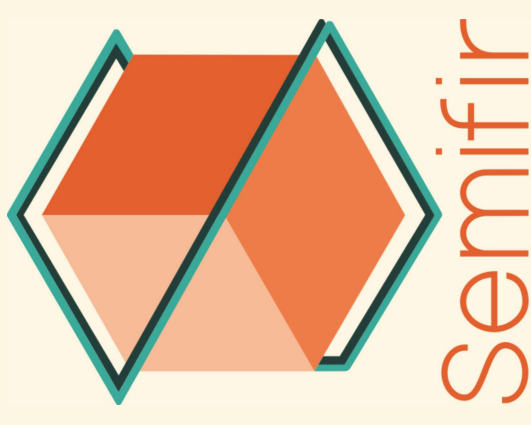


UML :

PRÉSENTATION



OBJECTIFS DU MODULE

- Concevoir des applications objets avec UML
- Décrire ce qu'est un design pattern
- Identifier les différents diagrammes
- Analyser un problème et le représenter avec UML

OBJECTIFS DU MODULE

- Formaliser les exigences sous forme de use cases
- Détailler les interactions entre objets avec les diagrammes UML
- Utiliser les dossiers de conception rédigés en UML.

PRÉREQUIS

- Idéalement : connaissance d'un langage de programmation

QU'EST-CE QUE L'UML ?

- Unified Modeling Language
- Publié en 1999 dans *The Unified Modeling Language User Guide*
- A été mis à jour en 2005 (UML V2)

BEFORE :

- Une cinquantaine de méthodes objet (années 90)
- Pas de méthodologie commune pour un même besoin
- Existait une "guerre" des méthodologies très contre productive

MÉTHODOLOGIES DE L'ÉPOQUE

- **OMT-2** : Basée sur l'analyse et l'abstraction
 - Par James Rumbaugh
- **Booch'93** : Basée sur les associations, traces d'événements
 - Par Grady Booch
- **OOSE** : Basé sur la modélisation d'objet
 - Par Ivar Jacobson

LA NAISSANCE DE L'UML

THE THREE AMIGOS



Grady Booch



Dr. James Rumbaugh



Dr. Ivar Jacobson

- Mélange et amélioration des trois méthodes
- Reprennent des idées d'autres méthodes connues

AUTRES INFLUENCES :

- Embley : Classes singletons et objets composites
- Fusion : Description des opérations, numérotation des messages
- Meyer : Pré/Post conditions
- Odell : Classification dynamique
- Shlaer-Mellor : Cycle de vie des objets

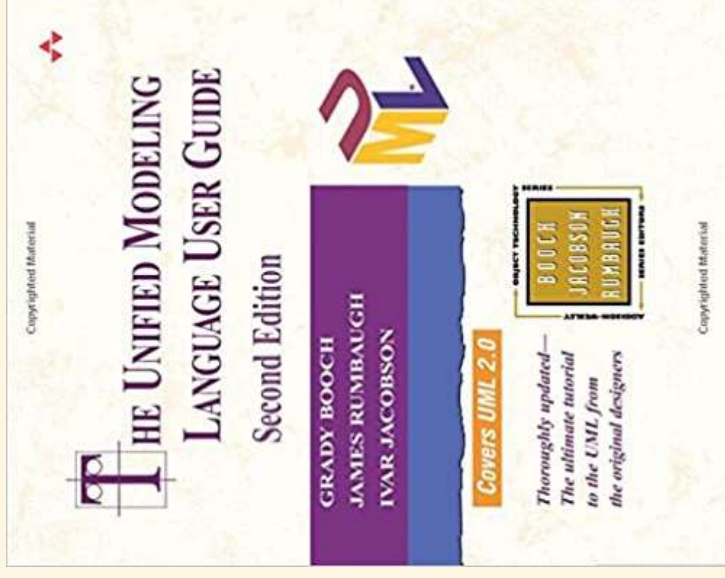
OBJECTIFS :

- Objectif : Répondre à l'augmentation de la complexité des systèmes informatiques :
 - Pour la conception,
 - Pour la compréhension informaticiens/non-informaticiens

OBJECTIFS :

- Etre capable de représenter la totalité d'un système,
- Créer un langage utilisable par les humains et les machines

THE HOLY BOOK



The Unified Modeling Language User Guide

L'UML À PROPREMENT PARLER :

- Langage de modélisation pensé comme langage de modélisation commun, à la fois visible et schématique,
- Destiné à la conception de logiciels et applications,
- Permet de spécifier et documenter les artefacts des systèmes

- Se présente sous forme de plans (diagrammes),
- Dispose de ses codes (syntaxe),
- Ce n'est pas un langage de programmation : il sert à concevoir et non à créer.

- Se concentre sur les artefacts du développement,
- N'a pas pour but de standardiser les processus

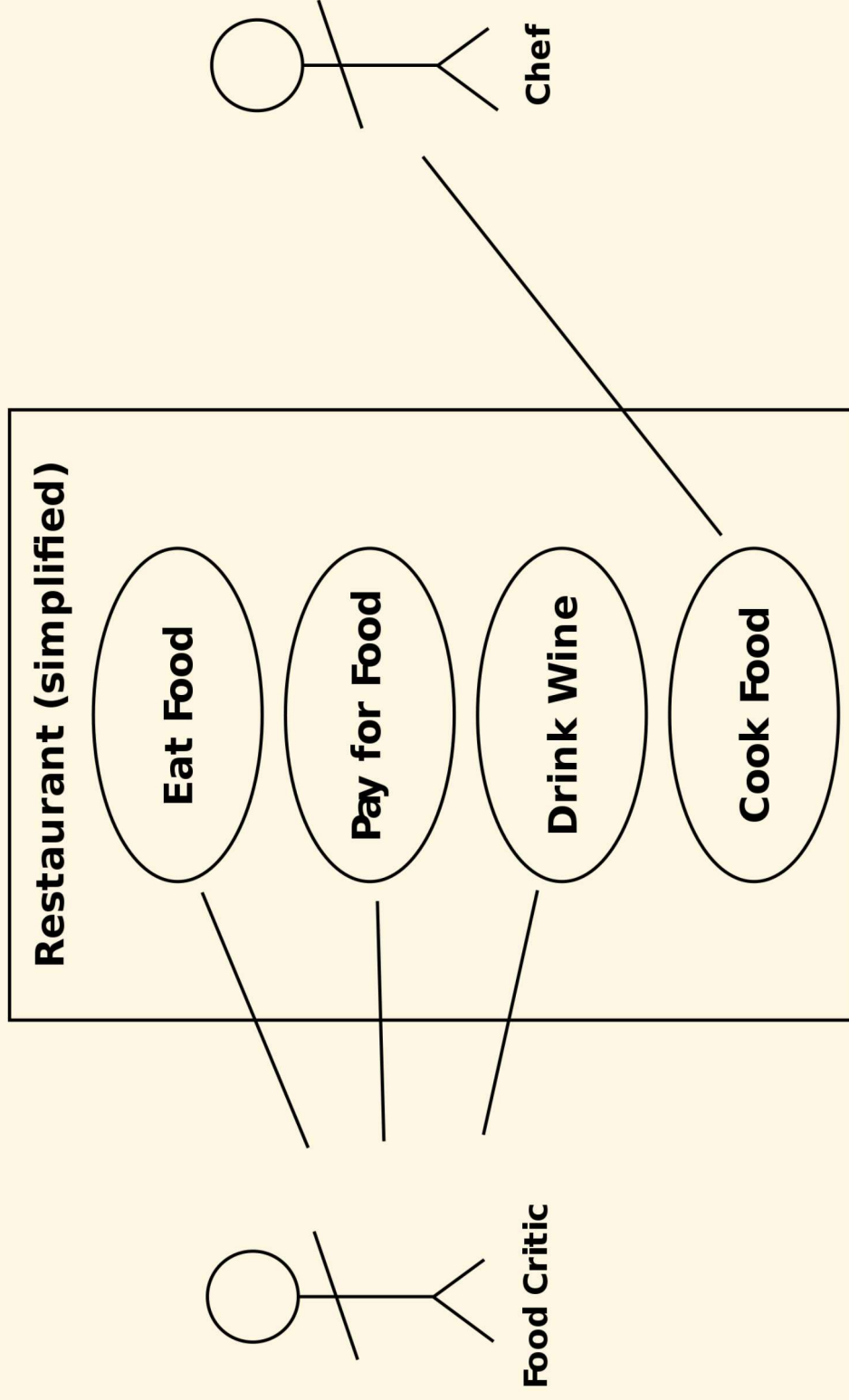
LES DIAGRAMMES

On distingue 13 types de diagrammes qui remplissent chacun des objectifs distincts. Voici la liste des plus utiles pour un développeur

DIAGRAMME DE CAS D'UTILISATION

- On y représente les différents acteurs (admins, clients, visiteurs...) et les interactions possibles entre eux et l'application
- On y tracera le cheminement des différents cas de la manière la plus exhaustive possible, en précisant ou non les prérequis des actions

EXAMPLE



*Finally, once the diagram
of activity is done, we can pull the list
of features of our application !*

DIAGRAMME D'ACTIVITÉ

- Représentation graphique qui représente le cycle de vie d'une activité émise par un acteur,
- Prend en compte l'aspect chronologique de l'activité,
- Similaire à un workflow

EXEMPLE

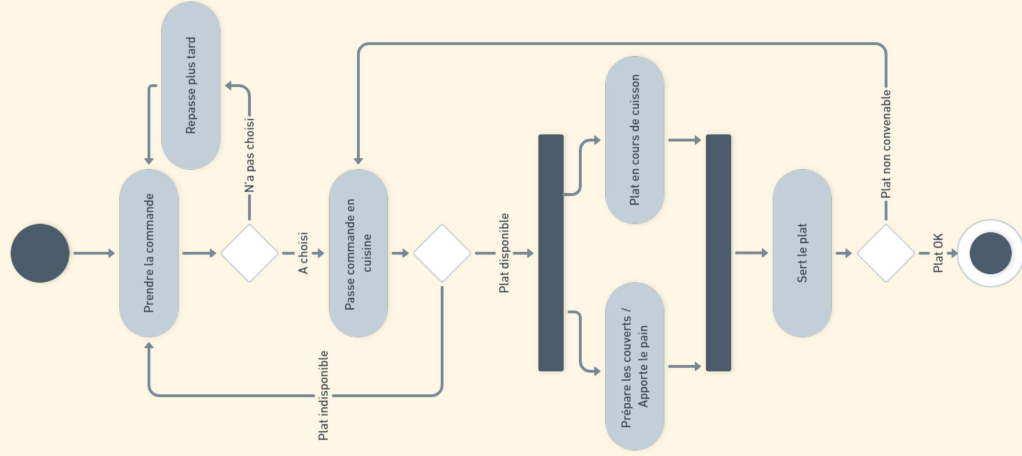


DIAGRAMME DE CLASSE

- Certainement un des diagrammes les plus utilisés par les développeurs.
- Sert à décrire la structure du système

ON Y MODÉLISE :

- Les classes,
- Les attributs,
- Les opérations,
- Les relations entre objets

EXAMPLE:

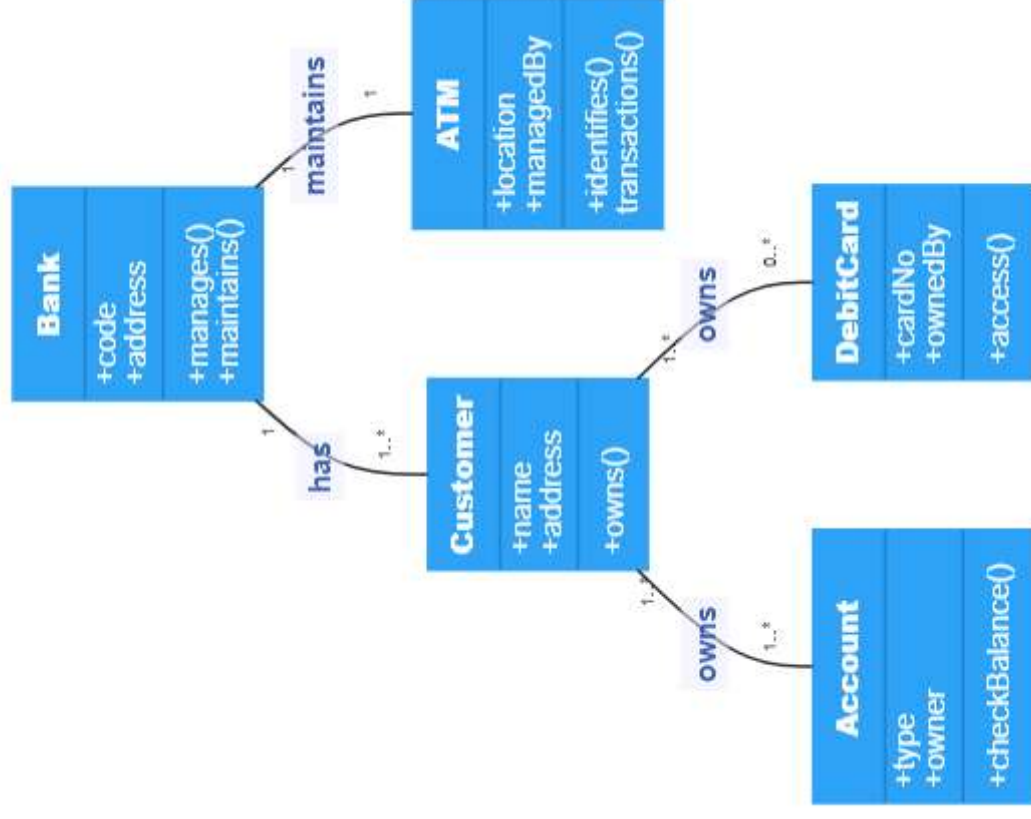


DIAGRAMME D'OBJET

Représente une instance spécifique d'un diagramme à un moment précis. Il ressemble beaucoup au diagramme de classe, si bien qu'il sont souvent mixés.

EXEMPLE

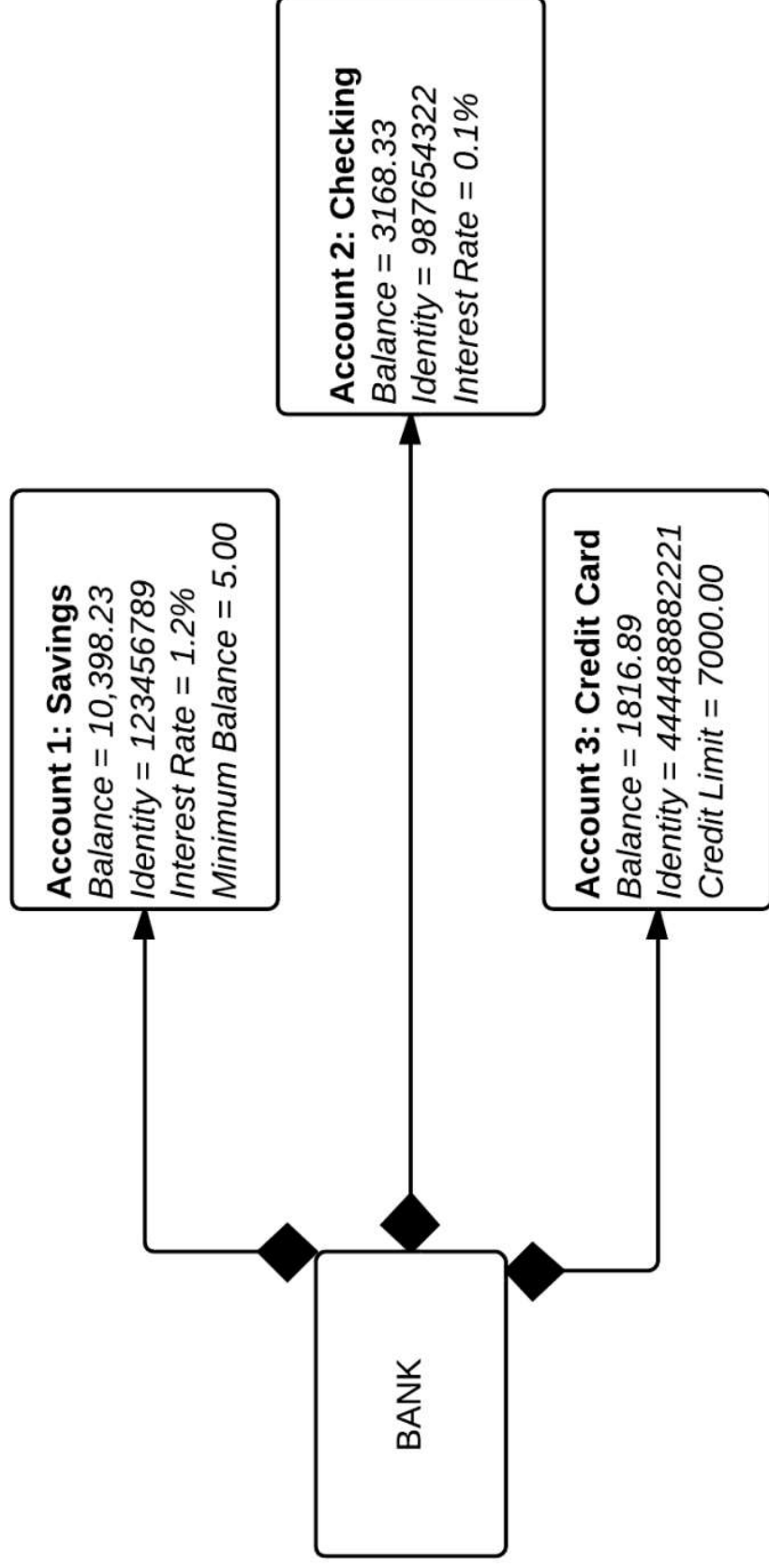
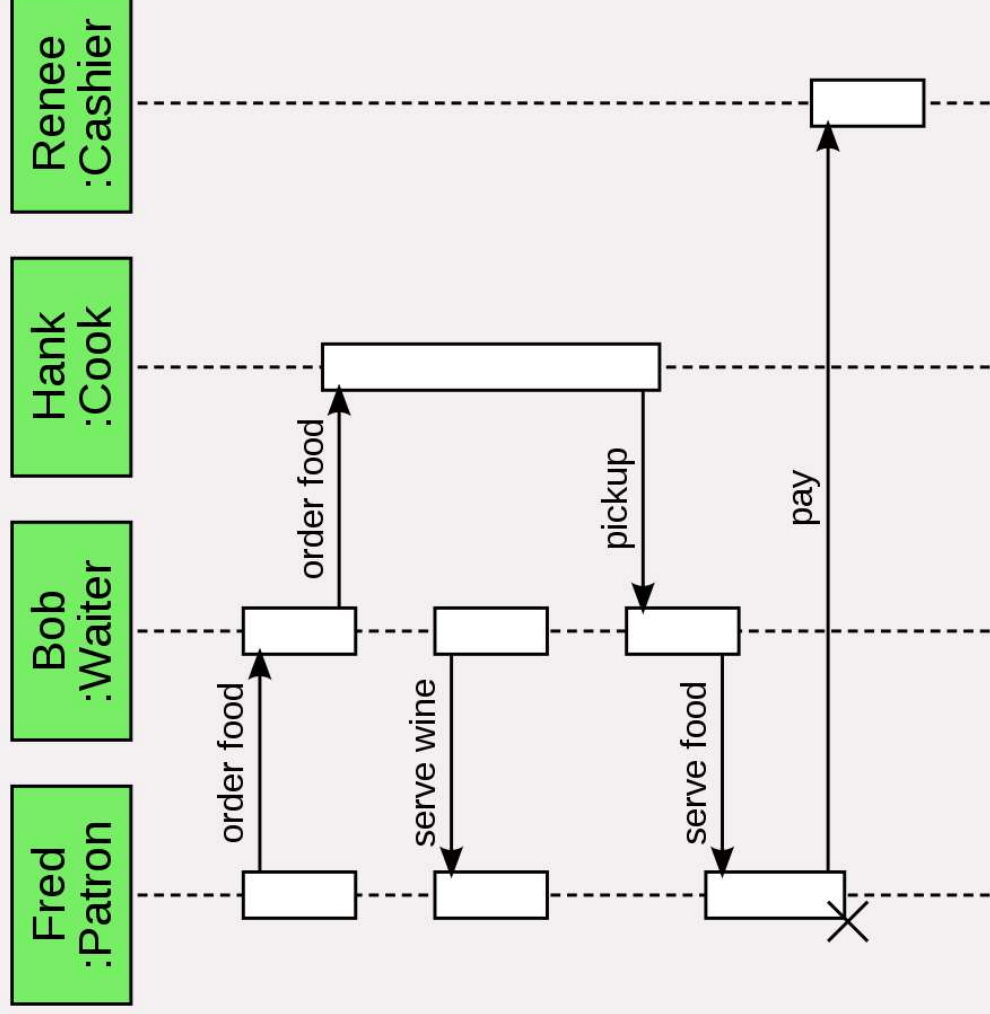


DIAGRAMME DE SÉQUENCE

- Sert à représenter les interactions entre acteurs,
- Prend en compte la gestion du temps
- Peuvent être appelés "Diagrammes d'événements" ou "Scénario d'événements"

EXEMPLE



AUTRES DIAGRAMMES

Voici les autres diagrammes UML existants qui ne seront pas abordé en détail durant ce cours.

DIAGRAMME DE COMPOSANT

- Permet d'identifier les composants du système
 - Structure physique,
 - Relations entre composants,
 - Comportement services/interface

EXEMPLE

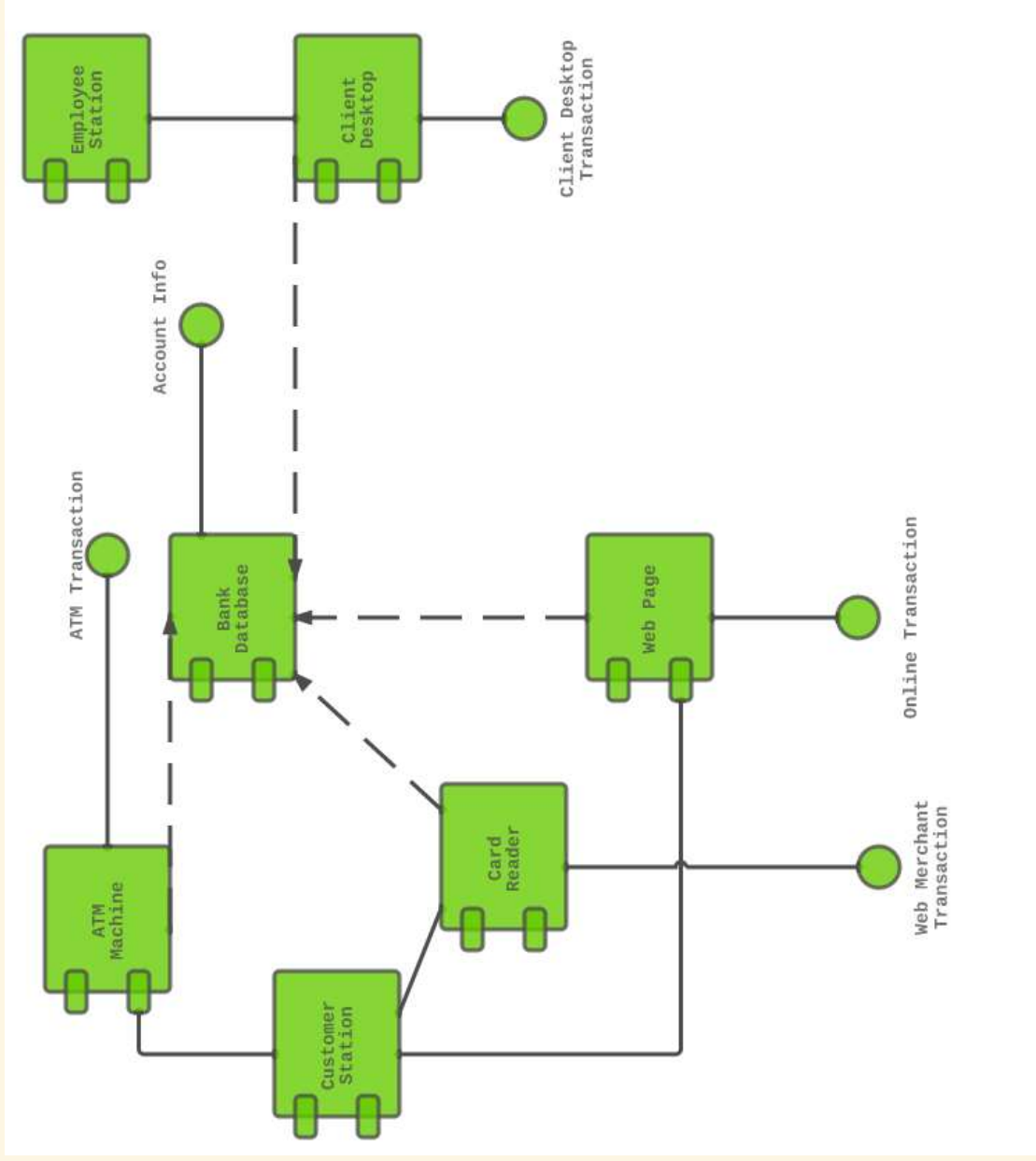


DIAGRAMME DE DÉPLOIEMENT

- Permet de faire le lien entre les composants logiciels et matériels,
- Illustre le traitement/l'exécution
- Permet de visualiser les composants matériels (serveurs, etc.)

EXEMPLE

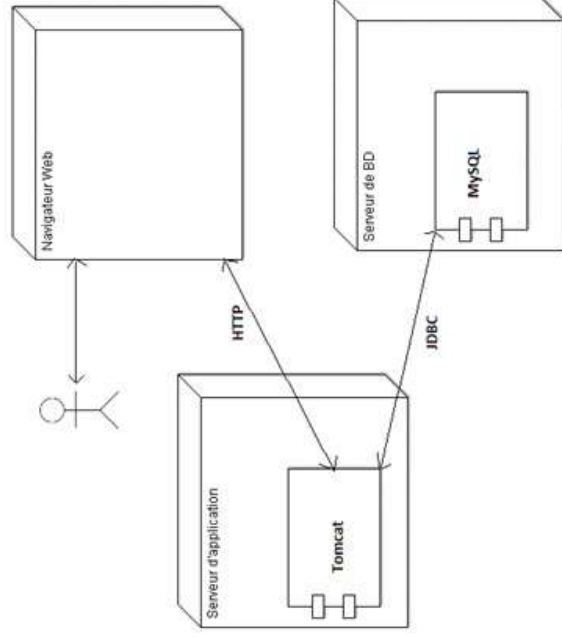


DIAGRAMME D'ÉTAT

- Se concentre sur un objet/ensemble d'objet,
- Fait état des différents cycles d'évolution possible
- Ressemble beaucoup au diagramme d'activité, si bien que les deux sont souvent mélangés

LES OUTILS DE RÉDACTION UML

Dans ce cours, nous utiliserons [PlantText](#) qui permet de traduire une notation en schéma. La documentation [peut être trouvée sur ce site](#)

PLANTEXT

- On utilisera des symboles pour illustrer les éléments que l'on souhaite afficher
- Plantext interprêtera les informations pour créer le diagramme en question

EXEMPLE

