La conception d'architecture

Le contenu est basé sur les transparents de la 10ème édition de "*Software Engineering*" de Ian Sommerville

Objectifs et activités

- Objectifs
  - Analyse du système
  - Communication avec les actionneurs
  - Réutilisation
- Activités
  - Décomposition
  - Spécifications des sous-systèmes
  - Spécifications des échanges (les interfaces)

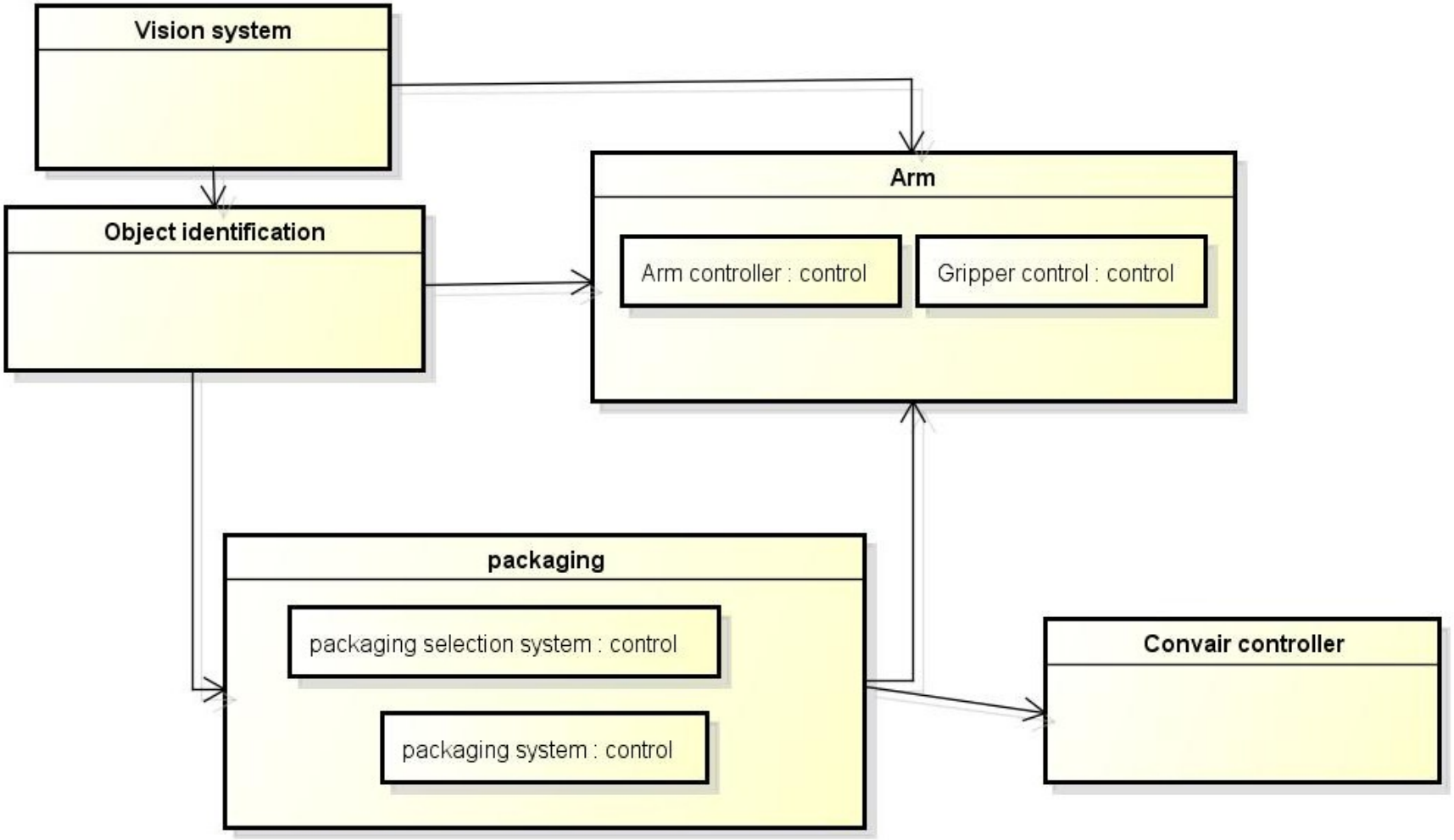
La gestion des caractéristiques

- Caractéristiques (besoins non-fonctionnels) de lesquelles l'architecture dépend
  - Performance
    - Localiser les opérations critiques et minimiser les communications.
  - Sécurité
    - Localiser les éléments critiques pour la sécurité dans peu sous-systèmes (dans les couches internes)
  - Disponibilité
    - Ajouter des composants redondants et mécanismes tolérant les fautes
  - Facilité pour maintenance
    - Utiliser des composants plus fines et réutilisables.
- Conflits
  - Sécurité contre performance
  - Facilité contre performance

Structuration et présentation

- Block diagrammes
- Diagrammes des classes
- Diagramme des composant avec interfaces

Exemple Robot de paquetage



Décisions de conception architecturale

- Y a-t-il une architecture générique à utiliser?
- Comment va le système être distribué?
- Quel style architectural est approprié?
- Quelle approche va être utilisée de structurer?
- Comment on va décomposer le système en modules
- Quelle stratégie de gestion on va utiliser?
- Comment on va évaluer le projet architectural?
- Comment on va documenter le projet?

Les modèles architecturals

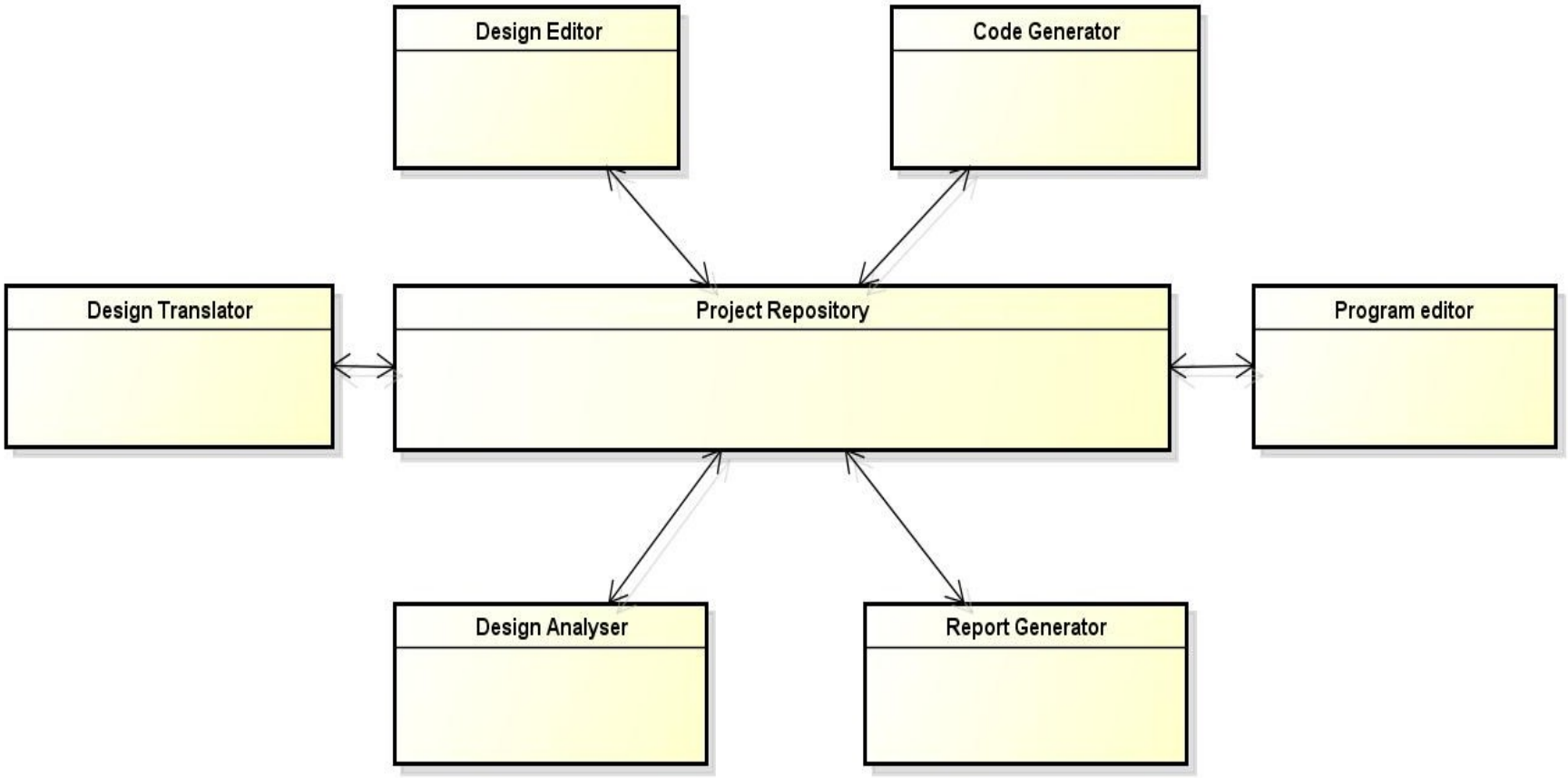
- Modèle statique de structure – les composants principaux
- Modèle dynamique des processus
- Modèle de l'interfaces des relations sous-système.
- Modèle des relations – DFD et sous systèmes.
- Modèle de distribution entre les nœuds.

Organisation du système

- 3 Styles d’organisation principaux
  - Donnés partagées (style d’entrepôt);
  - Services partagées (style serveur);
  - Machine abstraite (style en couches).

Le modèle entrepôt (de données)

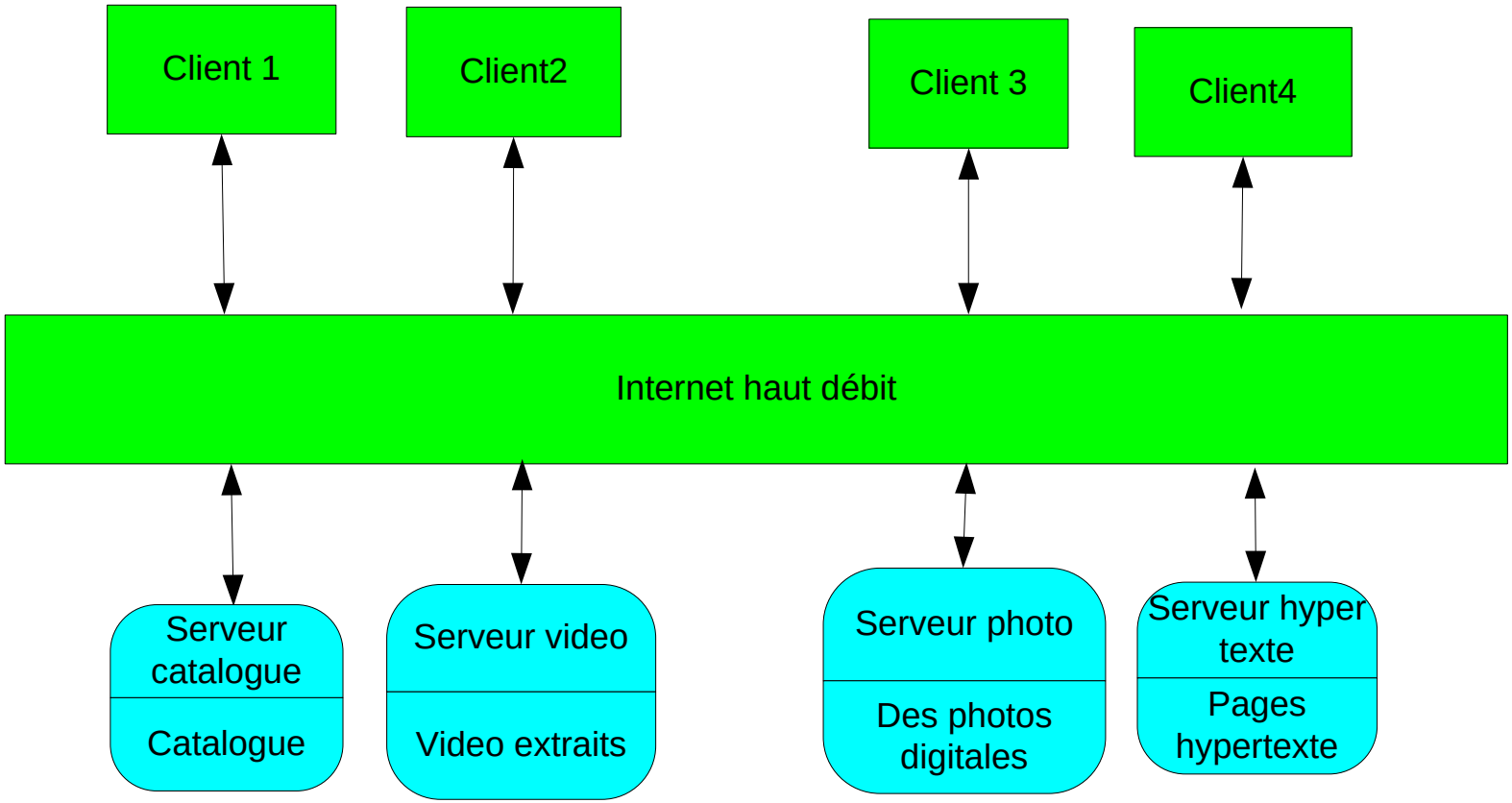
- CASE système



Le modèle entrepôt (de données)

- Deux moyens d'échange des données entre les sous-systèmes
  - Un entrepôt
  - Chaque sous-système a son propre dépositaire des données
- Avantages de style entrepôt
  - Efficace pour l'échange des grands quantités de données;
  - Les sous-systèmes n'ont pas le besoin de gérer les données.
  - Le modèle des données est unique.
- Désavantages
  - Tout sous-système doit utiliser le même modèle;
  - L'évolution des données est difficile;
  - Difficile pour distribuer.

Modèle client serveur



Modèle client serveur

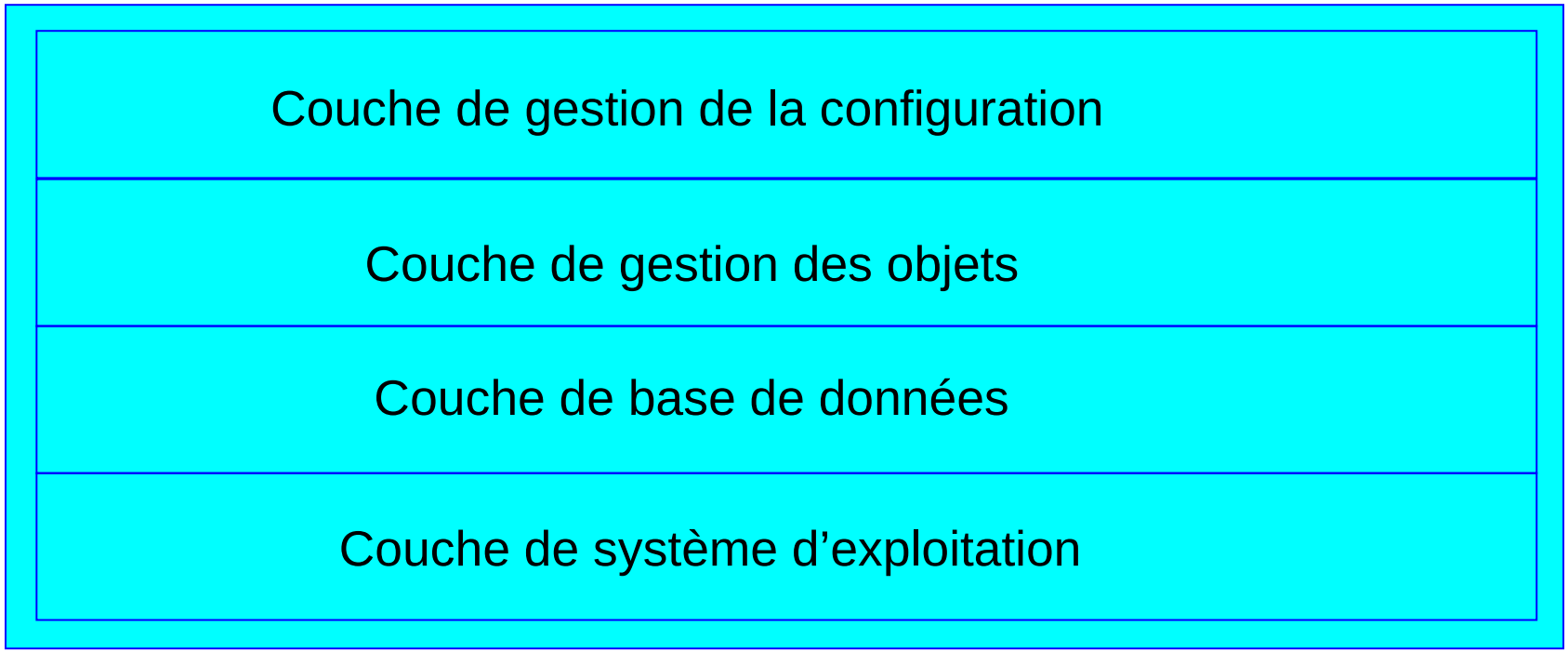
- Avantages
  - Distribution des données est simple;
  - On utilise effectivenent les réseaux;
  - C’est facile d’ajouter des nouveaux serveurs ou de moderniser les existants.
- Désavantages
  - Il n’y a pas un modèle unique des données et l’échange peut être ineffective;
  - Gestion redondante de chaque serveur;
  - Il n’y a pas un registre central des services et données. Ça peut poser des problèmes.

Machine abstraite

- Particularités
  - Modélise l'interface entre les sous-systèmes
  - Organisé en couches
  - Efficace quand on développe en incréments
- Désavantages
  - Structuration plus difficile et un peu artificielle

Machine abstraite

- Système de gestion des versions



Décomposer en modules

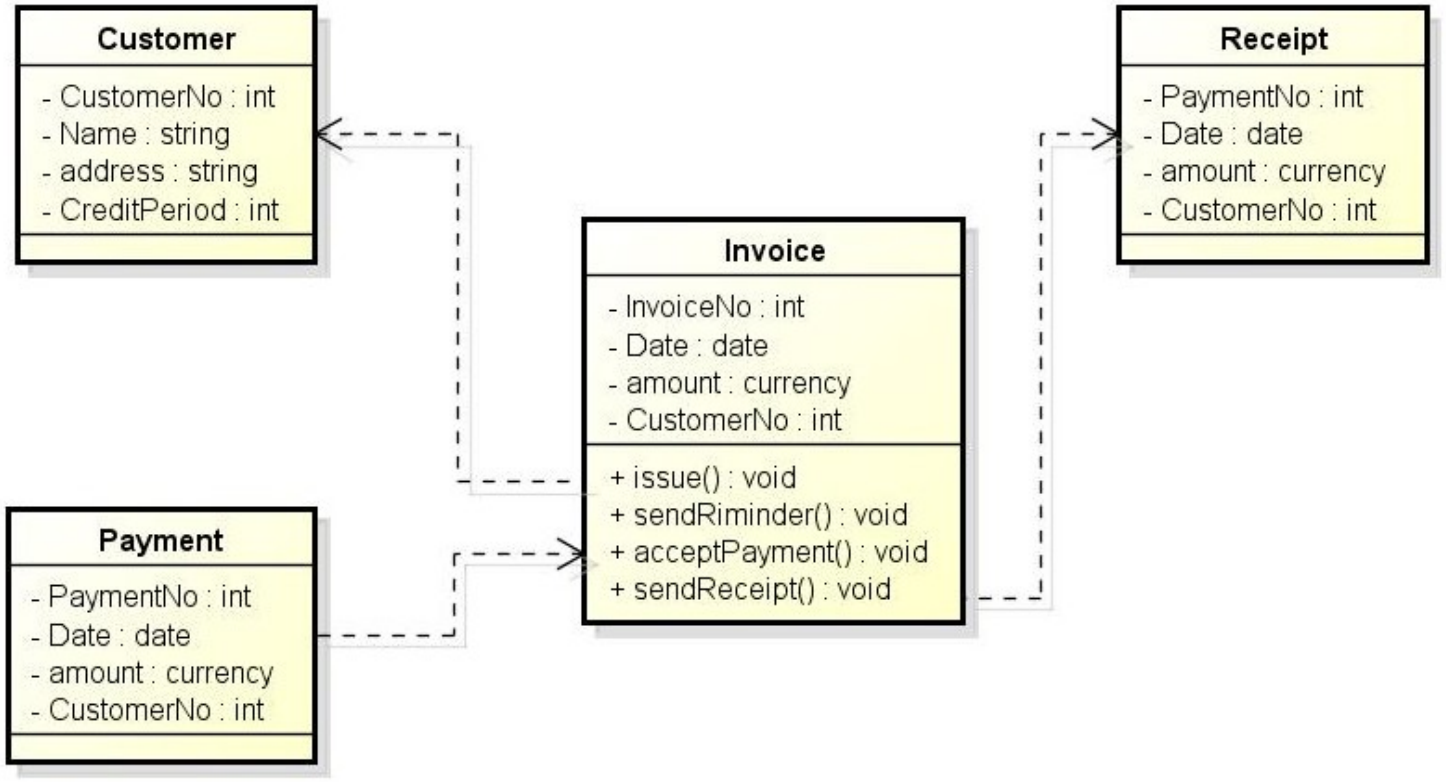
- Différence entre sous système et module?
  - Le sous système et un composant dont l'opération ne dépend pas des autres sous systèmes
  - Le module assures des services aux autres composants mais il n'est pas considéré comme un système séparé.
- Modèles de décomposition
  - Modèle objet
  - Modèle pipeline

Modèle objet

- Particularités
  - Il présente le système comme un ensemble d'objets qui sont faiblement connectés
  - Ils ont des interfaces bien définis.
- Avantages et désavantages
  - Les objets sont faiblement connectés et leur modification ne concerne pas les autres objets
  - Les objets sont souvent des entités réelles
  - On utilise des langages OO pour les implémenter
  - Quand les objets sont complexes la modélisation est difficile

Modèle objet

- Système de facturation

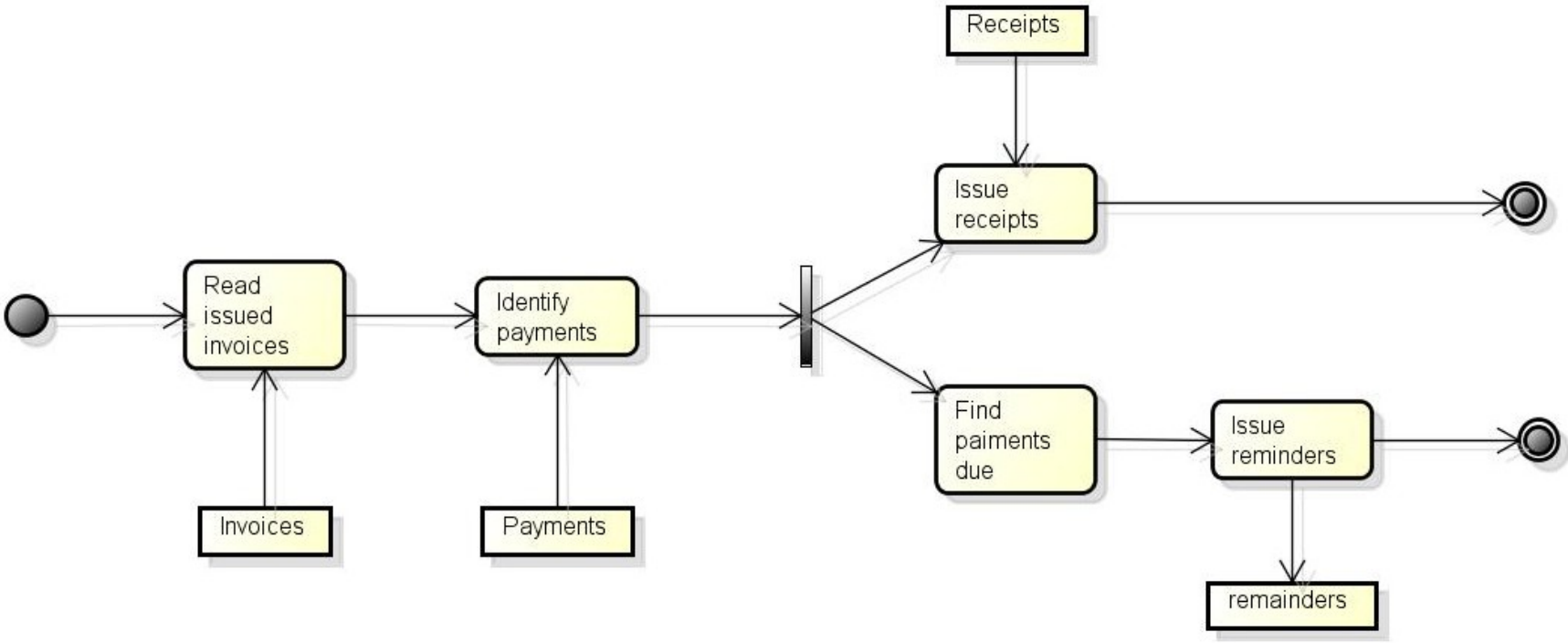


Pipeline orienté fonctionnellement

- Particularités
  - Il présente le système une pipeline de traitement des données
  - Très commun pour un traitement consécutif
- Avantages et désavantages
  - Approprié pour les processus en lots (batch)
  - Facile pour réutilisation tes transformation.
  - Organisation intuitive appropriée pour communiquer avec les actionneurs.
  - On peut ajouter facilement des nouvelles transformations.
  - Implémentation simple dans systèmes séquentielles et parallèles
  - Pas bon pour les systèmes interactifs

Pipeline orienté fonctionnellement

- Système de facturation



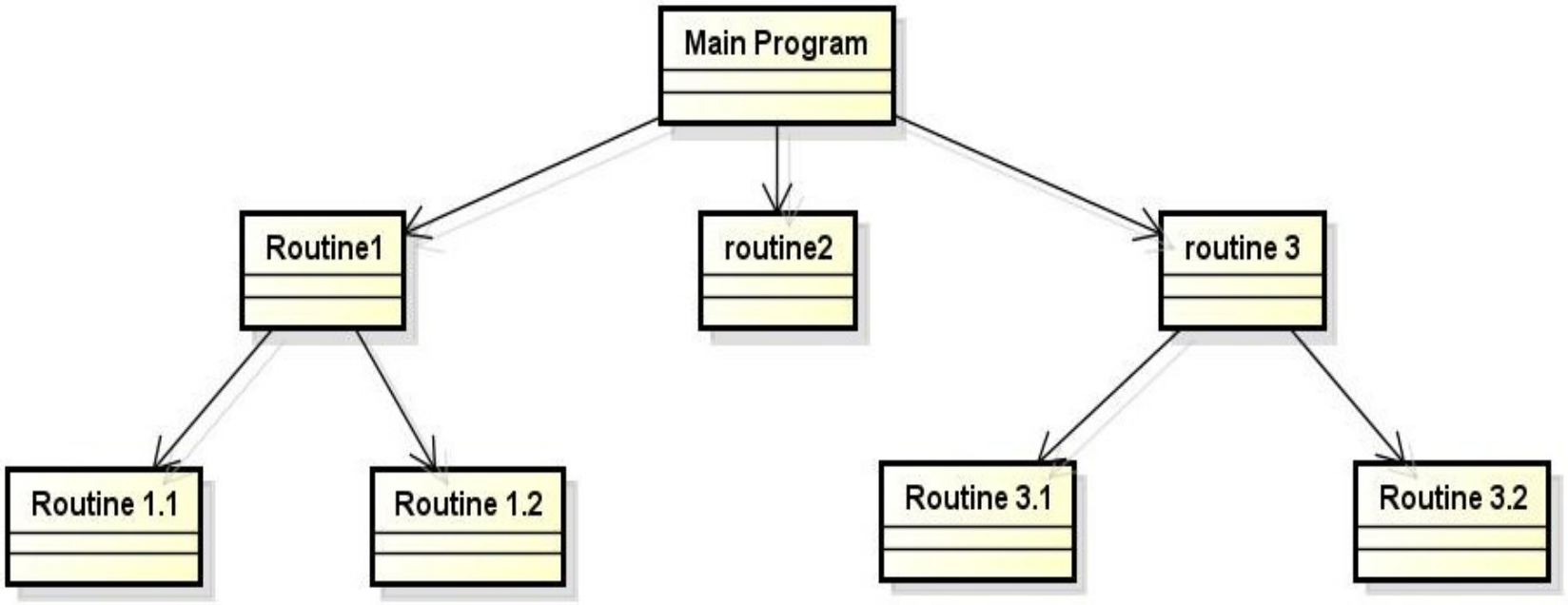
Styles de gestion

- Modèles de flux de contrôle
- Types
  - Contrôle centralisé
    - Un sous-système gérant gère les autres directement ou indirectement
  - Systèmes gérés par événements
    - Le système réagisse aux événements externes ou internes

Contrôle centralisé

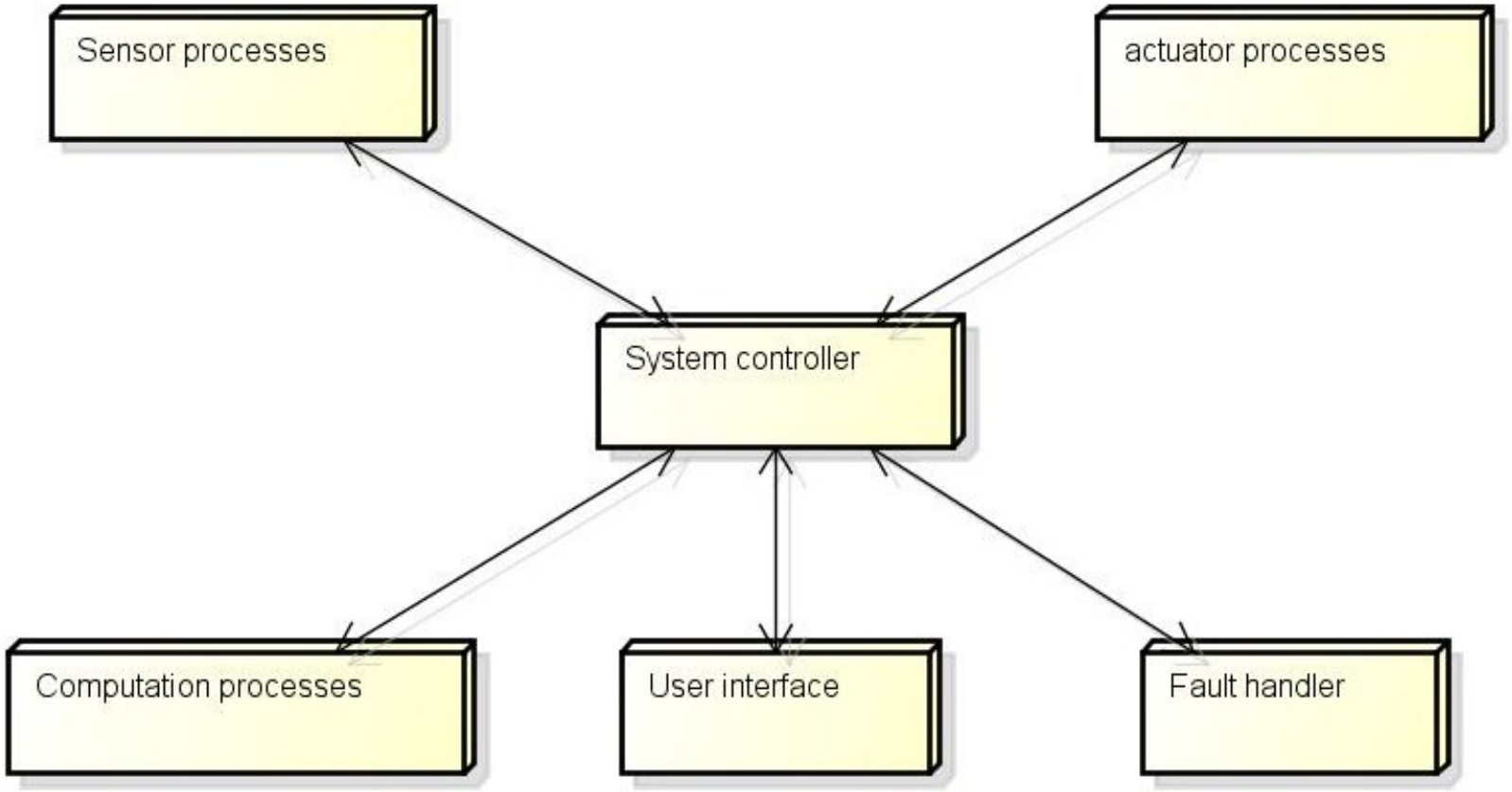
- Modèles
  - Modèle appel - retour (call-return)
  - Modèle de gérant centralisé (manager)

Modèle appel - retour (call-return)



Modèle de gérant centralisé

- Système de temps réel

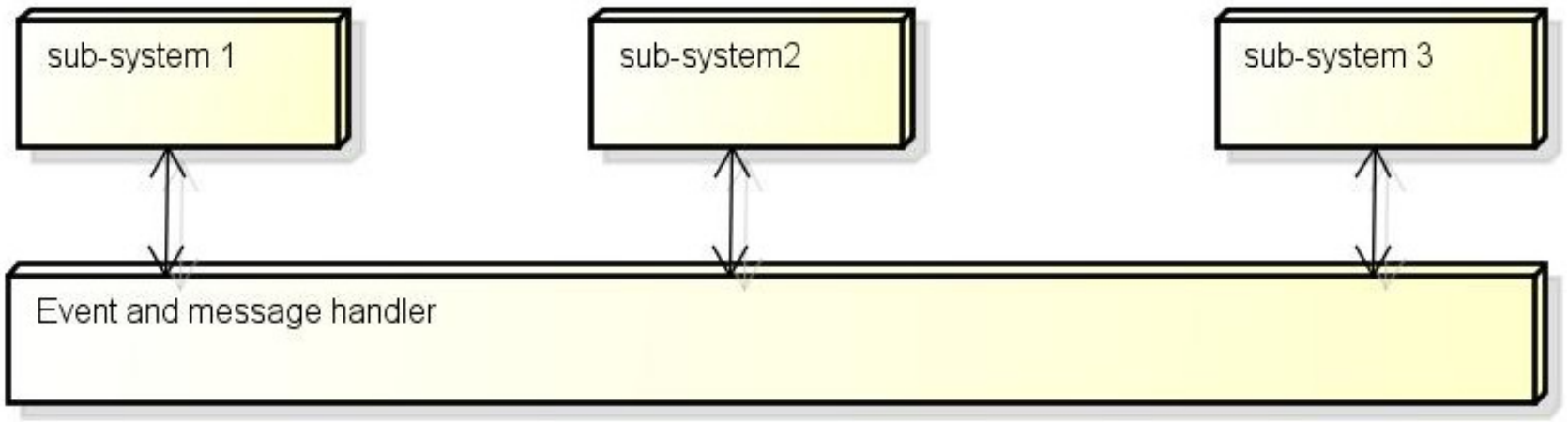


Systèmes gérés par événements

- Types
  - De diffusion (broadcasting) – l'événement est émis vers tous les sous systèmes et une d'eux le traite
  - Gérés par interruptions – les interruptions sont découverts par des pilotes qui les passent au quelque composant.

Le modèle de diffusion

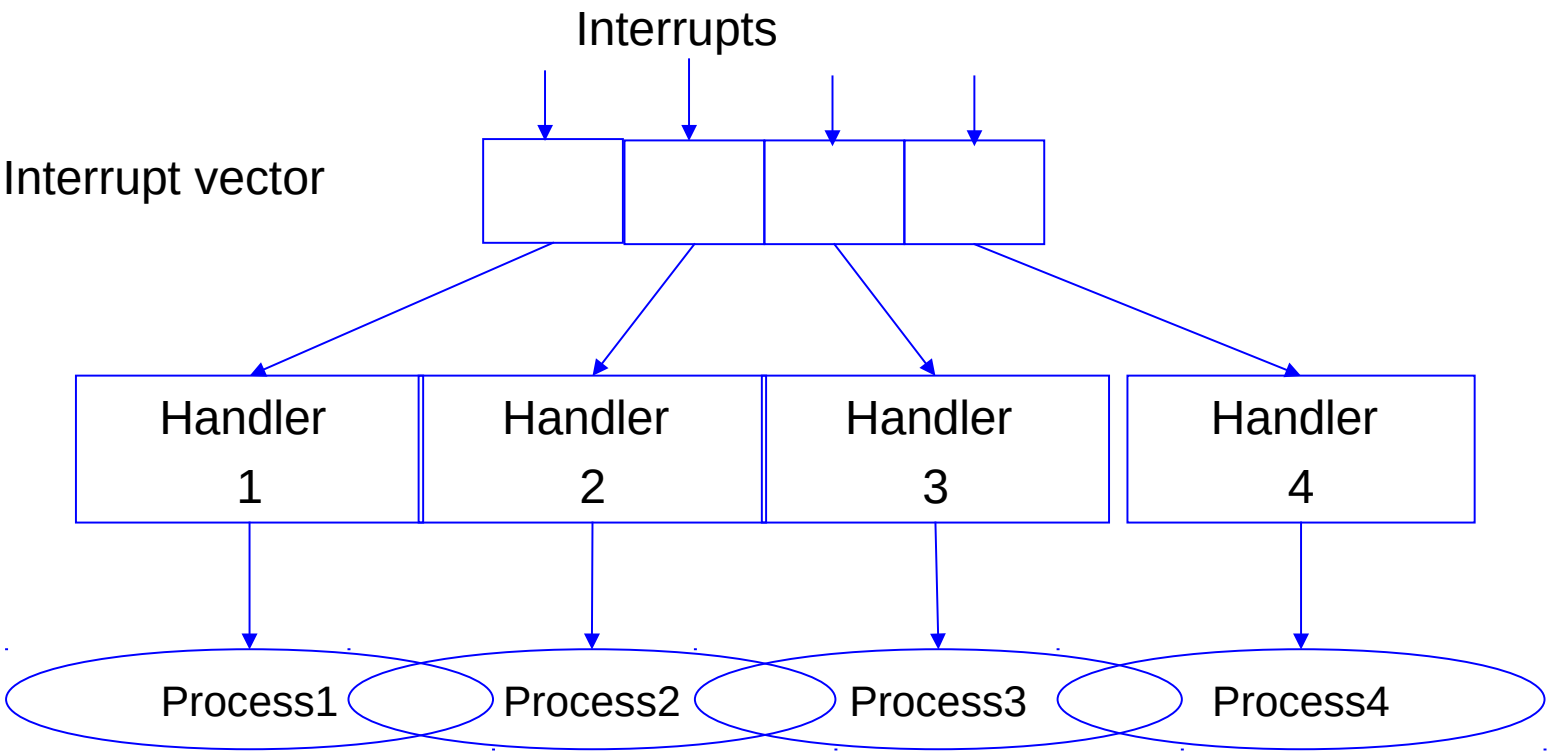
- Ce modèle est effective quand il y a beaucoup événement différents et chaque composants peut traiter certains d'eux.
- Le modèle de traitement n'est pas dans le pilote mais dans le composant traitant
- Diffusion sélective



La gestion d’interruptions

- Chaque interruption a son pilote qui est appelé immédiatement. Après le traitement le contrôle est retourné dans le point d’interruption.
- Utilisé dans les systèmes de temps réel.

La gestion d'interruptions

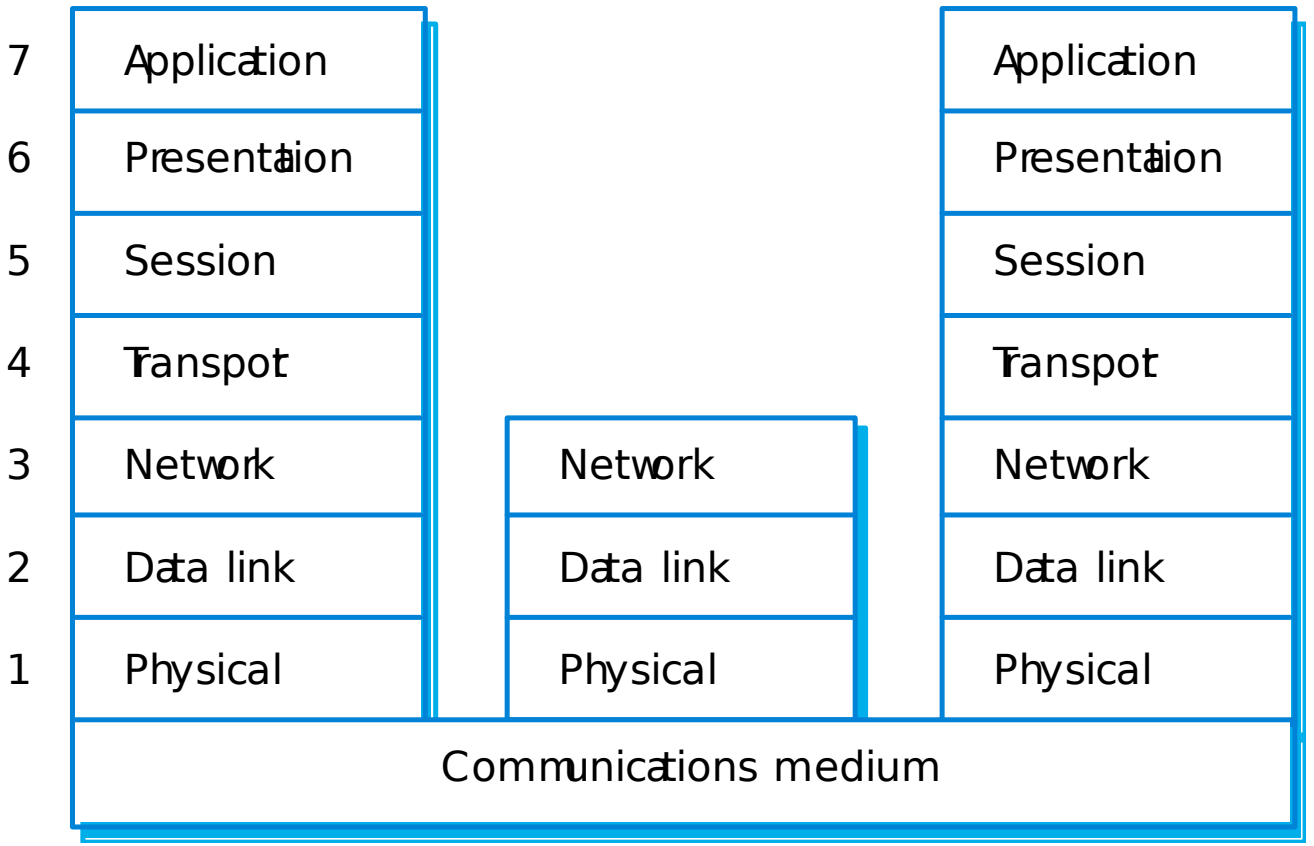


Architectures de référence

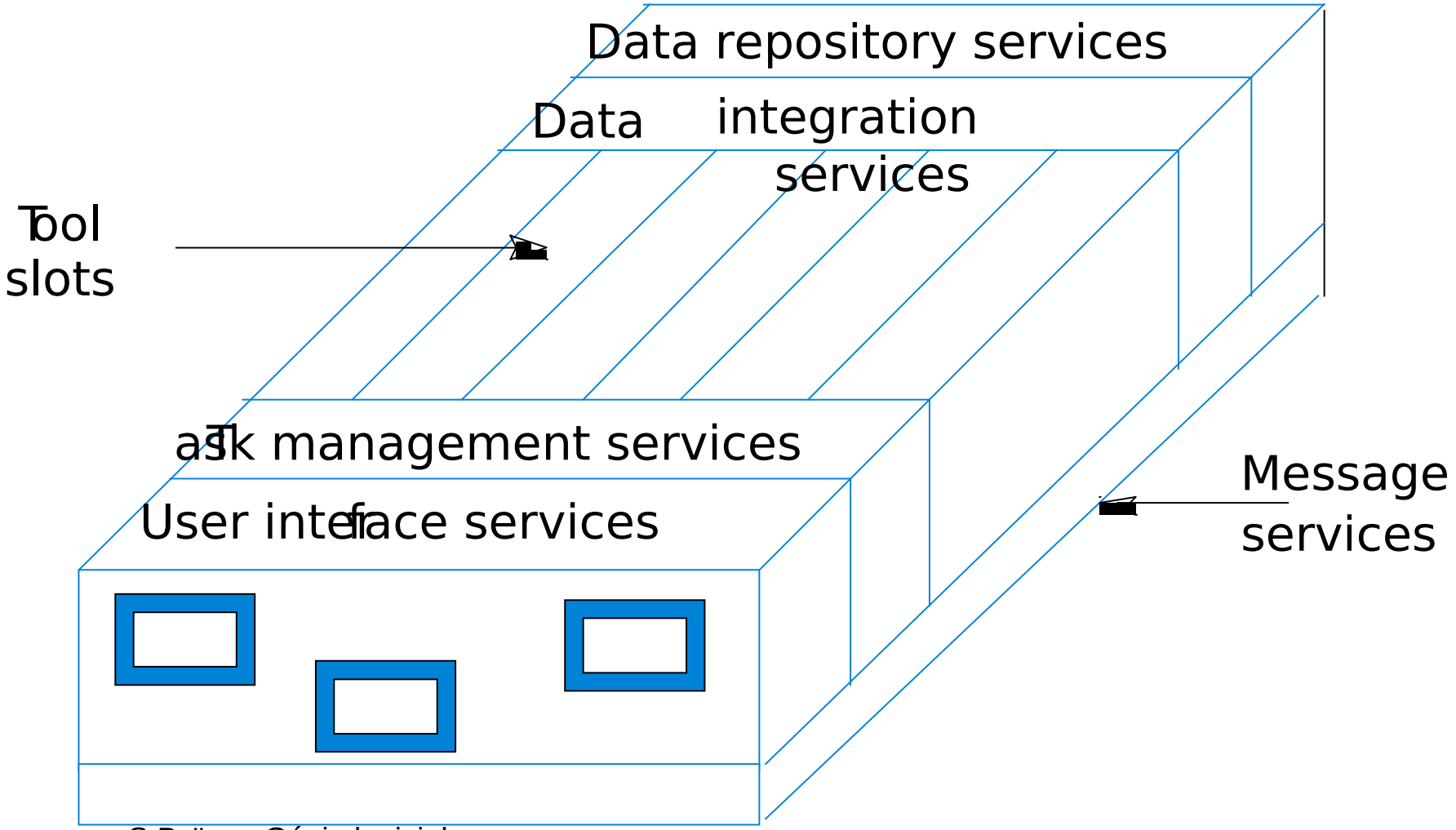
- Architectures spécifiques de certain domaine
- Types
  - Modèles génériques – généralisés de certains systèmes réels
  - Modèles de référence
    - abstrait et plus théoriques.
    - Dérivés du domaine
    - Peuvent servir comme standard pour validation et évaluation des architectures

Architectures de référence

- Modèle OSI



ECMA architecture pour CASE



Modèle de référence ECMA

- Data repository services
  - Gérer et stocker les données.
- Data integration services
  - Gérer des groupes d’entités et les relations entre eux.
- Services de gestion des tâches
  - Définition et énaction des modèles de processus.
- Messaging services
  - Communication Outil-outil et outil-environnement
- User interface services
  - Développement de l'interface utilisateur.