Master Informatique et Systèmes

Architecture des Systèmes d'Information

03 – Architecture Logicielle et Technique

Damien Ploix

2014-2015

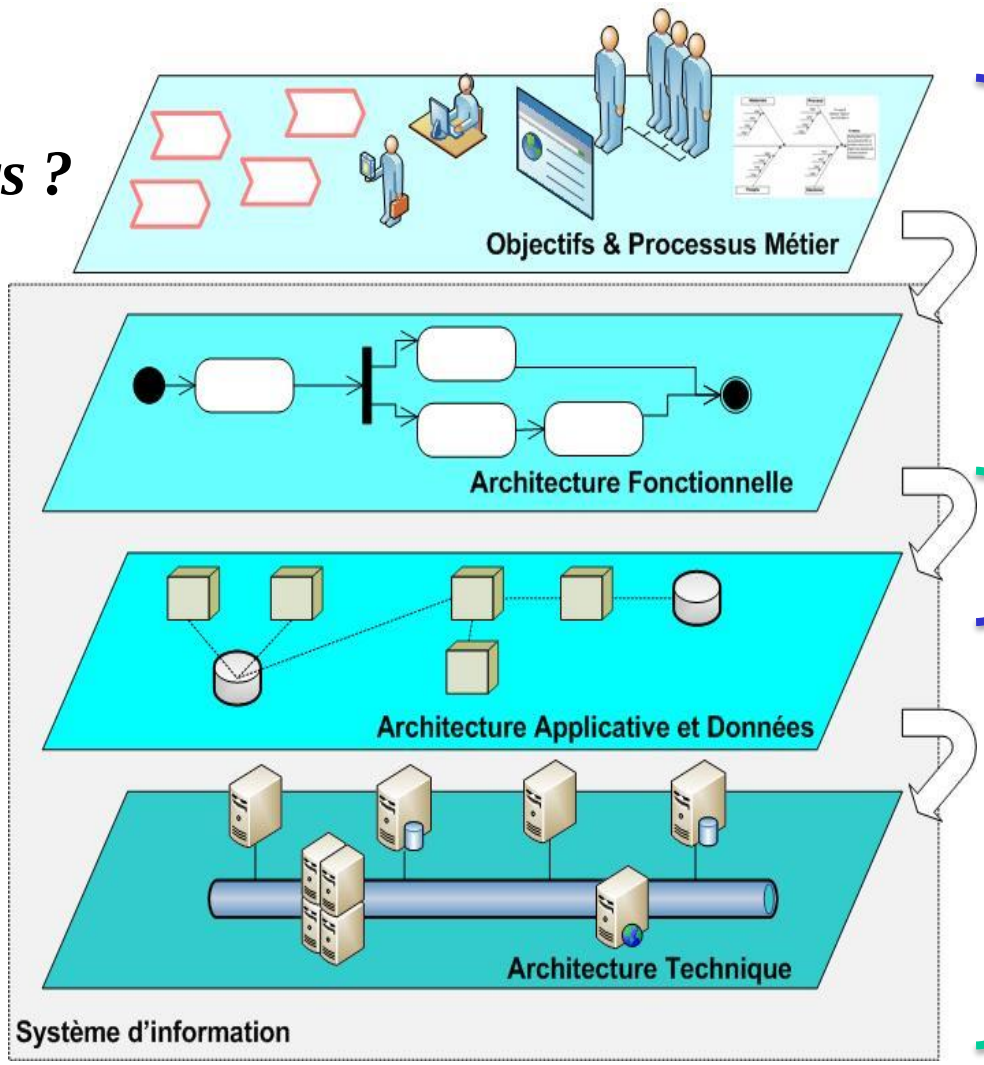
Démarche d'architecture SI : structuration en vues

Quels métiers ?

Quoi?

Comment?

Avec quoi?



Urbanisme

Architecture du SI

Architecture applicative

Architecture Logicielle

Architecture Technique

Démarche d'architecture Logicielle et Technique

- **Les architectures logicielles et techniques sont en « couplage fort » : les choix fait sur structurent les choix possible pour l'autre. En sus des problématiques d'intégration, leurs contraintes sont de garantir :**
  - **Sécurité**
  - **Disponibilité**
  - **Performance**

- **Architecture Logicielle**
  - **Elle structure les solutions en œuvre dans une application les mieux à même de répondre aux spécifications fonctionnelles**
  - **Elle structure et décompose de façon logique chaque partie de l'application via les notions et concepts de découpage en couches, composants, framework et design patterns**

- **Architecture Technique**
  - **Elle structure les solutions d'infrastructure technique du SI**
  - **Elle décrit et organise**
    - les différents moyens matériels (serveur, poste client …)
    - les logiciels de base (systèmes d'exploitation, SGBD, AGL …)
    - les moyens de communication (réseaux, middleware…)

Démarche d'architecture Logicielle et Technique

- **Définir les « solutions » d'architecture logicielle et technique pour chacun des *blocs applicatif* :**
  - **Identifier les référentiels de solution applicable**
  - **Définir le modèle d'architecture en couches et en tiers à mettre en œuvre, leur motif de conception, les librairies, framework et outils à utiliser**
  - **Logiciels de base (système d'exploitation, SGBD, middleware, serveurs d'applications, annuaires…) et leur mode d'intégration**
  - **Plates-formes matérielles (poste de travail, serveurs départementaux, serveurs d'entreprises, stockage, …) et leur dimensionnement**
  - **Réseaux et télécommunications (réseau locaux (LAN) ou longues distances (WAN), débit, switch, routeurs, proxies, firewalls, …) et leur dimensionnement**

Plan du chapitre

Modèle d'architecture Logicielle : FrameWork

- **Choix du Design Pattern global sous jacent (et ses variantes)**
  - **MVC (Push / Pull)**
    - Décomposition en modèle, vue et contrôleur
    - MVC Push :
      - Le contrôleur interprète les actions de la vue et lui « pousse » les informations
      - Exemple : la mise à jour de l'interface après la saisie d'une information
    - MVC Pull :
      - La vue va « tirer » les informations du contrôleur
      - Exemple : liste déroulante des fournisseurs de tel ou tel produit
  - **Orienté composants**
    - Décomposée en *composants* métier regroupant l'ensemble des éléments le concernant (vue, données, logique métier)

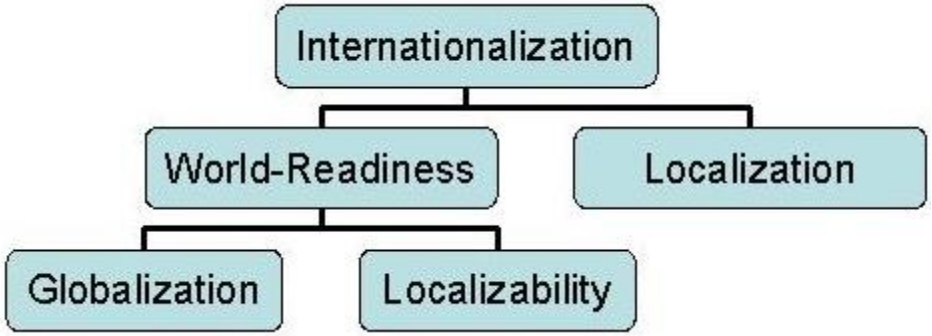
Modèle d'Architecture logicielle

- **La sécurisation d'une solution logicielle est fonction de son implémentation**

- **Deux cas possibles :**
  - **« sur étagère » :**
    - progiciel éditeur dont l'architecture de composants est connue
      - L'éditeur publie alors les *contraintes d'architecture* permettant l'intégration du progiciel
  - **« spécifique » :**
    - Sans FrameWork :
      - Modèle à 5 couches
      - Micro services
    - Dépendante du FrameWork utilisé
      - FrameWork public
      - FrameWork d'entreprise
    - Dépendante de contraintes particulières :
      - Volumétrie (big data)
      - Disponibilité (gestion de la HA)
      - D'urbanisation/d'intégration (MDM, briques, …)
      - …

Modèle d'architecture logicielle : FrameWork

- ***Flexibilité* et intégration**
  - **Internationalisation**
    - i18n, L10n, g11n

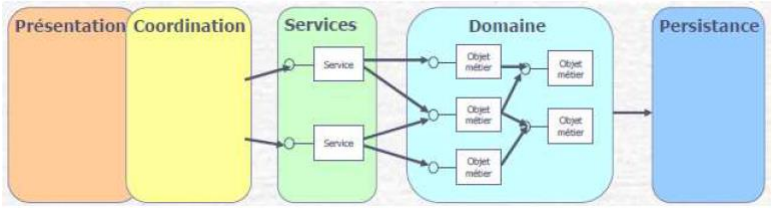


  - **Type de média (web, mobile, tablette, lunettes (VR – OR))**
  - **Templates**
    - Existence de modèles vs librairies
- **Gestion de la sécurité (cf cours SSI)**
  - **Gestion des identités**
  - **Gestion des authentifications**
  - **Gestion des autorisations**
  - **Sécurisation du transport des informations**

Modèle d'architecture logicielle : FrameWork

- **Gestion des données**
  - **Validation des données (M → V et V → M)**
    - Implémentation des règles de « qualité », par exemple issues de l'analyse MDM
  - **Interface entre la couche domaine et la couche persistence**
    - ORM :
      - Interface entre le modèle Objet et le modèle relationnel
    - Utilisation des répertoires virtuels et de services externes
    - Gestion de la persistance
      - Des données (hibernate)
      - Des pages (cache)
- **Outillage du processus de développement**
  - **Testing**
    - JUnit, …

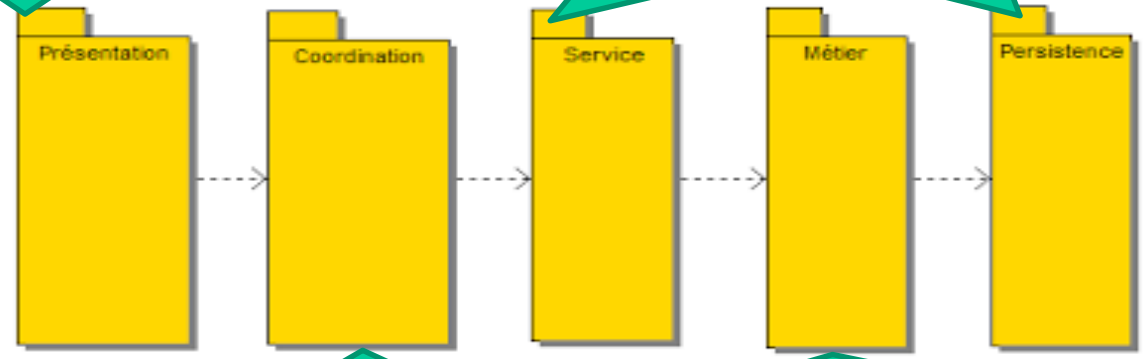
Modèle d'Architecture logicielle : modèle à 5 couches J2EE



Vue(s)

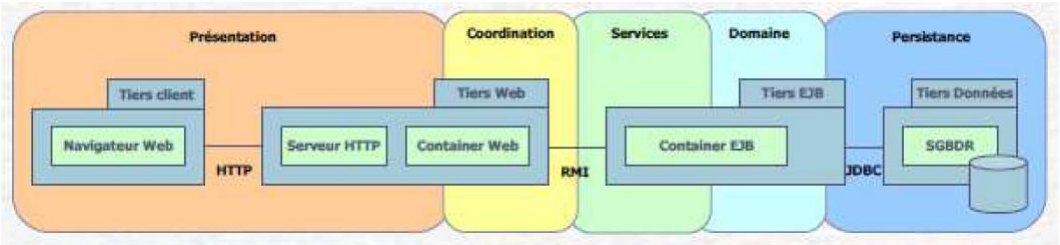
SOA (RESTfull)
EJB (Sessions)

Accès aux données



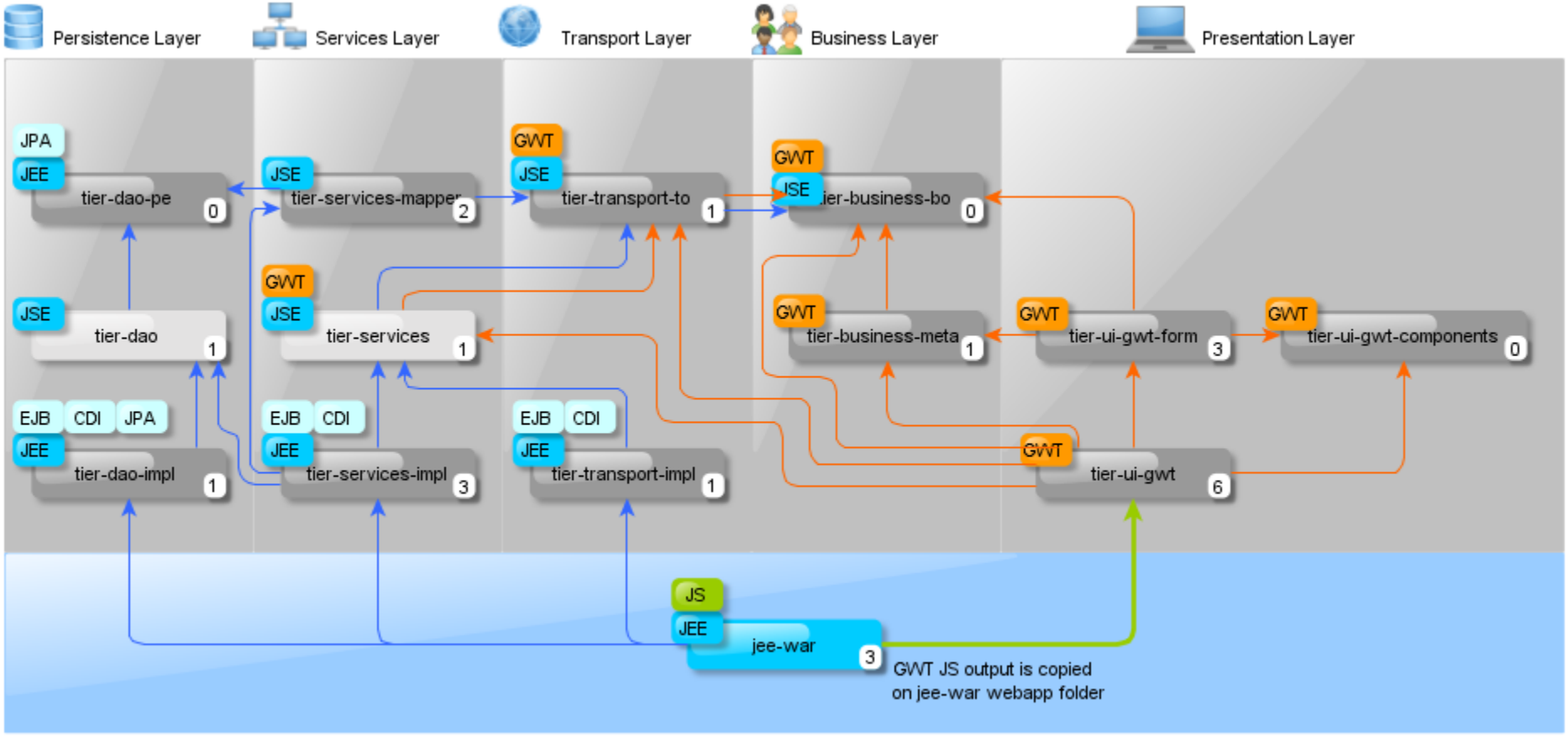
Contrôleur

EJB
(dont sessions internes)



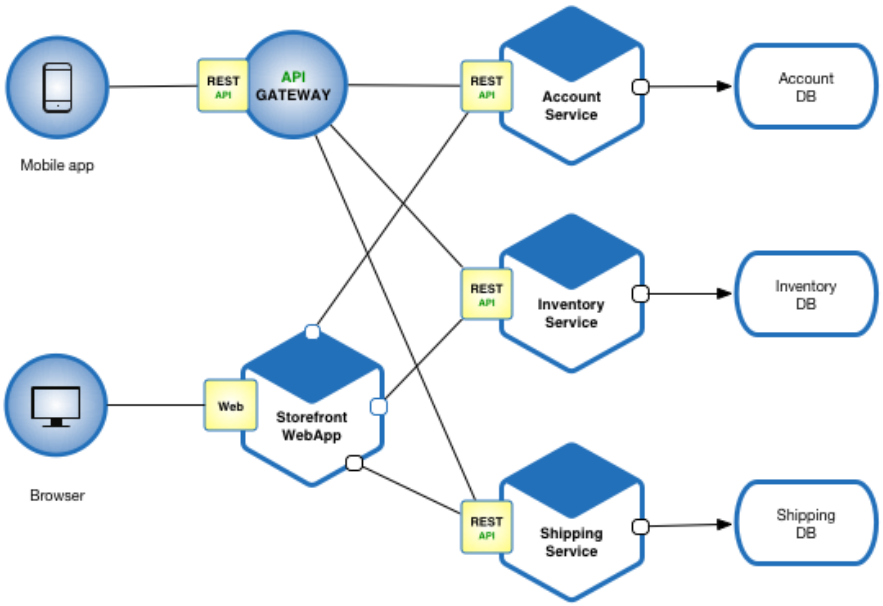
Modèle d'Architecture logicielle : modèle à 5 couches J2EE

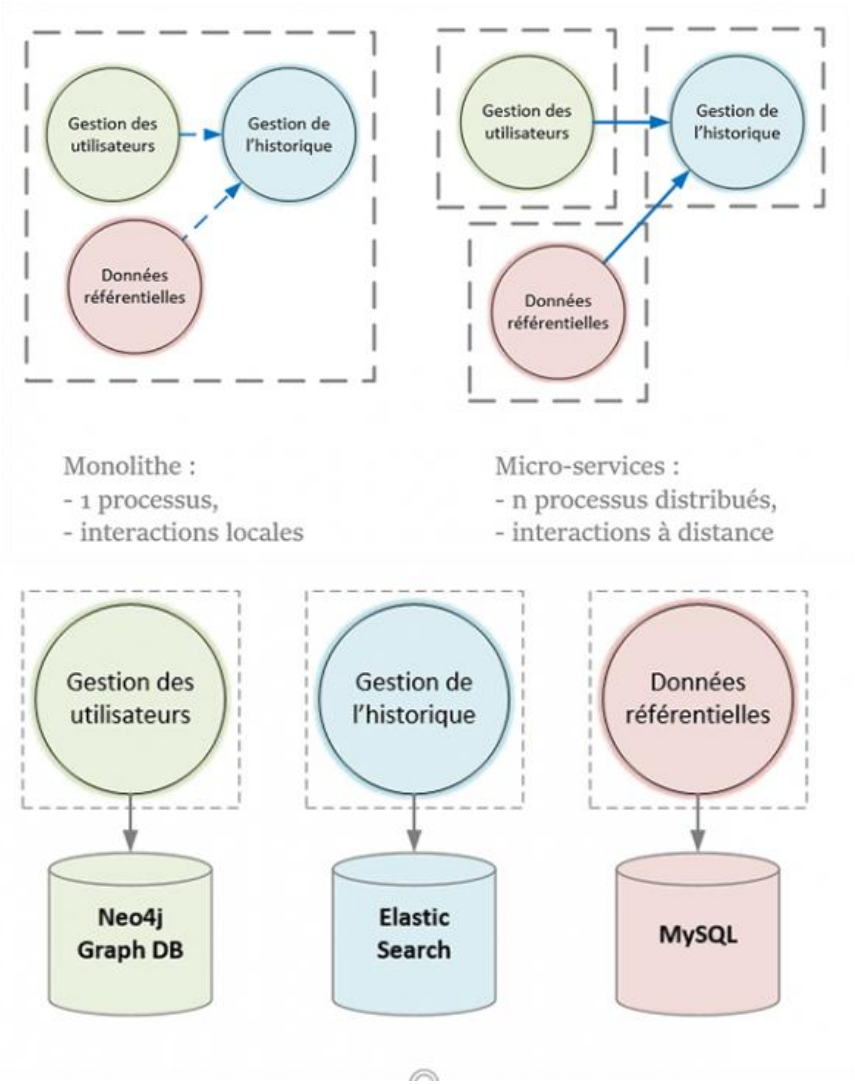
- **Structuration du code en artefacts Maven**
- **Composant JSE , partie serveur. Utilise les normes EJB, CDI**
- **Composant GWT (Google Web Toolkit – AJAX) : partie client**



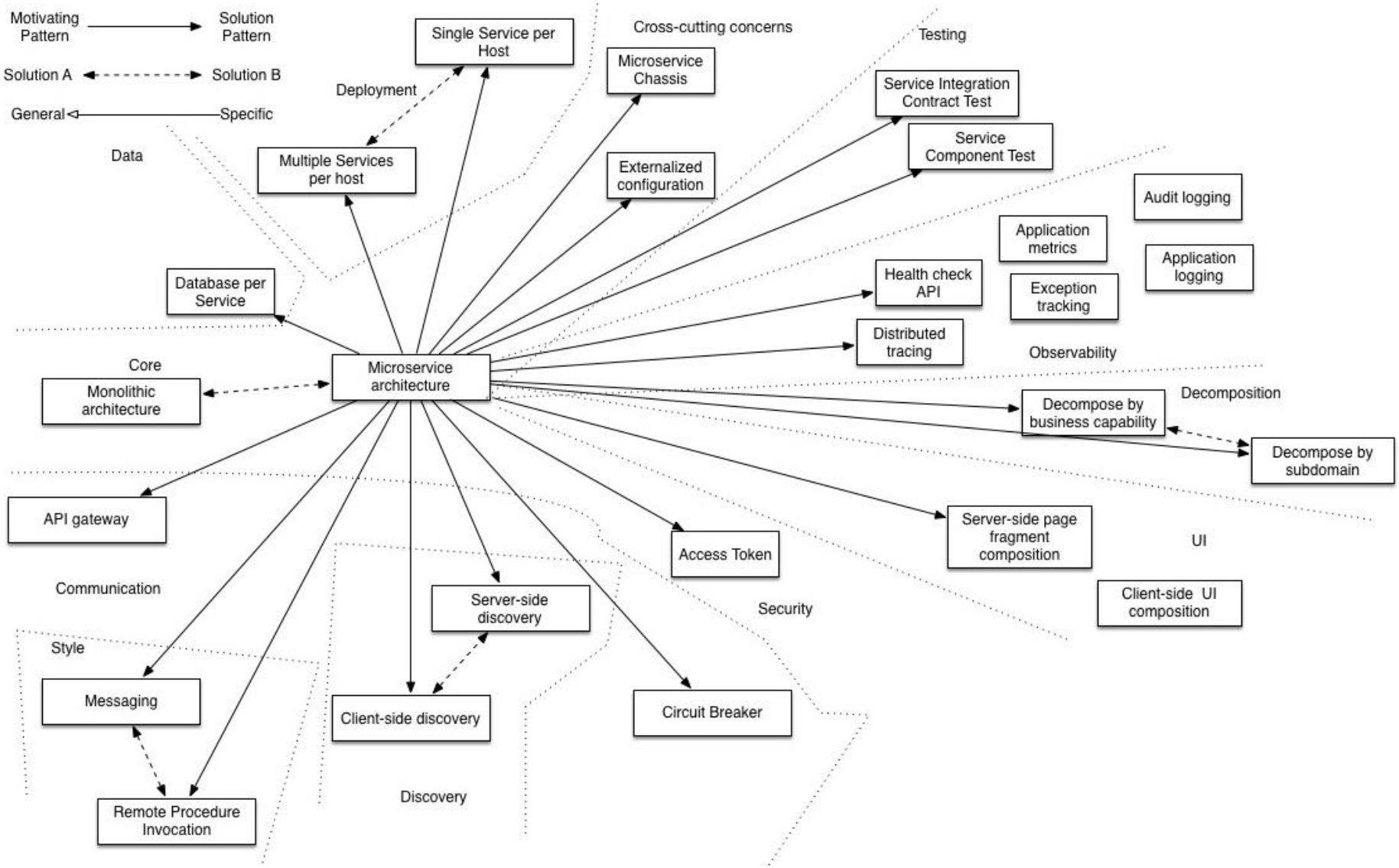
- **Source : *http://lyonjee.blogspot.fr/2012/03/le-diagramme-ci-dessus-presente.html***

Micro services (principe)

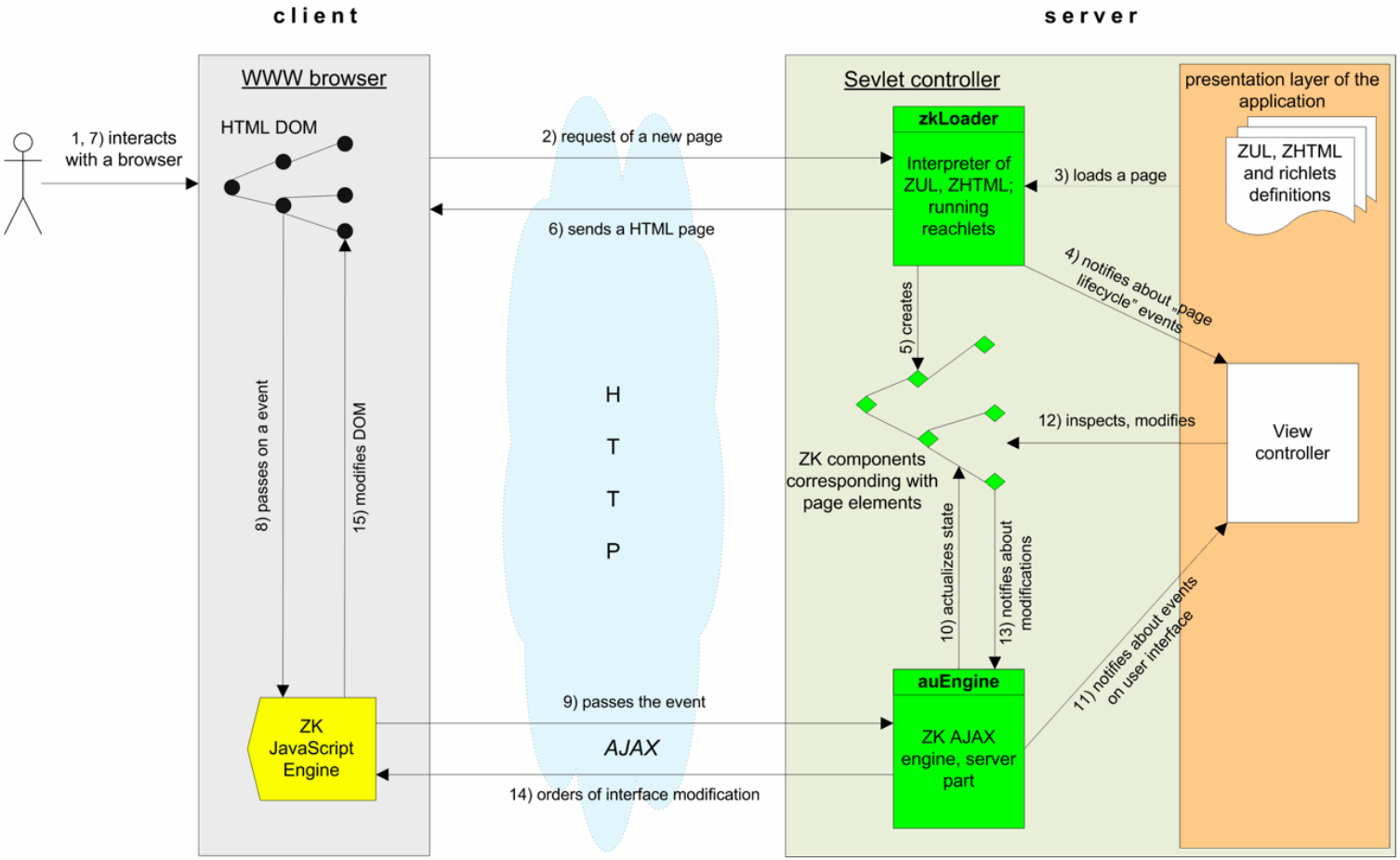




Décomposition des Microservices (http://microservices.io/patterns/microservices.html)



Modèle d'architecture logicielle : exemple Zk (http://www.zkoss.org/)



Modélisation(s)

- **Micro services :**
  - https://www.technologies-ebusiness.com/enjeux-et-tendances/architectures-micro-services-objectifs-benefices-defis-partie-1 **(et -2)**
  - http://microservices.io/patterns/microservices.html
- **OO (from Stupid to Solid) :**
  - http://williamdurand.fr/2013/07/30/from-stupid-to-solid-code/

Plan du chapitre

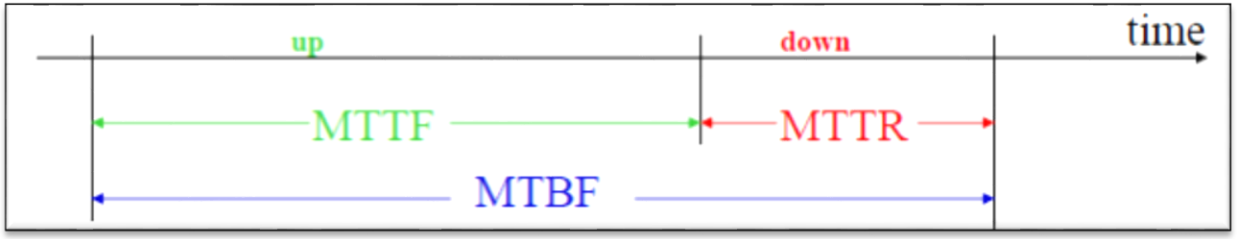
Contraintes et solution de disponibilité et d'intégrité

Disponibilité : définition du SLA

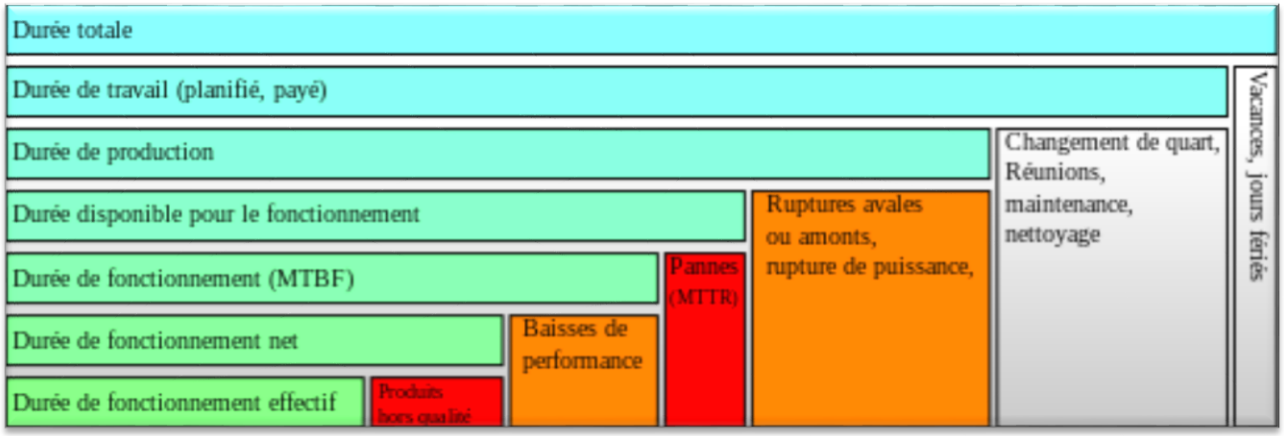
- **Service Level Agreement :**
  - accord *contractuel* de niveau de service
- **Définit en fonction des éléments à prendre en compte dans le calcul des *indisponibilités* :**
  - Plage horaire de calcul ?
  - Temps des opérations de maintenance ?
  - Temps de service dégradé ?
  - Quel outil est utilisé pour mesurer le temps de disponibilité ?
  - Quelle solution *métier* pour palier l'absence de l'outil informatique ?

Mesure de la disponibilité

- **Calcul de la disponibilité :**
  - **MTTF : Mean Time To Failure** — durée de fonctionnement effectif
  - **MTTR : Mean Time To Recover** — durée de rupture du fonctionnement
  - **MTBF : Mean Time Between Failures** — durée totale
  - **Disponibilité = MTTF/MTBF** — (durée de fonctionnement) / (durée totale)



- **Elle doit être relativisée par rapport à un « service » et donc être ramené à un besoin**
  - Question : quelle durée est prise en référence du bon fonctionnement / du disfonctionnement.

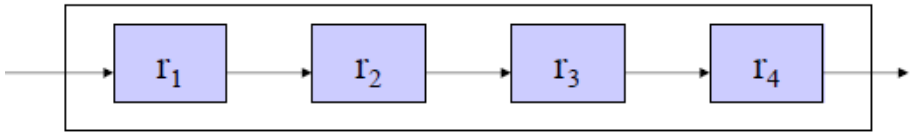


Mesure de la disponibilité

Disponibilité	Indisponibilité (minutes / an)	Niveau de disponibilité
99.99999%	0,053min (3 secondes)	Ultra disponible
99.9999%	0,53 min (31 secondes)	Très hautement disponible
99.999%	5,26 min	Hautement disponible
99.99%	52,6 min	Tolérant aux pannes
99.9%	526 min (9 heures)	Bon niveau de service
99.0%	5 256 min (3, 65 jours)	Service fournit
90.0%	52 560 min (36,5 jours)	Heures Ouvrées

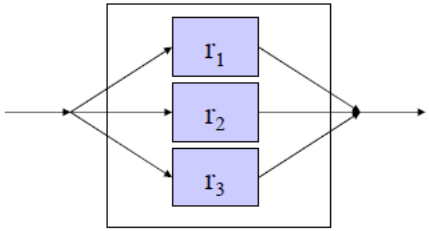
Mesure de la disponibilité

- **La disponibilité d'un système dépend de l'ensemble de ses composants :**



***Disponibilité du système* = $\prod r_i$**

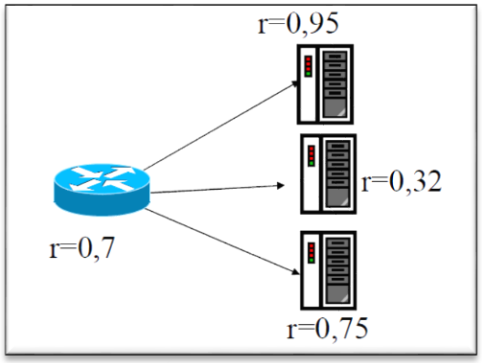
Tous les composants doivent être disponibles



***Disponibilité du système* = $(1 - \prod(1-r_i))$**

Le système est fiable tant que tous les éléments ne sont pas indisponibles

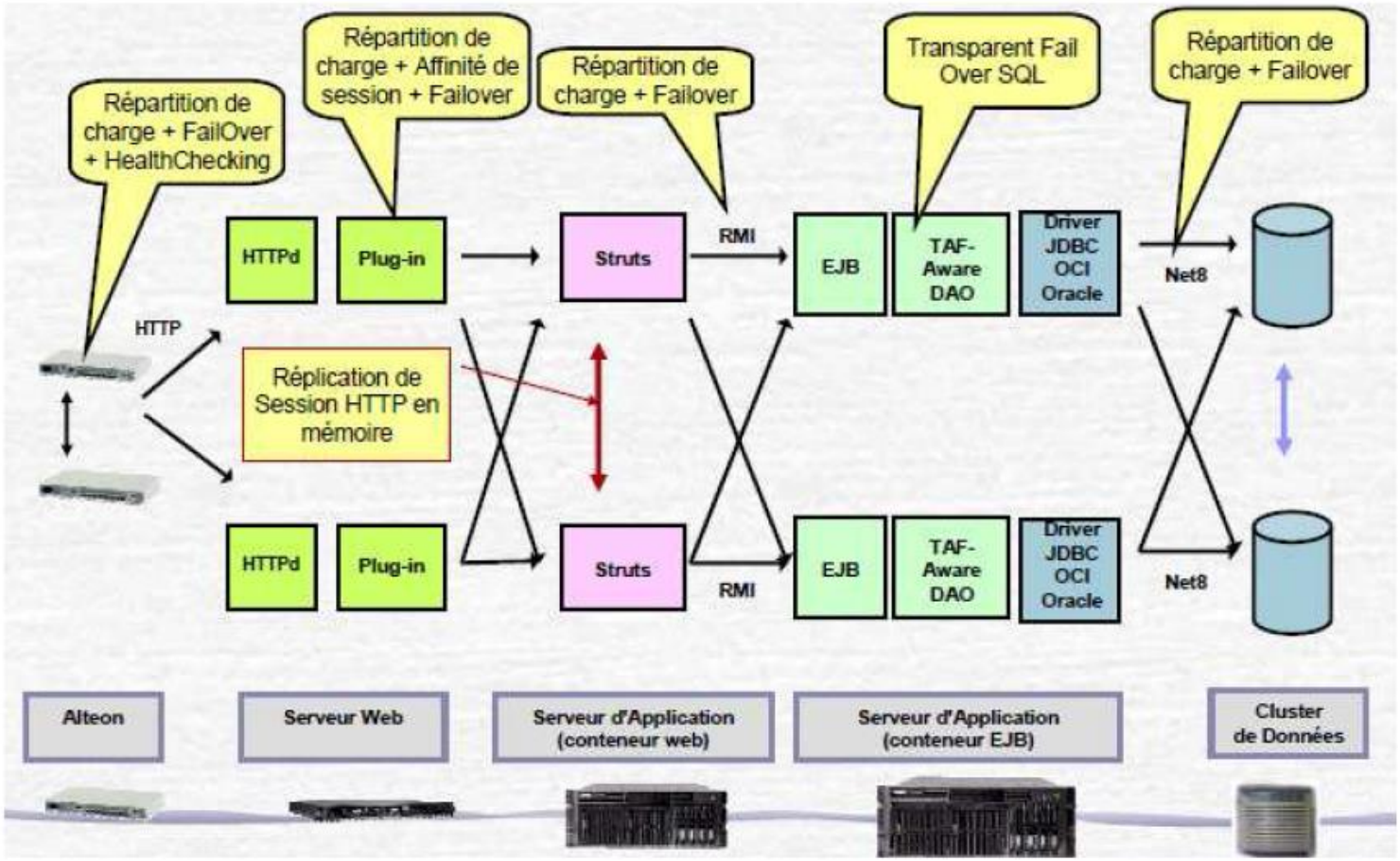
- ***Calculez le taux de disponibilité du système suivant :***



- **Ces calculs ont amené la généralisation de la mise en place de systèmes de redondance dans l'architecture logicielle et technique des applications.**

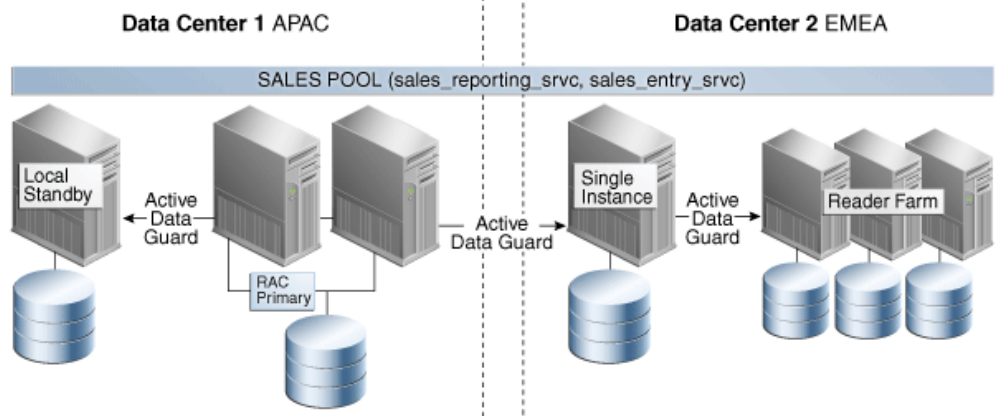
Contraintes et solution de disponibilité et d'intégrité

- Architecture web redondée
- Article conseillé sur le clustering J2EE : http://www.theserverside.com/news/1364410/Under-the-Hood-of-J2EE-Clustering



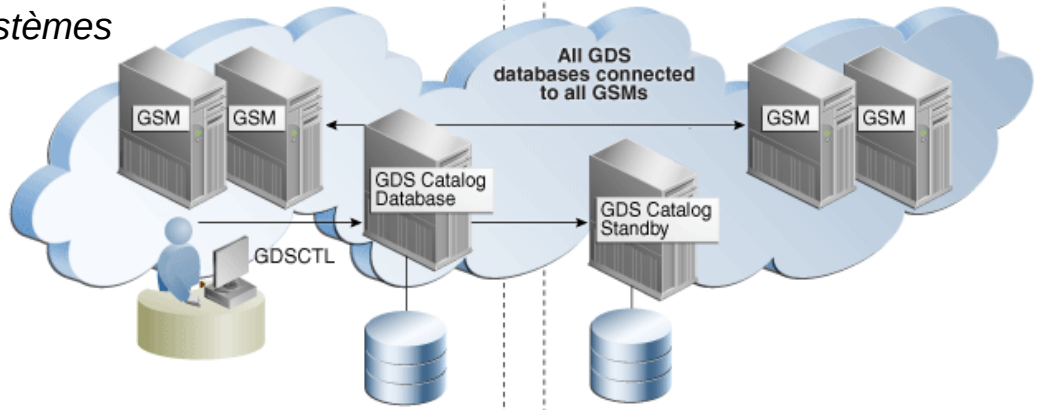
Contraintes et solution de disponibilité et d'intégrité : HD Oracle

Système commercial : haute disponibilité haute performance



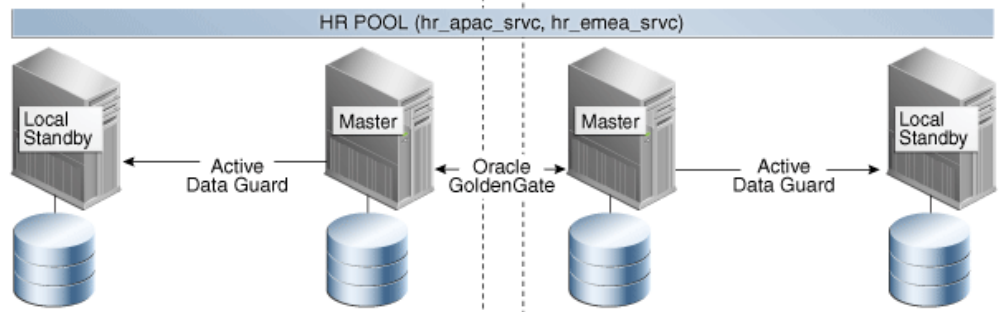
Production RW Locale
Production RO Distante
Standby Locale
Standby Distante

Administration des systèmes



Global Service Management
Global Data Service

Système RH : haute disponibilité



Production RW Répartie
Standby sur les deux DC

Plan du chapitre

Architecture L&T : contrainte de performance TP

- **Caractéristique de la performance d'un système transactionnel :**
  - **Utilisateurs concurrents = utilisant le système « en même temps »**
  - **Utilisateurs connectés = utilisateurs ayant demandés une page les 10 à 15 dernières minutes**
  - **Utilisateurs déclarés = utilisateur pouvant se connecter.**
  - **Ratio d'utilisation (RU) = utilisateurs concurrents / utilisateurs déclarés**
    - Pour les applications support, le RU mesuré est de l'ordre de 1 à 2 %
    - Le RU utilisé pour garantir la performance de ces systèmes est de 10 %
    - Pour les applications « cœur de métier », le RU peut être plus important et doit faire l'objet d'analyse précise.
  - **Les temps seront mesurés sur un scénario de navigation :**
    - Temps moyen d'affichage d'une page (réponse à une requête)
    - Temps maximum d'affichage d'une page
- **Le dimensionnement des composants physiques est donné par les constructeurs de matériel (évolution constante)**

Architecture L&T : performance

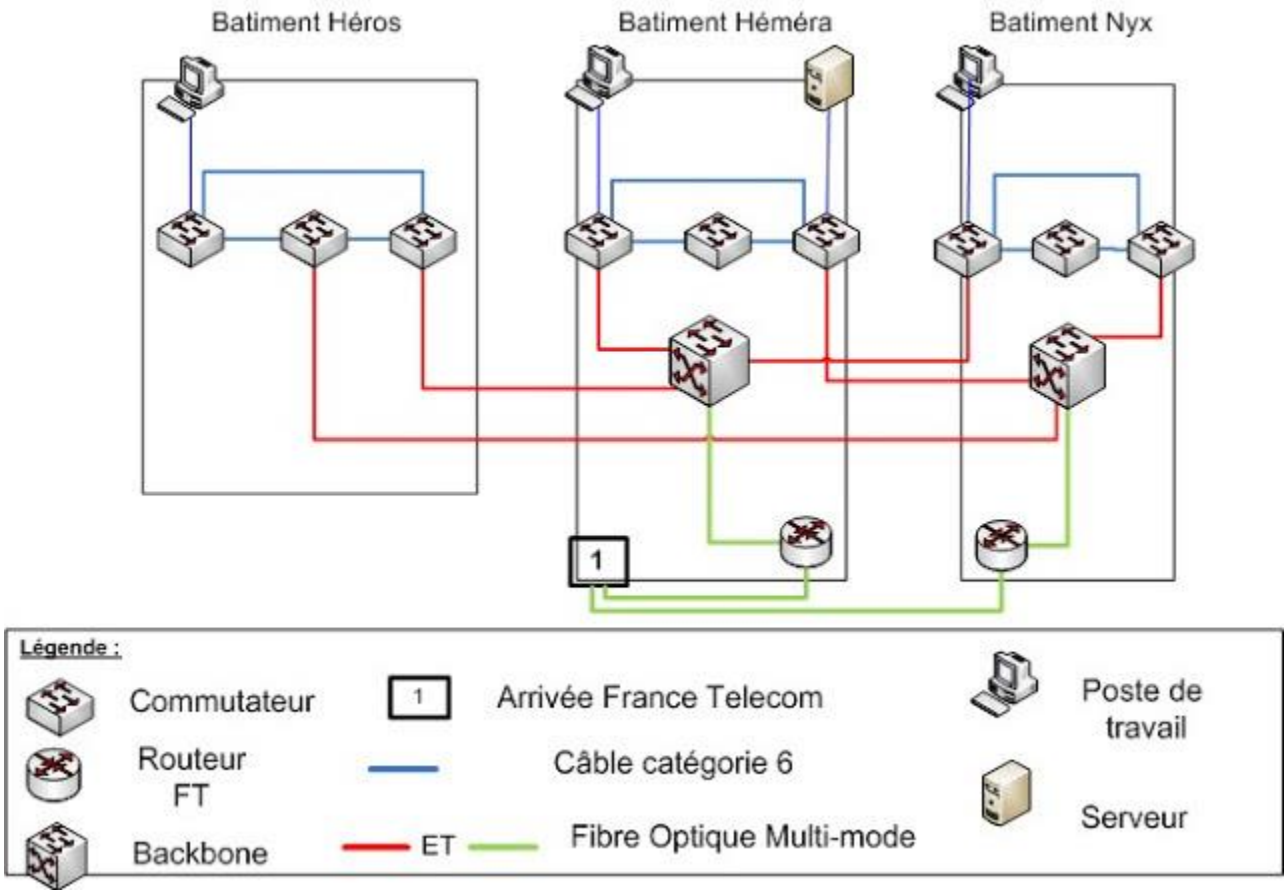
- **Caractéristiques d'architecture impactant la performance**
  - **Planification et solution de gestion de la montée en charge (scalabilité) en fonction des traitements :**
    - Horizontale : ajout de nouveaux serveurs de traitement
    - Verticale : ajout de la capacité au serveur de traitement
  - **Mécanismes de cache (navigateur, proxy, serveur web, cache serveurs d'applications, cache base de données…)**
  - **Gestion de la réplication :**
    - Gestion de l'affinité de session
    - Haute disponibilité RAC/GG
    - Modalité de synchronisation des bases de secours (HP/HD/HA)

Plan du chapitre

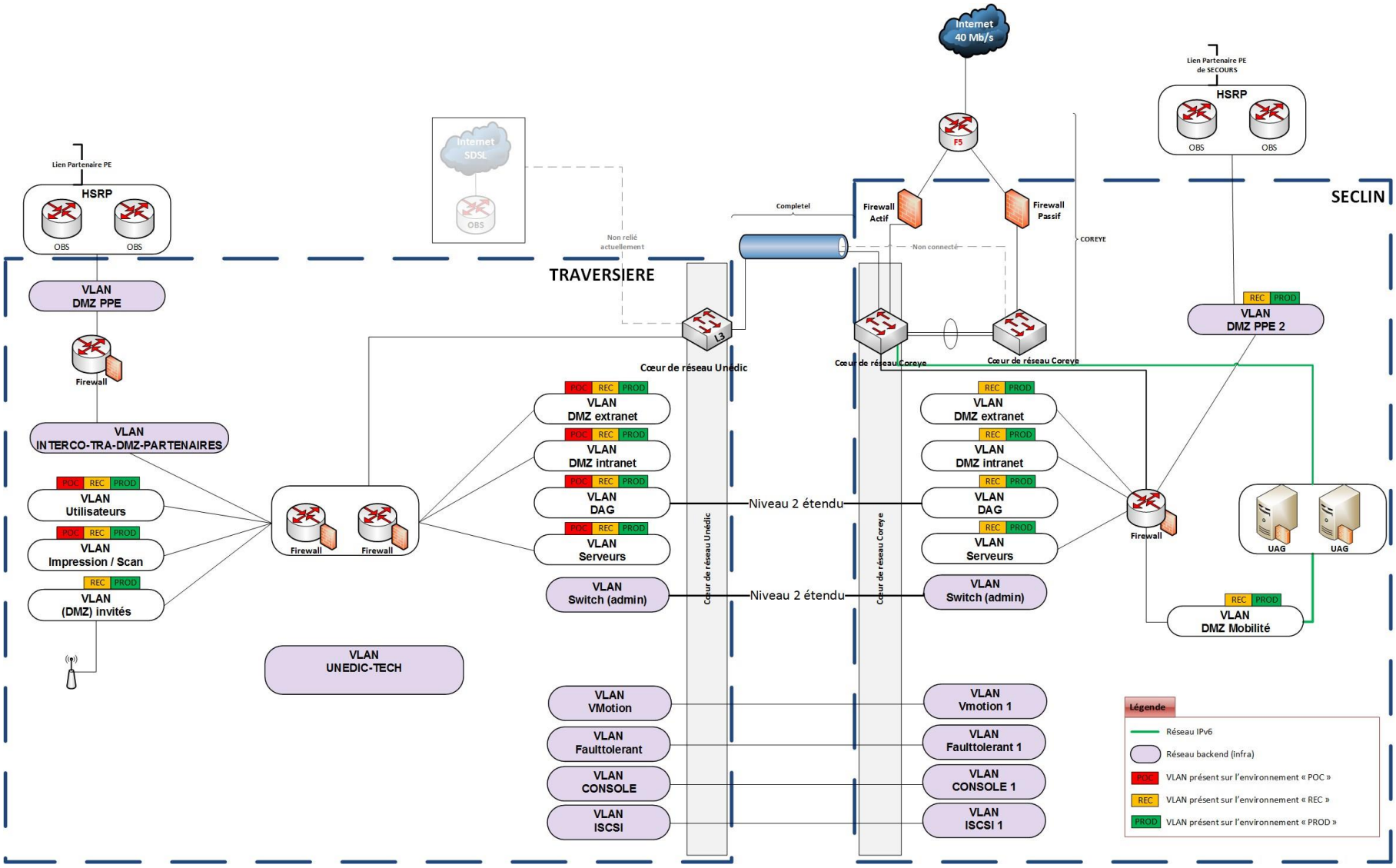
Hebergement

- **Contraintes sur l'architecture d'hébergement**
  - **Protection des zones réseau**
  - **Hébergement externe (cloud)**

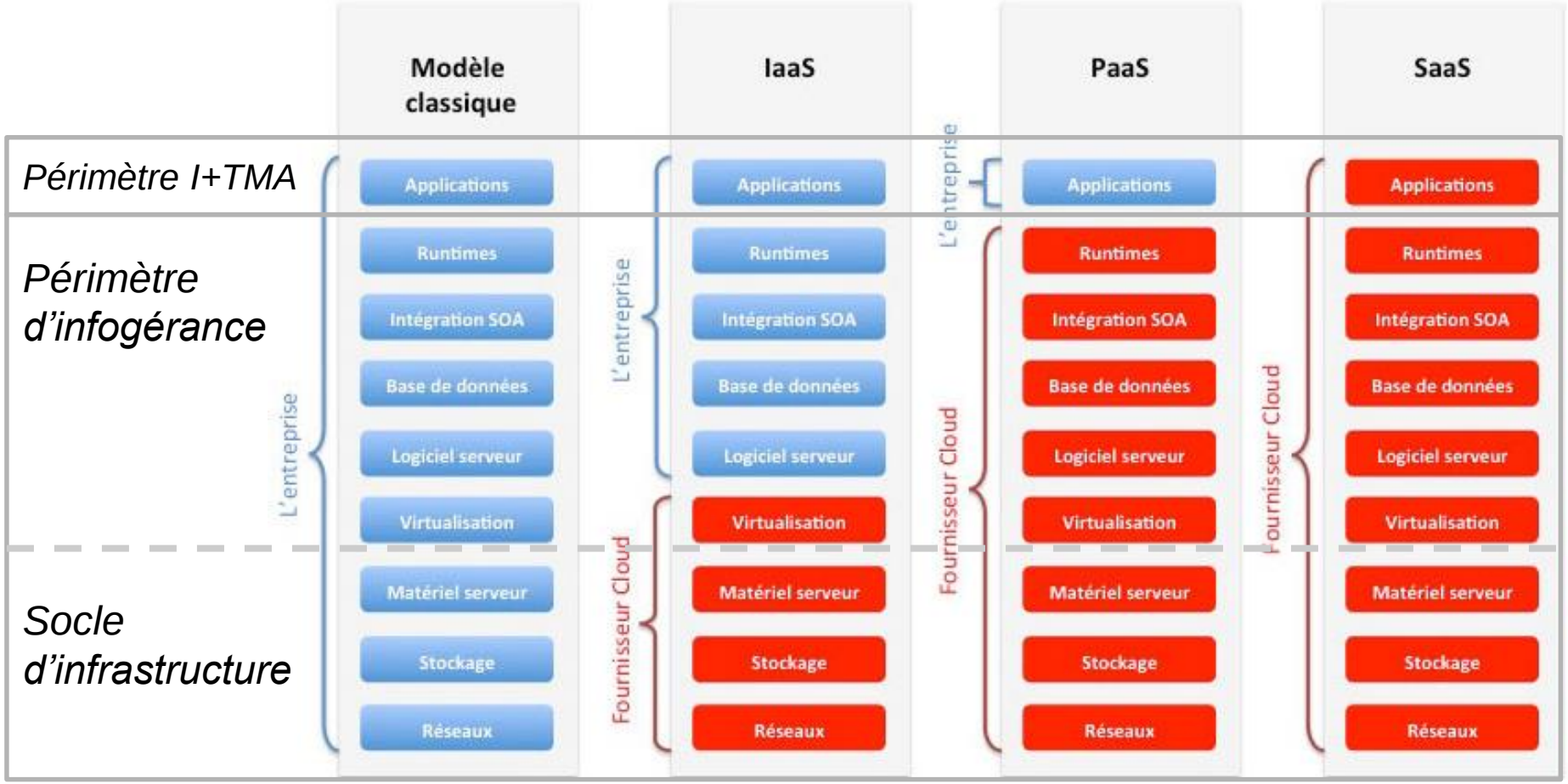
Architecture réseau



Protection des zones réseau : exemple



Modèles d'hébergement externe (cloud)



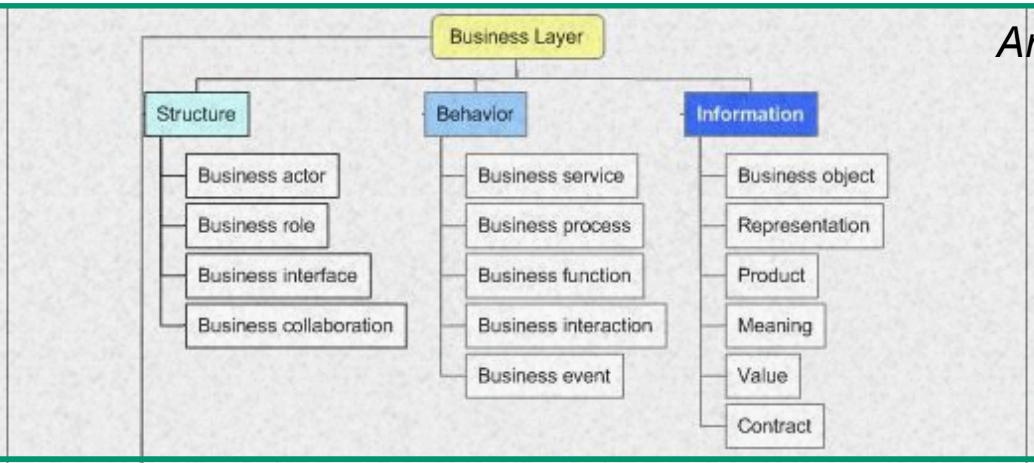
Plan du chapitre

Démarche d'AA : méta-modèle Archimate

Quels métiers ?

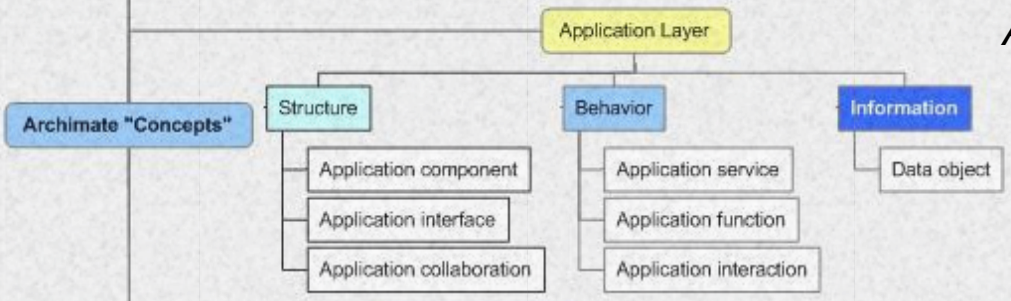
Quoi?

Architecture d'entreprise
Urbanisme



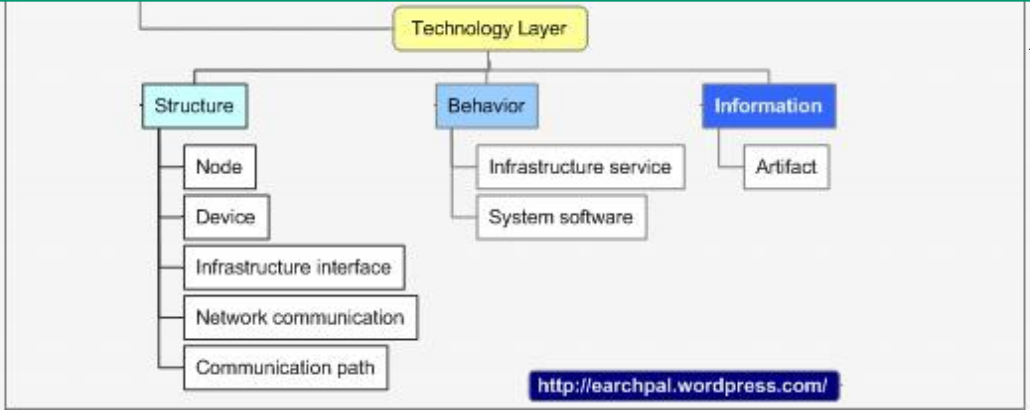
Comment?

Architecture applicative

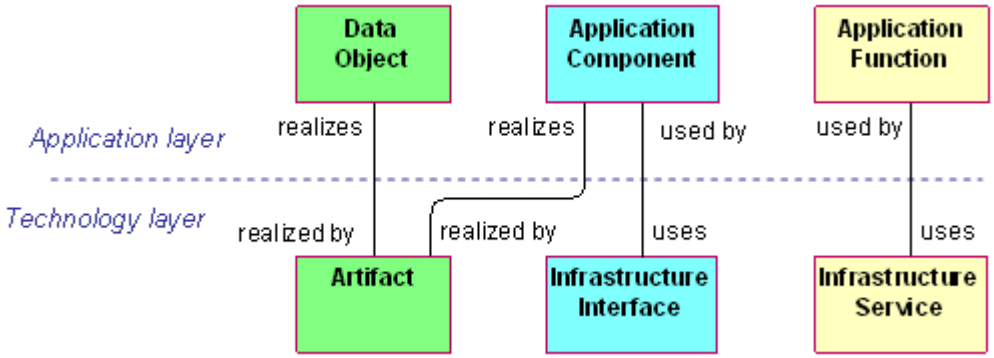


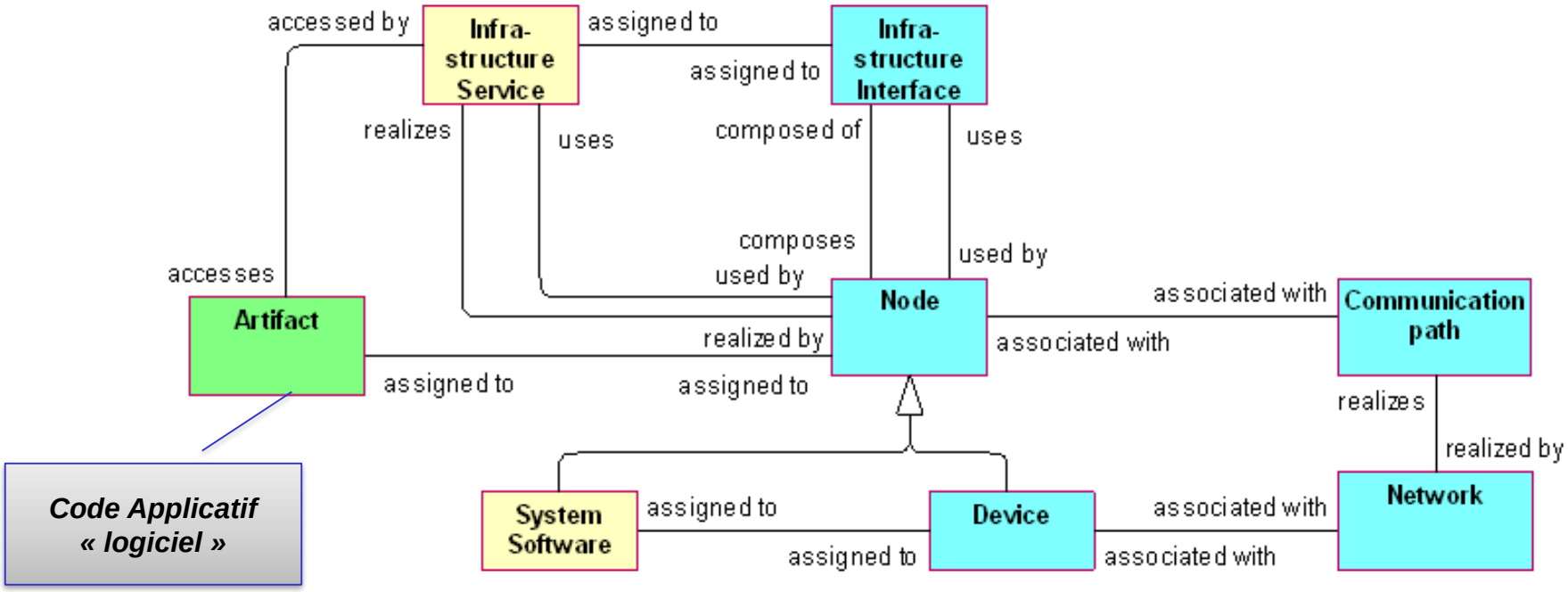
Avec quoi?

Architecture technique

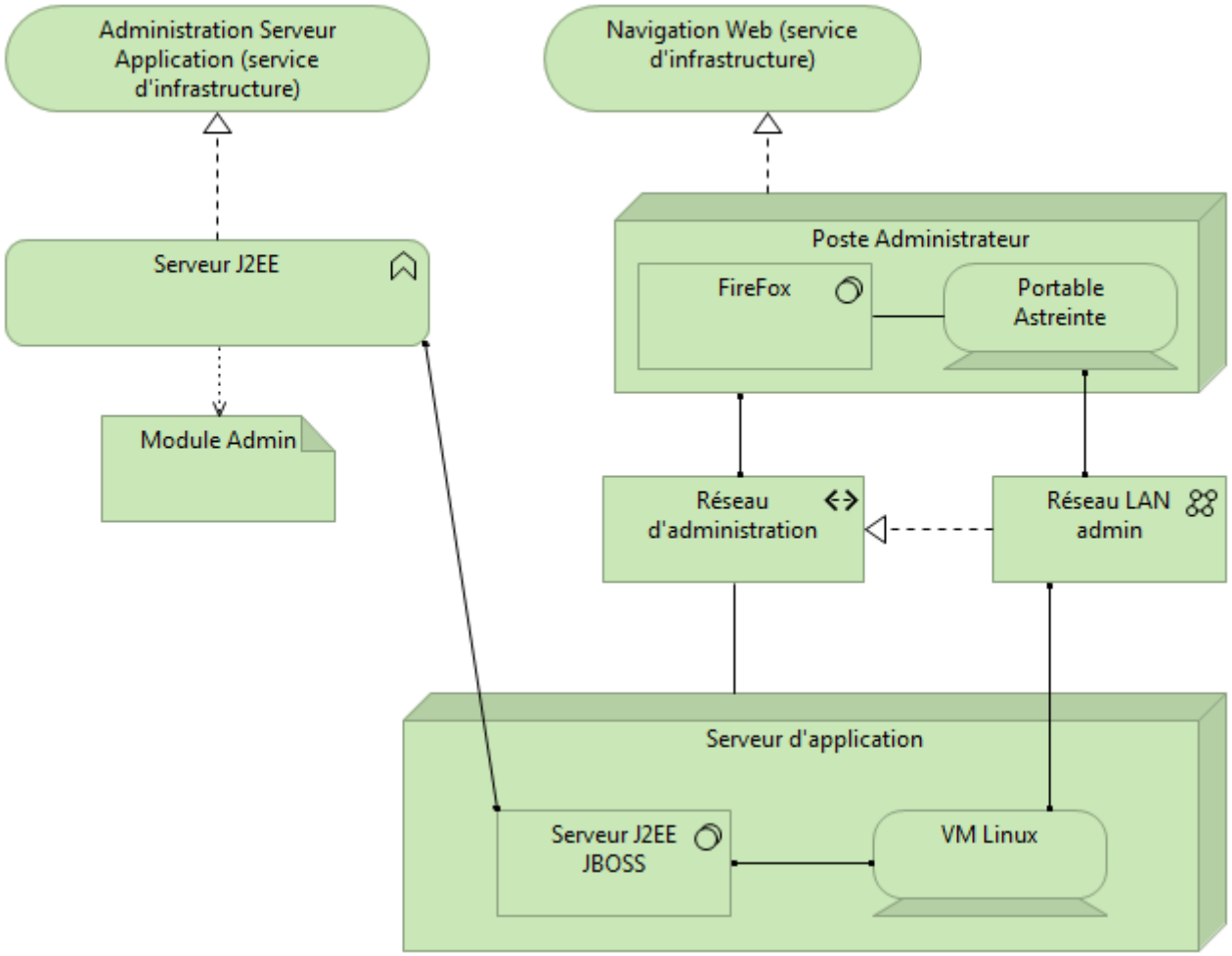


Architecture Technique : méta-modèle

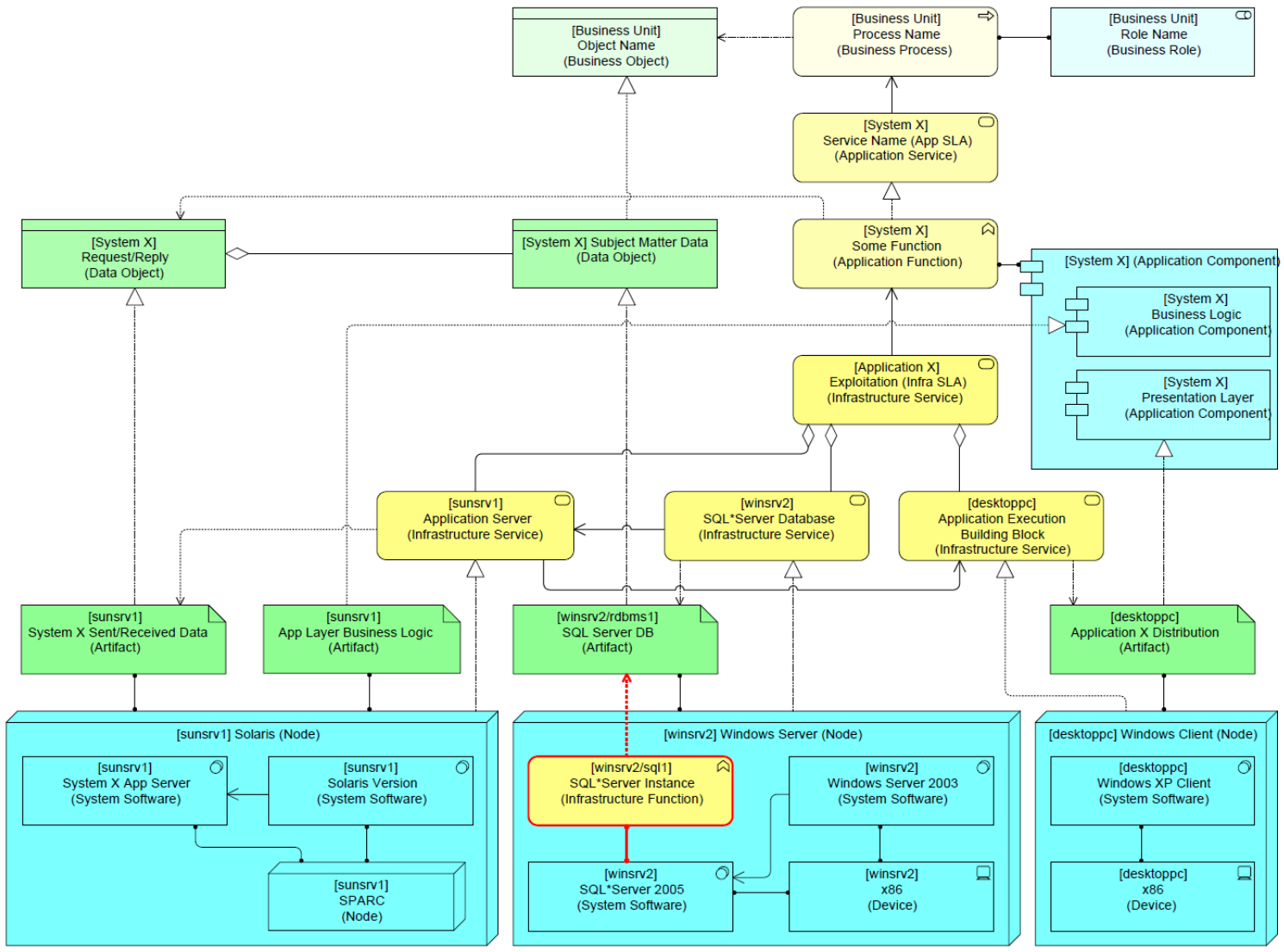




Composants de l'architecture technique



Par l'exemple : une application 3 tiers



Mise en pratique : de l'ULM à Archimate