Cours 6 – Architectures de Logiciels

MODULE INTRODUCTION AU GÉNIE LOGICIEL

MOSTEFAI MOHAMMED AMINE – BATATA SOFIANE

COPYRIGHT © ESI 2015-2016

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Objectifs du Cours

Présenter la notion d'architecture logicielle

Donner un aperçu des principaux styles architecturaux

Découvrir le lien entre l'architecture et le déploiement physique à travers le diagramme de déploiement

Plan du Cours

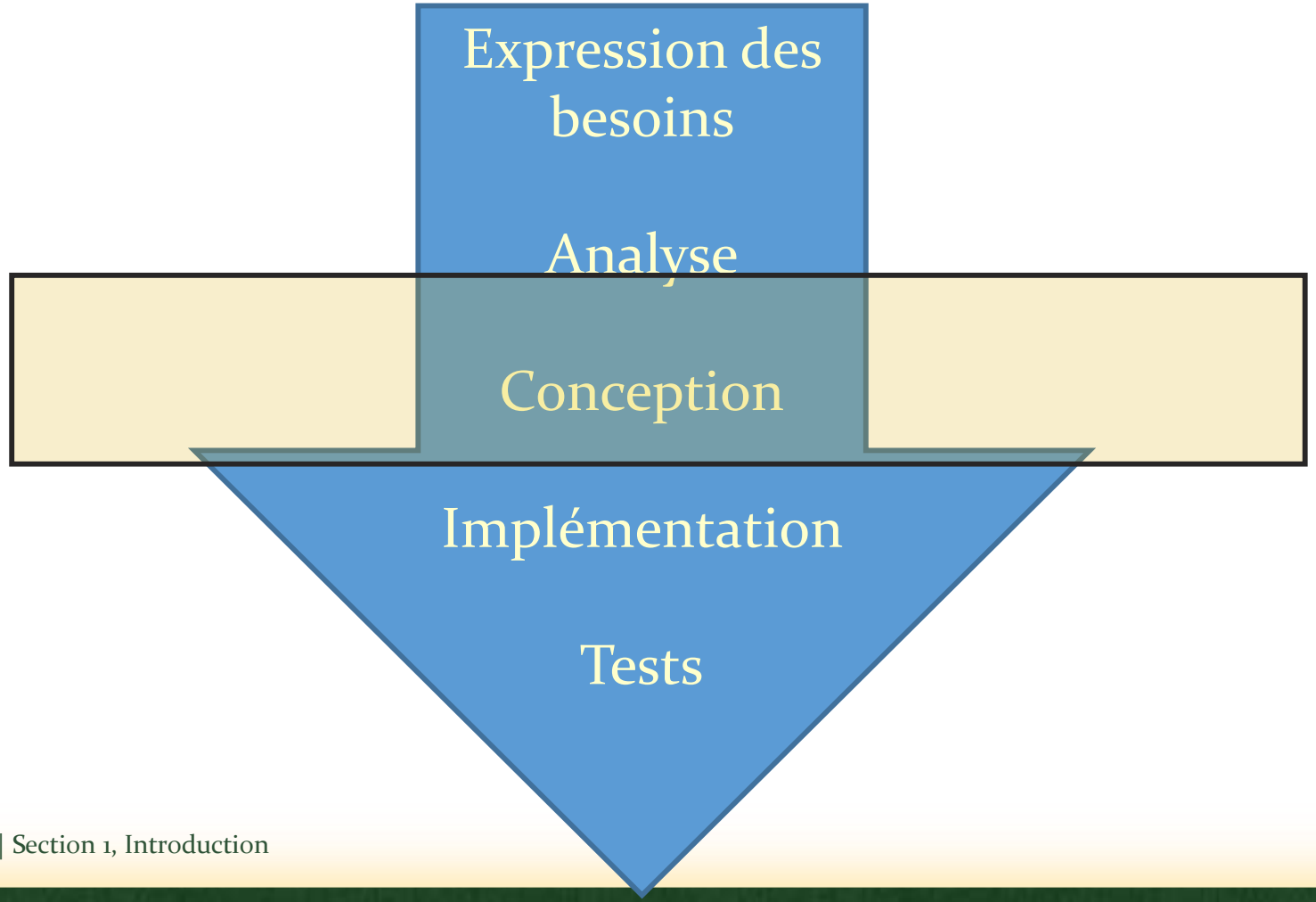
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Introduction

SECTION 1

Cycle de Vie



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Définition

L’architecture d’un programme ou d’un système informatique est la structure (ou les structures) du système qui comprend les éléments logiciels, leurs propriétés visibles et leur relations

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Introduction

- L'architecture d'un système est sa ***conception*** de haut niveau
- N'importe quel système complexe est composé de s***ous-systèmes*** qui interagissent entre eux
- La conception de haut niveau est le processus qui consiste à ***identifier*** ces sous-systèmes ainsi que les ***relations*** qu'ils entretiennent entre eux
- ***L'architecture*** d'un système est le résultat de ce processus

Introduction – Suite

- L'architecture implique plusieurs choix dont les ***technologies***, les ***produits*** et les ***serveurs*** à utiliser
- Il n'y a pas une architecture unique permettant de réaliser le système, il y en a plusieurs.
- Le ***concepteur*** ou ***l'architecte*** tâchera de choisir la meilleure architecture possible selon plusieurs critères dont la nature du projet, les compétences de l'équipe, les budgets et outils disponibles, …etc.

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Représentation des architectures

- Il existe ***plusieurs*** représentations graphiques des architectures
- Une des représentations les plus utilisées est la représentation ***C&C*** : ***Composants*** et ***Connecteurs***
- Un composant est un ***module logiciel*** (application, bibliothèque, module, …etc.) ou un ***entrepôt de données*** (base de données, système de fichiers, …etc.)
- Le connecteur représente les ***interactions*** entre les composants
- La représentation C&C est un ***graphe*** contenant des composants et des connecteurs

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Composants

- Un composant est un module logiciel ou un entrepôt de données
- Un composant est identifié par son ***nom*** qui indique son ***rôle*** dans le système
- Les composants communique entre eux en utilisant des ***ports*** (ou ***interfaces***)
- Les architectures utilisent des composants standards : ***serveurs***, ***bases de données***, ***application***, ***clients***, …etc.

Serveurs et clients

- Un serveur est un ***module*** logiciel qui répond aux requêtes d'autres modules appelés ***clients***
- Généralement, les services et les clients sont hébergés dans des machines différentes et communiquent via le ***réseau*** (intranet / internet)
- Par exemple, le service http répond aux requêtes des clients qui sont les navigateurs web

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Application

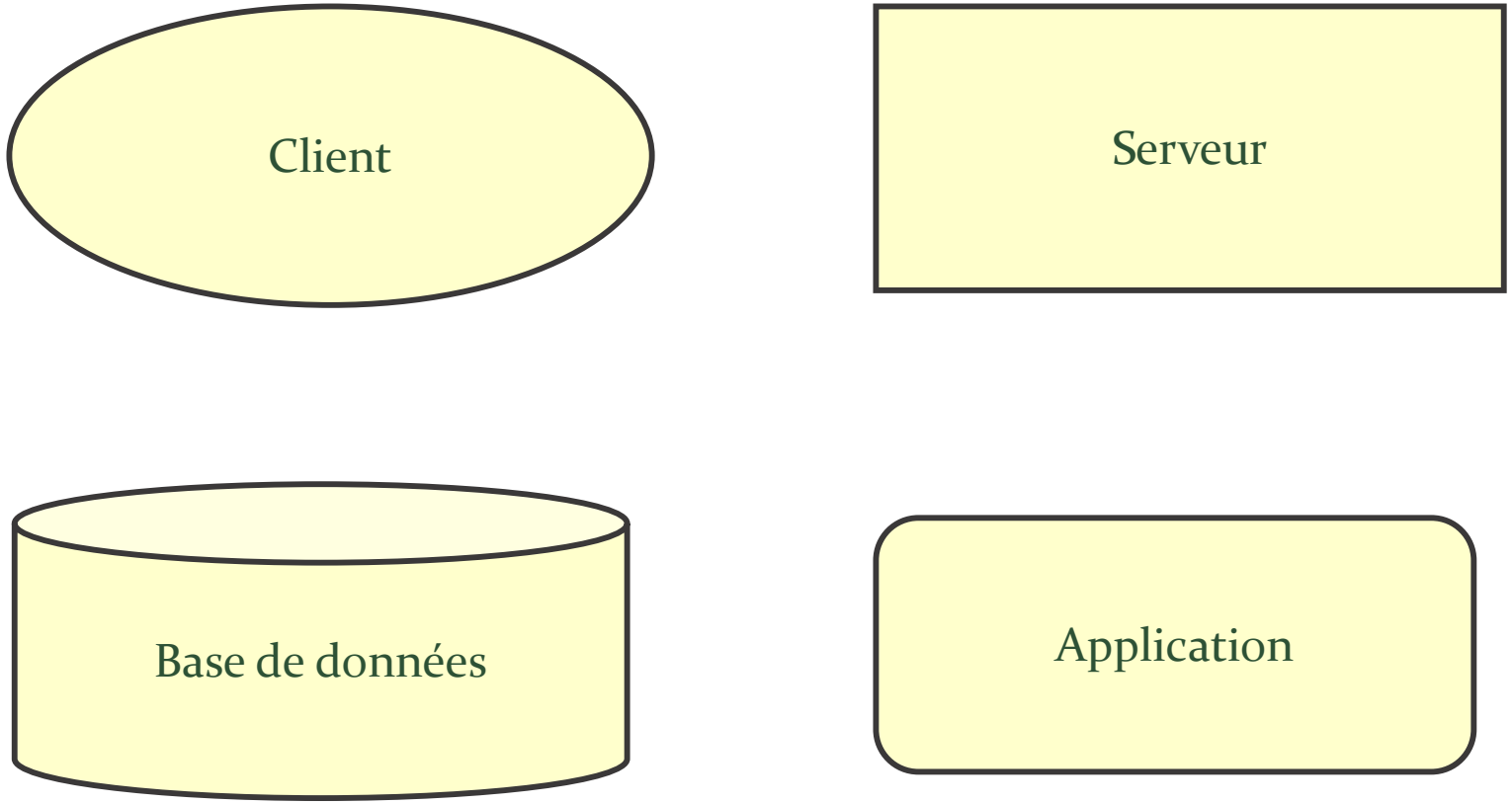
- Une application est un ***module logiciel*** qui a un rôle défini dans le système logiciel
- Par exemple, une application d'envoi de mails

Base de données

- Une base de données est un ***entrepôt*** stockant les données sous un format ***normalisé***
- Il y a deux grandes familles de bases de données : relationnelles et NoSQL
- L'interrogation et la modification des données relationnelles se fait en utilisant un langage spécial appelé ***SQL***
- Un SGBD (Système de Gestion de Base de Données) est une base de données puissante et accessible sur le ***réseau*** conçue généralement pour les gros systèmes
- SQL Server, Oracle, MySQL, PostgreSQL sont des exemples de SGBD relationnels
- MangoDB, CouchDB, Cassandra sont des exemples de BDD non-relationnelles

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Vue C&C



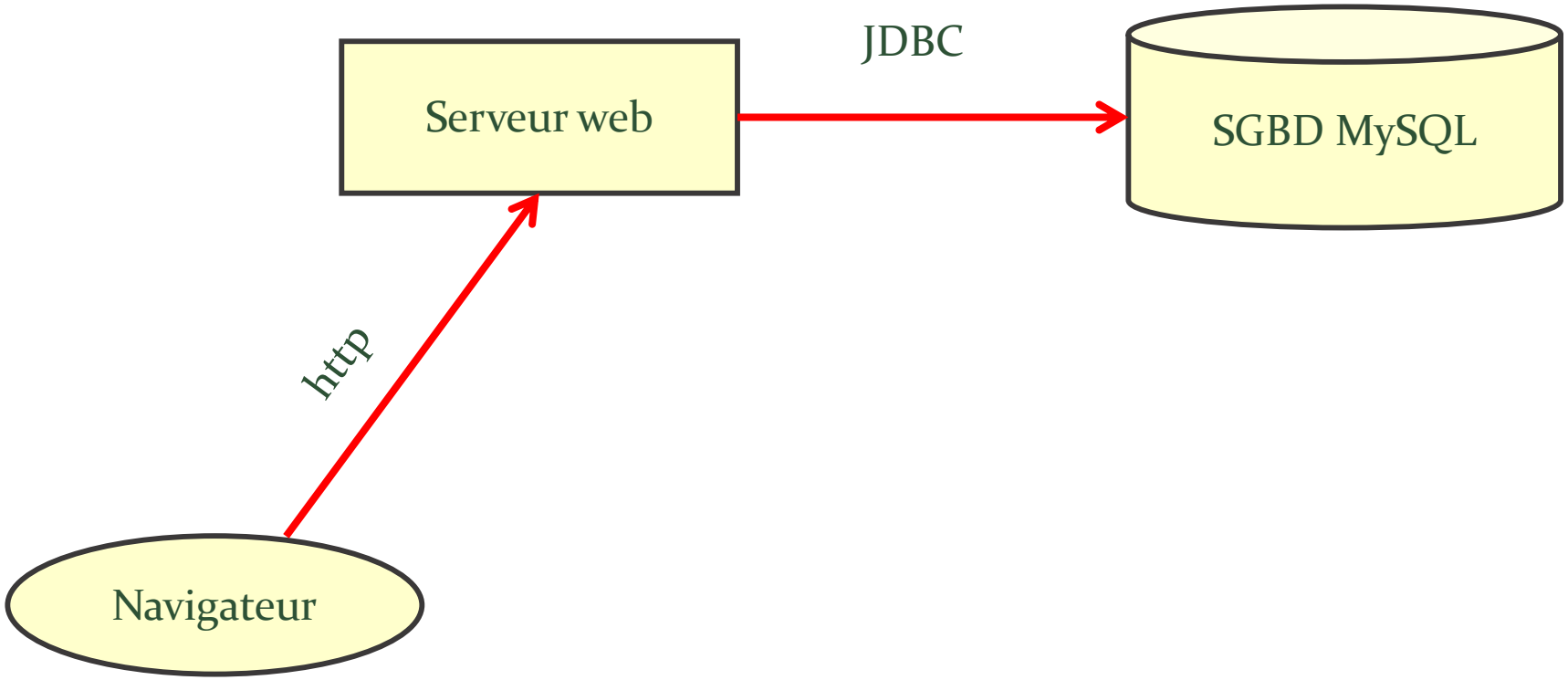
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Connecteurs

- Le connecteur modélise une ***interaction*** entre deux composants
- Un connecteur peut modéliser une interaction ***simple*** (appel de procédure) ou une interaction ***complexe*** (par exemple utilisation d'un protocole comme http)

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Exemple – Application JSP



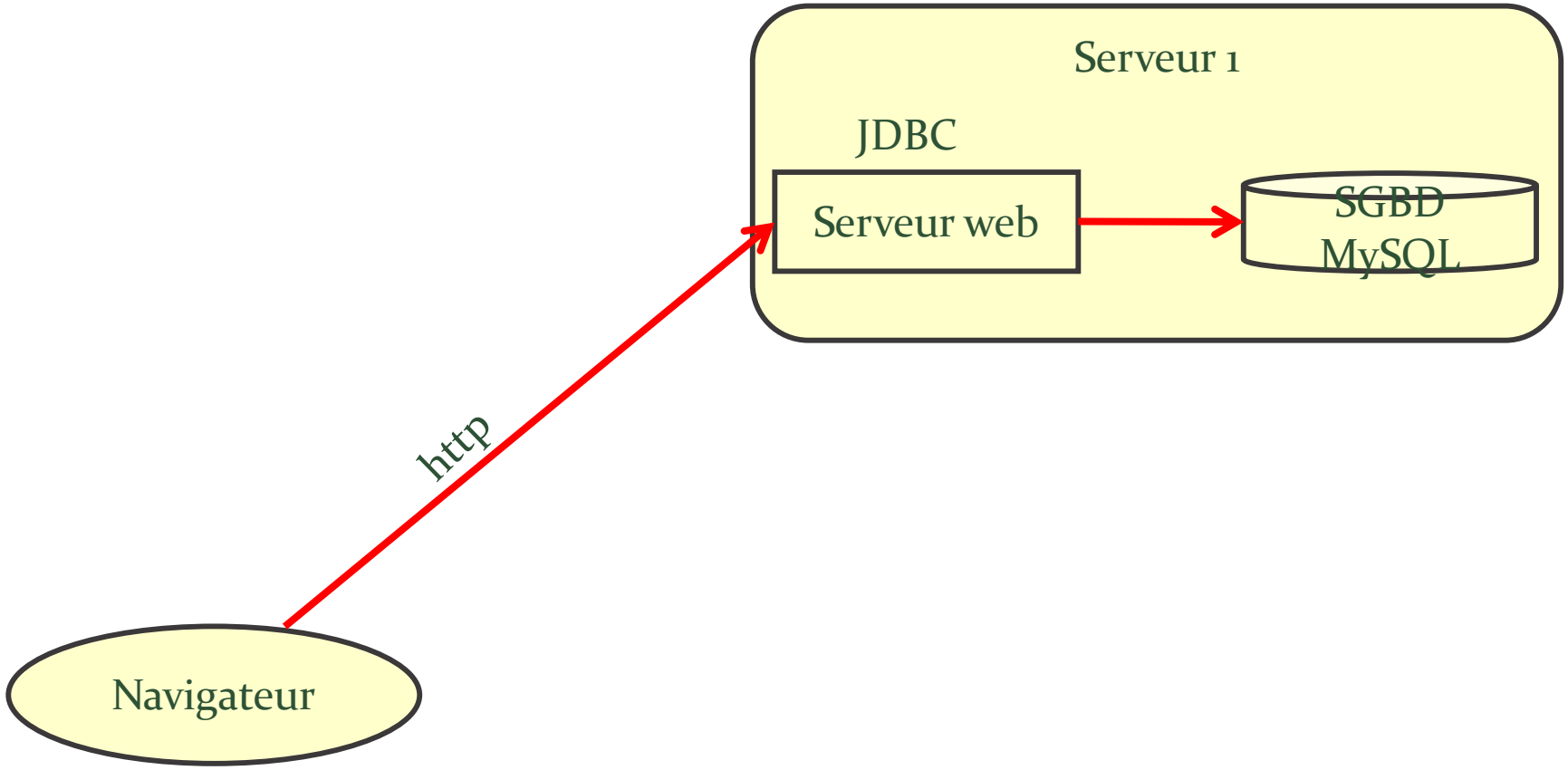
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Vue Physique et Vue Logique

- La vue logique d'une architecture logicielle définit les principaux ***composants*** d'une architecture sans se soucier des détails physiques (équipements, machines, …etc.)
- La vue physique s'intéresse au ***déploiement physique*** des différents services
- La vue physique est peu précise lors de la conception. Elle devient concrète lors de la phase de déploiement.

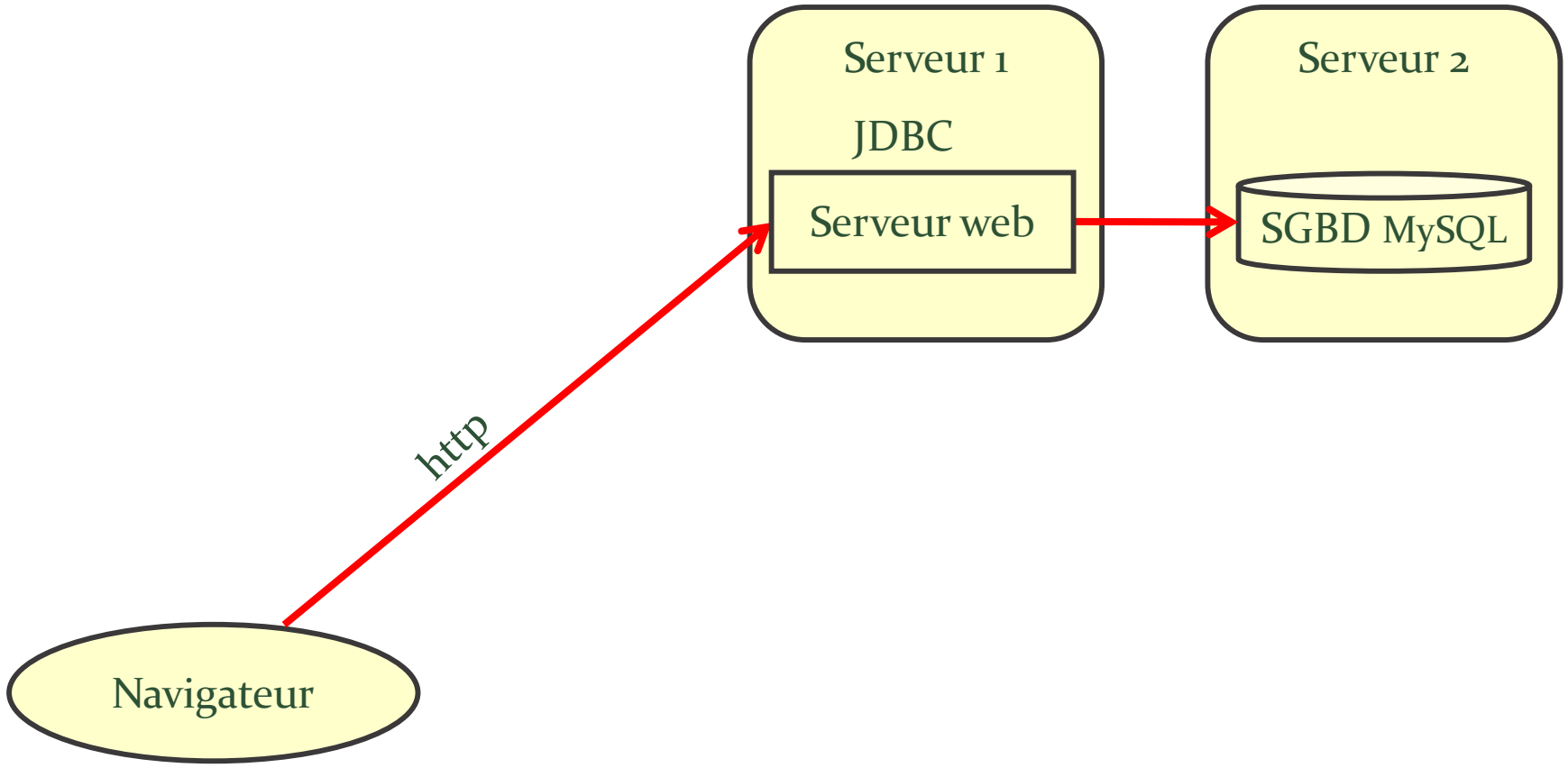
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Exemple 1 : services déployés dans le même serveur



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Exemple 2 : services déployés dans des serveurs différents



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Utilisation de l'architecture

- Donne un aperçu de haut niveau du système qui va faciliter la communication et la compréhension
- Aide à comprendre des systèmes existants
- Décompose le système en sous-systèmes et sous-modules ce qui réduit la complexité et facilite la distribution des tâches
- Facilite l'évolution du système en remplaçant uniquement le sous-système désiré

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

UML et les architectures logicielles

- Plusieurs formalismes peuvent décrire une architecture logicielle
- UML 2 est un bon moyen de représenter une architecture logicielle
- Le ***diagramme de composants*** peut servir à représenter la ***vue logique*** d'une architecture
- Le ***diagramme de déploiement*** peut servir à représenter la ***vue physique*** d'une architecture

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Introduction

SECTION 1 – DÉBAT (10 MNS)

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Diagramme de Composants

SECTION 2

Composant

- Un composant est une ***unité autonome*** dans un système
- Un composant définit un ***système*** ou un ***sous-système*** de n'importe quelle taille ou complexité
- Les diagrammes de composants permettent de modéliser les ***composants*** et leurs ***interactions***
- Les composants d'un système sont facilement ***réutilisés*** ou ***remplacés***

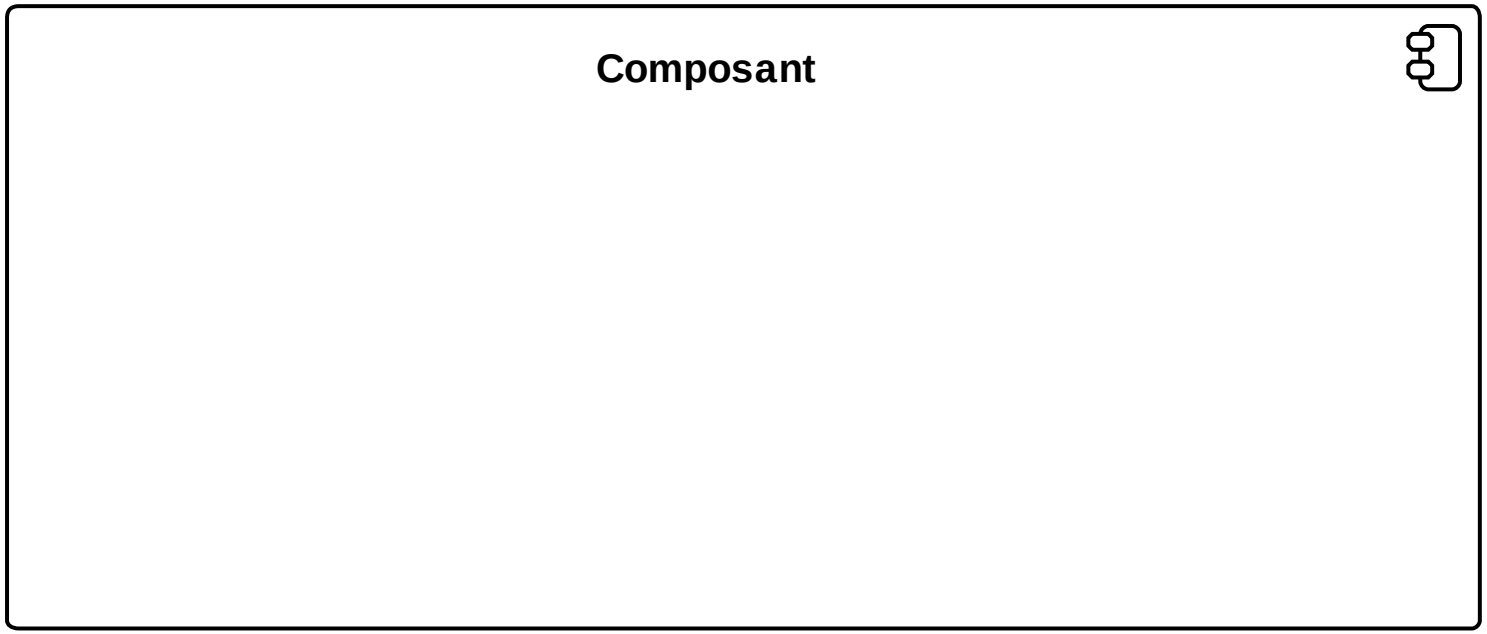
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Caractéristiques d'un composant

- Un composant est une ***unité modulaire*** avec des interfaces bien définies
- Les interfaces définissent comment le composant peut être ***appelé*** ou ***intégré***
- Le composant est ***remplaçable*** et ***autonome***
- L'implémentation du composant est ***cachée*** (encapsulée) aux entités externes

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Représentation UML



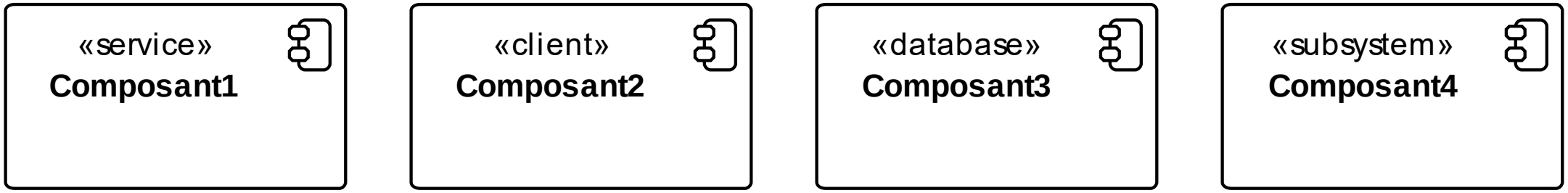
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Caractéristiques d'un composant

- Un composant a un nom ***unique*** dans son contexte
- Un composant peut être étendu par un ***stéréotype***
- Il existe des stéréotypes standard pour les composants comme « subsytem », « database » ou « executable »
- L'utilisateur peut ajouter ***ses propres stéréotypes*** à condition que ça soit consistant avec l'objectif du diagramme
- Dans le cadre d'architectures logicielle, ces stéréotypes peuvent être utilisés : « service », « client », … etc.

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Exemples de stéréotypes



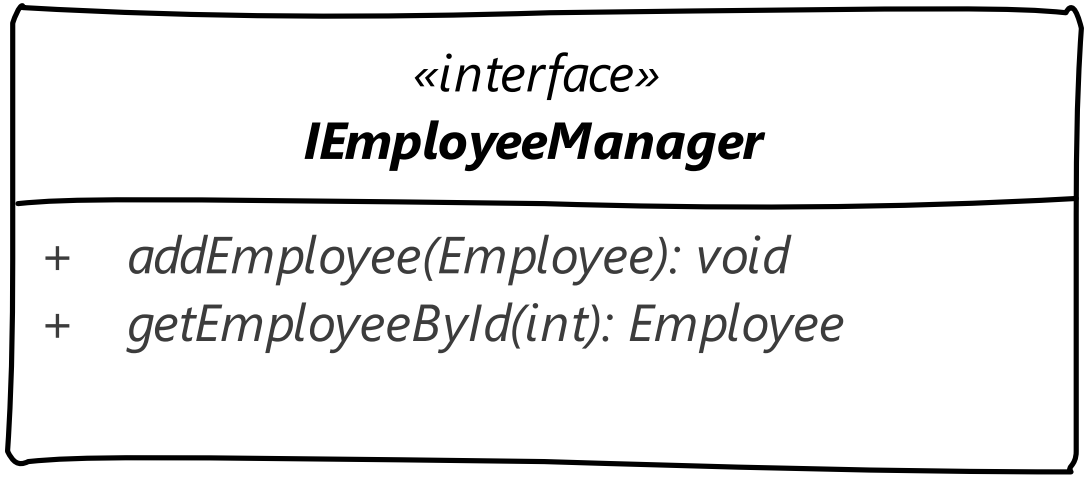
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Interfaces

- Un composant définit son comportement en terme d'***interfaces fournies*** et ***interfaces requises***
- Une interface est une ***collection d'opérations*** ayant un ***lien sémantique*** et qui ***n'ont pas d'implémentation***
- L'implémentation des interface se fait par une ou plusieurs ***classes*** implémentant le composant
- Les noms d'interfaces commencent par « I » (convention)
- Un ***contrat*** entre C1 et C2 est défini quand C1 fournit une interface I qui est requise par C2

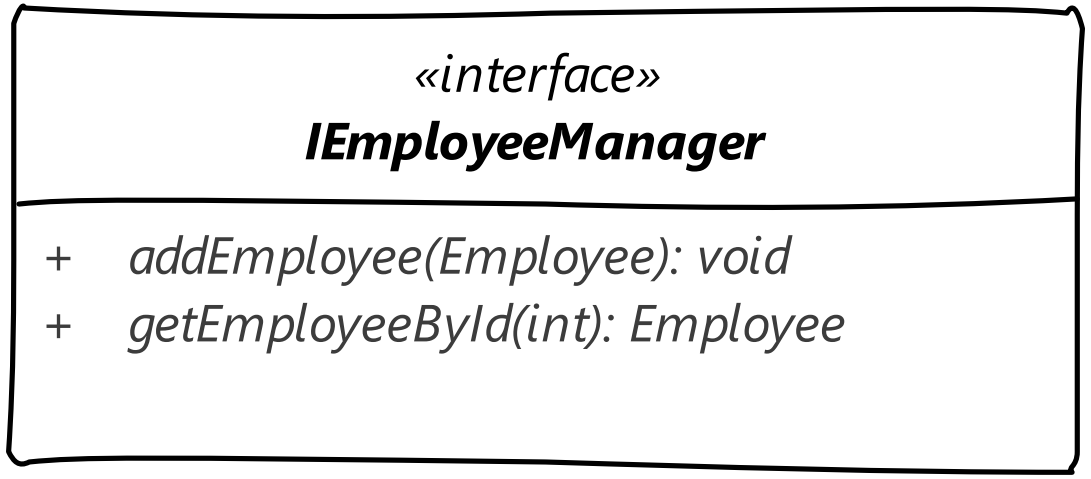
Interfaces - Exemple



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Interfaces - Membres

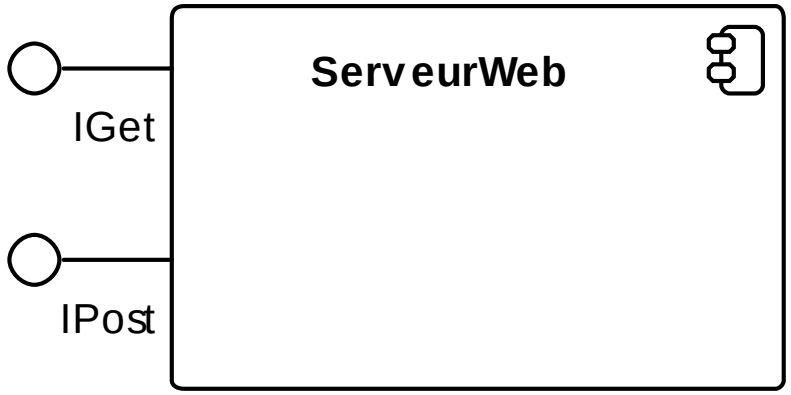
- Cours 6 – Introduction Aux Architectures de Logiciels



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Interfaces fournies

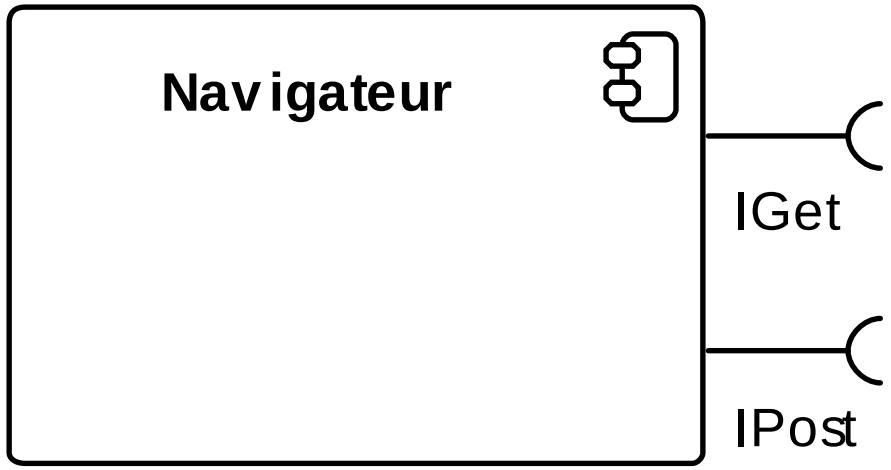
- Une interface fournie définie les fonctions qu'un composant ***pourrait faire***
- Exemple : un serveur web peut gérer les requêtes HTTP de type get ou post



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Interfaces requises

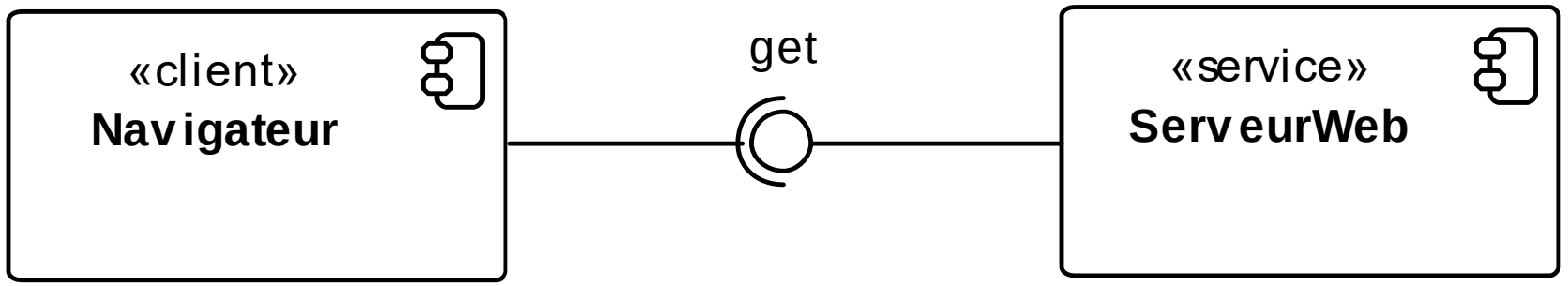
- Définit la (ou les interfaces) qu'un composant ***attend*** de son environnement



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Assemblage

- Un assemblage entre deux composants est lorsqu'une même interface est requise pour un composant et fournie par l'autre

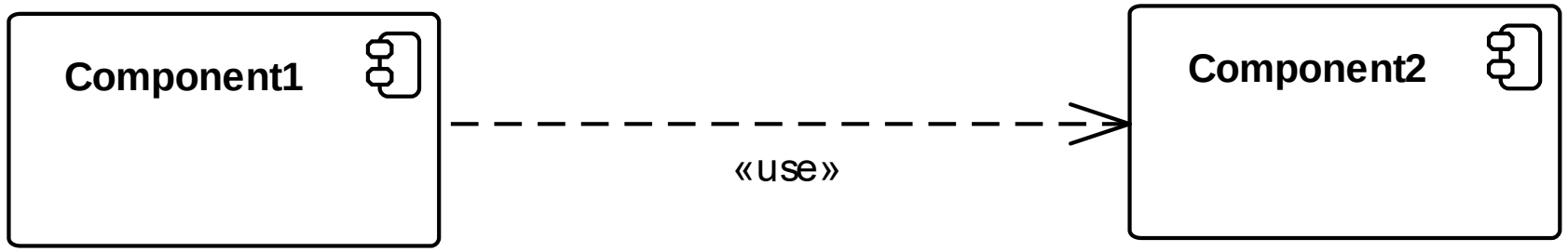


Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Utilisation

- Un composant C1 dépend d'un autre composant C2 lorsque C1 requiert C2 pour son implémentation (C1 appelle un des services de C2)
- En d'autres mots, l'exécution de C1 requiert la ***présence*** de C2

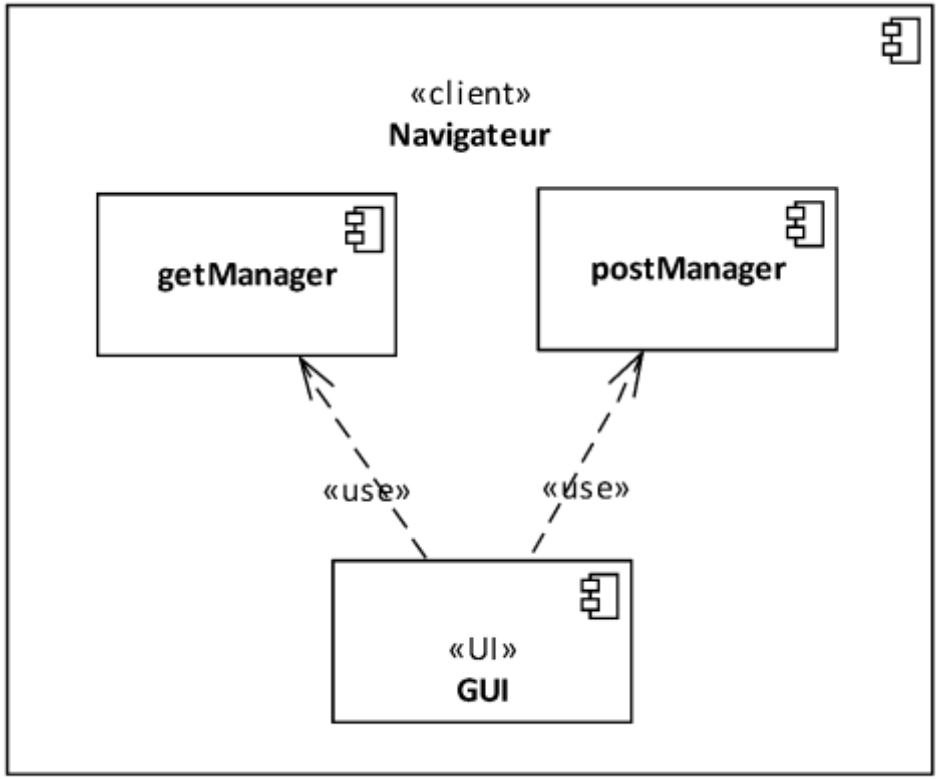


Composition

- Un composant peut être lui-même composé d'autres composants. On parle alors de composition.
- Par exemple, le navigateur est composé de getManager (gestionnaire des requêtes get), postManager (gestionnaire des requêtes POST) et GUI (interface)

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Composition



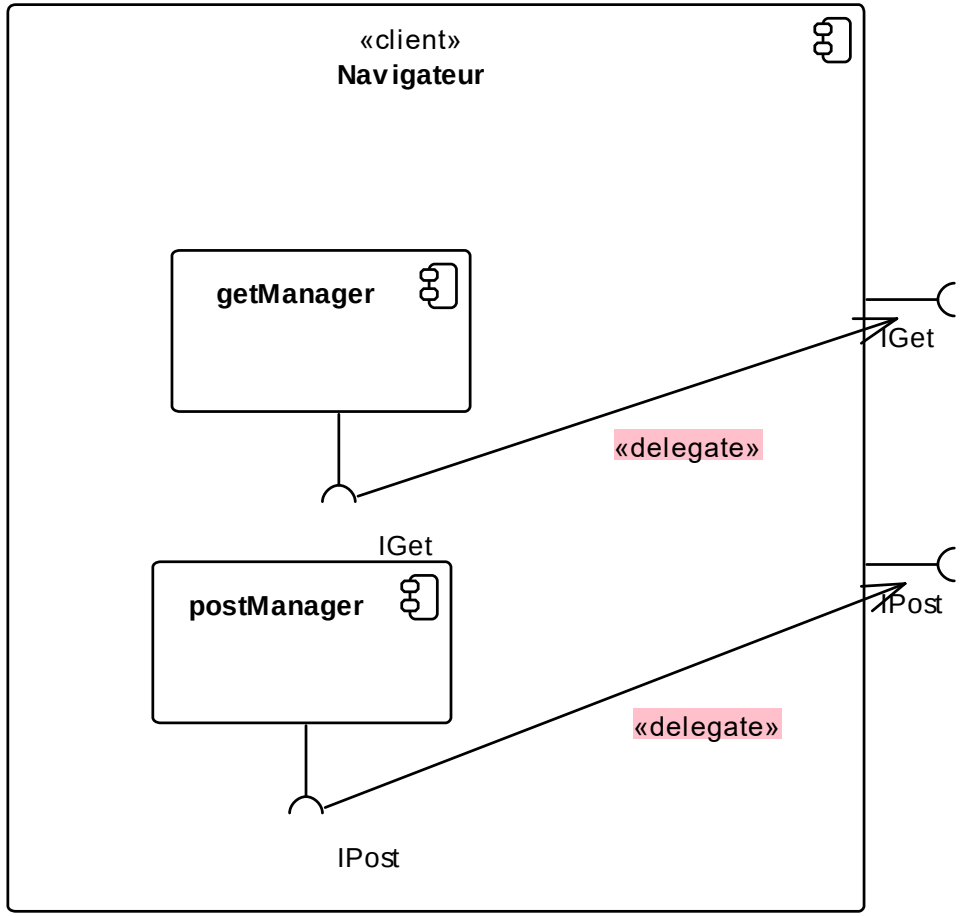
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Délégation

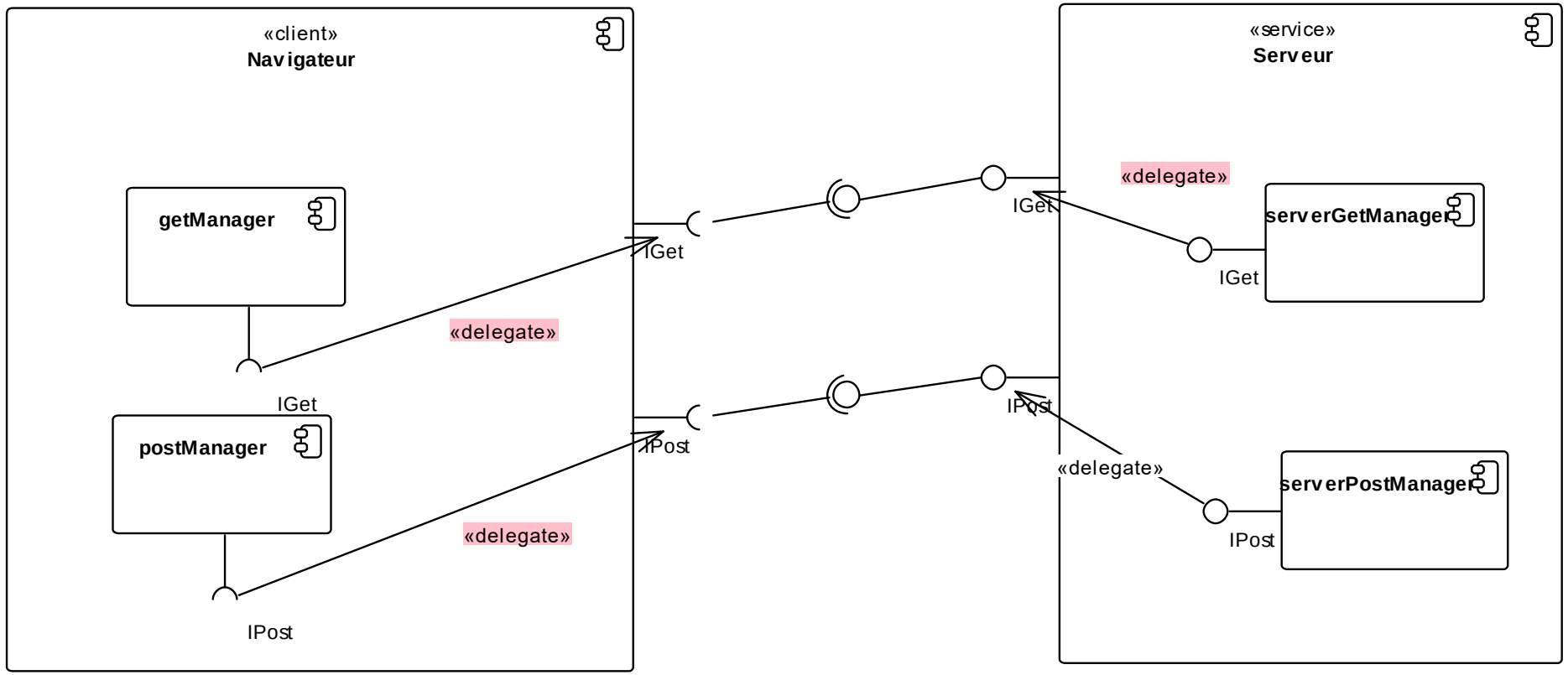
- Un composant peut avoir des sous-composants qui incluent des interfaces fournies ou des interfaces requises
- La délégation consiste à ***transférer*** les interfaces fournies / requises du composant interne vers le composant externe

Délégation



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

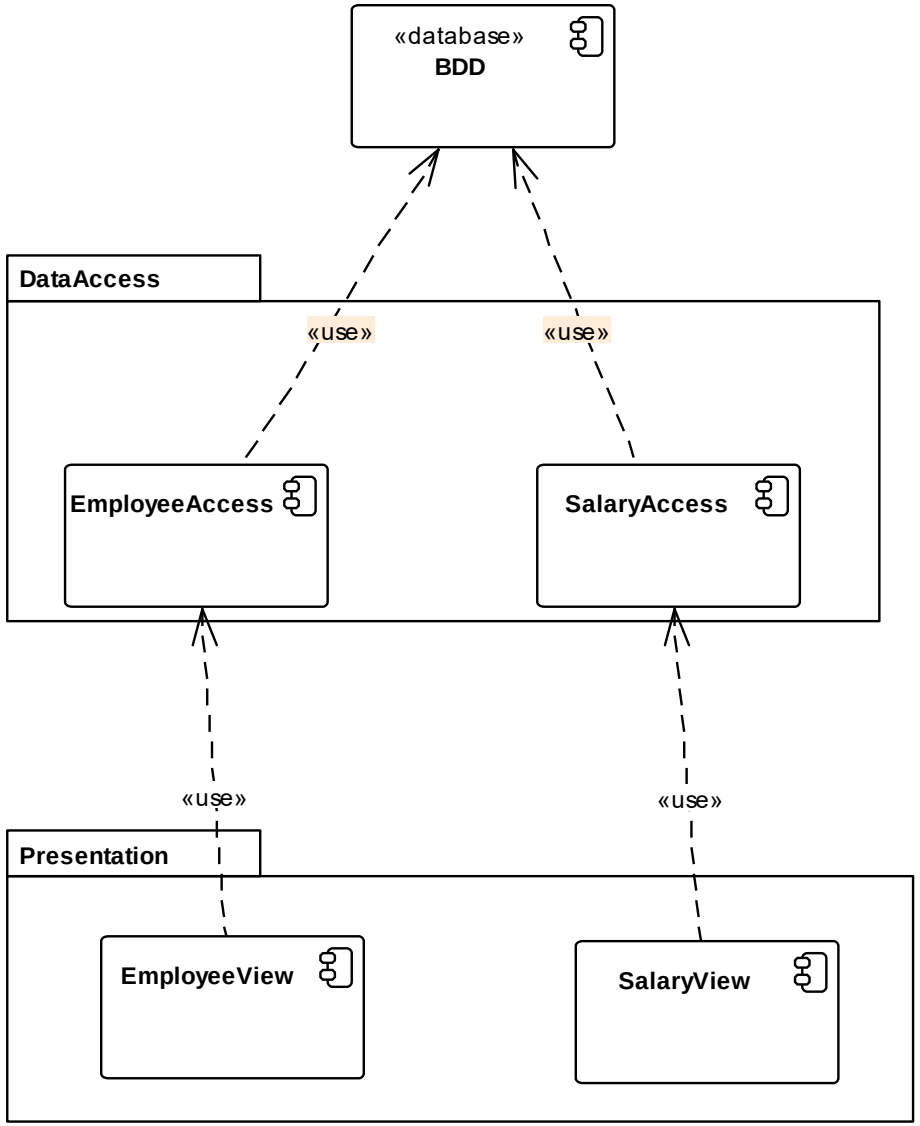
Délégation – Exemple 2



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Paquets

- Les paquets peuvent être aussi utilisés dans les diagrammes de composants pour organiser les composants



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Composants et classes

- Les classes sont « packagées » dans des bibliothèques
- Par exemple une bibliothèque est un fichier jar (java) ou une assembly dll (.NET)
- Le diagramme de composants définit aussi le packaging des classes du système
- Le connecteur liant les classes aux composants est le connecteur « realize »

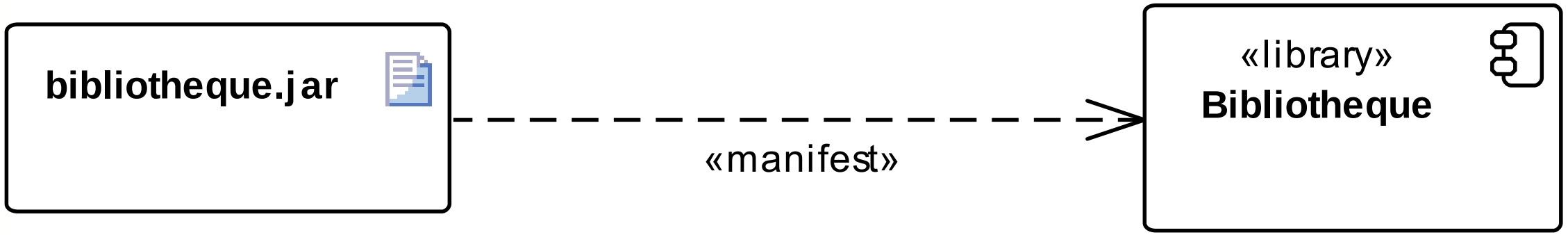
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Artifacts

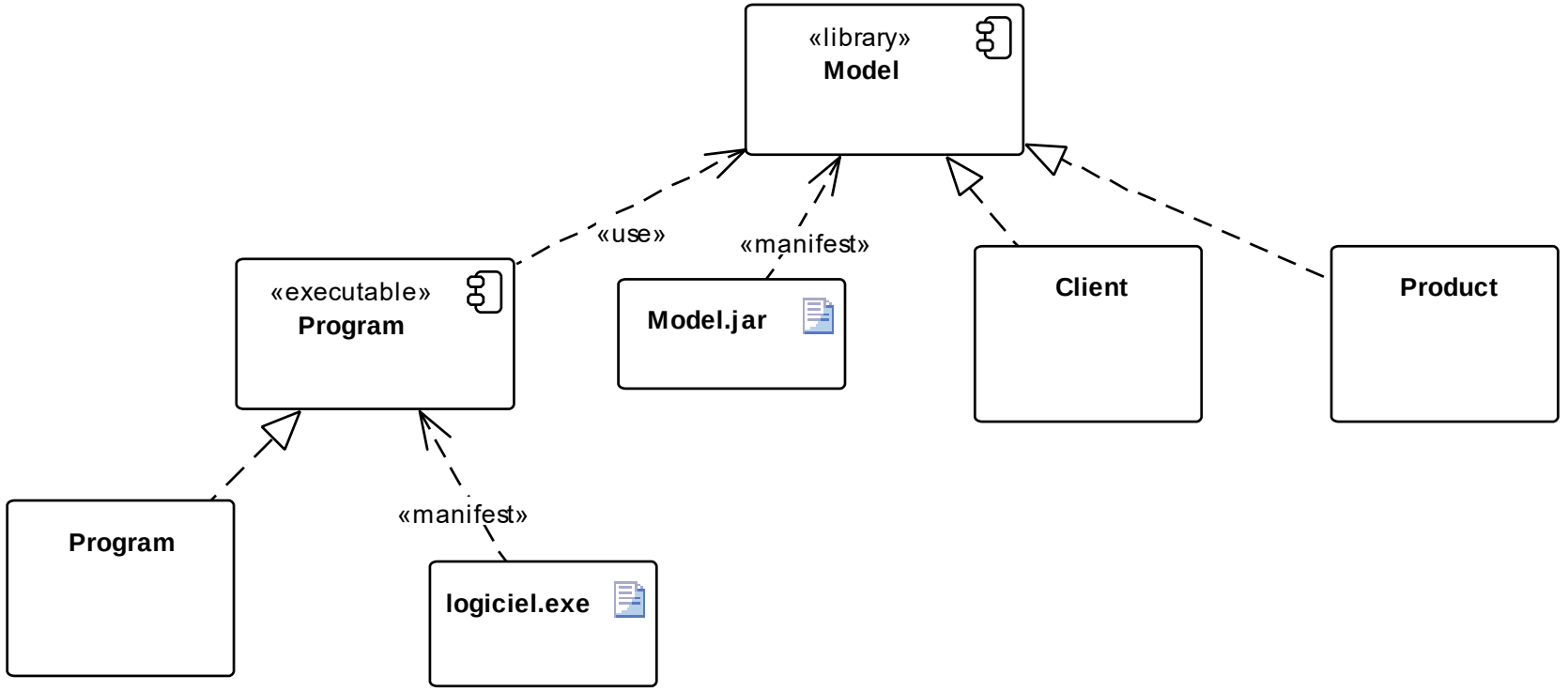
- Un artifact est une ***pièce physique*** utilisée par le système
- Un artifact peut être un document, un fichier, un code source ou n'importe quel élément ayant une relation avec le système
- La dépendance avec le stéréotype « manifest » indique qu'un artifact est la représentation physique d'un composant.
- Par exemple, un fichier jar est une représentation physique d'une classe java

Artifacts – Suite



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Composants et classes



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Diagramme de Composants

SECTION 2 – DÉBAT (10 MNS)

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Styles Architecturaux

SECTION 3

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Introduction

- Un style architectural est un ***modèle*** définissant comment sera le système
- Comme les systèmes ont des points communs, ces systèmes auront des architectures qui se ressemblent. Le regroupement de ces architectures est appelé « ***style architectural*** »
- Un style architectural définit quels sont les ***composants***, les ***connecteurs*** et les ***contraintes*** définissant l'architecture d'un système

Bénéfices d'un style architectural

- Un style architectural aide à avoir un ***aperçu*** du système avant son développement
- Le styles sont ***indépendants*** des technologies.
- ***Plusieurs technologies*** peuvent réaliser un certain style. Par exemple des serveur sous Linux et des clients sous Windows.
- Facilite la ***réutilisation***
- Un système peut s'appuyer sur ***plusieurs styles***

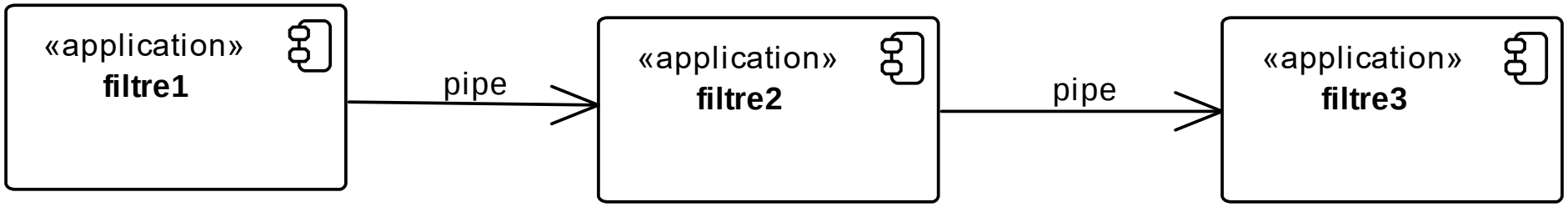
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Pipe / Filtre - Définition

- Permet à l'information d'être traitée par ***plusieurs composants*** d'une manière ***séquentielle***
- La ***configuration*** détermine l'ordre des traitements
- Le filtre est un composant qui ***traite*** l'information
- La pipe est un canal par lequel ***transite*** l'information

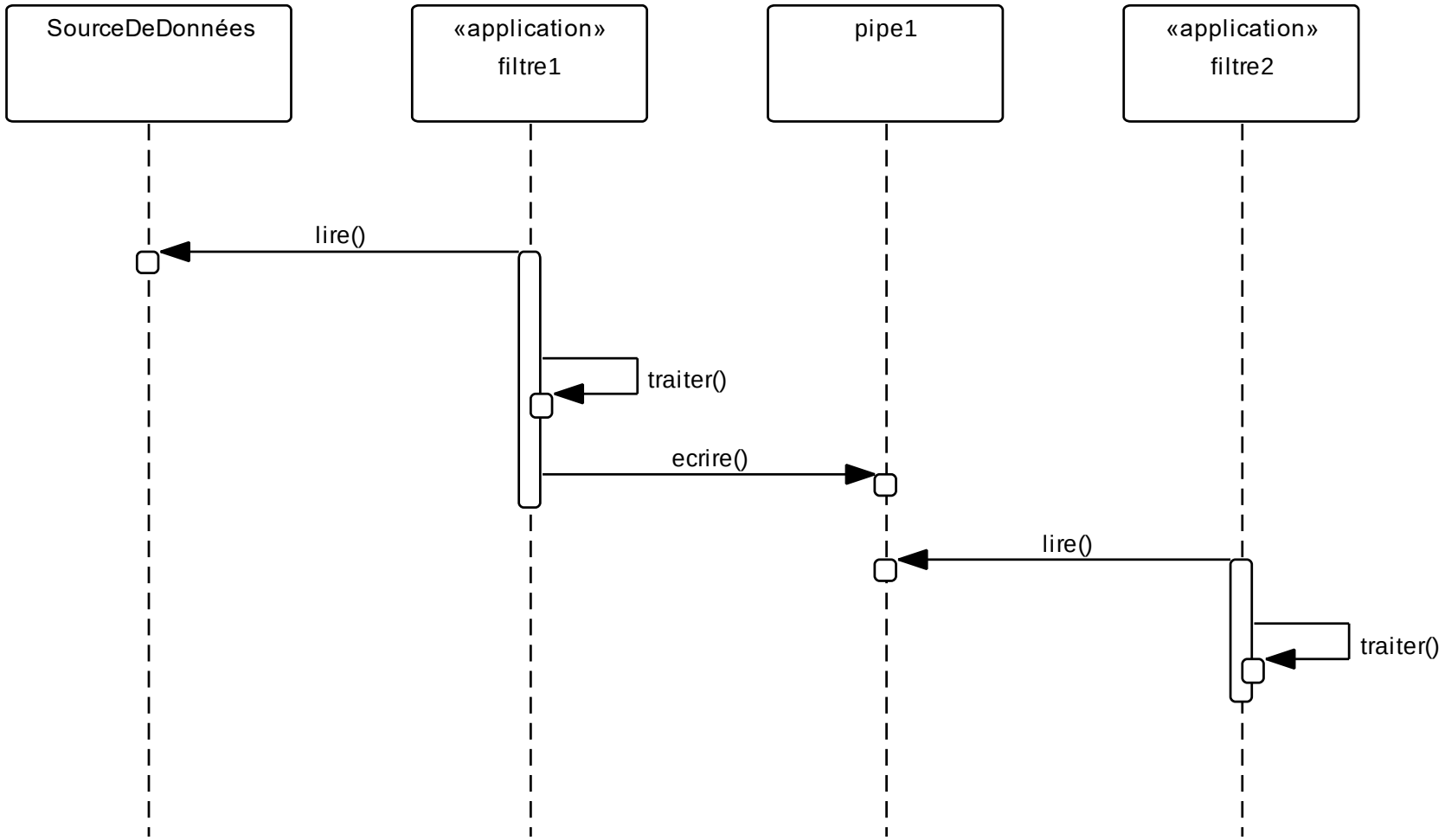
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Pipe / Filtre - Définition



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Pipe / Filtre - Fonctionnement



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Pipe / Filtre - Suite

Exemples

- Unix Shell
- Windows Powershell
- Unix Shell : cat fichier.txt | grep logiciel | wc : compte le nombre de mots logiciel dans le fichier fichier.txt

Avantages

- Forte décomposition du systèmes
- Filtres faciles à réutiliser
- Facilite la maintenance
- La dépendance entre les filtres est faible

Inconvénients

- Ne convient pas aux applications à haute interactivité
- Les performances dépendent des pipes

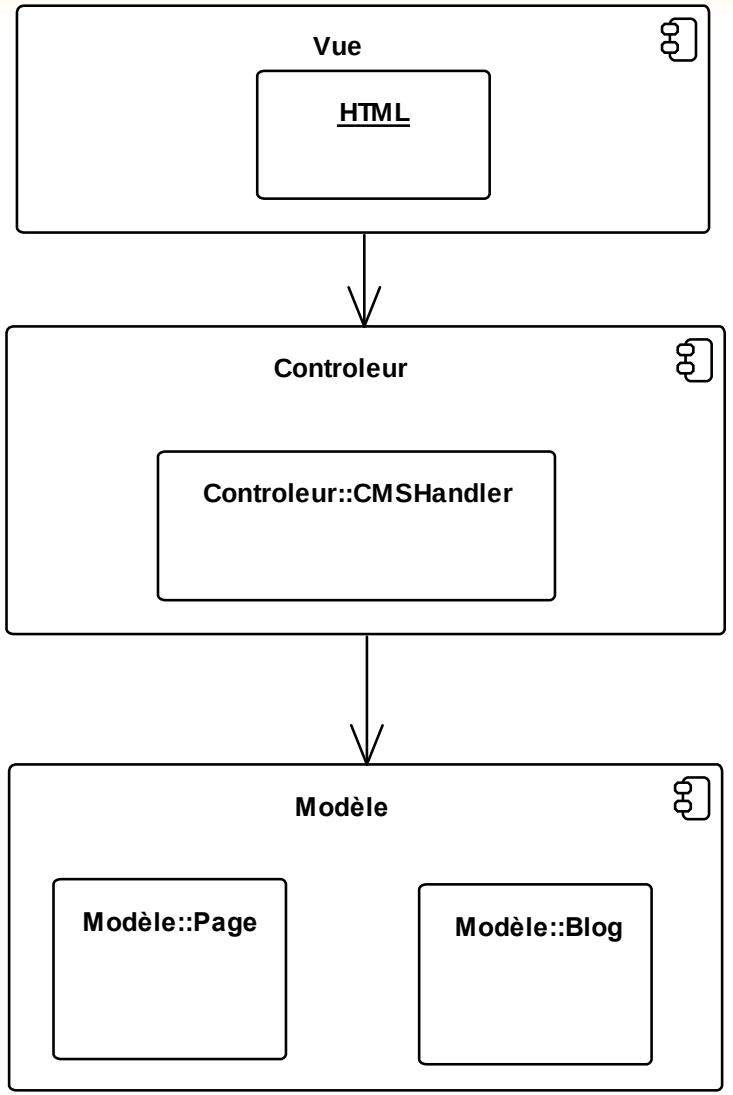
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

MVC - Définition

- MVC = ***Model View Controller***
- Déclinaison client appelée ***MVVM***
- Le modèle représente les ***entités du système***
- Le contrôleur implémente la ***logique métier*** et la ***logique des interactions***
- La vue représente ***l'interface utilisateur***

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

MVC - Exemple



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

MVC – Suite

Exemples

- .NET : ASP.NET MVC, MonoRail
- Java : JavaServer Faces (JSF), Struts
- Ruby On Rails
- Python : Zope
- Javascript: AngularJS, Knockout.JS

Avantages

- Modèle de conception largement apprécié de la communauté de développeurs
- Séparation de la logique de l'interface
- Testabilité accrue (les tests unitaires supportent le modèle et le contrôleur)

Inconvénients

- Assez complexe
- Plus d'efforts de développement car chaque tâche concerne les trois couches

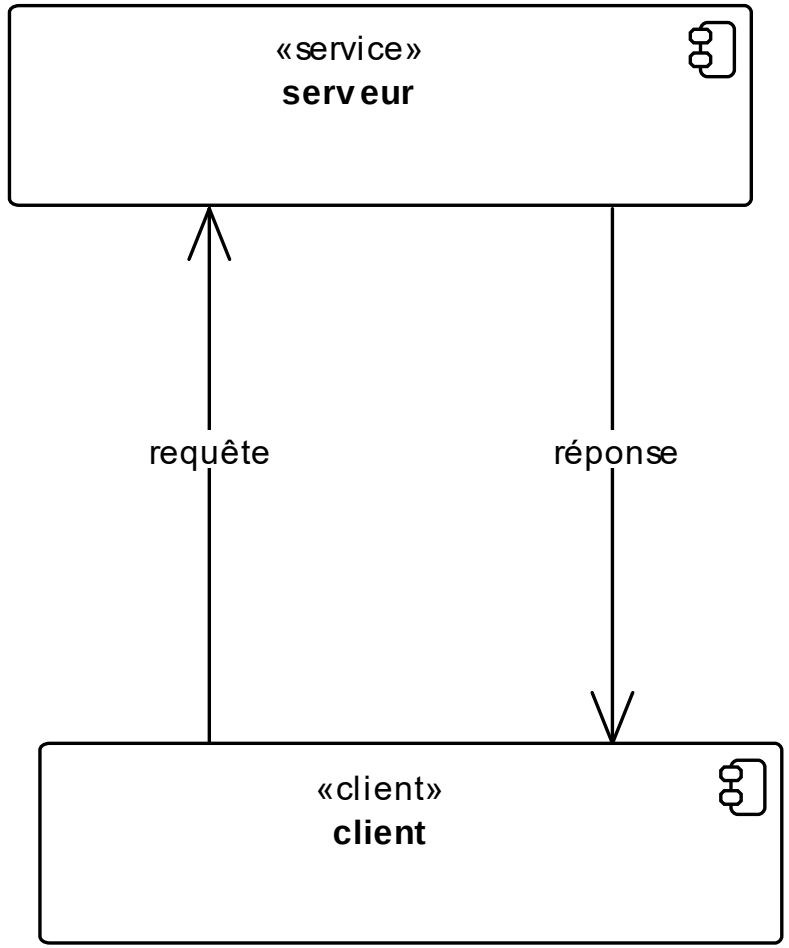
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Client-Serveur - Définition

- Le système est composé de deux composants principaux se trouvant généralement dans des machines séparées : le ***client*** et le ***serveur***
- Le client envoie des ***requêtes*** au serveur
- Le serveur réagit au requêtes en renvoyant des ***réponses***
- L'interface utilisateur se trouve au niveau du client

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Client-Serveur – Fonctionnement



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Client – Serveur , Suite

Exemples

- Serveur web (IIS / Apache), Client web (FireFox / Chrome / Internet Explorer)
- Serveur FTP (ftpd) / Client FTP (FileZilla)

Avantages

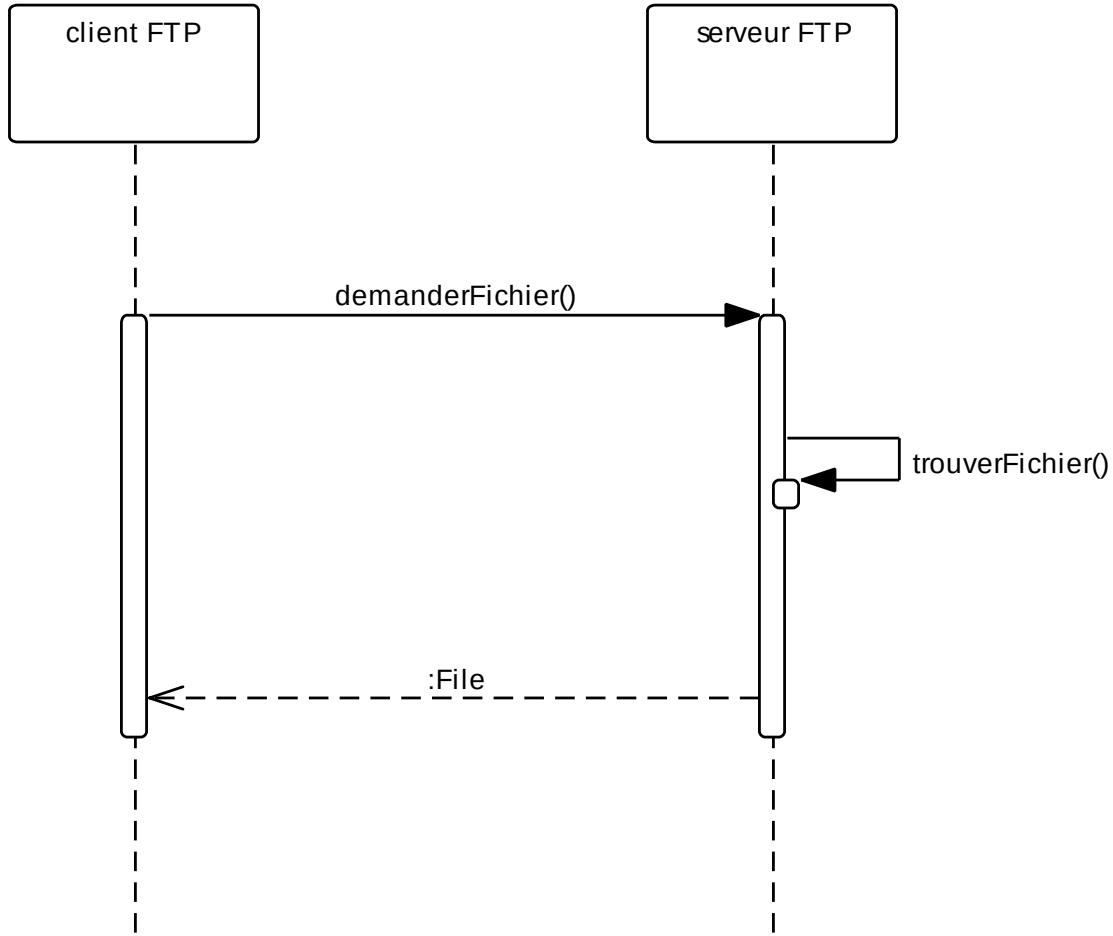
- Séparation des tâches
- Simple à utiliser

Inconvénients

- Ne convient plus aux nouvelles générations d'appliacations

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Client-Serveur - Exemples



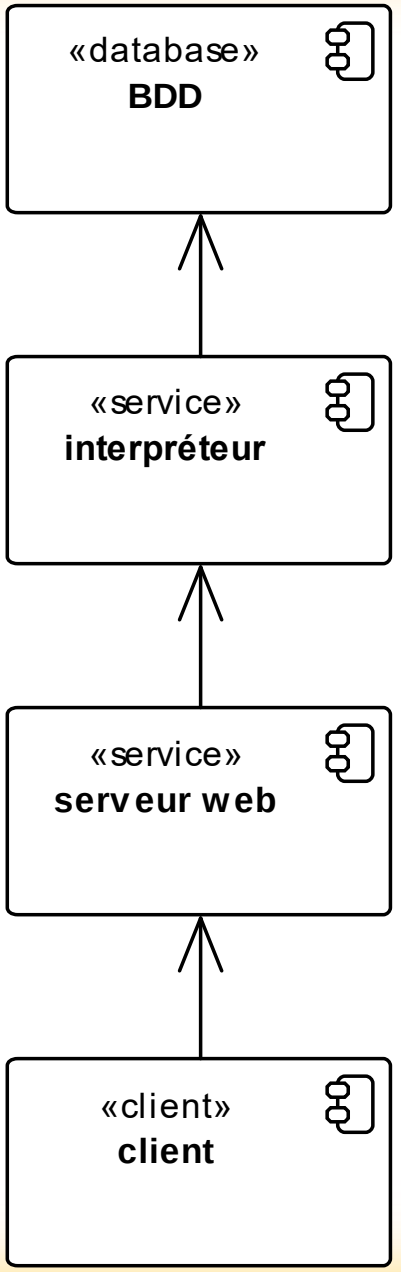
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Architecture N-Tiers - Définition

- Fragmente le système en plusieurs ***niveaux***
- Le niveau présentation, le niveau logique ou le niveau données sont des exemples de niveaux
- Chaque niveau dépend uniquement du niveau qui est au dessus
- Exemple : applications web modernes

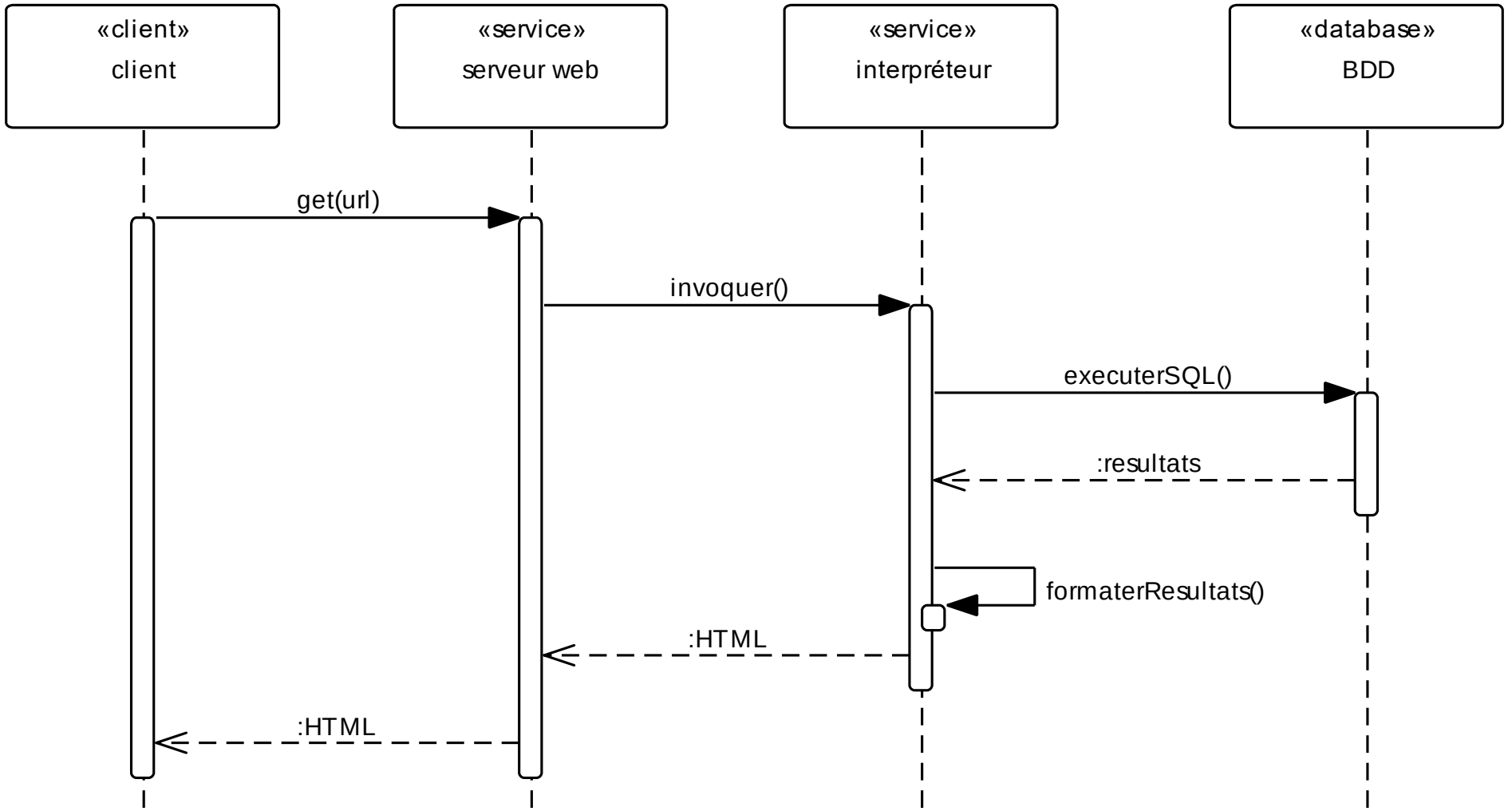
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Architecture N-Tiers - Exemple



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Architecture N-Tiers - Exemple



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

N-Tiers – Avantages et inconvénients

Avantages

- Séparation poussée des tâches
- Haute flexibilité

Inconvénients

- Nécessite des ressources matérielles importantes

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

SOA - Définition

- SOA ou (***Service-Oriented Architecture***) est une évolution du modèle client-serveur
- SOA est basée sur des services ***faiblement couplés***, ***indépendants*** des protocoles, basés sur les ***standards*** et distribués
- Chaque ressource disponible sur le réseau est utilisée comme un service

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Caractéristiques des services

- Les service sont ***autonomes***
- Les services sont ***composables*** : créer un service à partir d'autres services
- Les services sont ***réutilisables***
- Les services permettent leur ***découverte***

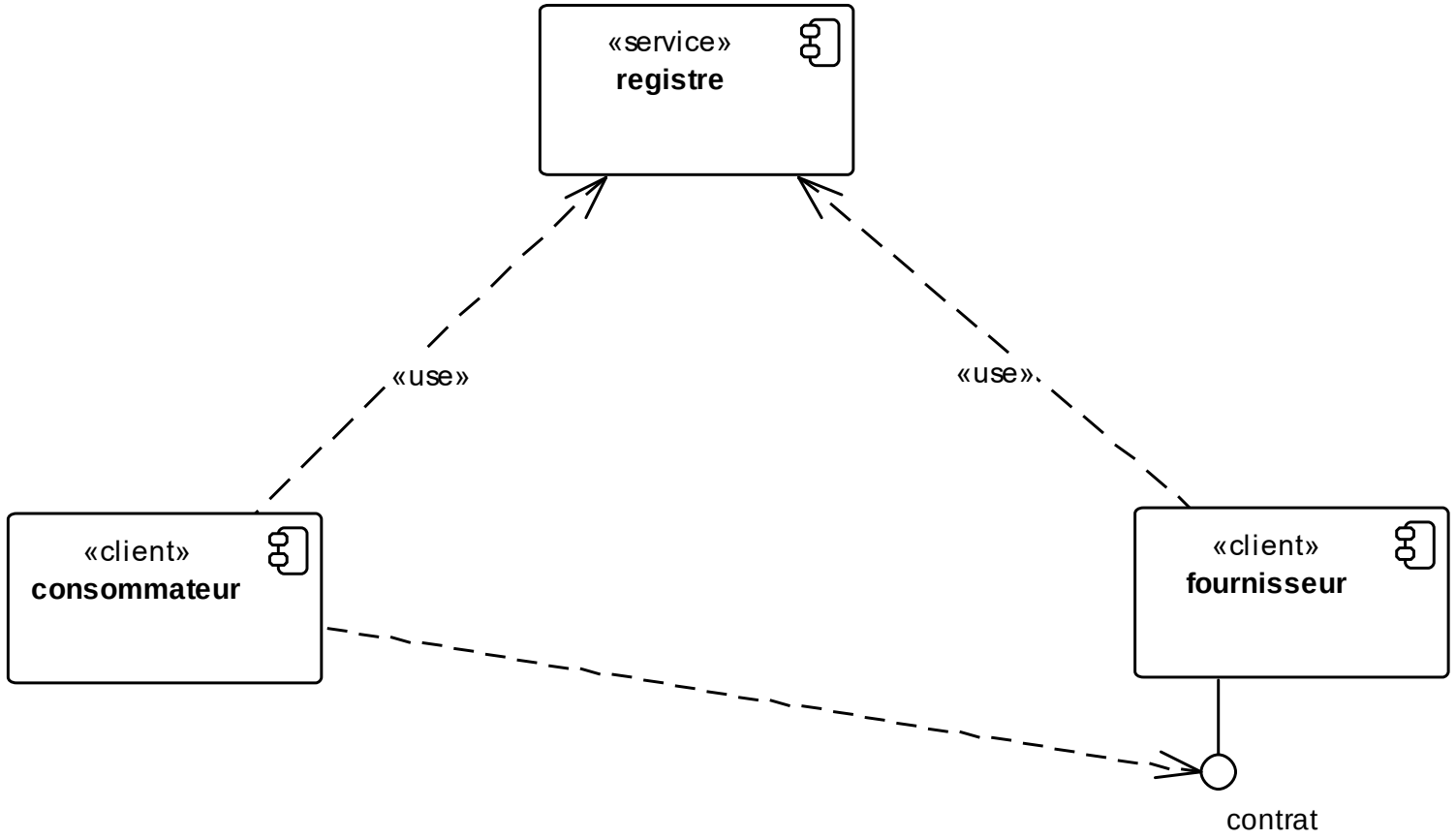
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Composants de SOA

- Une architecture SOA est basé sur un ***consommateur de service***, un ***fournisseur de service*** et un ***registre de services*** (Service Broker)
- Le consommateur ***utilise*** le service
- Le fournisseur ***assure*** le service
- Le registre fait ***le lien*** entre le fournisseur et le consommateur

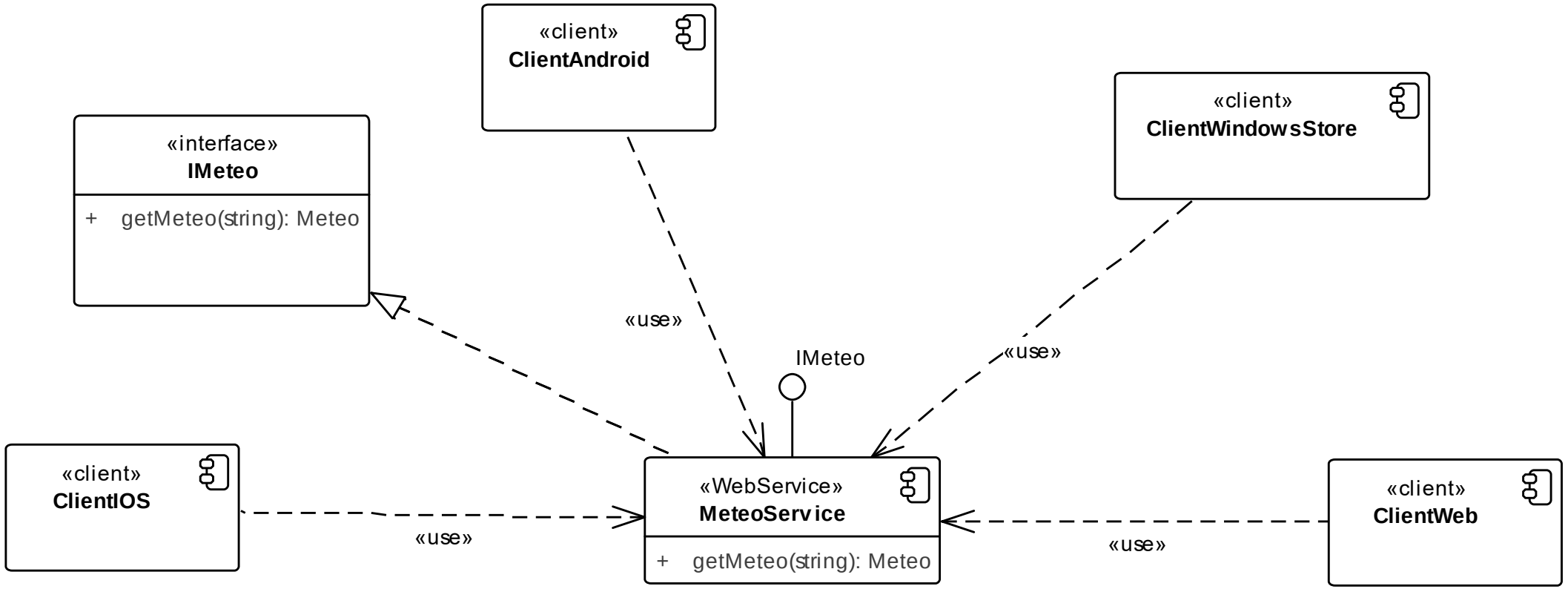
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Composants de SOA



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

SOA - Exemple



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

SOA – Technologies

- Deux tendances permettent d'implémenter SOA : ***SOAP*** / ***WSDL*** / ***UDDI*** ou ***REST***
- SOAP est un protocole basé sur ***XML*** permettant de ***véhiculer*** des données via ***HTTP*** en utilisant XML
- WSDL permet de ***décrire*** un service web
- UDDI permet de ***découvrir*** un service web
- SOAP / WSDL / SOAP sont utilisées conjointement
- REST est un protocole basée sur ***HTTP uniquement***
- Dans REST, HTTP est le protocole de transmission et aussi le service web en même temps

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

SOA

Exemples

- Google Search Engine (web + application android + application iOs)
- Youtube (web + application android + application blackberry + application iOs)
- Facebook (web + applications mobiles)

Avantages

- Basés sur les standards (JSON / HTTP)
- Indépendance et facilité de découverte
- Permettent à l'utilisation des applications depuis n'importe quel équipement (PC, mobile, etc…)

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Cloud Computing / PaaS

- Le Cloud Computing est une technologie basée sur ***internet*** qui permet de fournir des ressources d'une manière ***évolutive*** sur internet
- Le Cloud Computing décharge le client de l'infrastructure IT puisqu'elle fournit le matériel et l'infrastructure
- Le Cloud Computing est la base du ***SaaS*** (Software As A Service)
- Le Cloud Computing est la base du PaaS (Platform as a Service) qui permet de créer et de déployer des applications sur le cloud
- Avec le PaaS, les utilisateurs ne se soucient plus de l'élasticité et de la consommation des apps
- Avec le SaaS, les utilisateurs ne se soucient plus de l'évolution et de la maintenance des logiciels

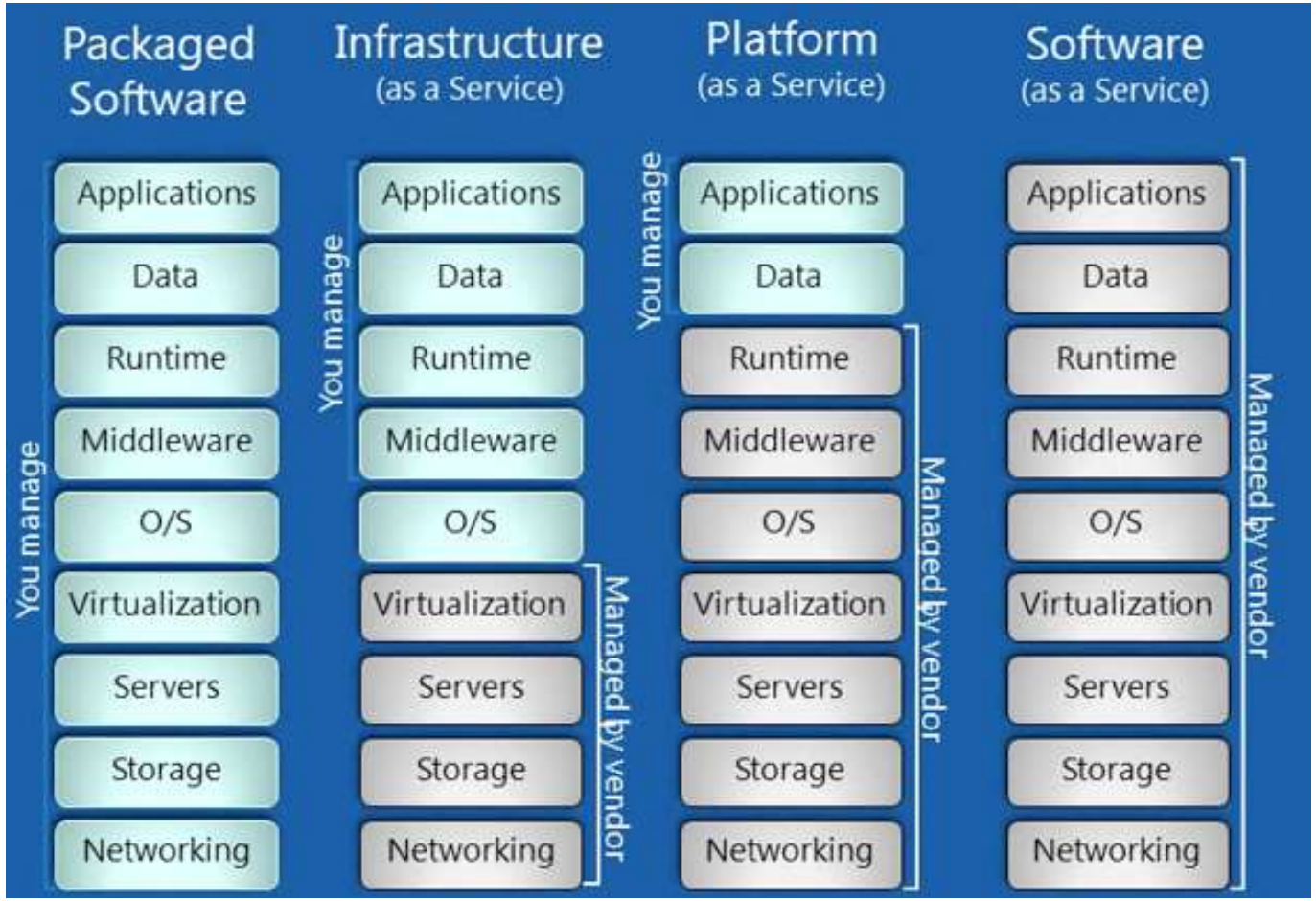
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Cloud Computing

- Le Cloud Computing permet aux entreprises une ***réduction des coûts*** car le client ne paye que le stockage et l'utilisation des processeurs
- Une structure basée sur le cloud est ***théoriquement infaillible*** car lorsque les serveurs actuels ne peuvent plus répondre à la demande, un nouveau serveur virtuel est automatiquement créé
- Le cloud est dit privé lorsqu'une entreprise décide de mettre en œuvre le cloud dans sa propre infrastructure IT

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Cloud Computing



Source : http://www.e-liance.fr/saas-paas-iaas-cloud-computing-quelles-differences/

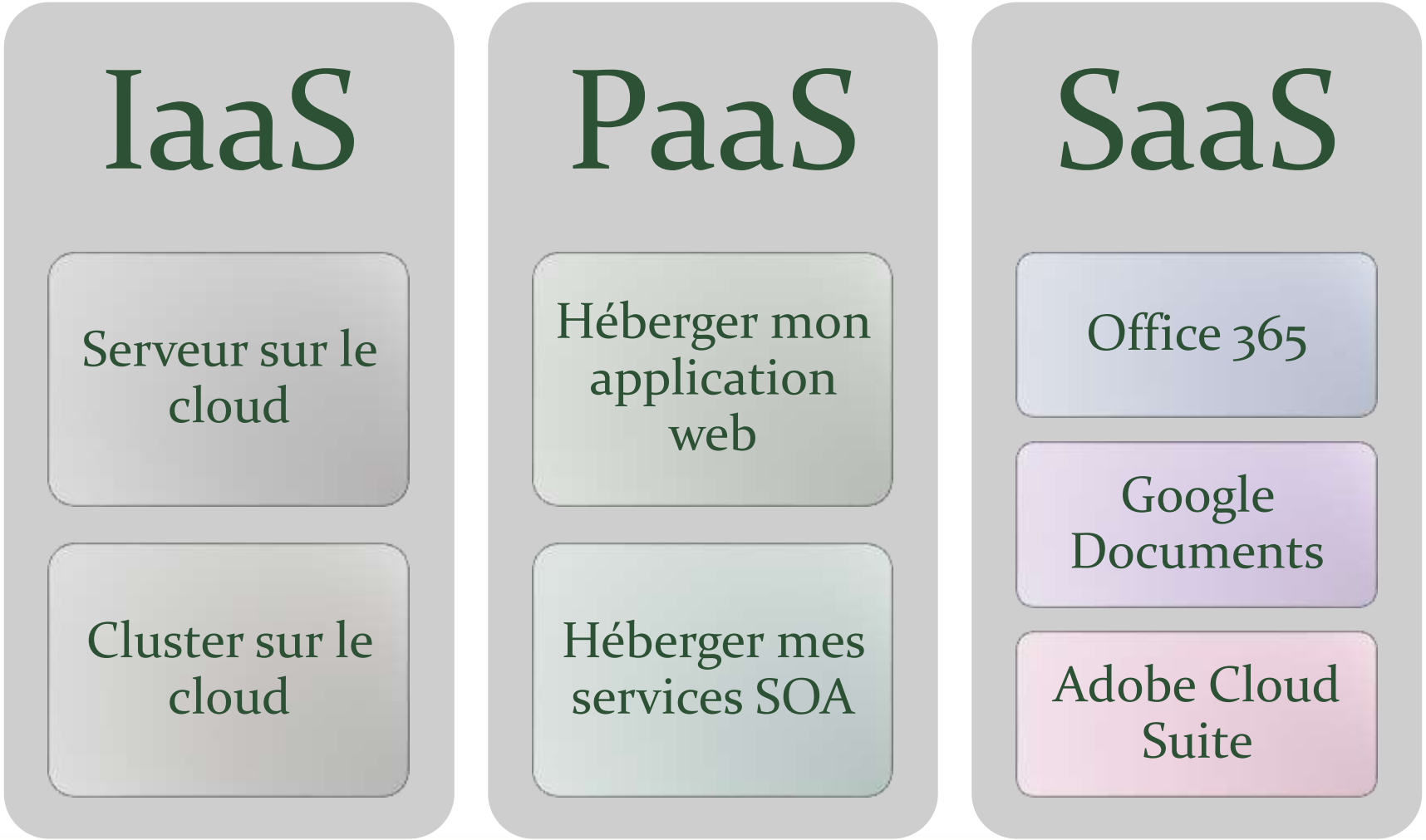
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Cloud Computing – Infrastructures PaaS

- Google AppEngine
- Amazon
- Windows Azure

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Cloud – Exemple



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Avantages & Inconvénients

Avantages

- Idéal lorsqu'une entreprise ne veut pas (ou ne peut pas) se charger de l'infrastructure IT
- Peut répondre à n'importe quelle charge
- Evolution instantanée des SaaS (pas de besoin de redéploiement)

Inconvénients

- Grosses problématiques de sécurité : le client ne sait pas où sont ses données et si elles sont vraiment sécurisées
- Souveraineté des données

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Styles Architecturaux

SECTION 3, DÉBAT 05 MNS

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Diagramme de Déploiement

SECTION4

Introduction

- Le diagramme de composants s'intéresse à l'architecture d'un point de vue ***logique*** tandis que le diagramme de déploiement s'y intéresse d'un point de vue ***physique***
- Le diagramme de déploiement s'intéresse aux relations entre les relations entre les composants et les ***équipements***
- Les équipements hébergeant des unités logicielles sont appelés ***nœuds*** (nodes)

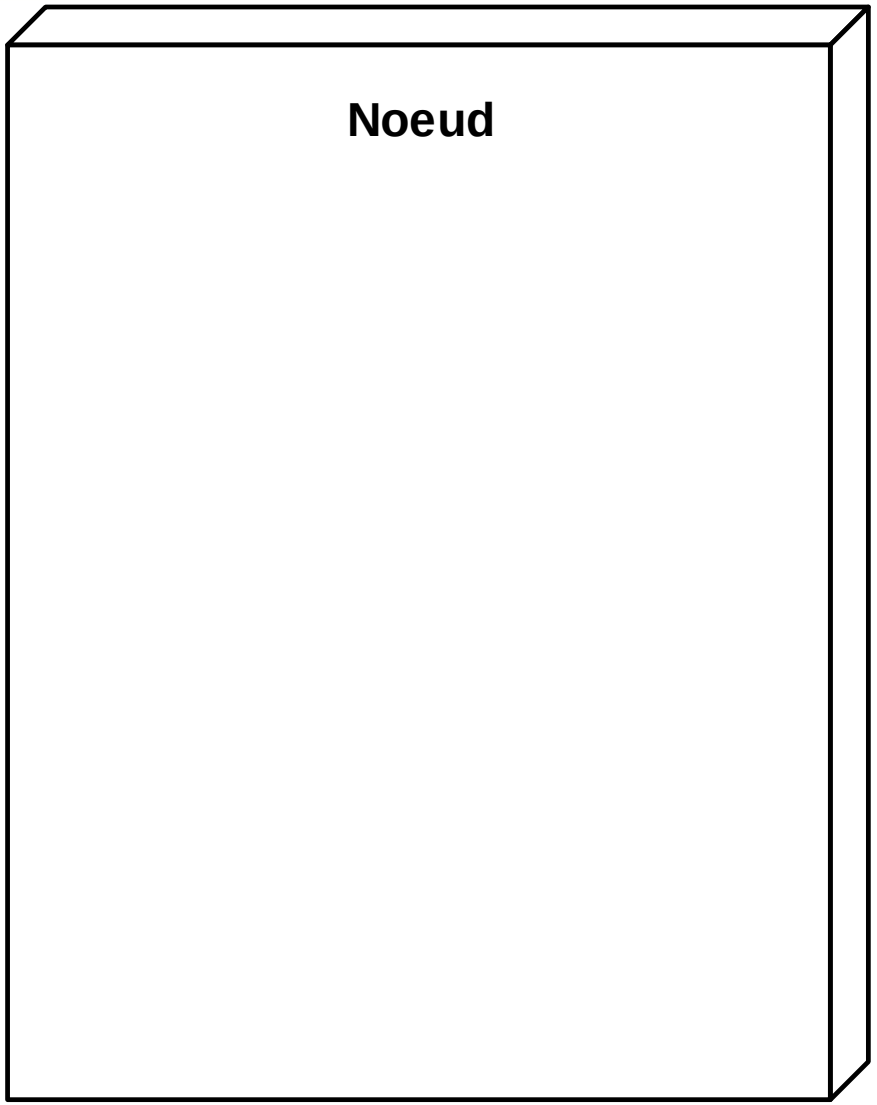
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Introduction

- Le diagramme de déploiement est composé de ***nœuds*** et de connecteurs
- Un nœud représente un ***équipement*** dans le système
- Un ***connecteur*** représente une communication entre les noeuds

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Noeud



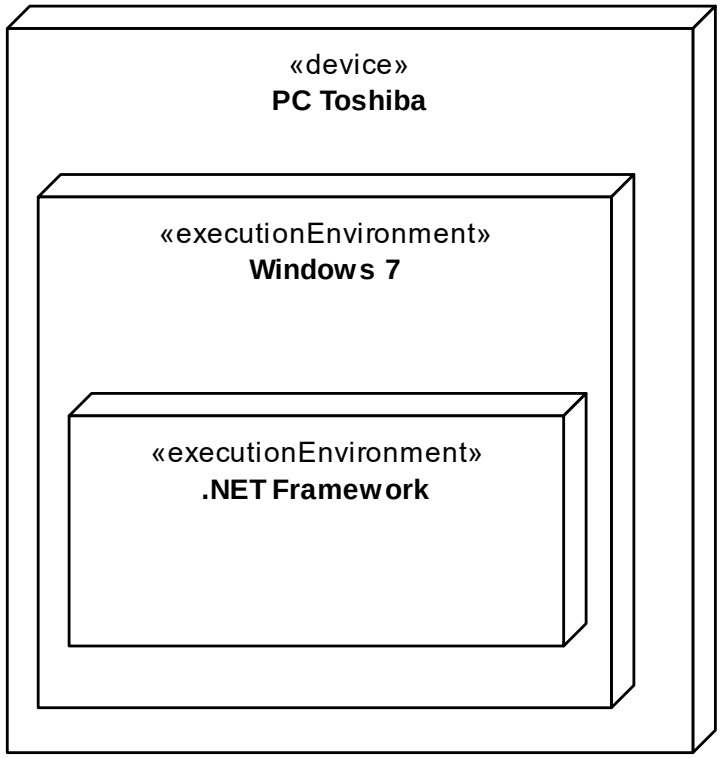
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

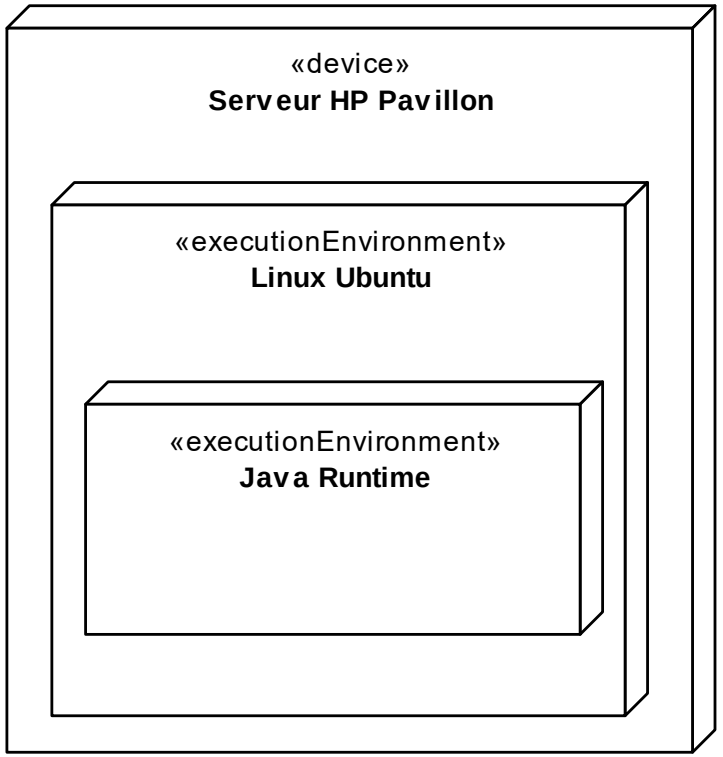
Nœuds

- Dans UML, un nœud peut aussi avoir un ***stéréotype*** pour préciser la nature du nœud
- «cdrom», «cd-rom», «computer», «disk array», «pc», «pc client», «pc server», «secure», «server», «storage», «unix server», «user pc» sont des exemples de stéréotypes
- Deux stéréotypes très importants : « ***device*** » et « ***execution environment*** »
- Le stéréotype « device » représente un équipement ***hardware***
- Le stéréotype « execution environment » détermine un ***environnement où les processus s'exécutent*** : par exemple un framework ou un système d'exploitation
- Les nouds peuvent être ***imbriqué***

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Nœud - Exemple





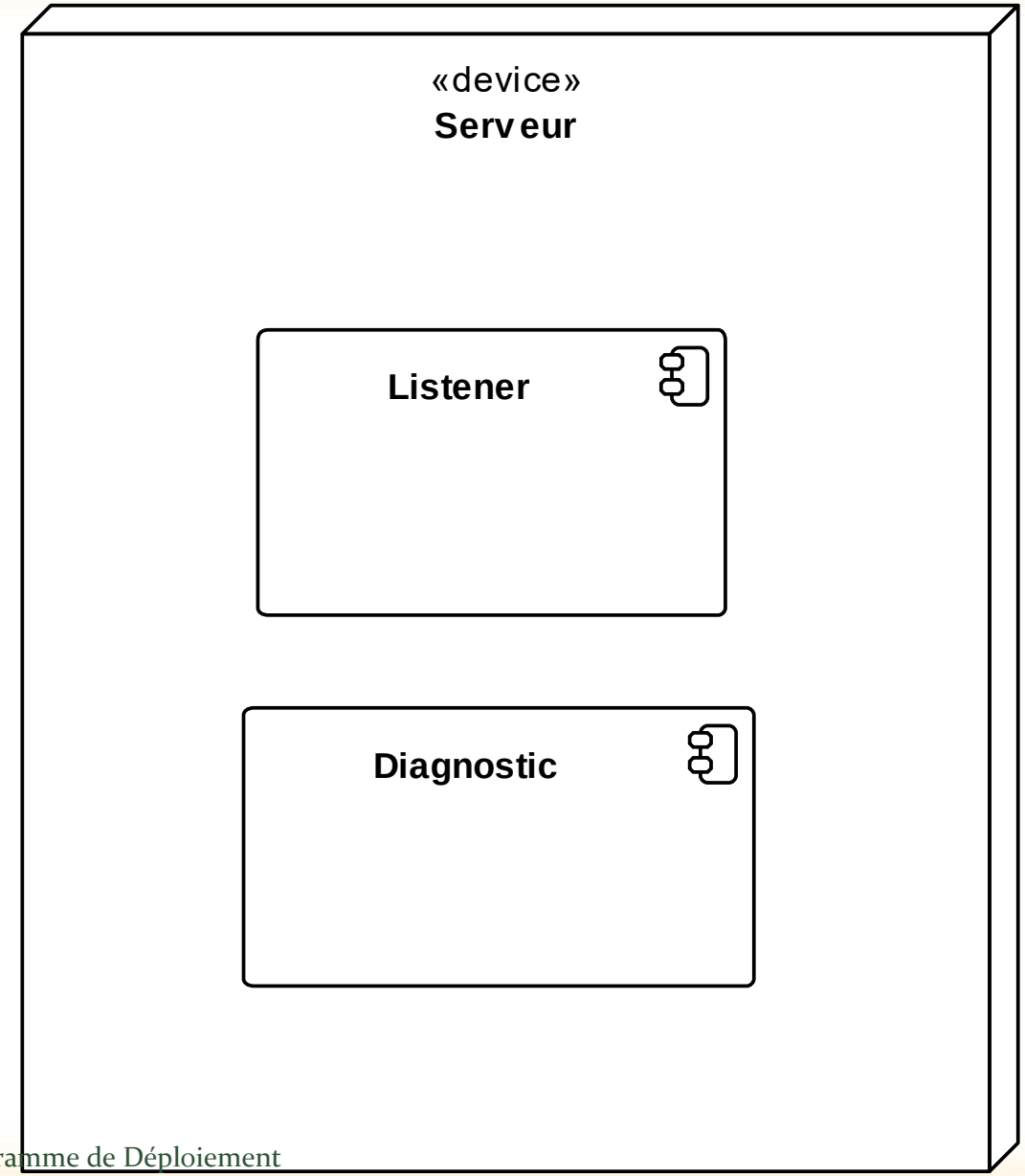
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Nœuds, composants et artifacts

- Un composant ***réside physiquement*** dans un nœud
- L'artifact est la manifestation ***physique*** d'un composant où tout autre élément physique (document, exécutable, code source,…)

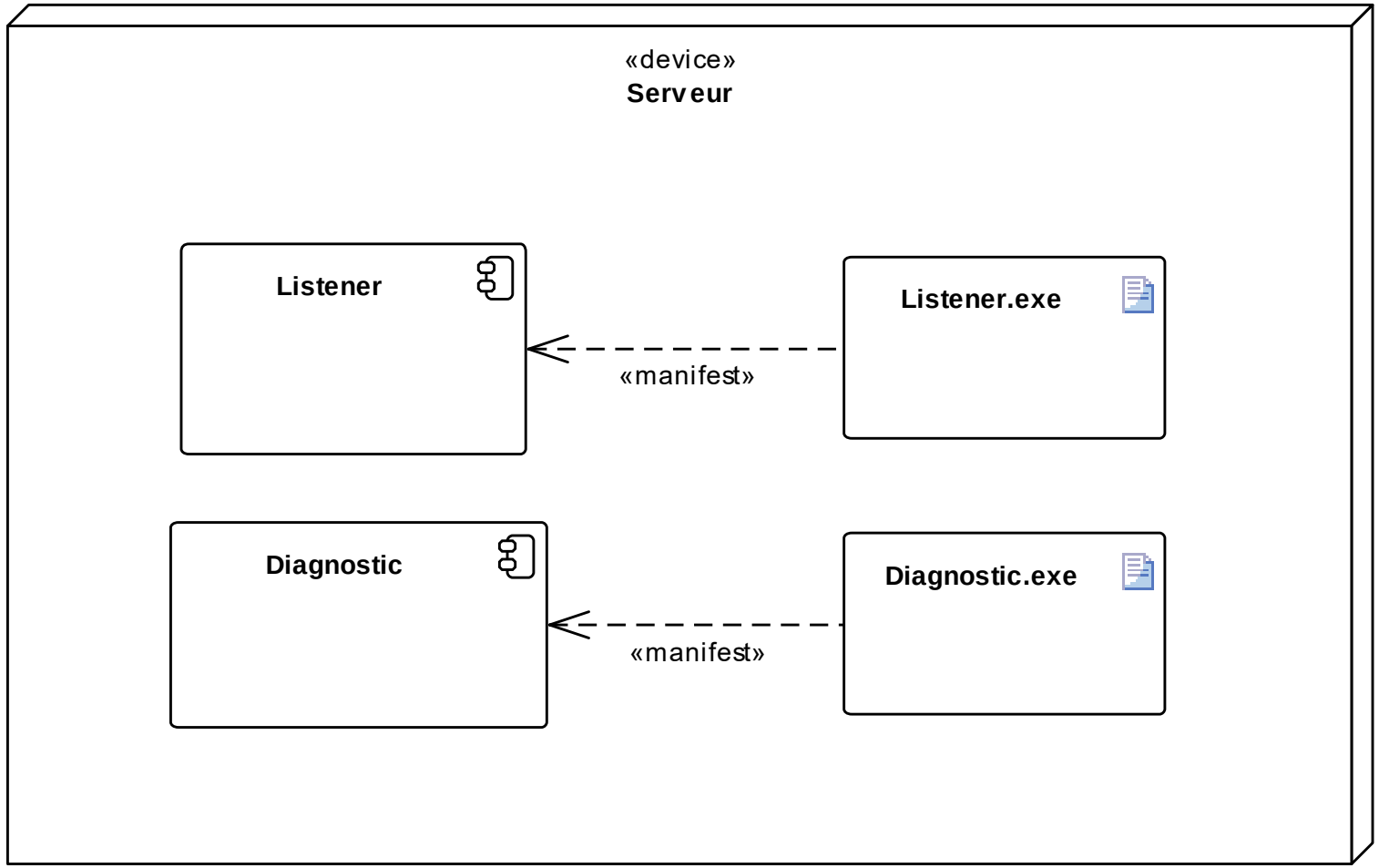
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Exemple 1



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Exemple 2



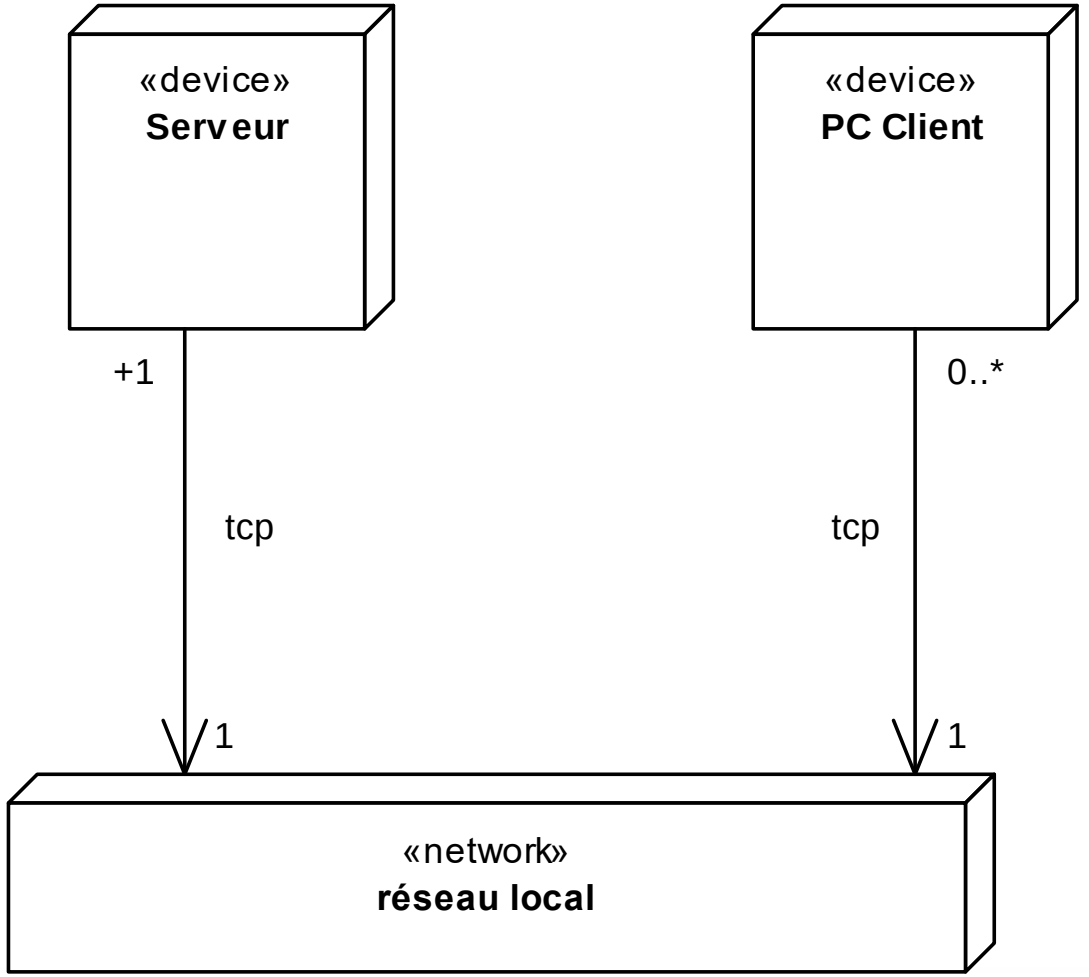
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Lien de communication

- Le lien de communication est une ***association*** entre les nœuds modélisation la communication entre ces nœuds

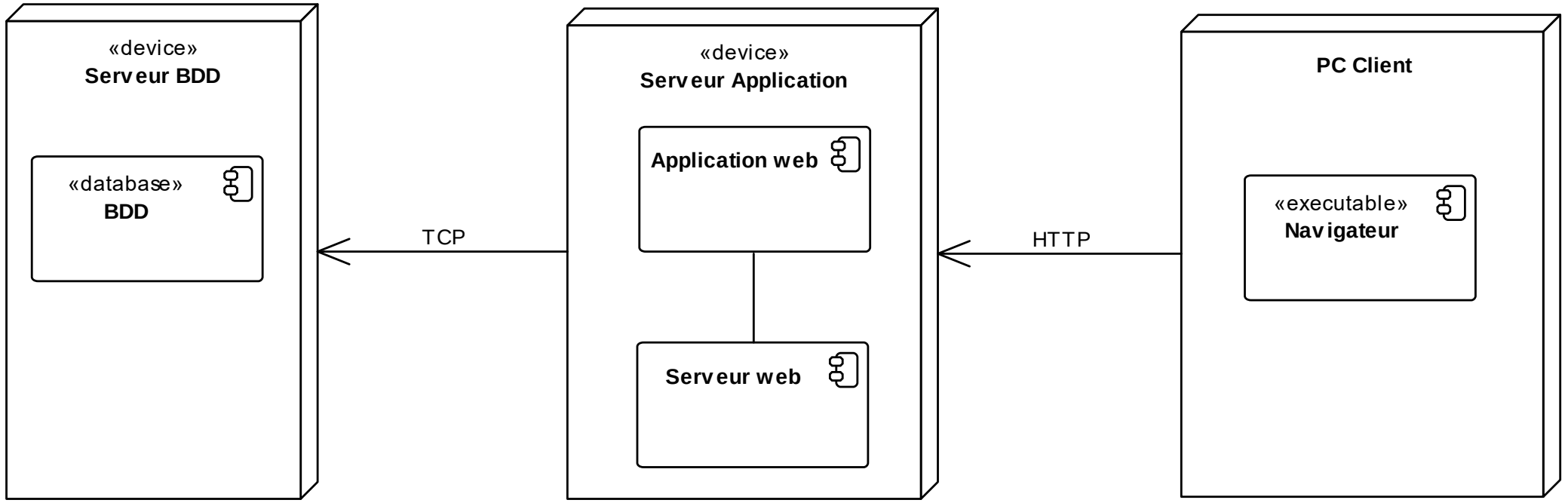
Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Lien de communication - Exemple



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Exemple d'architecture N/Tiers



Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Diagramme de Déploiement

SECTION 4, DÉBAT 05 MNS

Mostefai Mohammed Amine – Batata Sofiane | Copyright © ESI 2015-2016

Bibliographie

- UML Component Diagrams, Veronica Carrega, 2004
- Introduction to Software Architecture”David Garlanand Mary Shaw, January 1994
- Analyse, Conception Objet
- Diagrammes de déploiement, SIMMO/ENSM.SE, 2002