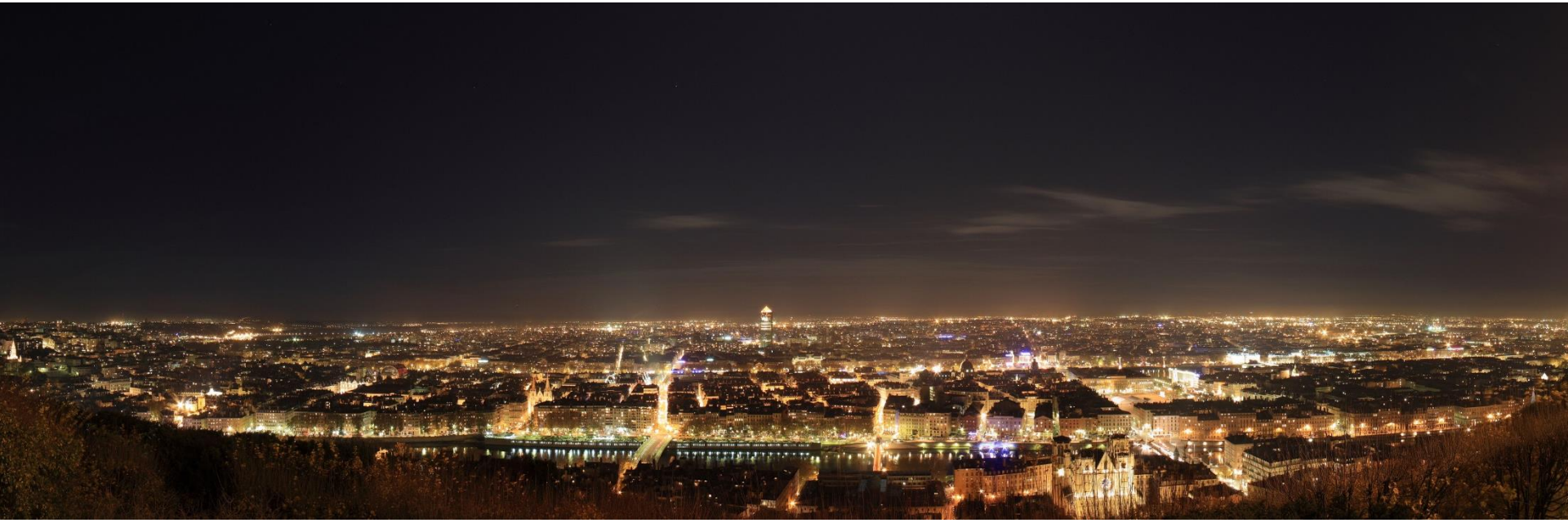
Urbanisation des systèmes d'information



Université Lyon 1, 6 Novembre 2014

Présentation

Julien VILLANTI (*julien.villanti@worldline.net*) *@JulienVillanti*

- **Unité**
  - **M**obility **&** e**T**ransactional **S**ervices (Business Center - **Contact**)

- **Fonctions**
  - Responsable d'application de 2009 à 2011
  - Expert technique depuis 2011
  - Membre **I**nnovation **Labs** depuis 2011

- **Missions**
  - Missions d'architecture, Lead Developer, Industrialisation
  - Support, Formation interne, Bonnes pratiques, …

- **Formation**
  - Master 2 TI (**T**echnologie de l'**I**nformation)

Business Center - Contact

Des projets critiques à forte volumétrie

2,2 Milliards **d'appels vocaux** gérés par nos SVI

20.000 **positions ACD** déclarées sur nos infrastructures ACD

3,0 Milliards **de SMS** entrants et sortants

28.000 **lignes télécom** absorbées par notre télécom center

worldline
e-payment services

Commençons par

Nous finirons par

Définitions

Urbanisation

Définition selon Wikipédia

L'urbanisation du système d'information de l'entreprise **est une discipline** informatique consistant à faire **évoluer le système d'information** d'une entreprise **dans son ensemble afin de garantir sa cohérence** vis-à-vis des **objectifs et du métier** de cette entreprise, en prenant en compte ses **contraintes externes et internes**, tout en tirant parti des opportunités de l'état de l'art informatique.

Cette discipline s'appuie sur une série de **concepts calqués sur ceux de l'urbanisation de l'habitat humain** (organisation des villes, du territoire), concepts qui ont été réutilisés en informatique **pour formaliser ou modéliser** le système d'information,

L'urbanisme définit des règles ainsi qu'un cadre cohérent, stable et modulaire, auquel **les différentes parties prenantes se réfèrent pour toute décision d'investissement** relative au management du système d'information.

En résumé, urbaniser, c'est diriger la transformation continue du système d'information en vue de le simplifier et garantir sa cohérence.

Système d’information

Définition selon Wikipédia

Un système d'information (SI) est un **ensemble organisé de ressources** (matériels, logiciels, personnel, données et procédures) **qui permet de regrouper, de classifier, de traiter et de diffuser de l'information** sur un environnement donné.

MOA : Métier / Utilisateurs

MOE : Système informatique

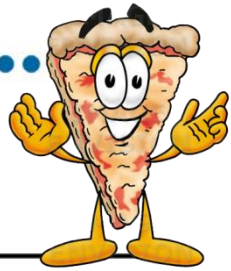
Système d’Information

worldline
e-payment services


fil rouge

Pizza 3000

Cas d’étude fictif



- **Petite entreprise de livraison de pizza**

- **Actuellement**
  - Les commandes se font par téléphone
  - 3 équipes : téléphone / cuisine / livraison

- **Souhaite**
  - S’ouvrir sur internet
  - Trouver des nouveaux moyens de communication avec ses clients

- **Objectif**
  - Doubler de taille en 2 ans

La démarche d’urbanisation

La démarche d'urbanisation

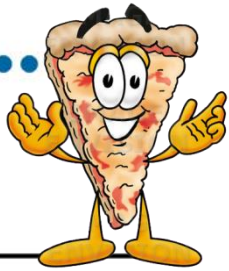
Les 4 grandes étapes

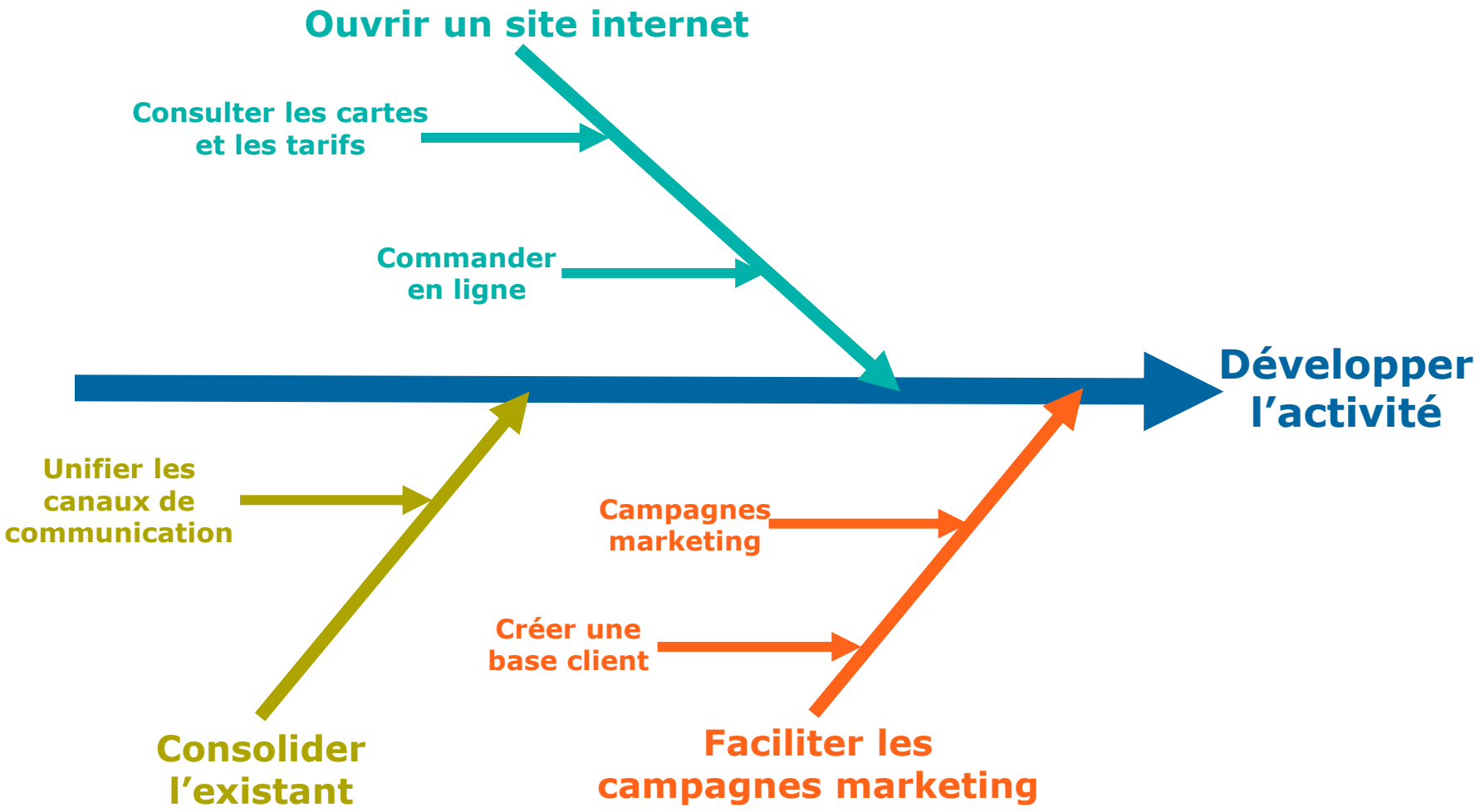
- **Définition des objectifs**
  - Stratégie commerciale
- **Analyse de l'existant**
  - Lister le patrimoine
  - Cartographier les différentes couches (métier, fonctionnelle, applicative, …)
- **Identification du SI cible**
  - Impact sur le différentes couches
  - Prise en compte des contraintes (humaine, matériels, …)
  - Définition des livrables
- **Mise en évidence de la trajectoire**
  - Définition des paliers
  - Planning


Goals

Pizza 3000

Objectifs stratégiques





worldline
e-payment services

L’état des lieux

Analyse de l'existant
Lister le patrimoine

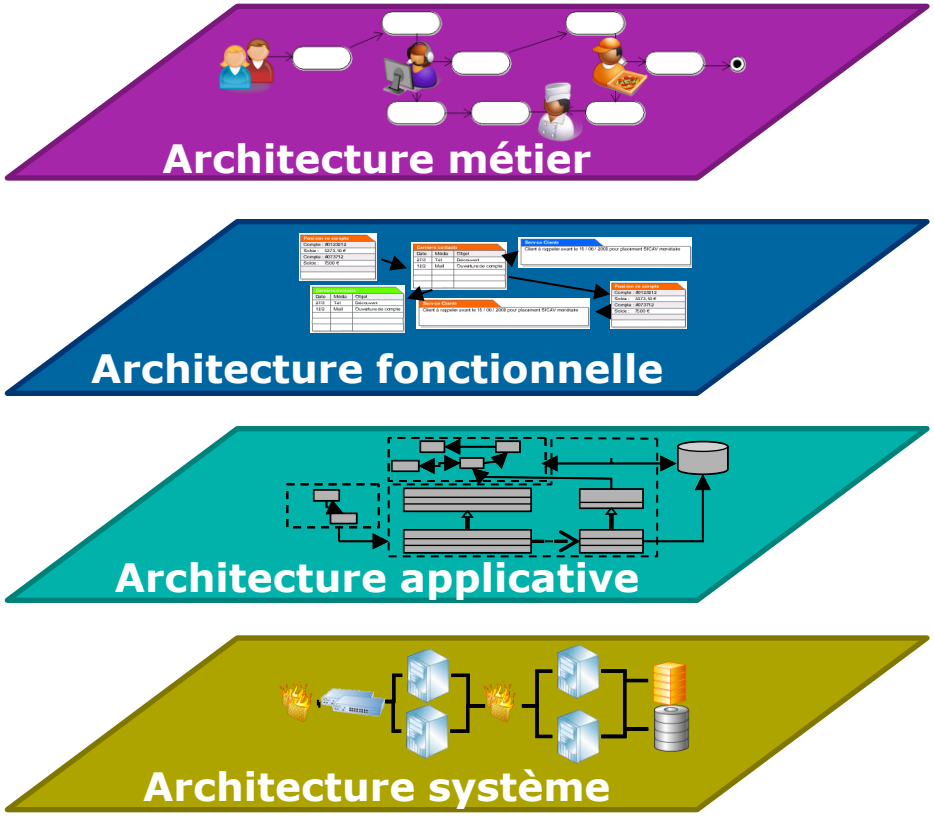
- **Capital matériel**
  - Logiciels (commerciaux, open source, progiciels, …)
  - Matériels (serveurs, réseaux, postes de travail, …)

- **Capital immatériel**
  - Savoir faire (procédure, formation, expertise, …)
  - Contribution à l'activité

Remarques : Suivant la taille de l'entreprise, le SI représente d'une dizaine à plusieurs milliers d'applications

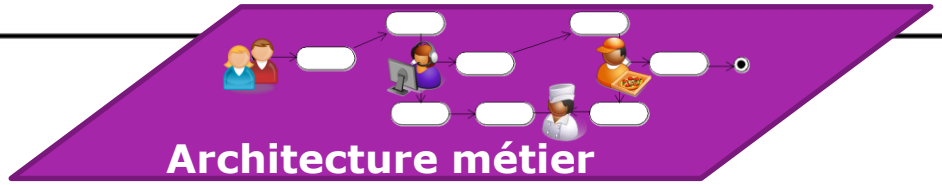
Analyse de l’existant

Cartographier les différentes couches



worldline
e-payment services

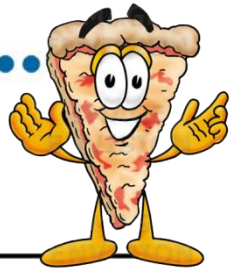
Architecture métier

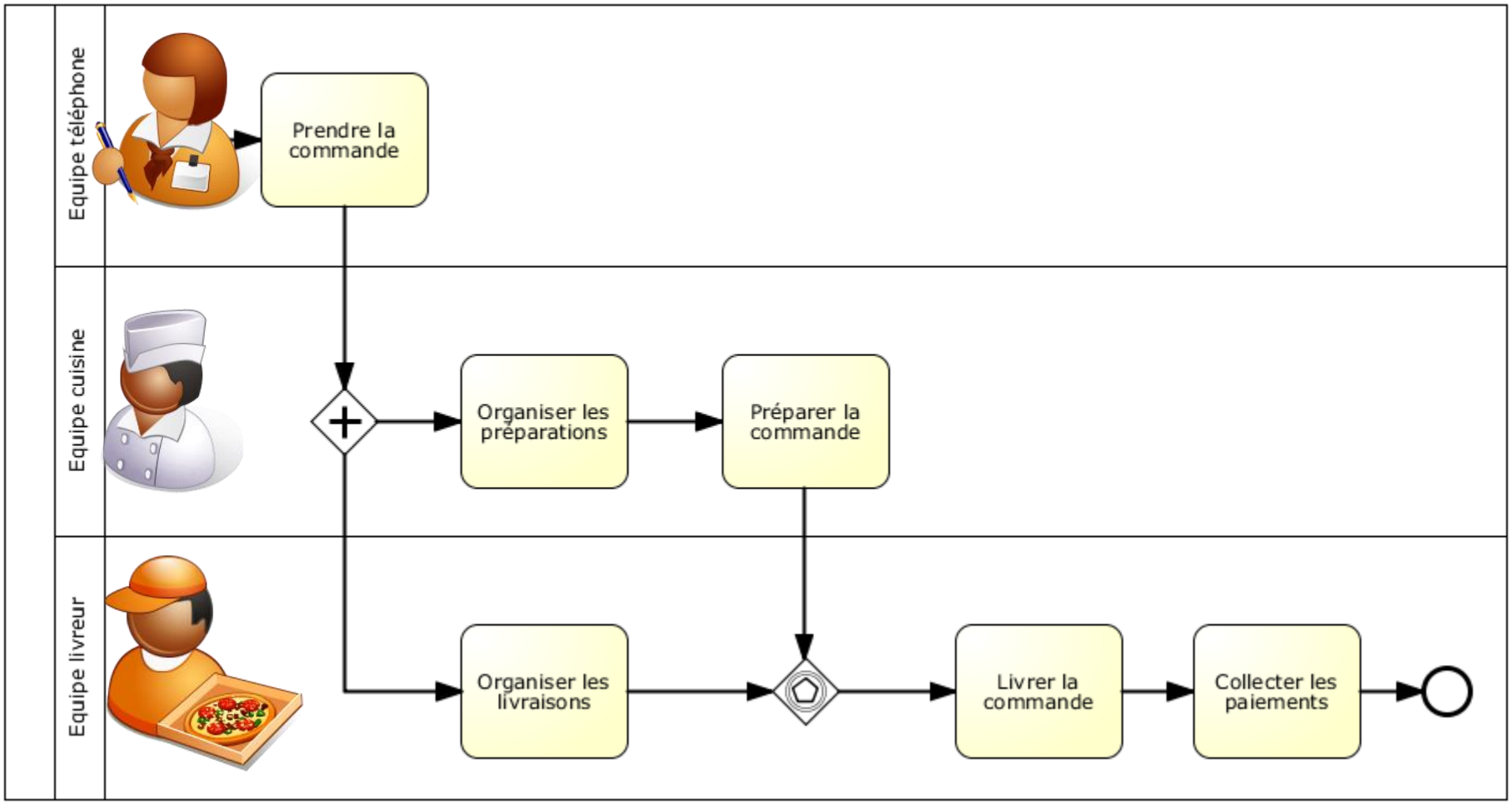


- **Identifier les « processus métiers »**
  - Qui fait quoi et pourquoi ?
  - Concept simple mais en général, personne n'a la vision complète
- **Forme**
  - Différents formats BPMN, EPC, ...
- **Difficultés**
  - Sensible, touche à l'organisation et aux personnes
  - Son importance est souvent sous-estimée
- **Valeur ajoutée**
  - Améliore la compréhension globale, bon terrain de dialogue MOA / MOE
  - Démultiplie les possibilités d'optimisation

Pizza 3000

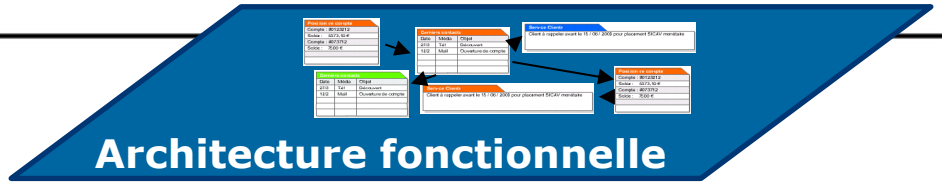
Architecture métier





worldline
e-payment services

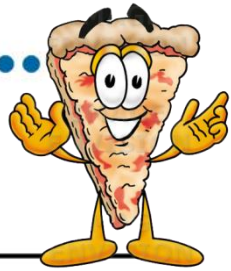
Architecture fonctionnelle



- **Identifier les « blocs fonctionnels »**
  - De quoi a-t-on besoin pour réaliser les processus métiers ?
- **Forme**
  - Basé sur un découpage classique en zones (échanges, cœur de métier, données de référence, données de production, activités support, pilotage)
- **Difficultés**
  - Etre cohérent avec les processus métiers
  - Choisir le bon niveau de détail
- **Valeur ajoutée**
  - Présentation hiérarchique, facilite le découpage du travail

Pizza 3000

Architecture fonctionnelle



Zone d'échanges

- Accueil téléphonique

Pilotage

- Suivi d'activité
- Suivi des incidents

Activités opérationnelles

- Prise de commande
- Préparation des commandes
- Suivi livraisons
- Facturation et encaissements

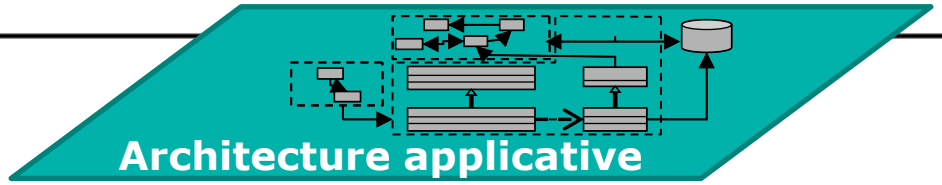
Activités de support

- RH
- Stocks
- Comptabilité

Gisements de données

- Information livreurs
- Information cuisine

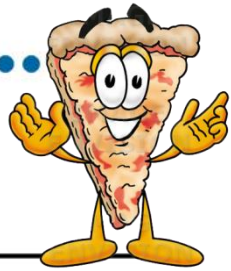
Architecture applicative

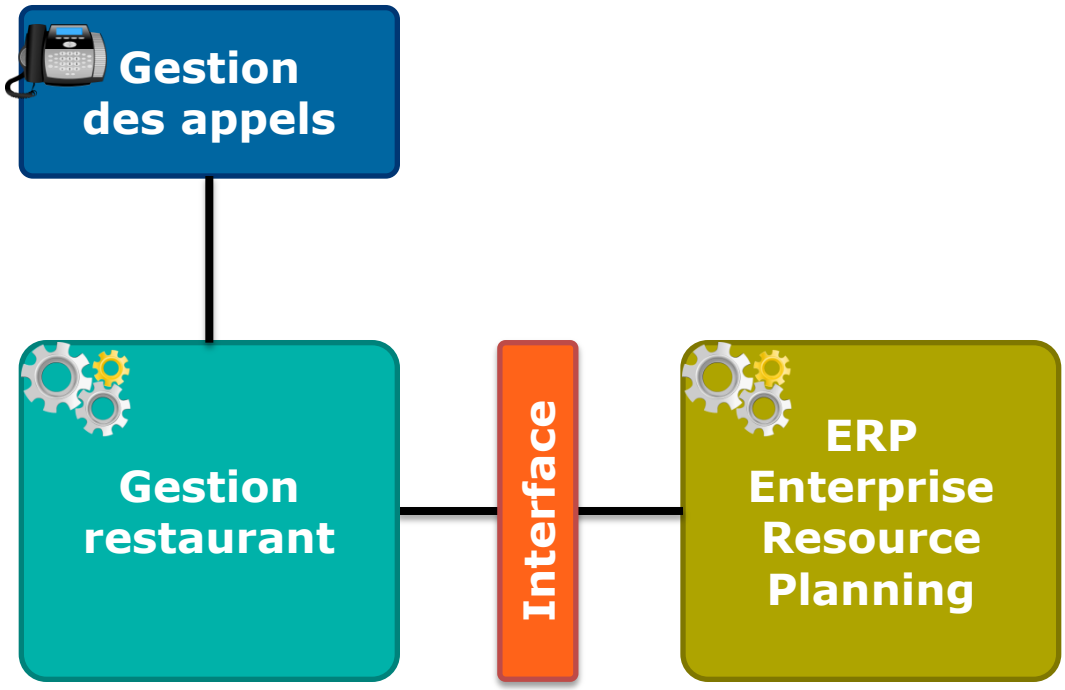


- **Identifier les applications**
  - Comment va-t-on réaliser les fonctionnalités ?
- **Forme**
  - Basé sur un découpage classique N-Tiers
- **Difficultés**
  - Apporter de la valeur et des solutions par rapport à l'architecture fonctionnelle
  - Rester compréhensible par le commun des mortels
- **Valeur ajoutée**
  - Jette les bases de la réalisation (grands choix technologiques, etc)

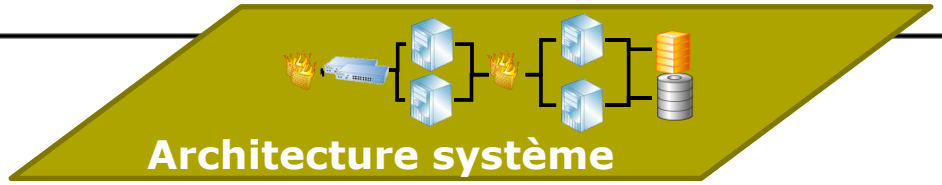
Pizza 3000

Architecture applicative





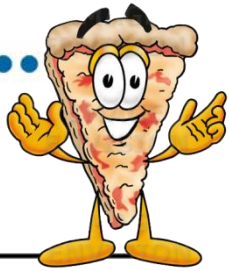
Architecture système

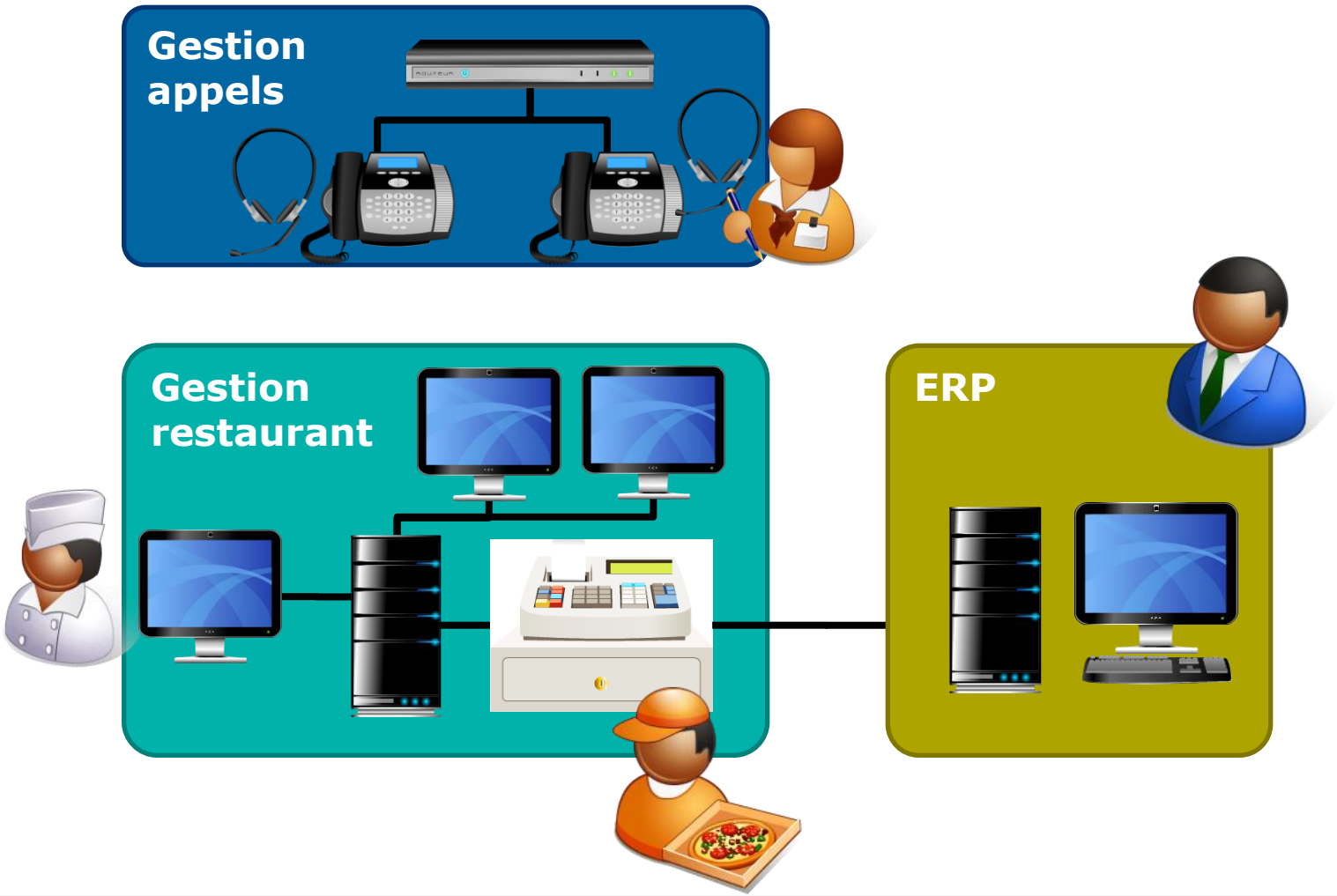


- **Identifier les composants techniques**
  - Avec quoi et où fonctionnent les applications
- **Forme**
  - Basé sur un découpage classique en zones techniques (sécurité, stockage, …)
- **Difficultés**
  - Faire le lien entre les applications et les serveurs
  - Rester compréhensible par le commun des mortels
- **Valeur ajoutée**
  - Apporte du concret et du structurant
  - Indispensable pour évaluer le coût du système

Pizza 3000

Architecture système



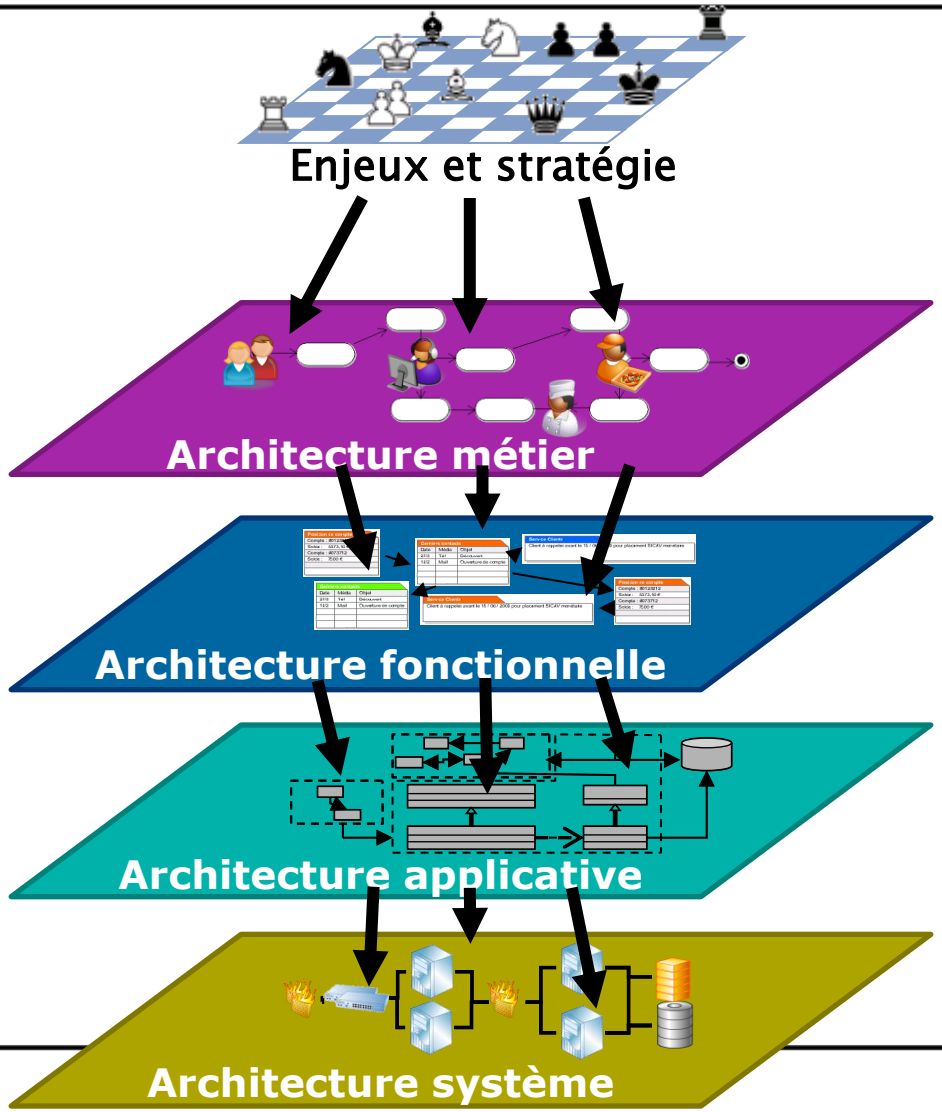


worldline
e-payment services

La cible

SI cible

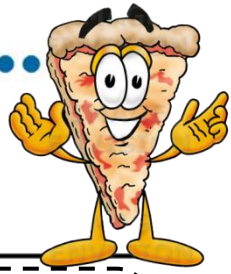
Impact sur le différentes couches



worldline
e-payment services

Pizza 3000

Cible fonctionnelle



Pilotage

- Suivi d'activité
- Suivi des incidents
- Marketing

Zone d'échanges

- Accueil téléphonique
- Accueil Internet
- Sms / Mail

Activités opérationnelles

- Prise de commande
- Préparation des commandes
- Suivi livraisons
- Facturation et encaissements

Gisements de données

- Information livreurs
- Information cuisine

Données marketing

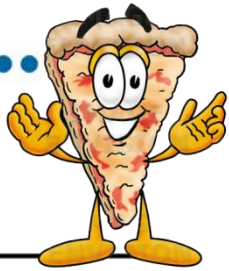
- Clients
- Cartes et menus

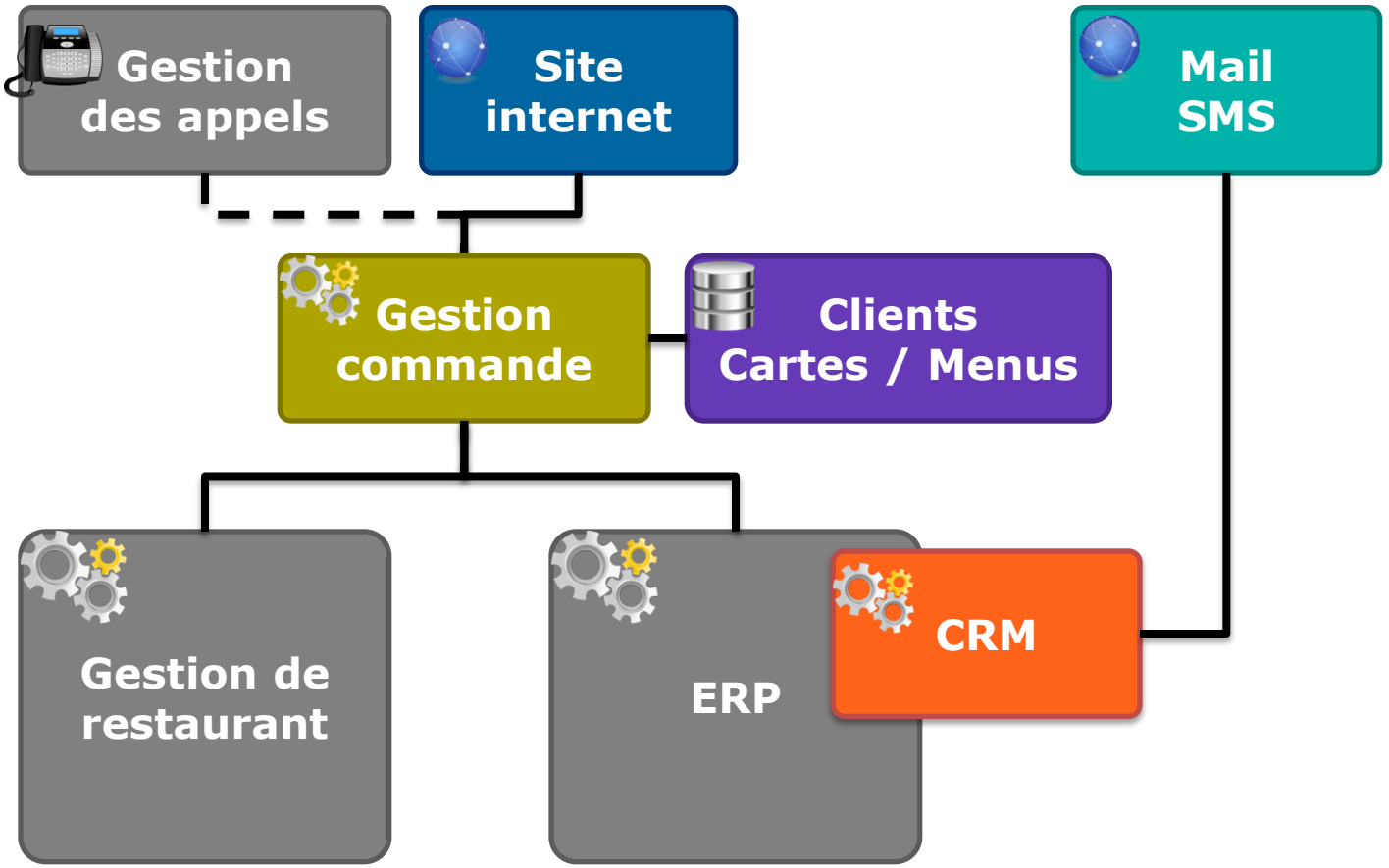
Activités de support

- RH
- Stocks
- Comptabilité

Pizza 3000

Cible applicative

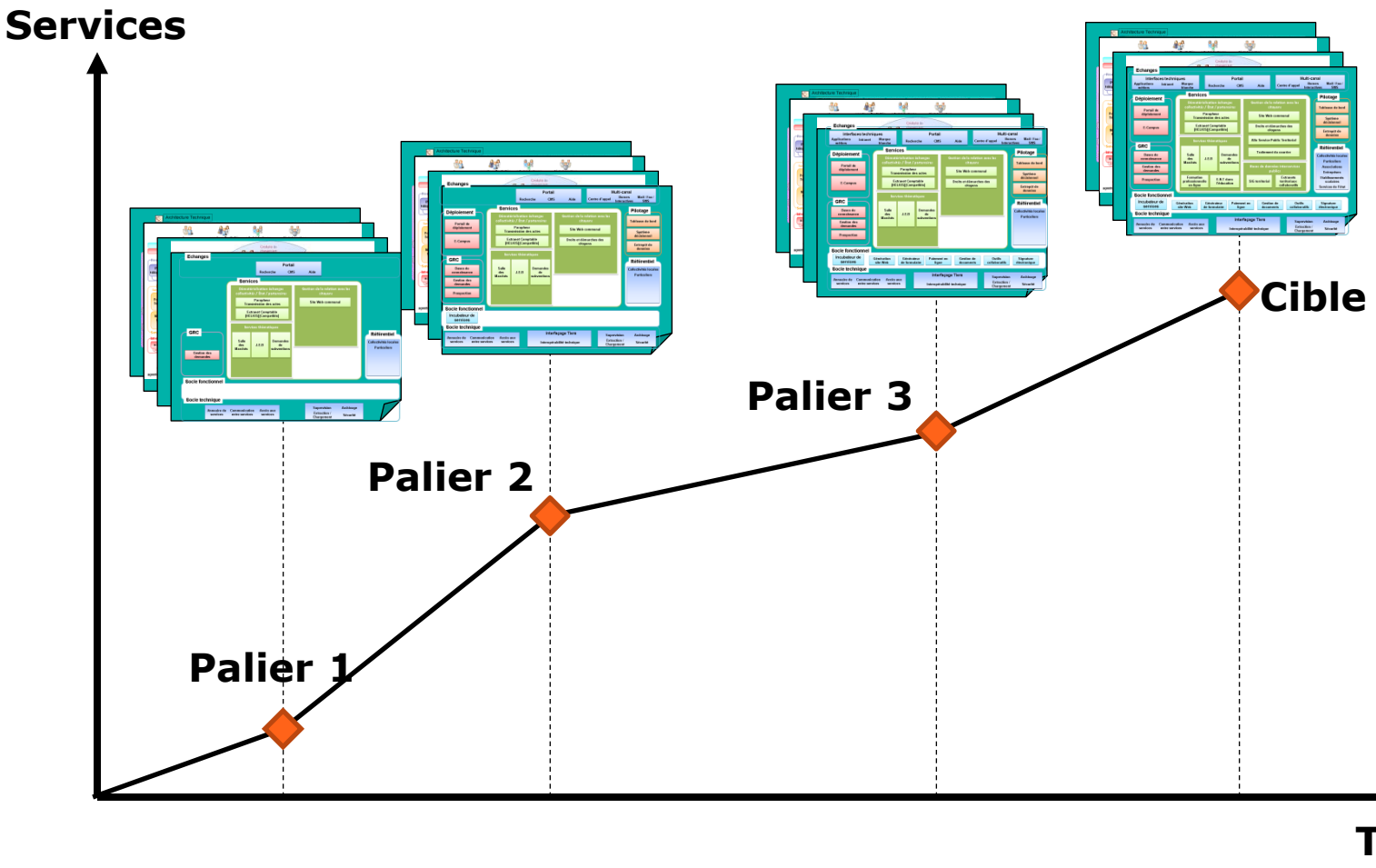




La trajectoire

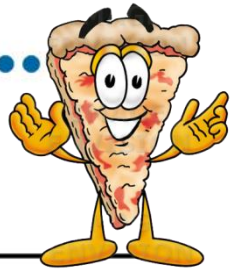
Définition de la trajectoire

Palier par palier



Pizza 3000

Les paliers, le planning



- **Etape 1 : TO + 3 mois**
  - Mise en place de la gestion de commandes et du référentiel client
- **Etape 2 : TO + 4 mois**
  - Ouverture du site internet
- **Etape 3 : TO + 6 mois**
  - Nouvelles fonctionnalités CRM : campagne de promotion par mail, …

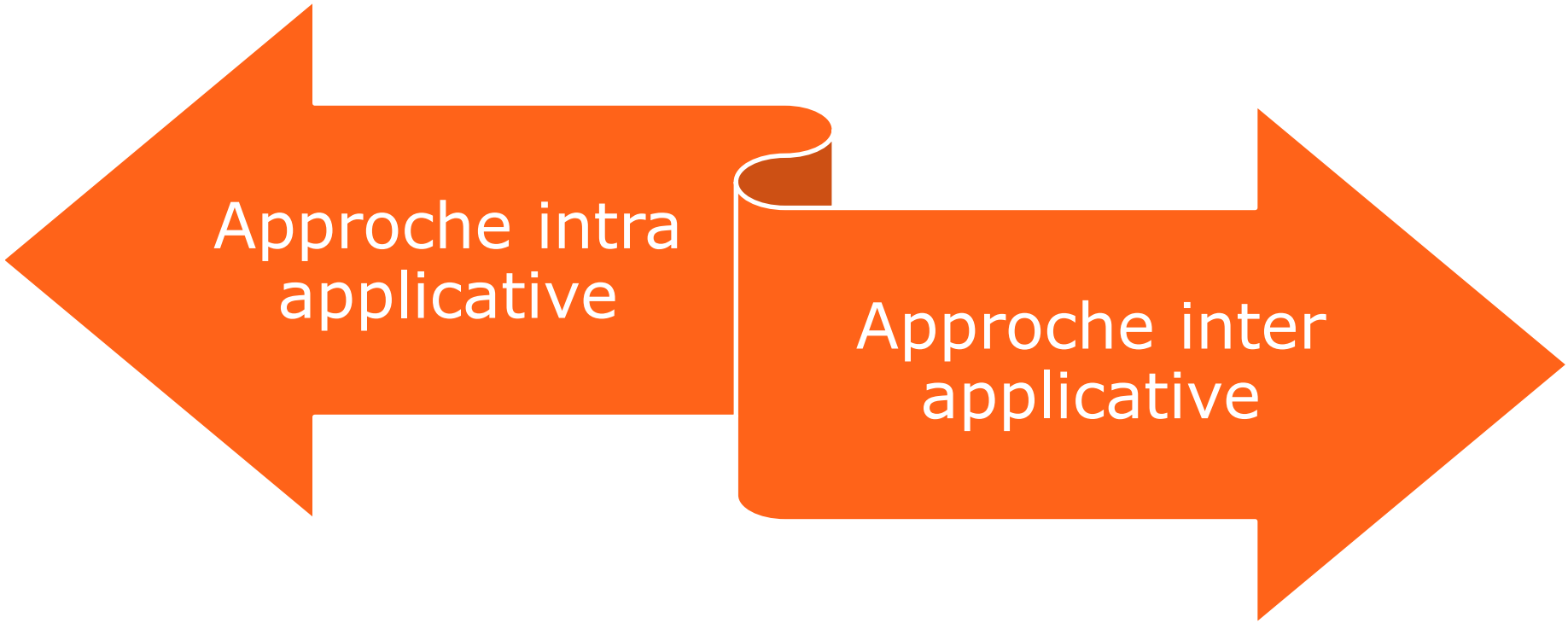
Pour aller plus loin …

Les livres

Approche par les développements

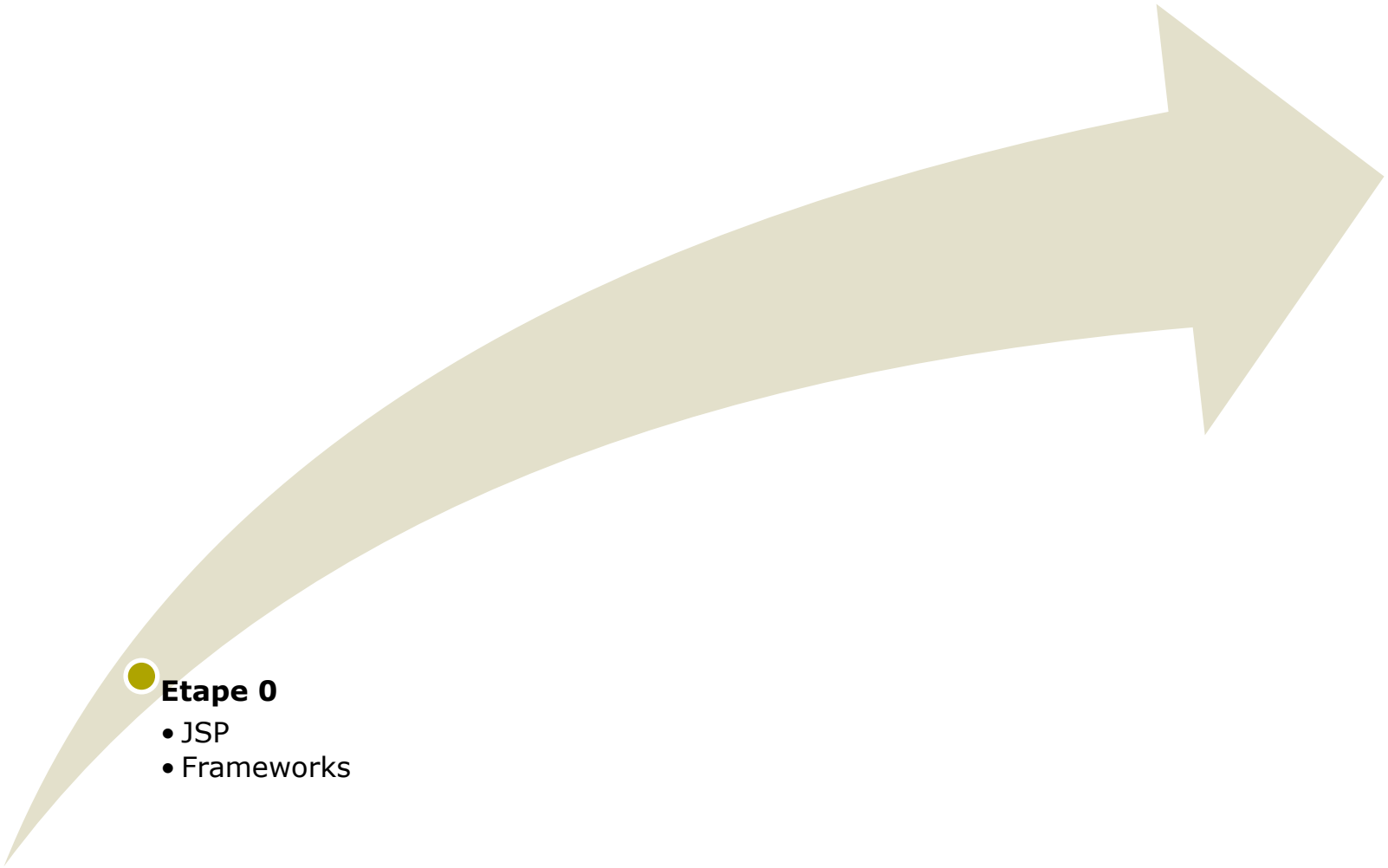
Les différentes approches

Intra / Inter



Approche intra applicative

Etape 0



worldline
e-payment services

Pizza 3000 - Avancement

"La" JSP

```
<html>
    <head>
        <title>Obtaining a Connection</title>
    </head>
    <body>
        <h1>Liste des pizzas</h1>
        <%
        Connection conn = null; ResultSet result = null; Statement stmt = null;
        try {
            Class c = Class.forName("com.mysql.jdbc.Driver");
        }
        catch (Exception e) { System.out.println("Error occurred " + e); }
        try {
            conn = DriverManager.getConnection("jdbc:mysql://localhost/urba", "urba","Ur8@");
        }
        catch (SQLException e) { System.out.println("Error occurred " + e); }
        try {
            stmt = conn.createStatement();
            result = stmt.executeQuery("SELECT * FROM Pizza");
        }
        catch (SQLException e) { System.out.println("Error occurred " + e); }
        %>
        <ul>
            <%
            while(result.next()) {
                out.println("<li>" + result.getString(2) + "</li>");
            }
            %>
        </ul>
    </body>
</html>
```

Les frameworks

La jungle

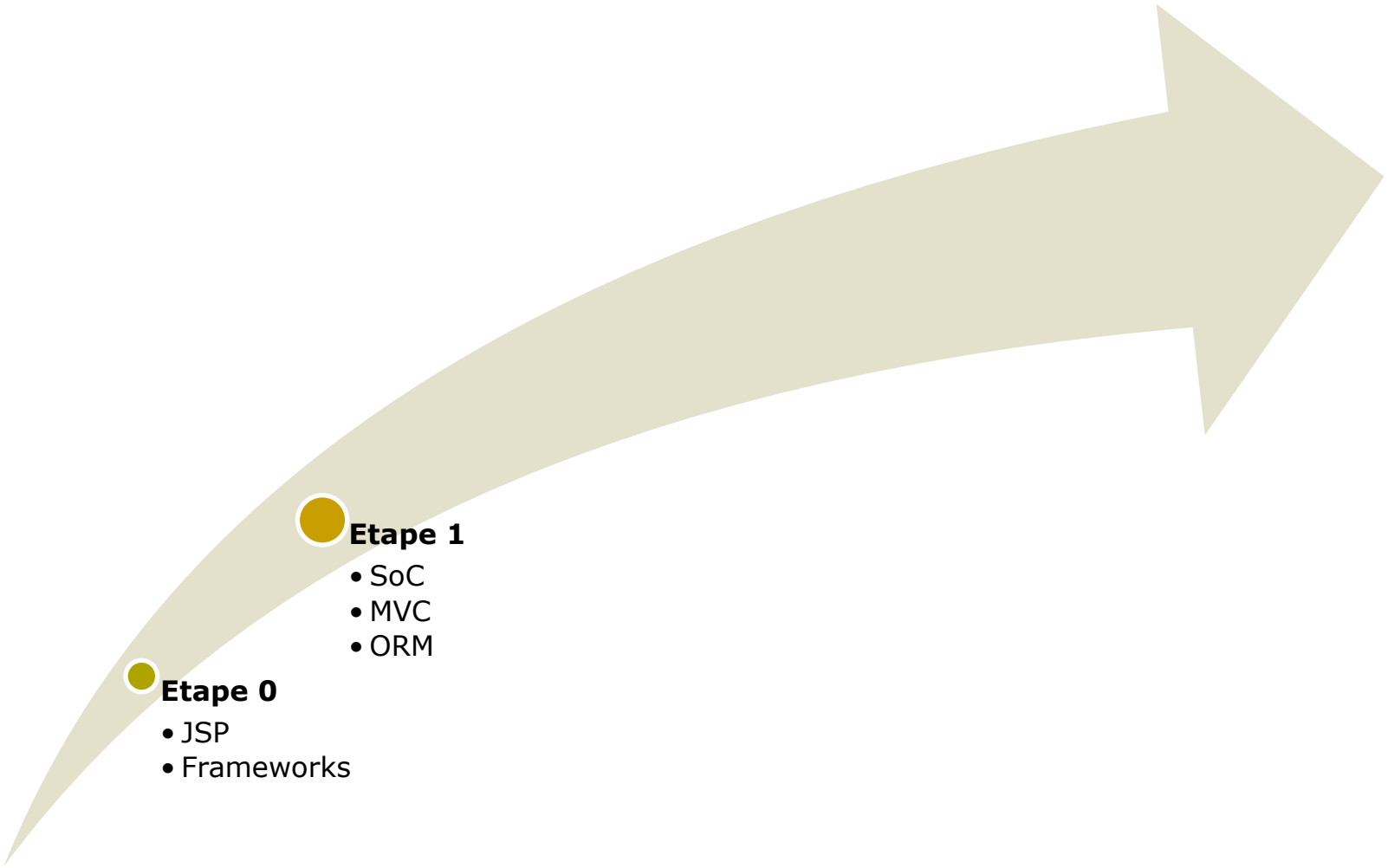
worldline
e-payment services

Les frameworks

Pourquoi ?

Approche intra applicative

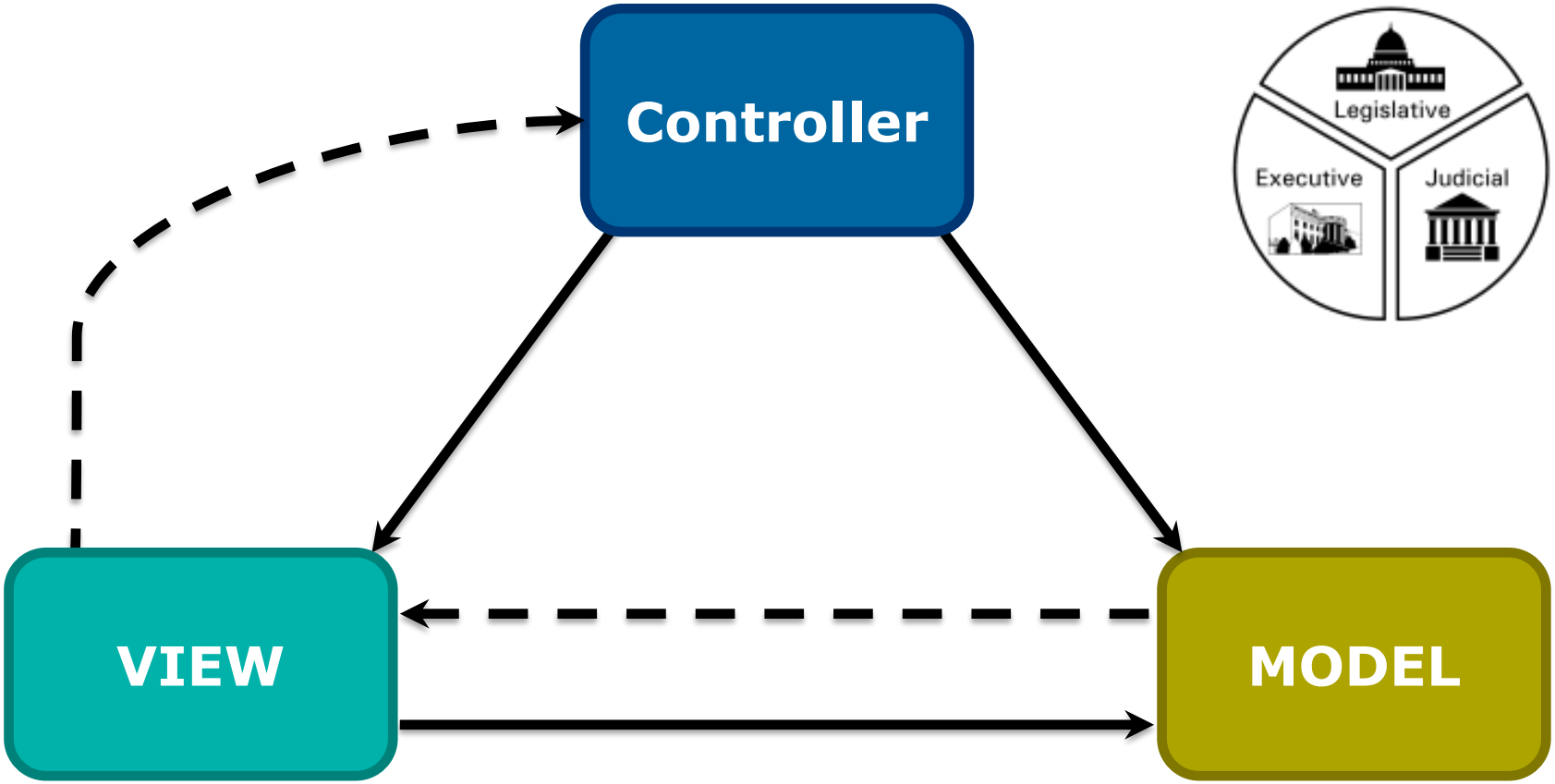
Etape 1



worldline
e-payment services

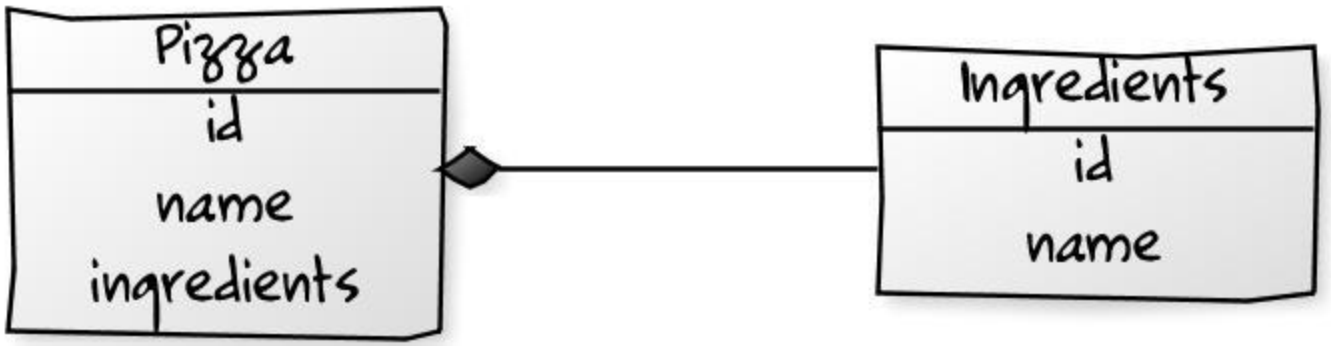
Separation of Concerns

MVC Pattern



Pizza 3000 - Avancement

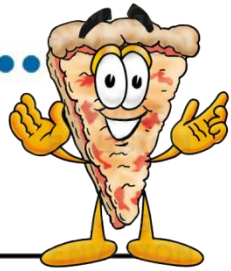
Model

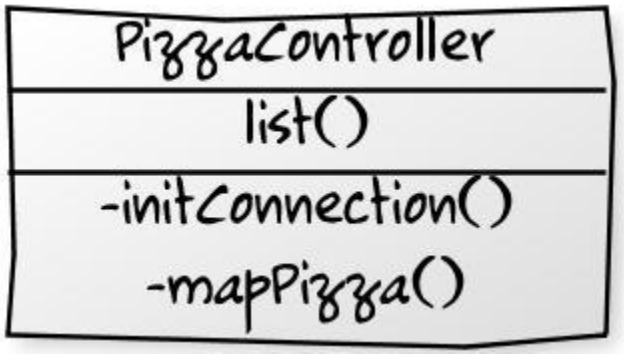


```java
public class Pizza
{
    private Integer id;

    private String name;

    private List<Ingredient> ingredients = new ArrayList<Ingredient>();

    public Pizza() {
    }

    public Pizza(String name) {
        this.name = name;
    }

    public Integer getId() {
        return id;
    }

    public Pizza setId(Integer id) {
        this.id = id;
        return this;
    }

    public String getName() {
        return name;
    }

    public Pizza setName(String name) {
        this.name = name;
        return this;
    }

    public List<Ingredient> getIngredients() {
        return ingredients;
    }

    public Pizza setIngredients(List<Ingredient> ingredients) {
        this.ingredients = ingredients;
        return this;
    }
}
```

Pizza 3000 - Avancement

Controller





```java
package org.losohome.pizza;

import java.util.List;
import java.util.ArrayList;

import java.sql.*;

public class PizzaController {
    private Connection conn;

    public PizzaController() {
        initConnection();
    }

    public void list() throws SQLException {
        Statement stmt = conn.createStatement();
        ResultSet result = stmt.executeQuery("SELECT * FROM Pizza");
        List<Pizza> pizzas = mapPizzas(result);
        for(Pizza pizza : pizzas) {
            System.out.println(pizza);
        }
    }

    private List<Pizza> mapPizzas(ResultSet result) throws SQLException {
        List<Pizza> pizzas = new ArrayList<Pizza>();
        Pizza pizza;
        while(result.next()) {
            pizza = new Pizza(result.getString(2));
            pizza.setId(result.getInt(1));
            pizzas.add(pizza);
        }
        return pizzas;
    }

    private void initConnection() {
        try {
            Class c = Class.forName("com.mysql.jdbc.Driver");
            conn = DriverManager.getConnection("jdbc:mysql://localhost/urba", "urba", "Ur8@");
        }
        catch (Exception e) {
            System.out.println("Error occurred " + e);
        }
    }
}
```

Le Controller joue son rôle de chef d'orchestre

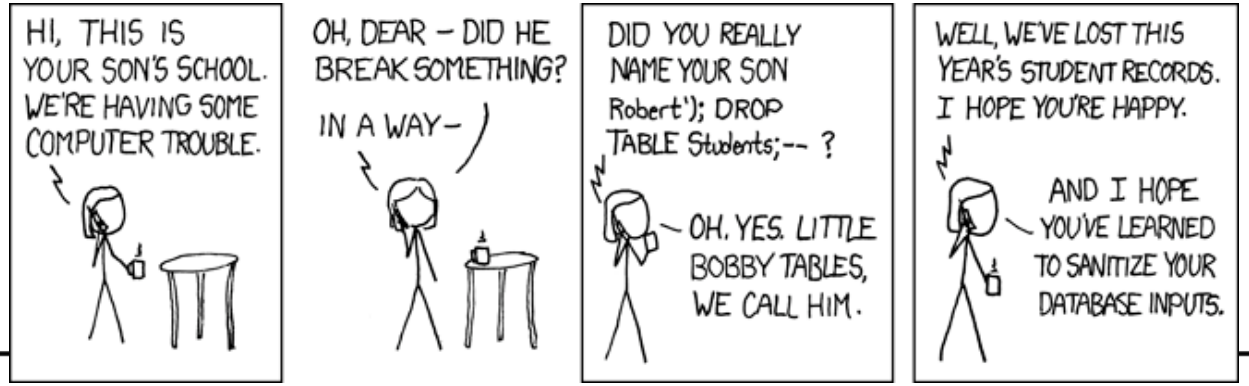
Utilisation du model Pizza

Gestion « manuelle » de l'accès à la BDD

ORM : Object Relational Mapping

Pourquoi ?

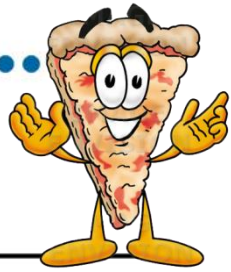
- Travail avec des objets
- S'abstraire du SQL, du SGBD
- Gestion des transactions
- Performances
- Sécurisation

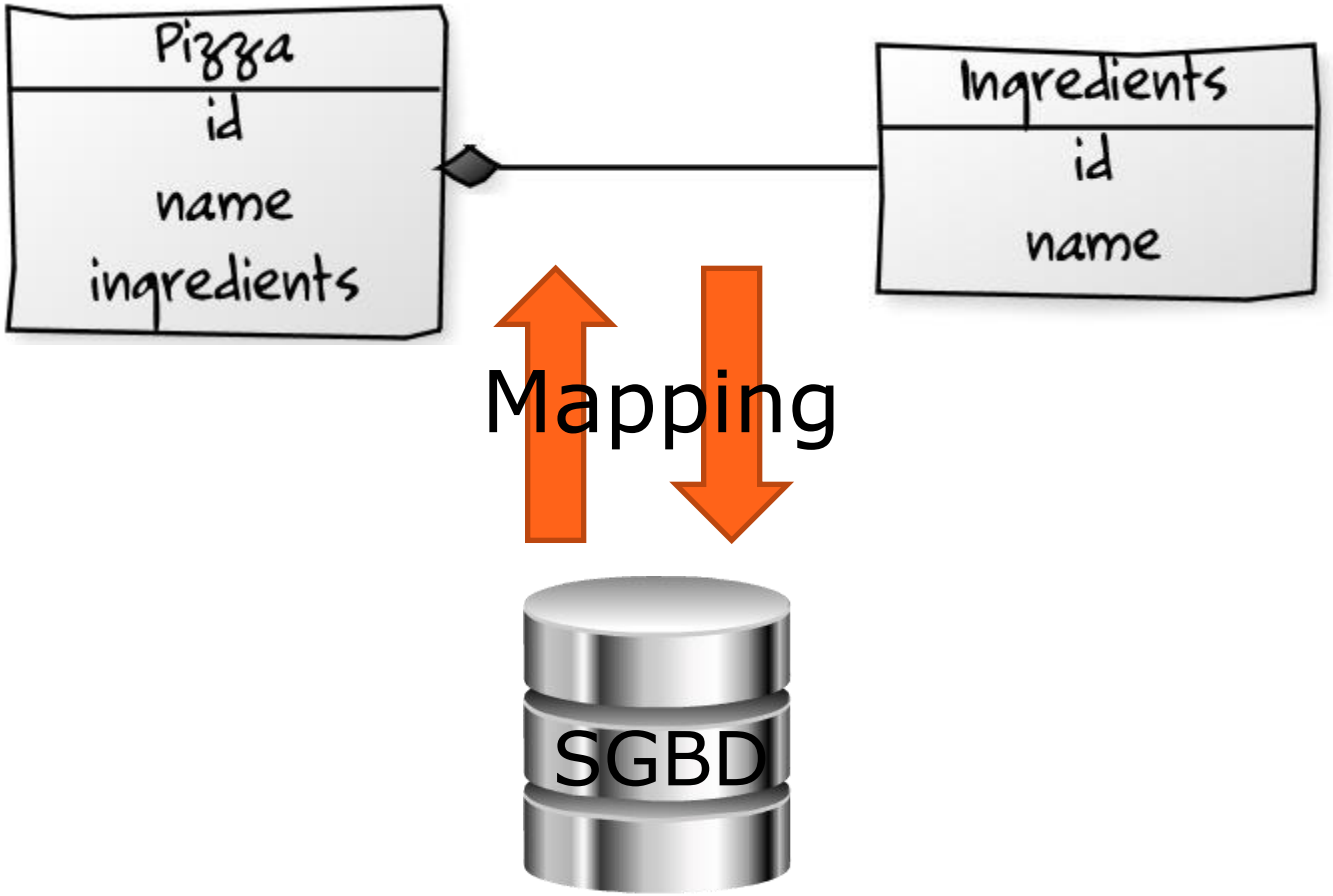


worldline
e-payment services

Pizza 3000 - Avancement

ORM : Object Relational Mapping

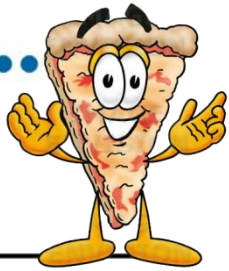


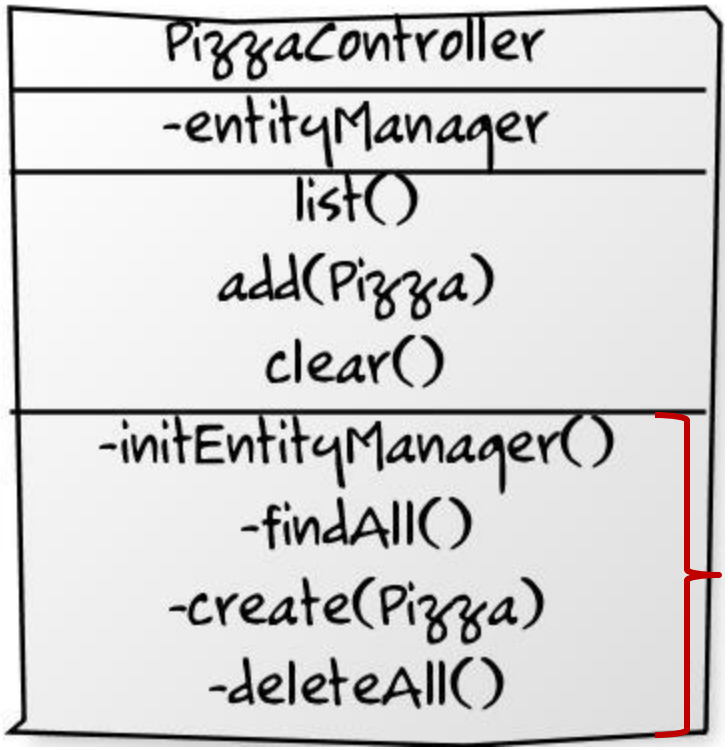


```java
@Entity
public class Pizza
{
    @Id @GeneratedValue
    private Integer id;

    @Column
    private String name;

    @OneToMany(mappedBy="pizza", cascade=CascadeType.ALL, fetch=FetchType.EAGER)
    private List<Ingredient> ingredients = new ArrayList<Ingredient>();

    public Pizza() {
    }

    public Pizza(String name) {
        this.name = name;
    }

    public Integer getId() {
        return id;
    }

    public Pizza setId(Integer id) {
        this.id = id;
        return this;
    }

    public String getName() {
        return name;
    }

    public Pizza setName(String name) {
        this.name = name;
        return this;
    }

    public List<Ingredient> getIngredients() {
        return ingredients;
    }
}
```

Pizza 3000 - Avancement

Continuons les améliorations

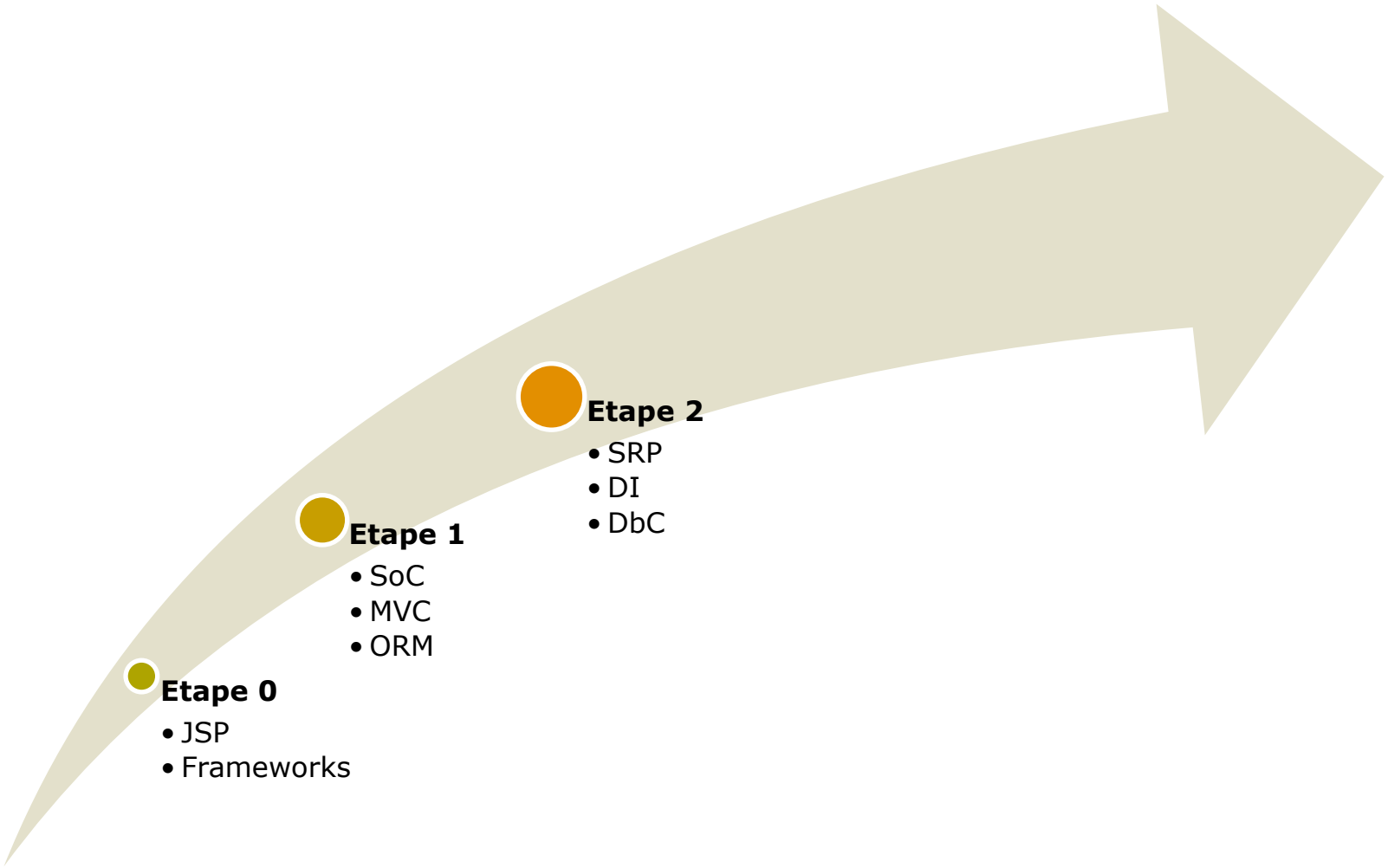




Encapsulation de l'accès aux données

Approche intra applicative

Etape 2



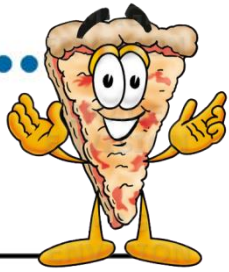
worldline
e-payment services

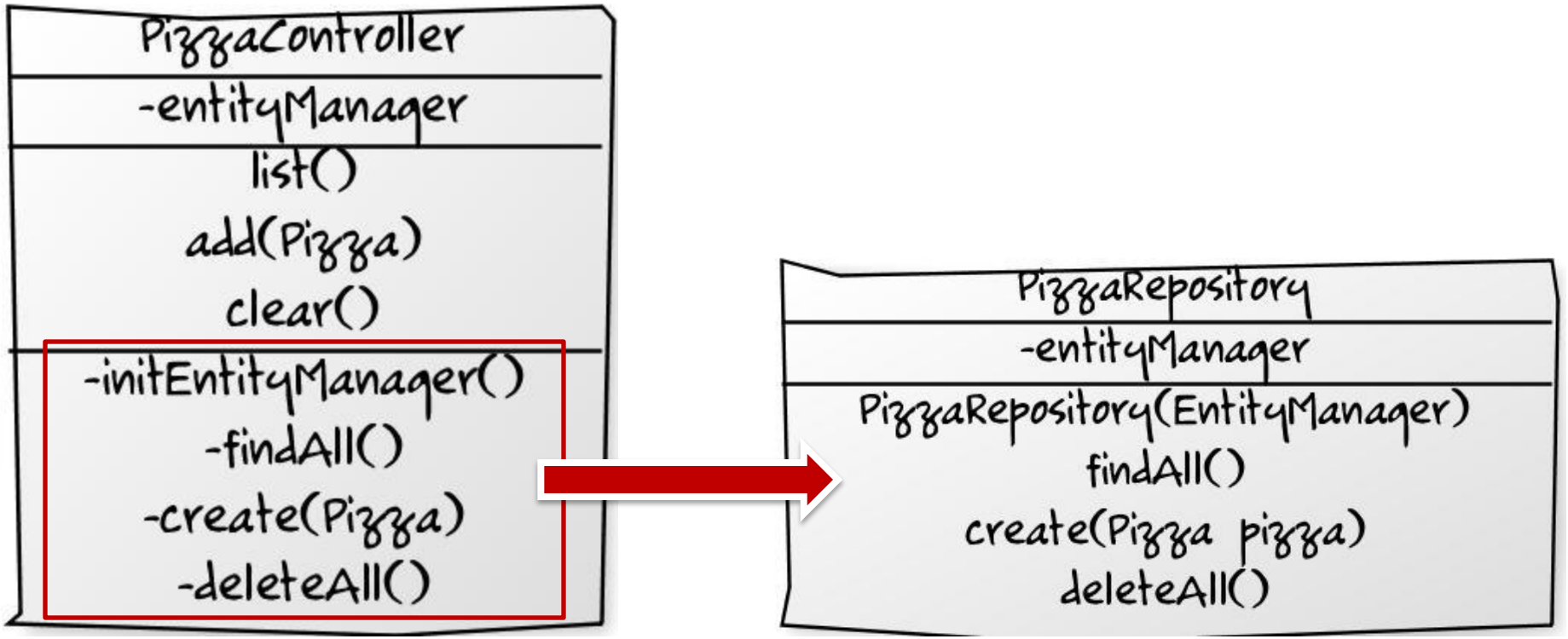
SRP : Single Responsibility Principle

worldline
e-payment services

Pizza 3000 - Avancement

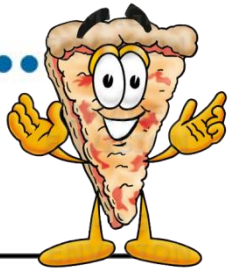
SRP : Single Responsibility Principle

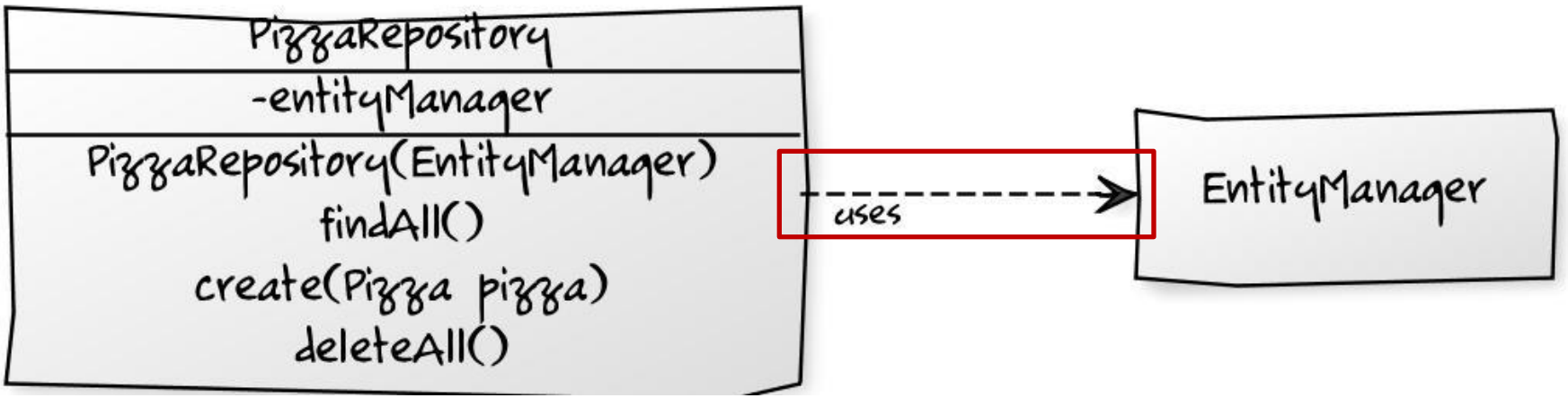




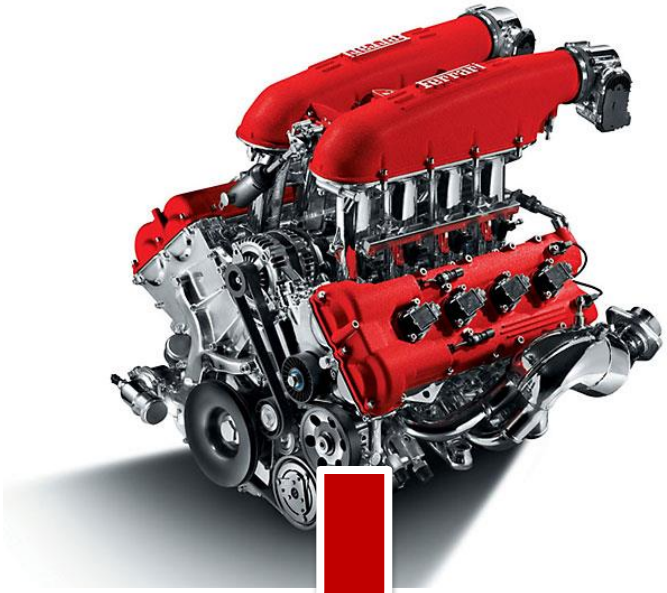
Pizza 3000 - Avancement

Continuons les améliorations





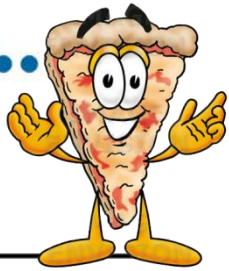
DI : Dependency Injection

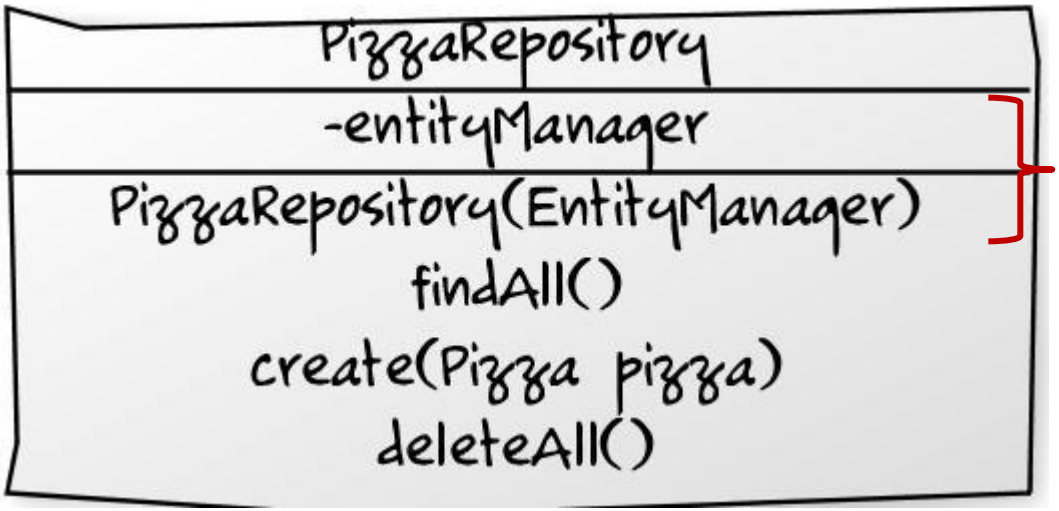


worldline
e-payment services

Pizza 3000 - Avancement

Continuons les améliorations





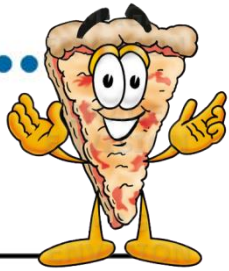
Technologie imposée

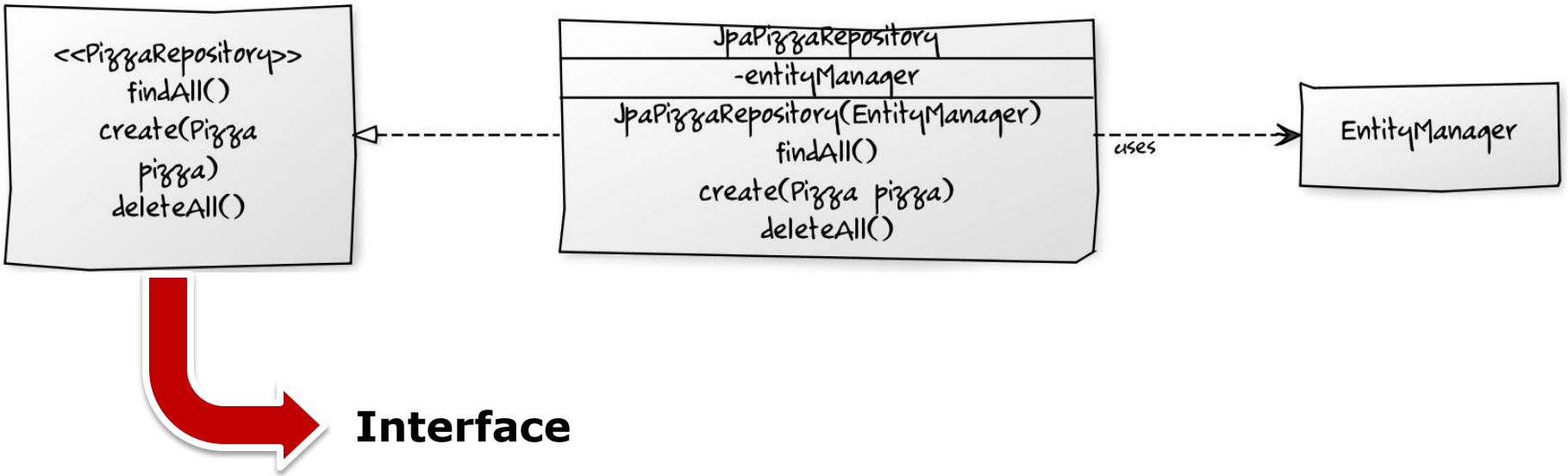
DbC : Design by Contract

worldline
e-payment services

Pizza 3000 - Avancement

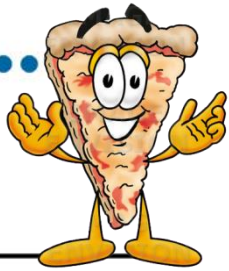
DbC : Design by Contract



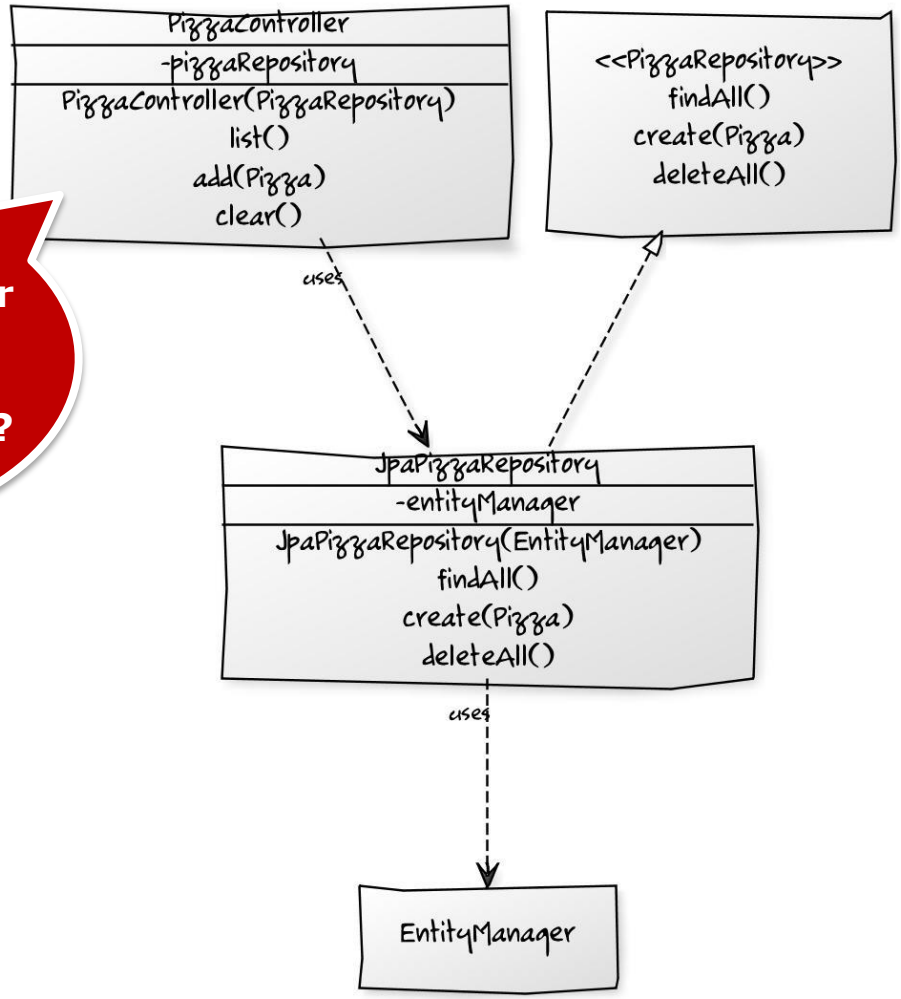


Pizza 3000 - Avancement

Problématique ?

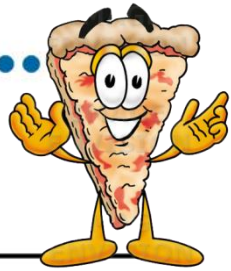


Comment récupérer une instance de PizzaController ?



Pizza 3000 - Avancement

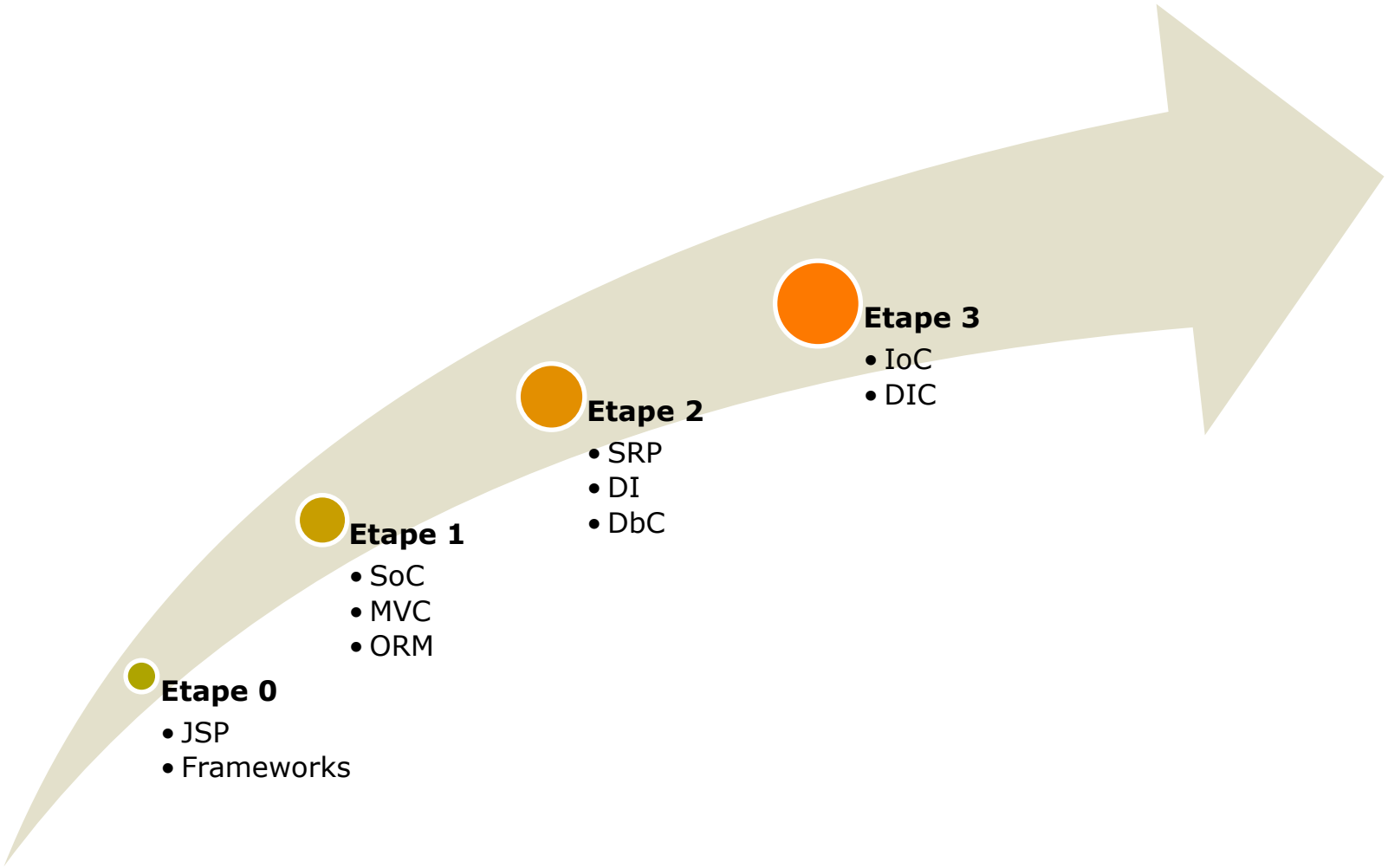
L'instanciation manuelle



```
EntityManager em = entityManagerFactory.createEntityManager();

PizzaRepository pizzaRepository = new JpaPizzaRepository( em );

PizzaController pizzaController = new PizzaController( pizzaRepository );
```

Approche intra applicative

Etape 3



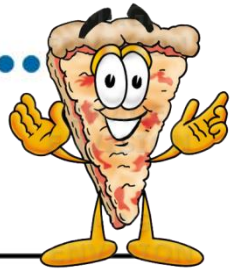
worldline
e-payment services

IoC : Inversion of Control

worldline
e-payment services

Pizza 3000 - Avancement

DIC : Dependency Injection Container



etc

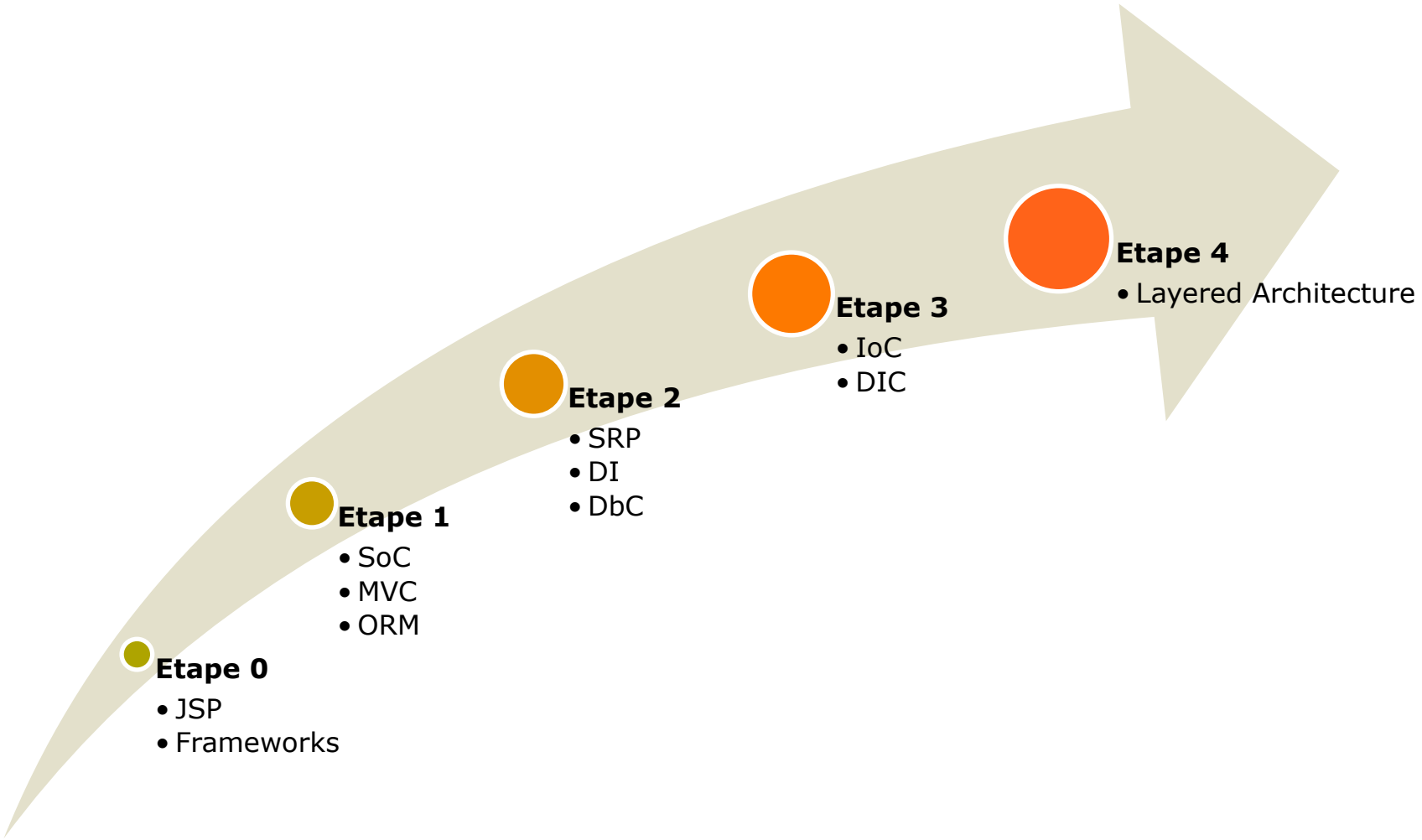
```
EntityManager em = entityManagerFactory.createEntityManager();
PizzaRepository pizzaRepository = new JpaPizzaRepository( em );
PizzaController pizzaController = new PizzaController( pizzaRepository );
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
           http://www.springframework.org/schema/context
           http://www.springframework.org/schema/context/spring-context-3.0.xsd
           http://www.springframework.org/schema/tx
           http://www.springframework.org/schema/tx/spring-tx-3.0.xsd">

    <bean id="emf" class="org.springframework.orm.jpa.LocalEntityManagerFactoryBean">
        <property name="persistenceUnitName" value="urba"/>
    </bean>

    <bean id="em" class="javax.persistence.EntityManager"
          factory-bean="emf" factory-method="createEntityManager" />

    <bean id="pizzaRepository" class="org.losohome.pizza.JpaPizzaRepository">
        <constructor-arg ref="em" />
    </bean>

    <bean id="pizzaController" class="org.losohome.pizza.PizzaController">
        <constructor-arg ref="pizzaRepository" />
    </bean>

</beans>
```

```java
@Service
public class PizzaController {
    private PizzaRepository pizzaRepository;

    @Autowired
    public PizzaController(PizzaRepository pizzaRepository) {
        this.pizzaRepository = pizzaRepository;
    }

    public void list() {
        List<Pizza> pizzas = pizzaRepository.findAll();
        for (Pizza pizza : pizzas) {
            System.out.println(pizza);
        }
    }

    public void add(Pizza pizza) {
        pizzaRepository.create(pizza);
        System.out.println("Pizza '" + pizza.getName() + "' successfully created!");
    }

    public void clear() {
        pizzaRepository.deleteAll();
        System.out.println("All pizzas cleared.");
    }
}
```

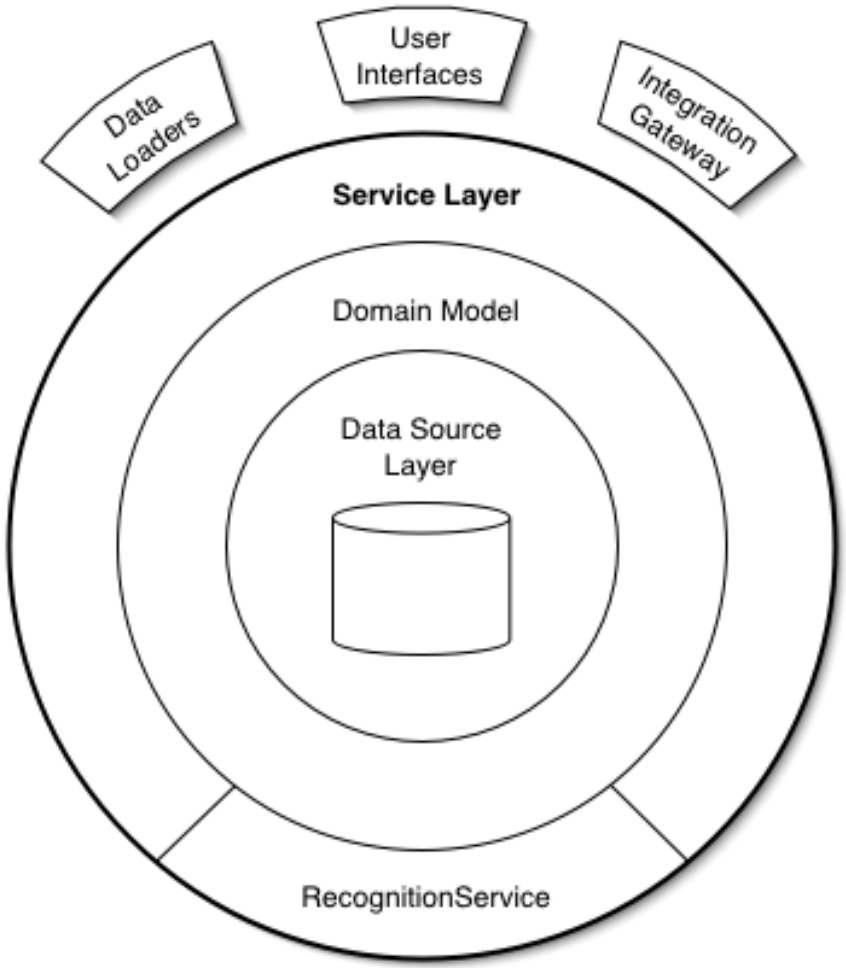
Approche intra applicative

Etape 4

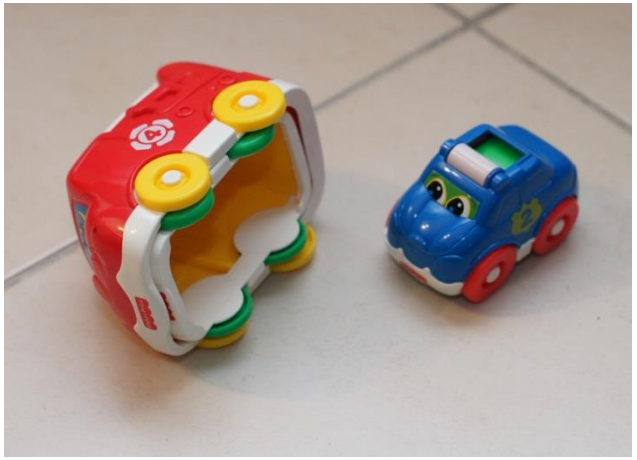


worldline
e-payment services

Layered Architecture



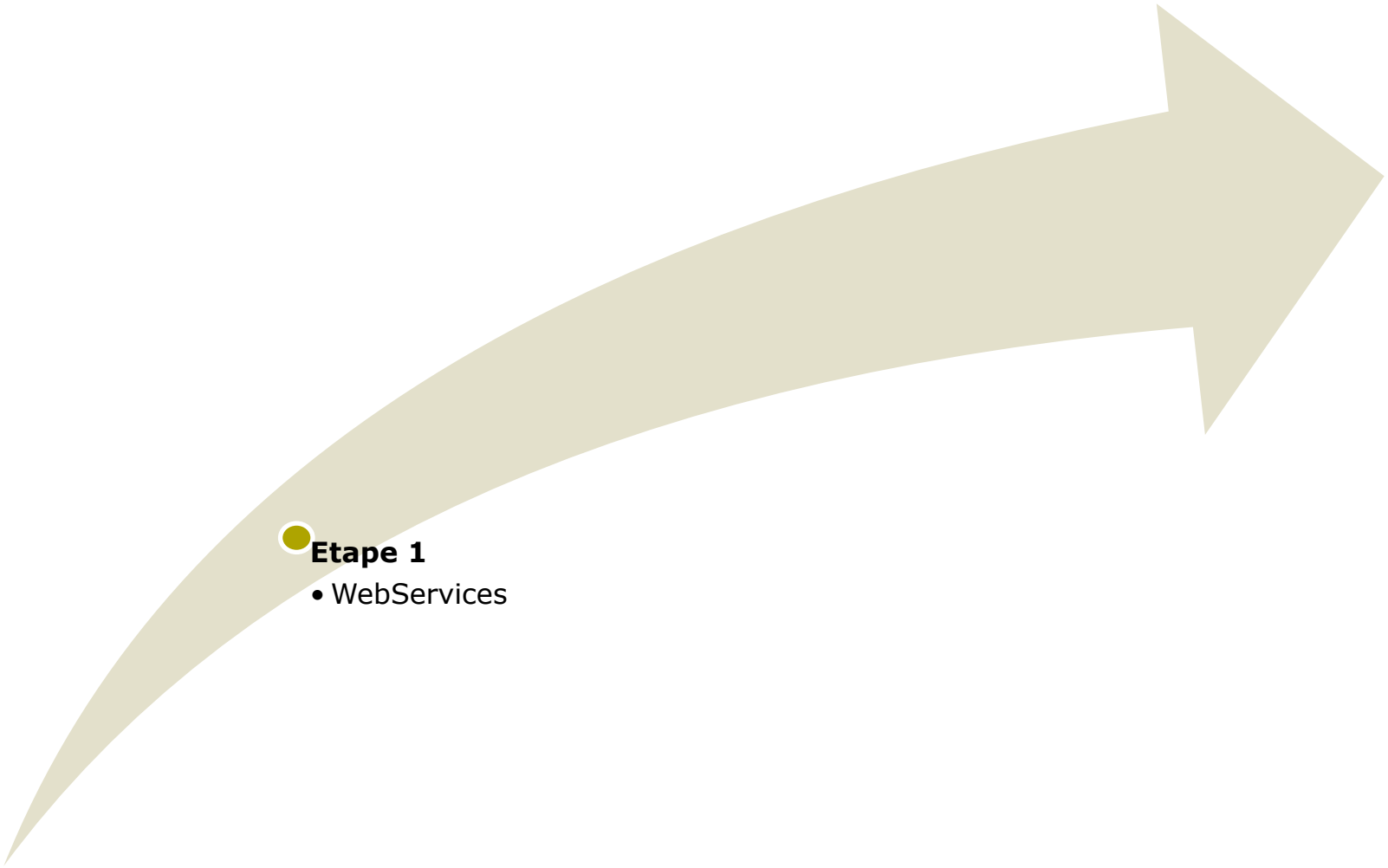
Layered Architecture & Inversion of Control



worldline
e-payment services

Approche inter applicative

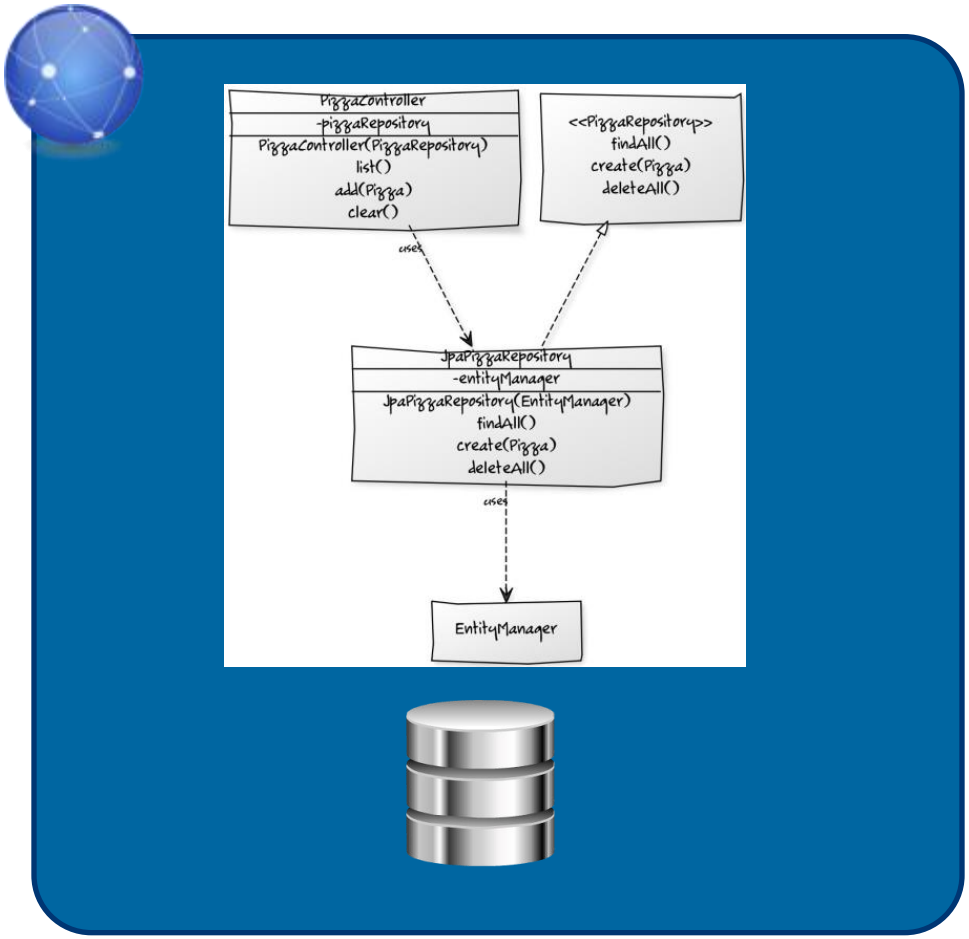
Etape 1



worldline
e-payment services

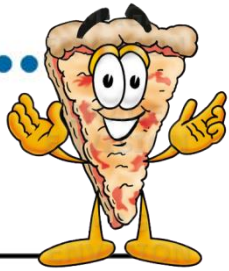
Pizza 3000 - Avancement

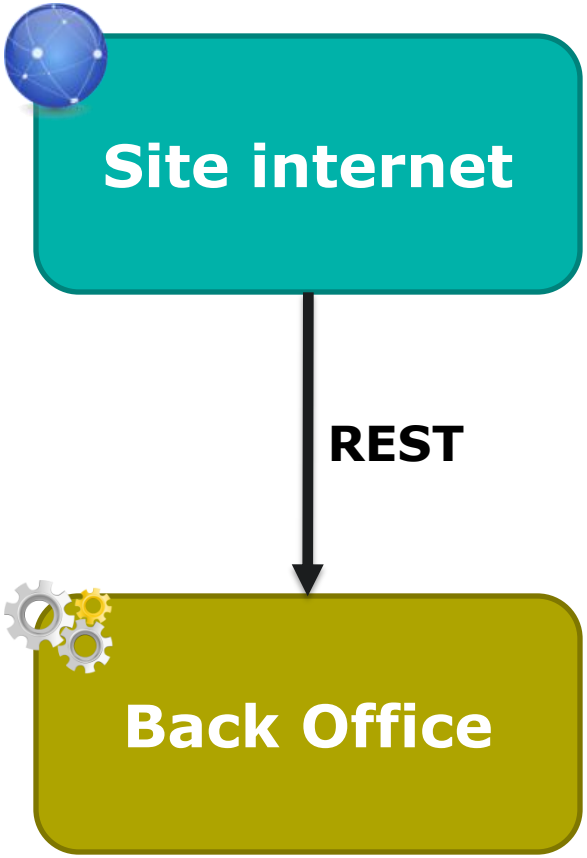
S'ouvrir



Pizza 3000 - Avancement

WebServices





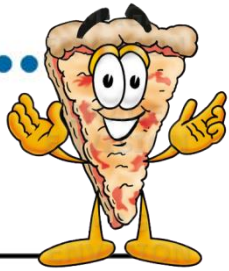
Approche inter applicative

Etape 2

worldline
e-payment services

Pizza 3000 - Avancement

SOA : Service Oriented Architecture



Gestion des appels

SOAP

Site internet

REST

JMS

ERP / CRM

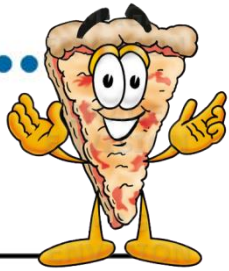
JMS

Back Office

worldline
e-payment services

Pizza 3000 - Avancement

SOA : Service Oriented Architecture

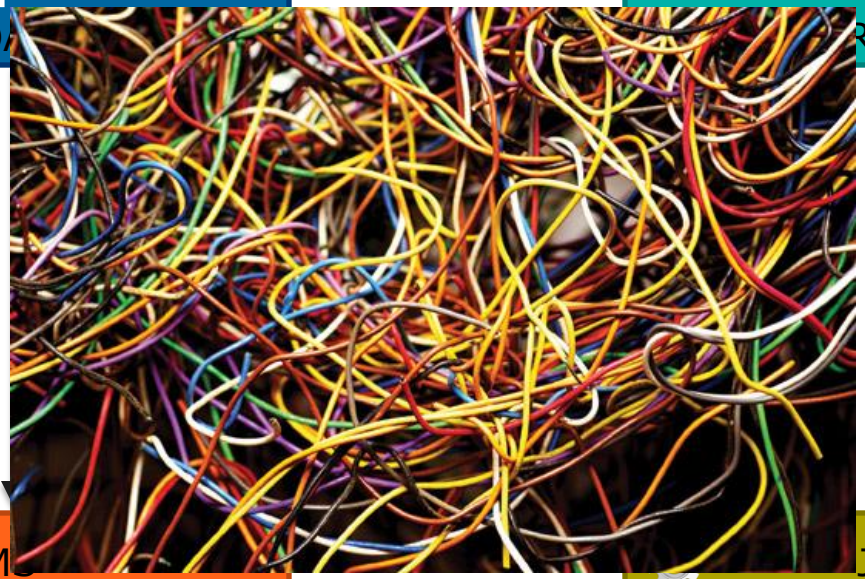


Gestion des appels

SO

Site internet

REST



JM

ERP / CRM

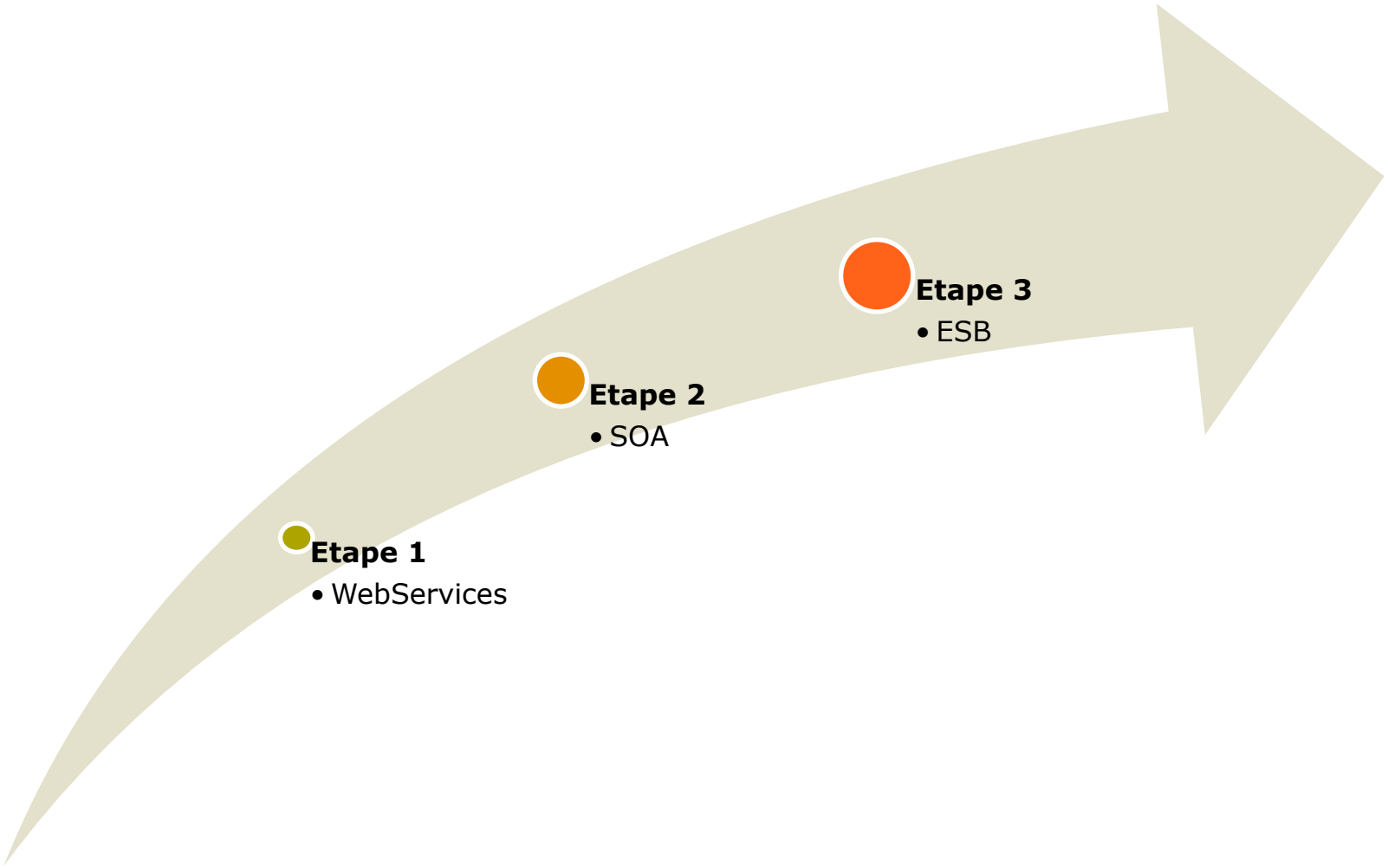
JMS

Back Office

worldline
e-payment services

Approche inter applicative

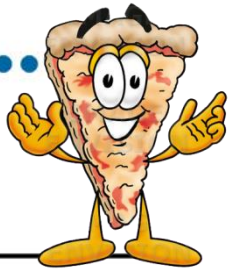
Etape 3



worldline
e-payment services

Pizza 3000 - Avancement

ESB : Enterprise Service Bus

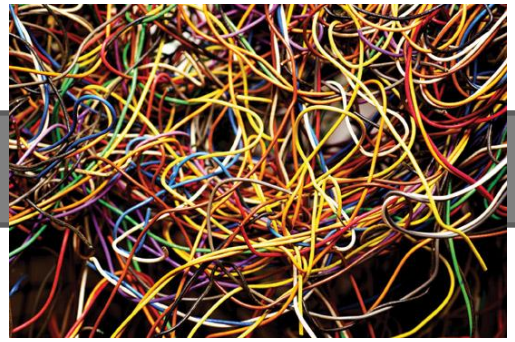


Gestion des appels

Site internet

SOAP

REST



JMS

JMS

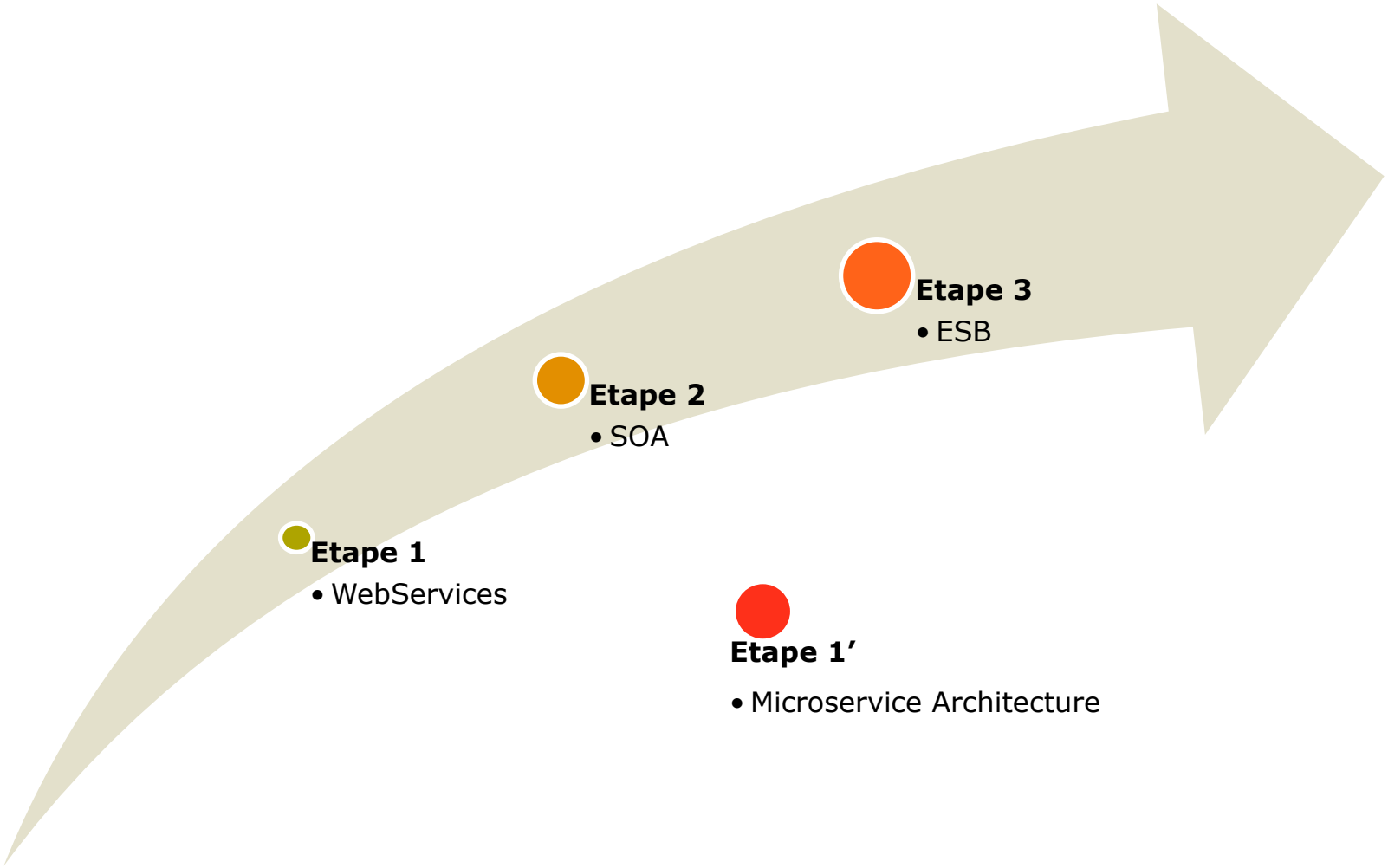
ERP / CRM

Back Office

worldline
e-payment services

Approche inter applicative

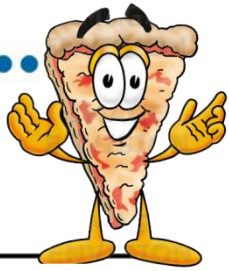
Etape 1’



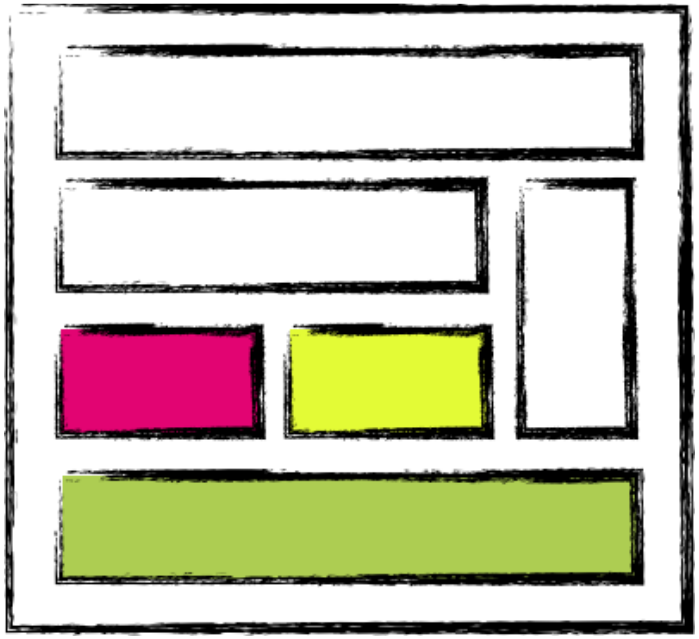
worldline
e-payment services

Pizza 3000 - Avancement

Microservice Architecture

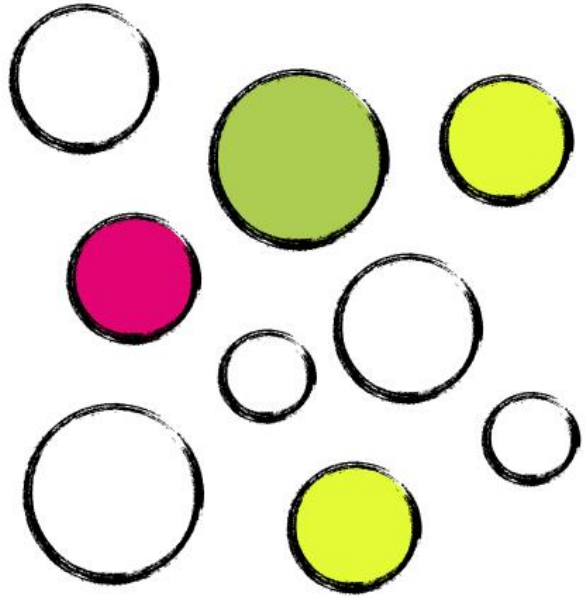


Back Office



Vision monolithique

Back Office



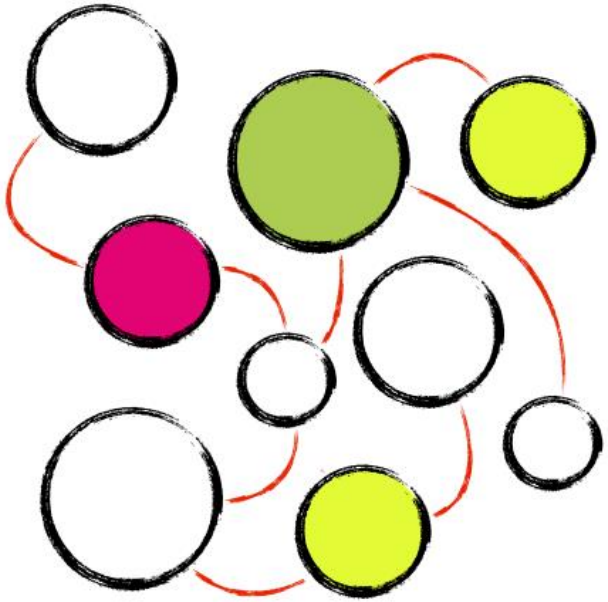
Vision microservice

1 des problématiques : l'interconnexion

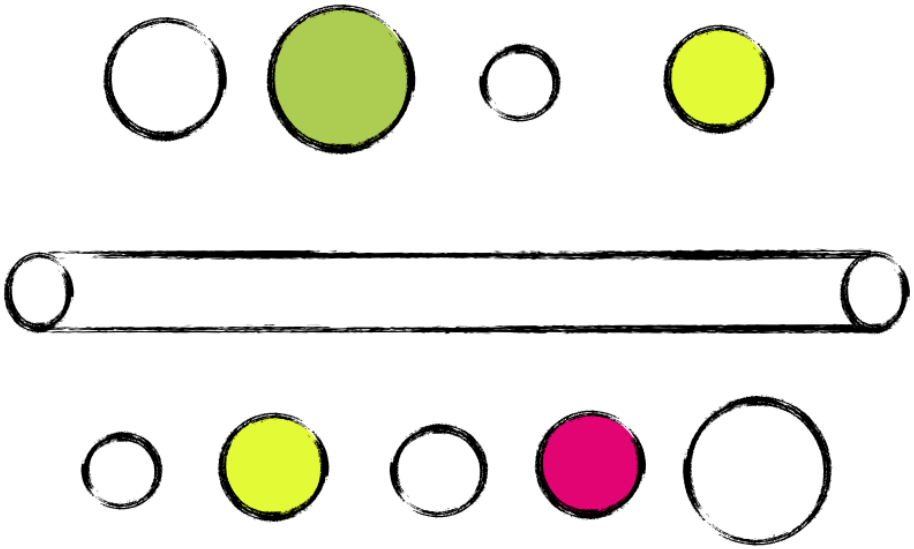
worldline
e-payment services

Pizza 3000 - Avancement

Microservice Architecture

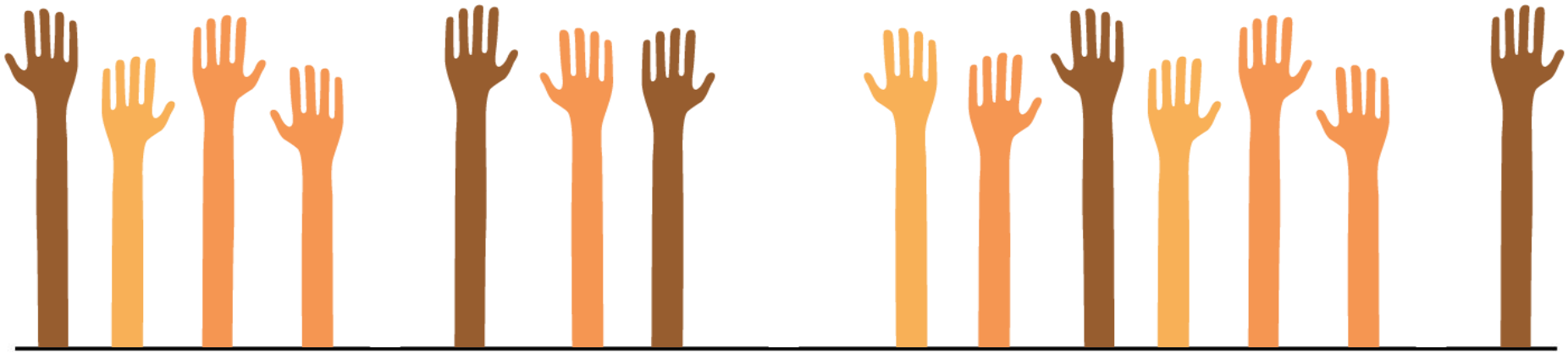


Fortement couplé



Bus de message

Question ?



worldline
e-payment services

Crédits

Suivez-nous

4 : Theory
5 : Word nature in dictionary
10 : The human gait
12 : Goals
46 : Xkcd - Exploit of a Mom
51 : Bill Brookman

- **Suivez le travail d'iLabs**
  - http://resthub.org
  - https://github.com/resthub

Merci