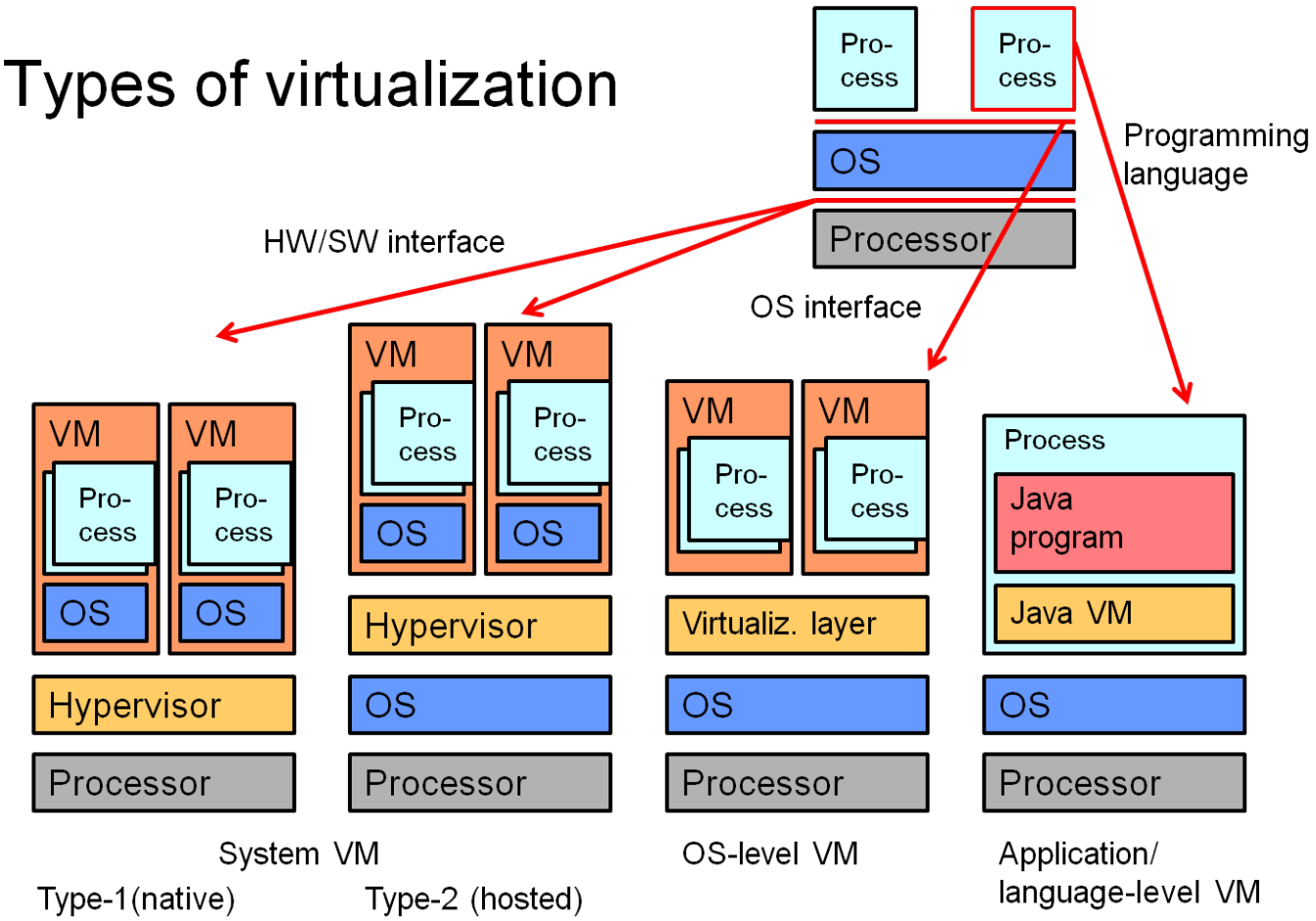
Machines virtuelles

- **Des intergiciels qui étendent les fonctions classiques d'un système d'exploitation**
  - Des approches et des niveaux d'abstraction différents pour différents besoins



Picture courtesy of Gernot Heiser, UNSW.

Machines virtuelles

1er exemple : hyperviseurs (1/2)

- **Un hyperviseur**
  - est une couche logicielle
  - fournit l'abstraction d'une ou plusieurs machines « nues » (processeurs, mémoire et périphériques) au dessus d'une véritable machine physique
- **Intérêt :**
  - Faire tourner plusieurs systèmes d'exploitation simultanément sur la même plateforme matérielle
  - Faciliter le déploiement pour le « cloud computing » (IAAS: Infrastructure as a service)
  - Faciliter le test/débogage/prototypage de systèmes d'exploitation et d'applications réparties
  - Sauvegarder/rembobiner l'état d'un système
  - Sécurité (renforcement de l'isolation de certaines applications)
  - ...

Machines virtuelles
1er exemple : hyperviseurs (2/2)

- **Deux types d'implémentations**
  - Type I : hyperviseur natif
  - Type II : hyperviseur hébergé
- **Hyperviseur de type I (natif)**
  - Un hyperviseur natif est la couche logicielle de plus bas niveau
  - C'est en fait un système d'exploitation spécialisé
  - Approche la plus efficace
  - Exemples : VMware ESX, Xen
- **Hyperviseur de type II (hébergé)**
  - Un hyperviseur hébergé est une application intermédiaire (intergiciel) qui s'appuie sur un système hôte sous-jacent
  - Moins efficace (plus de couches) mais plus simple à installer/utiliser
  - Exemples : VMware workstation/fusion/player, VirtualBox

Machines virtuelles

2ème exemple : machine virtuelle Java

- **Une machine virtuelle Java (JVM) :**
  - Est implémentée sous la forme d'un intergiciel qui s'appuie sur les services d'un système hôte
    - S'exécute sous la forme d'un processus applicatif
    - Les threads d'un programme java sont (généralement) implémentés à partir des threads du système sous-jacent
  - Joue le rôle d'un interprète :
    - lit en entrée le code généré par le compilateur javac (format bytecode portable, indépendant de la plateforme sous-jacente)
    - traduit les instructions de bytecode en séquences d'actions adaptées à la plateforme sous jacente (ABI)
  - Fournit des services aux programmes Java
    - Exemples : ramasse-miettes mémoire, gestion d'interface graphique ...
    - Ces services peuvent être implémentés sous forme de threads

3ème exemple : Conteneurs (*OS-level containers*)

- **Extensions intégrées au sein d'un système d'exploitation « classique » comme Linux**
- **Création de compartiments cloisonnés**
  - Isolation de sécurité : Espace de nom (fichiers, processus) distincts
  - Isolation de performances : Règles d'allocation des ressources globales
- **En complément du support au niveau de l'OS, des outils pour simplifier la création de paquets applicatifs légers et auto-suffisants, et leur déploiement**
  - Gestion des exécutables, des bibliothèques et des fichiers de configuration
  - Gestion des numéros de version et des dépendances
  - Système de fichiers imbriqué
  - Exemple : Docker

3ème exemple : Conteneurs (*OS-level containers*) - Détails

- **Comparaison avec un hyperviseur**
  - Un seul noyau partagé entre tous les conteneurs
  - Moins de sécurité
  - Moins de flexibilité (choix de la configuration logicielle)
  - Moins de contrôle (pause/reprise, migration, rollback)
  - Mais une architecture plus simple à déployer (moins de couches logicielles) et plus légère (meilleures performances)