ne le sont pas.

# *PHASES DE RÉALISATION D'UN SYSTÈME D'INFORMATION*



Unified Modeling Language

## PHASES DE DÉVELOPPEMENT D'UN LOGICIEL

- Définition des objectifs
- Analyse des besoins et faisabilité
- Spécifications ou conception générale
- Conception détaillée
- Codage (Implémentation ou programmation)
- Tests et validation
- Intégration
- Qualification (ou recette)
- Documentation
- Mise en production (déploiement)
- Maintenance

Unified Modeling Language

# DÉFINITION DES OBJECTIFS

- Cet étape consiste à définir la finalité du projet et son inscription dans une stratégie globale

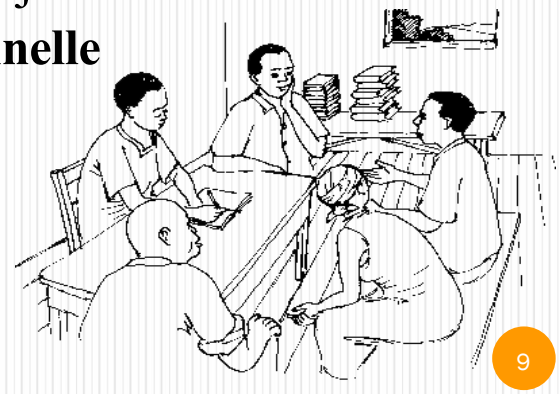




Unified Modeling Language

# ANALYSE DES BESOINS ET FAISABILITÉ

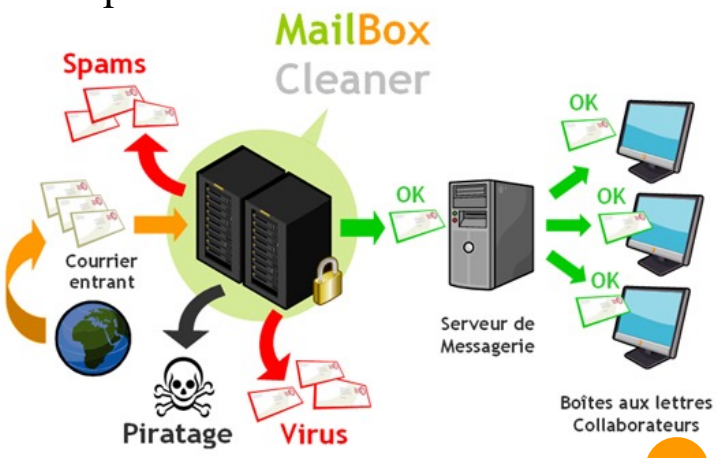
- Cerner les besoins des clients
  - clarifier, filtrer et organiser les besoins, ne pas chercher l'exhaustivité
- Délimiter les frontières du système
  - Spécifier le «quoi» fait par le logiciel
- Étudier la faisabilité du projet
  - **Faisabilité organisationnelle**
  - **Faisabilité technique**
  - **Faisabilité temporelle**
  - **Faisabilité financière**



Unified Modeling Language

# SPÉCIFICATIONS OU CONCEPTION GÉNÉRALE

- Il s'agit de l'élaboration des spécifications de l'architecture générale du logiciel
- De l'expression des besoins à la solution informatique
- Spécifications informatiques
  - **des fonctions**
  - **des données**
  - **des interfaces**



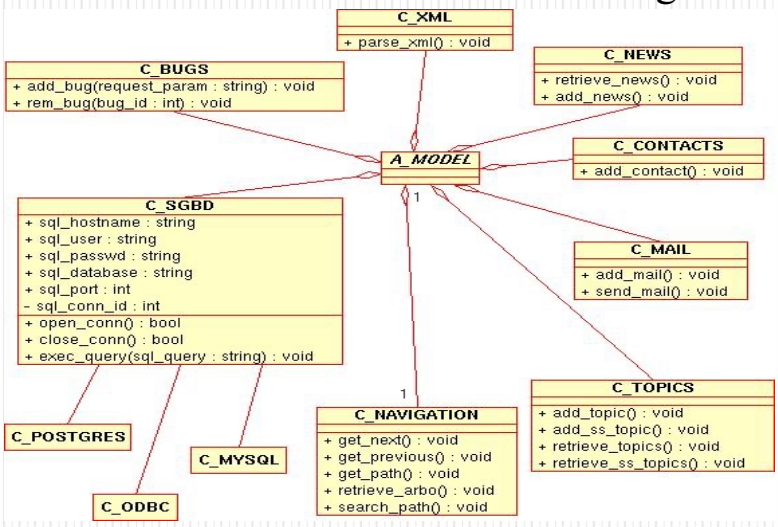





Unified Modeling Language

# CONCEPTION DÉTAILLÉE

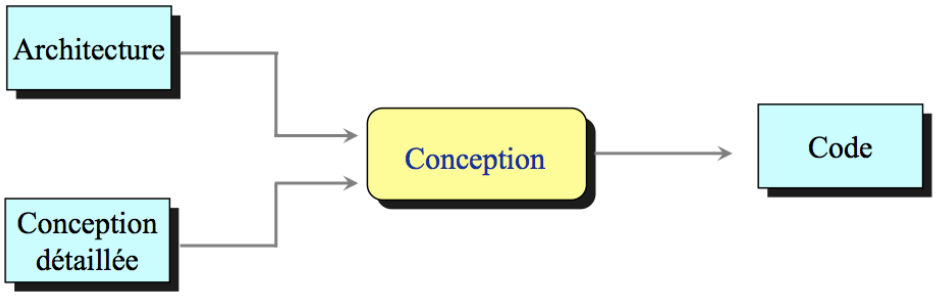
- Cette étape consiste à définir précisément chaque sous-ensemble du logiciel
  - Elaboration des objets du domaine et de ceux de l'application
  - Choix des structures de données et algorithmes



Unified Modeling Language

# CODAGE (IMPLÉMENTATION OU PROGRAMMATION)

- Traduction des fonctionnalités définies lors des phases de conception dans un langage de programmation
  - Création des modules et des bases de données
    - un module pour réaliser une fonctionnalité donnée
    - Adaptation et/ ou modification des modules existants
  - Intégration des différents modules.
    - Exemple : C++, Java, PHP, MySQL…



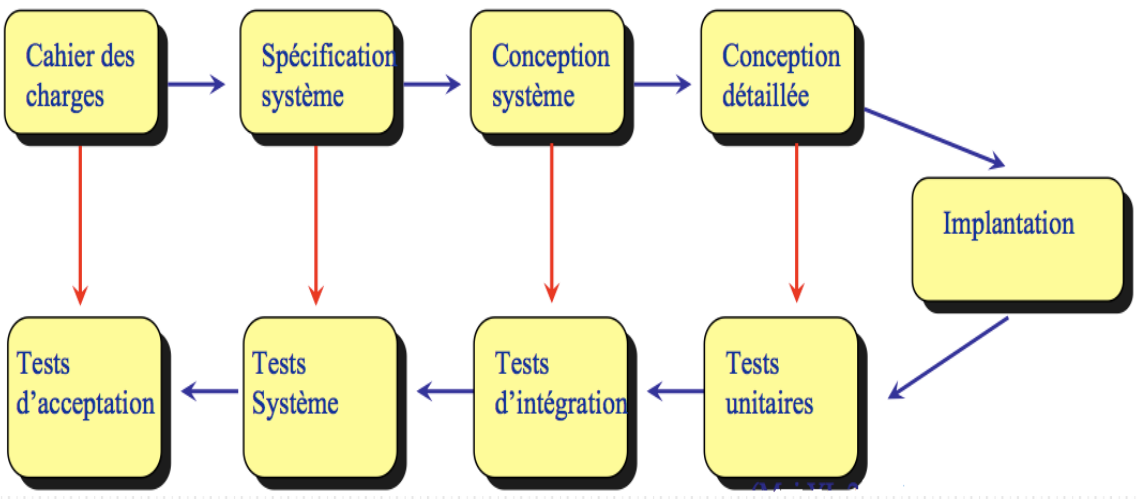

12



Unified Modeling Language

# TESTS VALIDATION

- Objectif de cette phase :
  - tester/valider les différents artefacts logiciels




13

Unified Modeling Language

# TESTS VALIDATION

- **Tests unitaires:** permettent de vérifier individuellement que chaque sous-ensemble du logiciel implémenté est conforme aux spécifications
- **Tests d'intégration:** permettent de vérifier que tous les programmes testés individuellement fonctionnent bien ensemble.
- **Tests systèmes:** permettent de vérifier que le système fonctionne correctement dans les conditions réelles d'utilisation.
- **Tests Alpha:** faire tester le logiciel par le client sur le site de développement
- **Tests Bêta:** faire tester le logiciel par le client sur le site de production





Unified Modeling Language

# TESTS VALIDATION- EXEMPLES DE TECHNIQUES

Audit

Revue/inspection

Test

Unified Modeling Language

# INTÉGRATION, QUALIFICATION

- Combiner les modules et vérifier le produit dans son ensemble
  - Essentiellement un problème de test

16



Unified Modeling Language

# QUALIFICATION (OU RECETTE)

- Vérification de la conformité du logiciel aux spécifications initiales

17

Unified Modeling Language

# DOCUMENTATION

- Elle vise à produire les informations nécessaires pour l'utilisation du logiciel et pour des développements ultérieurs






Unified Modeling Language

# MISE EN PRODUCTION ou DÉPLOIEMENT

- Le déploiement correspond à l'installation du produit fini chez le client
- Acteurs : programmeurs, ingénieurs du client, utilisateurs

Unified Modeling Language

# MAINTENANCE

- Elle comprend toutes les actions correctives (maintenance corrective) et évolutives (maintenance évolutive) sur le logiciel





# COMMENT RÉUSSIR UN LOGICIEL?

Unified Modeling Language

# UTILISER DES MÉTHODES

- **Objectifs**
  - Spécifier et planifier les étapes de l'analyse et de la conception
- **Composition**
  - **une démarche:** explique la procédure à suivre en exploitant au mieux les principes de modularité, d'abstraction, de réutilisation, etc.
  - **un formalisme de représentation**: facilite la communication, l'organisation et la vérification
  - **des modèles**: facilitent les retours sur la conception et l'évolution des applications





Unified Modeling Language

# NOTIONS DE MODÈLE

- C'est une représentation abstraite et simplifiée (i.e. qui exclut certains détails) d'une entité (phénomène, processus, système, etc.) du monde réel en vue de le décrire, de l'expliquer ou de le prévoir
- Un modèle est construit afin de mieux comprendre les systèmes développés
- Un modèle permet de réduire la complexité d'un système en éliminant les détails qui n'influencent pas son comportement de manière significative
- Un modèle reflète ce que le concepteur croit important pour la compréhension et la prédiction du phénomène modélisé

## EXEMPLES DE MODÈLES PRÉDICTIFS

- **Modèle météorologique:** à partir de données d'observation (satellite . . .), il permet de prévoir les conditions climatiques pour les jours à venir
- **Modèle économique:** peut par exemple permettre de simuler l'évolution de cours boursiers en fonction d'hypothèses macro-économiques (évolution du chômage, taux de croissance . . .)
- **Modèle démographique:** définit la composition d'un panel d'une population et son comportement, dans le but de fiabiliser des études statistiques, d'augmenter l'impact de démarches commerciales, etc.







## MÉTHODES D'ANALYSE ET DE CONCEPTION

- Elles fournissent une méthodologie et des notations standards qui aident à concevoir des logiciels de qualité
- Il existe de nombreuses méthodes:
  - **Méthodes fonctionnelles**: hiérarchie de fonction
    - SADT, SA-SD, etc.
  - **Méthodes systémiques**: séparation des données et des traitements
    - Merise, Entité Association, etc.
  - **Méthodes objets**: intégration des données et des traitements dans un objet unique
    - OMT, OOSE, etc.

Unified Modeling Language

# MÉTHODES ORIENTÉES OBJETS

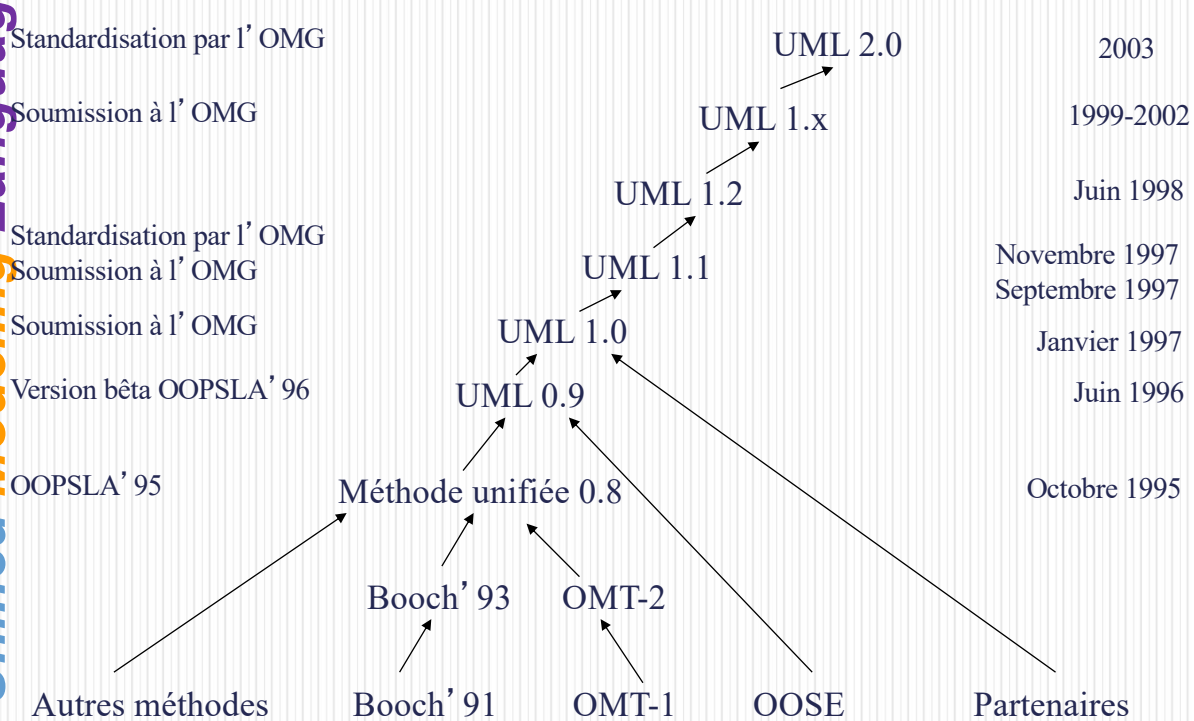
- Entre 89 et 94 : le nombre de méthodes orientées objet est passé de 10 à plus de 50
- Toutes les méthodes avaient pourtant d'énormes points communs (objets, méthode, paramètres, …)
- Au milieu des années 90, G. Booch, I. Jacobson et J. Rumbaugh ont chacun commencé à adopter les idées des autres. Les 3 auteurs ont souhaité créer un langage de modélisation unifié: Naissance d'UML





Unified Modeling Language

# HISTORIQUE



Unified Modeling Language

# Aujourd'hui

- UML est le langage de modélisation orienté objet le plus connu et le plus utilisé au monde
- UML s'applique à plusieurs domaines
- UML n'est pas une méthode
- Peu d'utilisateurs connaissent le standard, ils ont une vision outillée d'UML (Vision Utilisateur)
  - 5% forte compréhension, 45% faible compréhension, 50% aucune compréhension
- UML est fortement critiqué car pas assez formel
- Le marché UML est important et s'accroît
  - MDA, UML2.0, IBM a racheté Rational !!!





Unified Modeling Language