Système d'administration autonome adaptable: application au Cloud

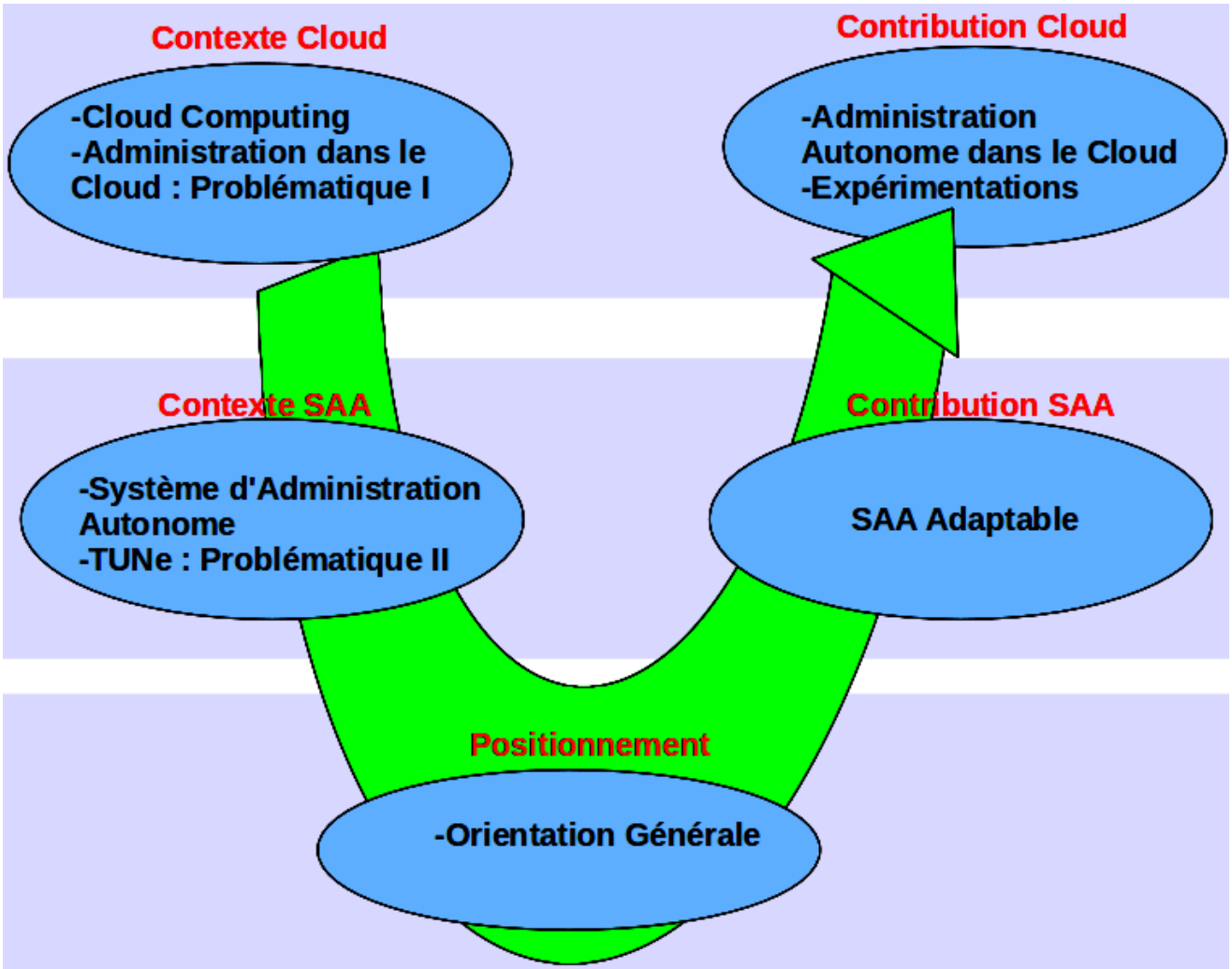
Alain TCHANA - atchana@enseeiht.fr

IRIT/ENSEEIHT, Equipe SEPIA

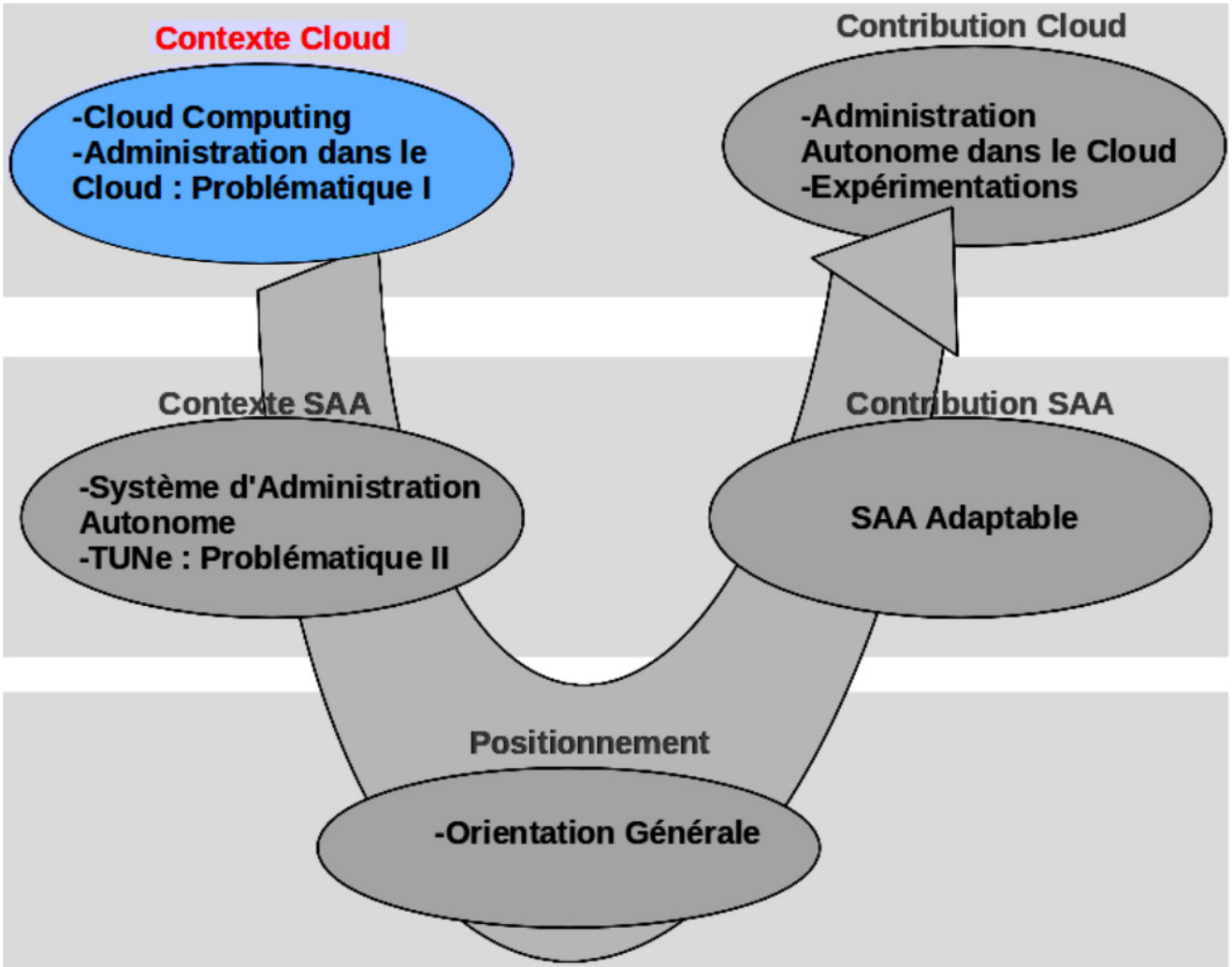
Directeur de thèse : Daniel HAGIMONT et Laurent BROTO
Rapporteurs : Jean-Marc MENAUD et Noël DE PALMA
Examinateurs : Michel DAYDE et Maurice TCHUENTE

29 Novembre 2011

Démarche Générale

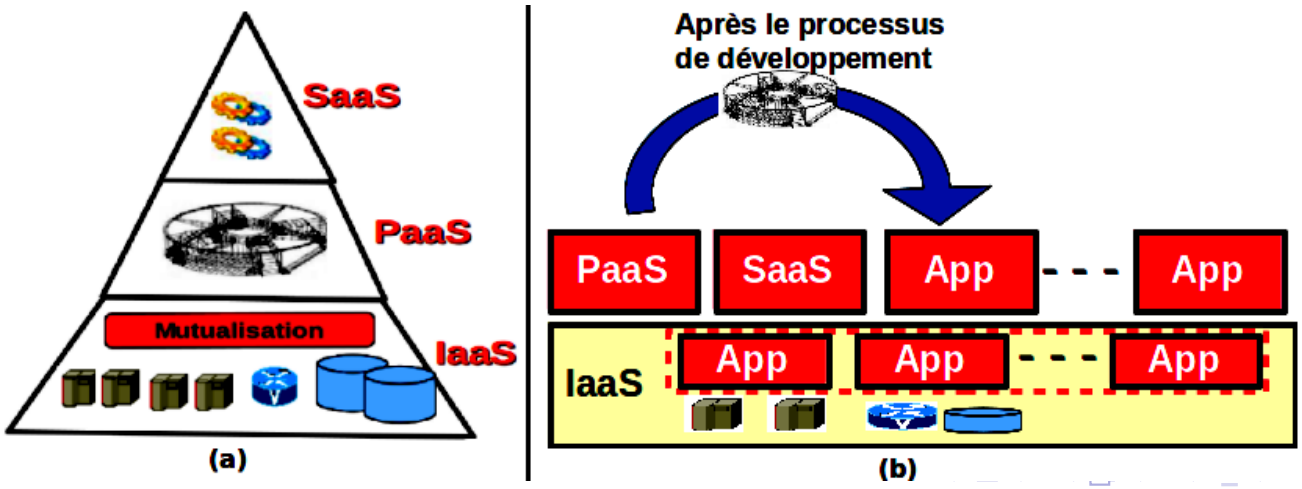


Étape 1


Contexte Cloud
-Cloud Computing
-Administration dans le Cloud : Problématique I
Contribution Cloud
-Administration Autonome dans le Cloud
-Expérimentations
Contexte SAA
-Système d'Administration Autonome
-TUNe : Problématique II
Contribution SAA
SAA Adaptable
Positionnement
-Orientation Générale

Cloud : Contexte

- Cloud Computing : nouvelle approche d'hébergement
- Avantages pour les Clients
  - Paiement à l'usage (Pay-As-You-Go)
  - Moins d'investissement
  - Économie
- Avantages pour le Fournisseur
  - Mutualisation de ressources
  - Donc : gain (et moins de gaspillage)



Cloud Computing : Défis

- Défis
  - Isolation (défaillance, ressources, performance) : par virtualisation
  - Portabilité et Interopérabilité : par standardisation
  - Administration
- Administration à deux niveaux
  - IaaS : administrateur du cloud
  - Applications clientes : administrateur entreprise

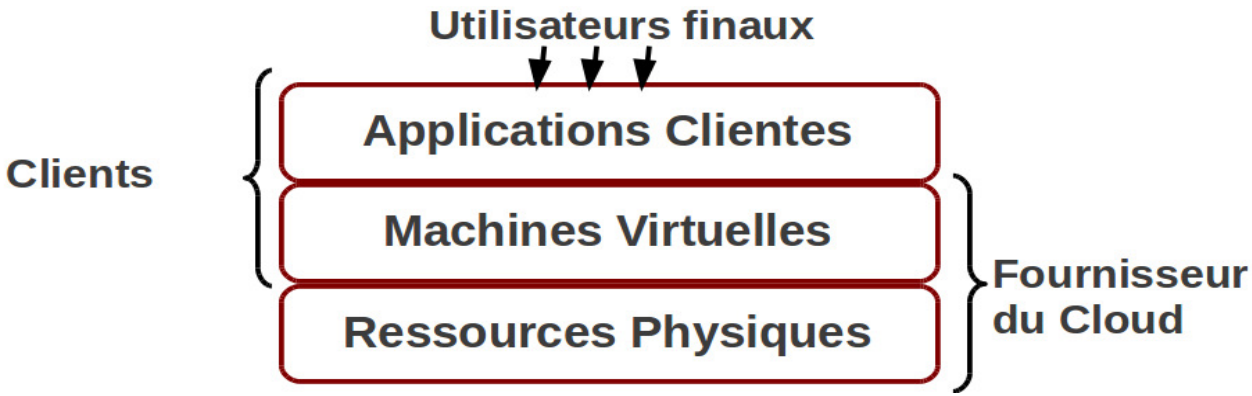
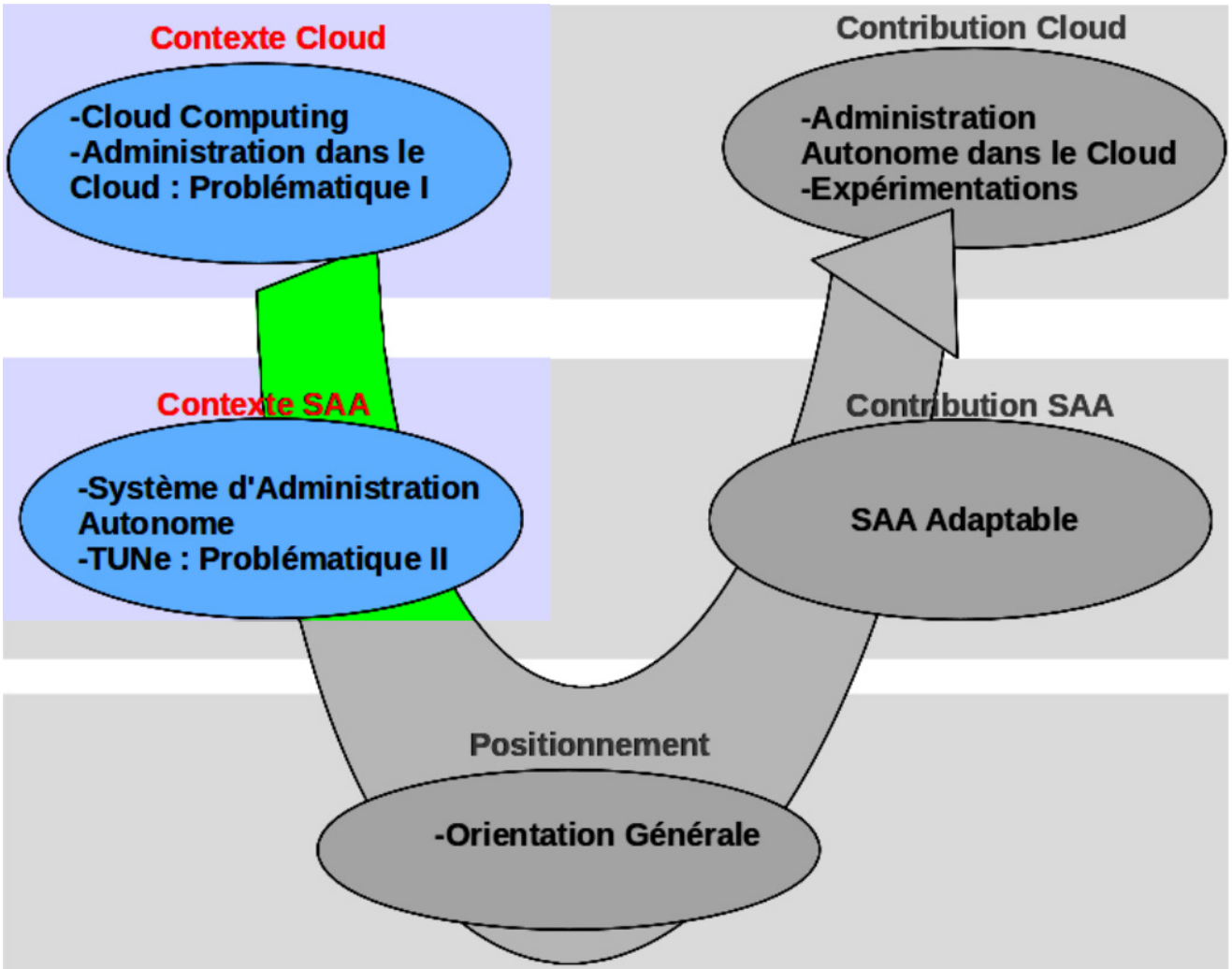


Figure – *Niveaux d'administration dans le cloud*

Cloud Computing : Administration

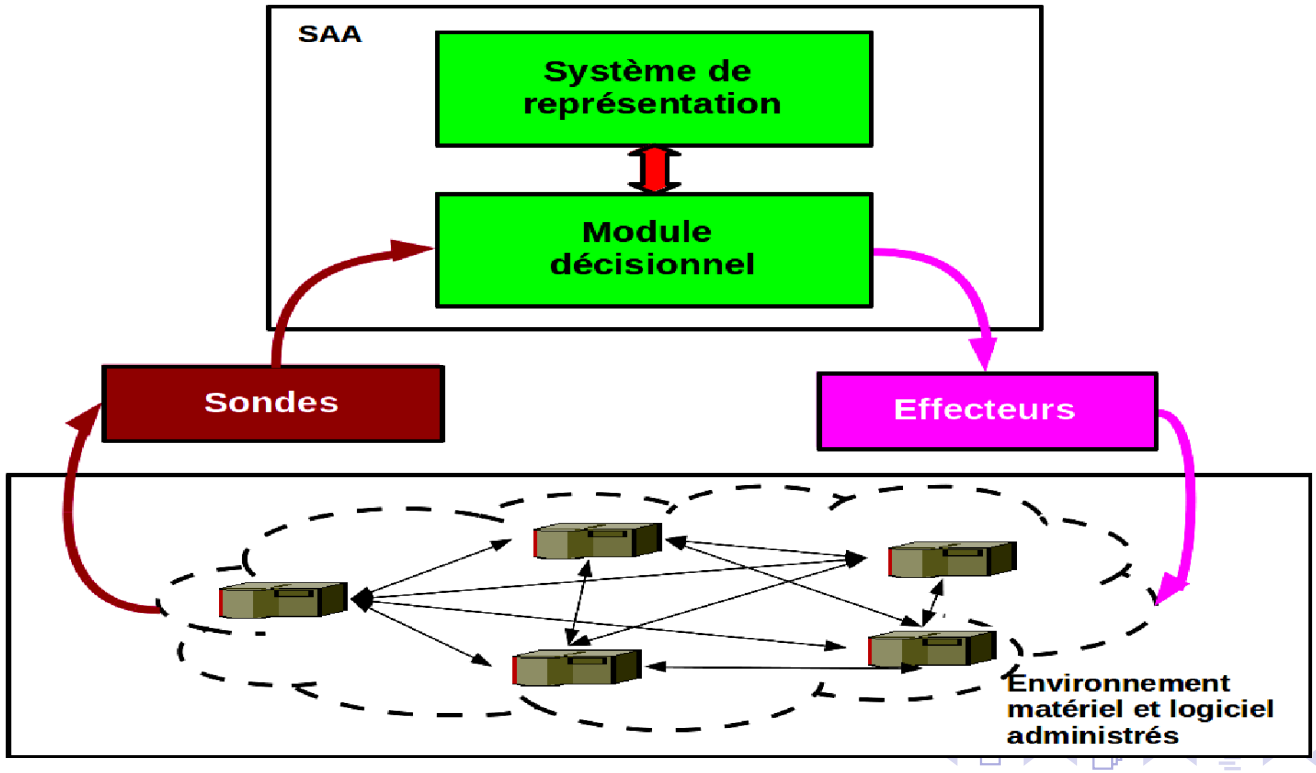
- Niveau IaaS
  - Déploiement : gestion des images (OS de VM), allocation de machines (placement)
  - Configuration et Démarrage : réseaux, VM
  - Gestion dynamique : monitoring, consolidation (migration), gestion des pannes
- Niveau application cliente
  - Déploiement : images de VM et logiciels
  - Configuration et Démarrage : allocation de VM et logiciels
  - Gestion dynamique : monitoring, réparation, scalabilité (réplication), etc.
- Solution : Système d'Administration Autonome (SAA)

Étape 2


Contexte Cloud
-Cloud Computing
-Administration dans le Cloud : Problématique I
Contribution Cloud
-Administration Autonome dans le Cloud
-Expérimentations
Contexte SAA
-Système d'Administration Autonome
-TUNe : Problématique II
Contribution SAA
SAA Adaptable
Positionnement
-Orientation Générale

SAA

- Système d'administration d'autres systèmes informatiques
- Fournit les services de
  - Déploiement de logiciels
  - Configuration et Démarrage
  - Monitoring et Reconfiguration
- Exemples : Unity, Jade, TUNe, ...



Cas du SAA TUNe

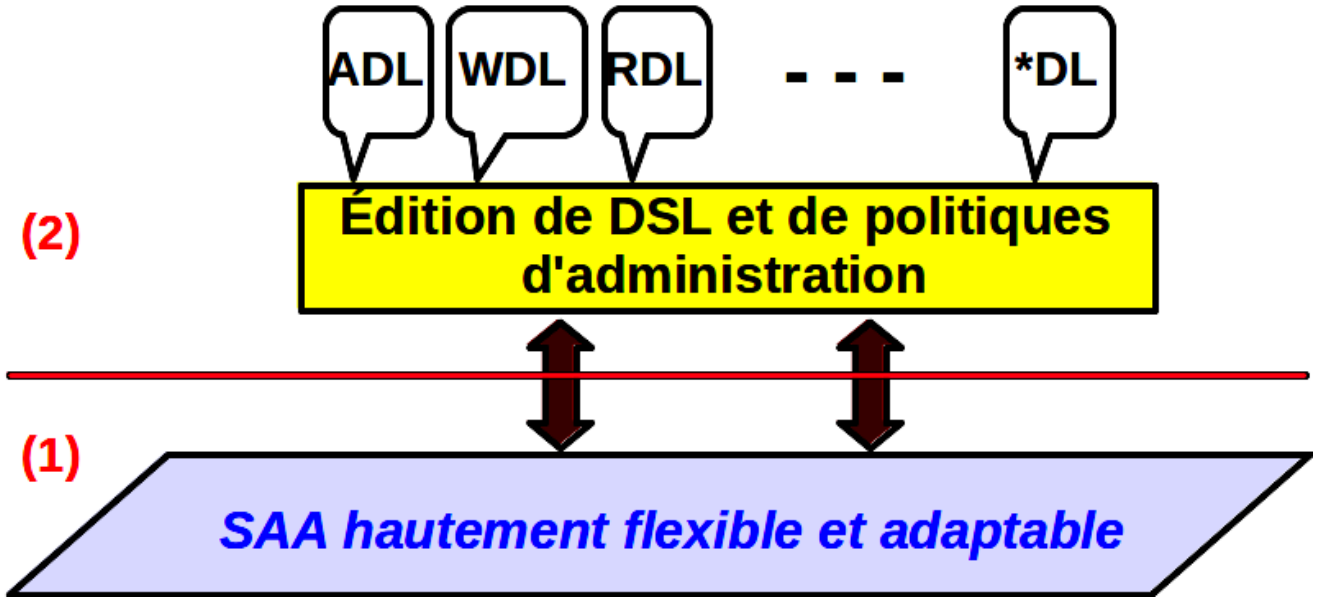
- Développé au sein de notre équipe
- Inspiré de Jade, à composants
- Machine à langages (profils UML)
  - ADL : Classes UML pour la description de logiciels et nœuds
  - WDL : XML pour les comportements des logiciels
  - RDL : Diagrammes d'état-transition pour les programmes de reconfiguration
- Plusieurs expérimentations
  - Applications J2EE : concluantes
  - Applications large échelle DIET : modification
  - Systèmes virtualisés : modification

Problématique : TUNe et ses limites

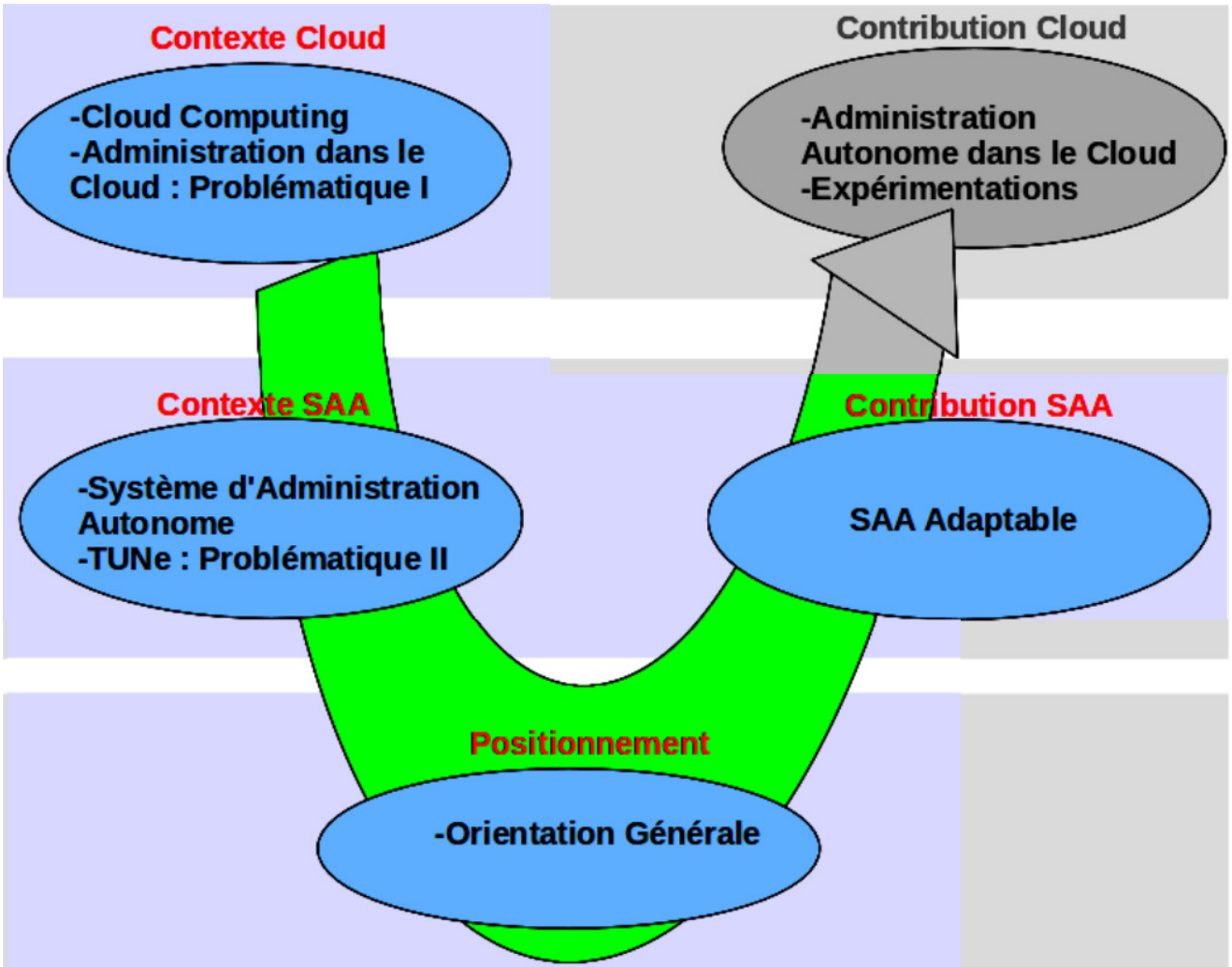
- Conçu pour des applications maître-esclave
- Hypothèses fortes sur les éléments administrés (Comportements câblés)
- Difficulté de personnalisation
- Difficulté de supporter de nouveaux langages
- Problèmes généralisés à d'autres
  - Machine langages (langages dédiés d'administration) : Rainbow, Accord, Unity ;
  - Pas de représentation de l'environnement matériel : Accord ;
  - Système de monitoring prédéfini : Accord ;
  - Types d'éléments administré câblés : Unity ;
  - ...

Problématique : TUNe et ses limites

- Une nouvelle orientation
  - Séparer les aspects langages et le corps métier du SAA
  - Adaptabilité à ces deux niveaux



Étape 3


Contexte Cloud
-Cloud Computing
-Administration dans le Cloud : Problématique I
Contribution Cloud
-Administration Autonome dans le Cloud
-Expérimentations
Contexte SAA
-Système d'Administration Autonome
-TUNe : Problématique II
Contribution SAA
SAA Adaptable
Positionnement
-Orientation Générale

TUNeEngine : Orientation générale

- Un nouveau SAA suivant 3 directives
  - Uniformité : pas d'hypothèses fortes sur les éléments administrés. Les éléments administrés utilisent la même représentation interne quelque soit leur nature.
  - Adaptabilité : les composants/services du SAA sont modifiables/remplaçables sans maîtrise complète du SAA.
  - Interopérabilité et Collaboration : Capacité à échanger des informations ou des ordres d'administration avec d'autres SAA distincts.
- Développé suivant une architecture à composants
  - Tout service est implanté par un composant clairement identifiable.
- TUNeEngine : est un SAA développé suivant ces directives

Rappel du modèle d'un SAA

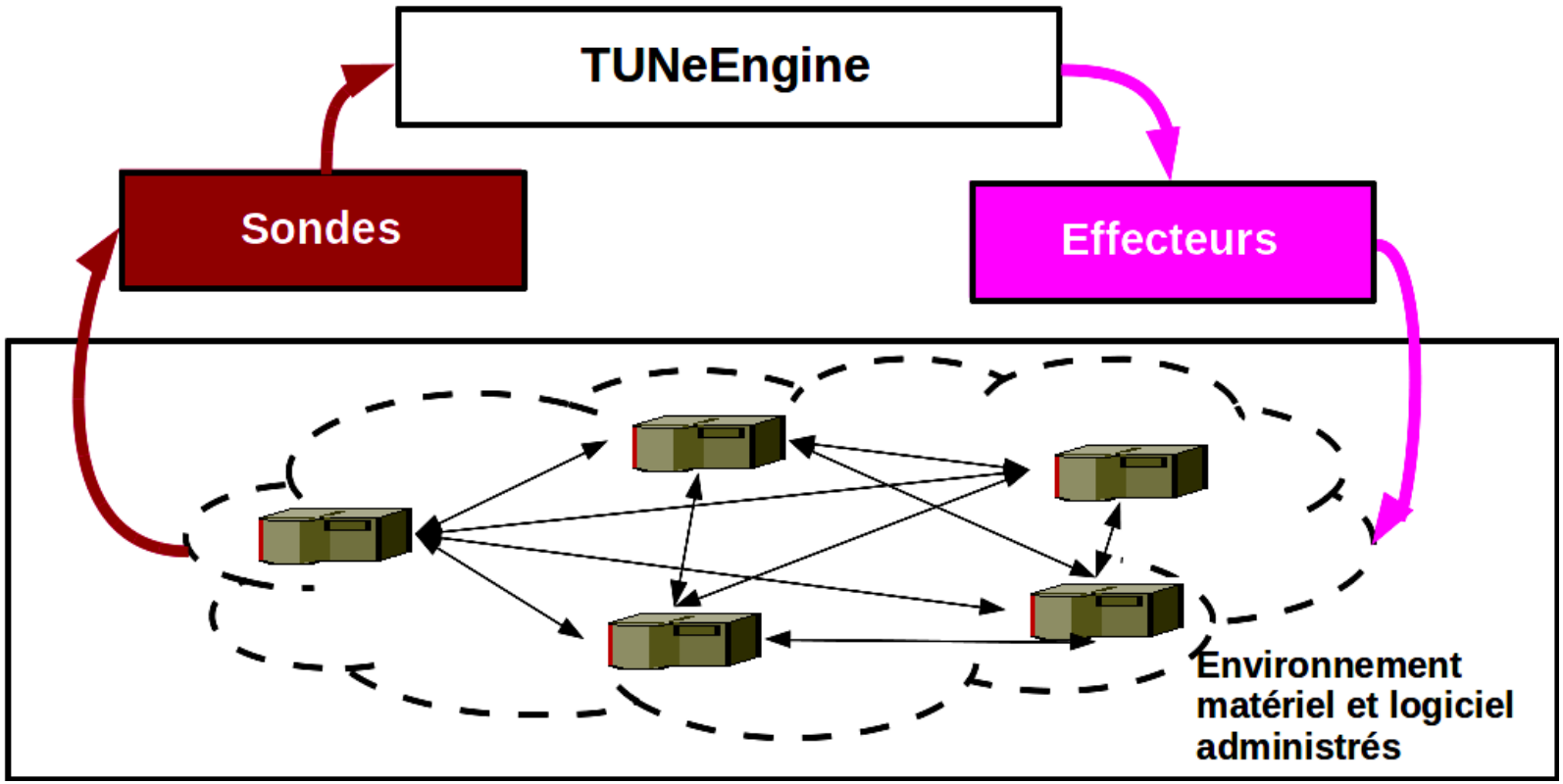
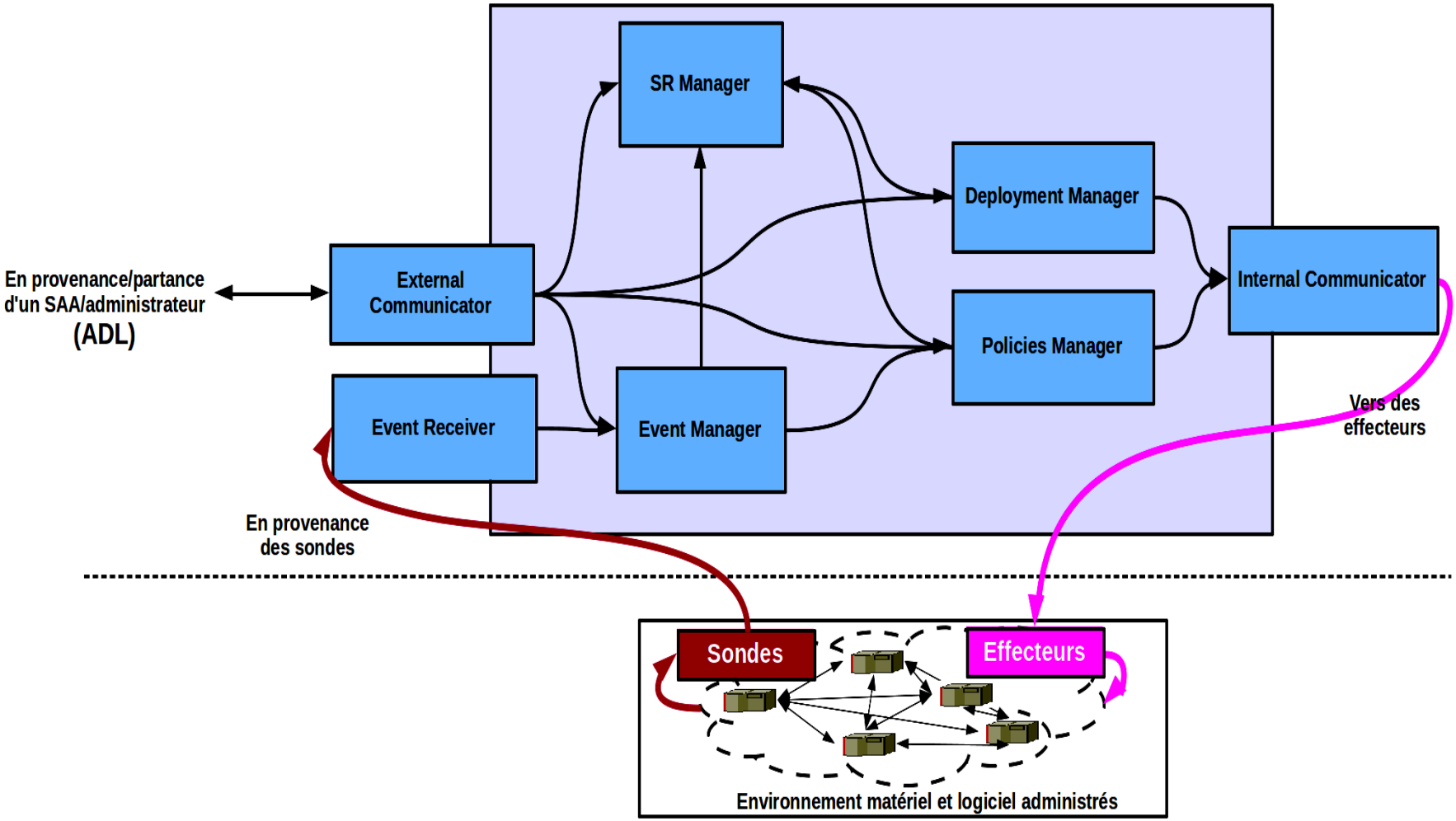
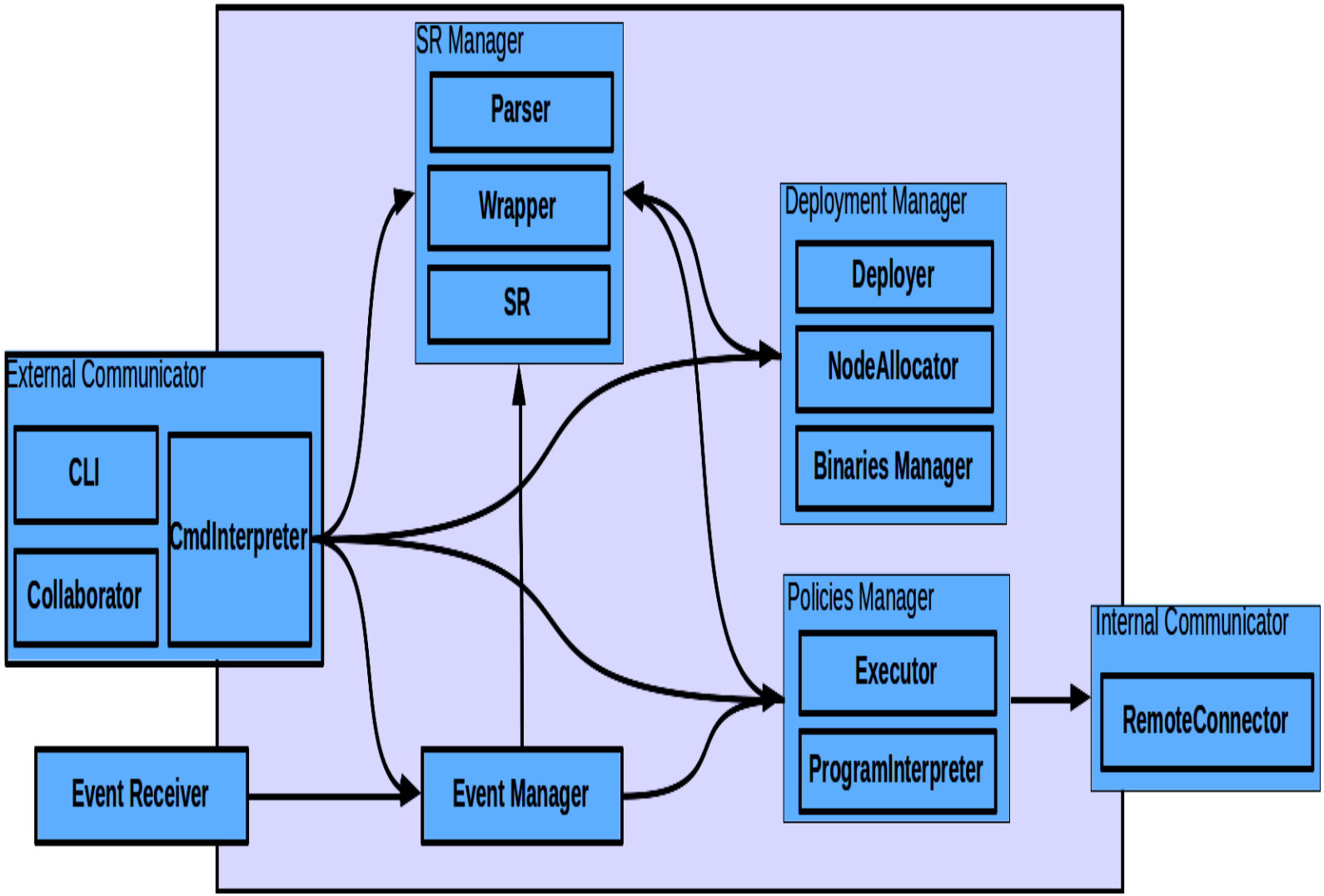


Figure – *Rappel du modèle d'un SAA*

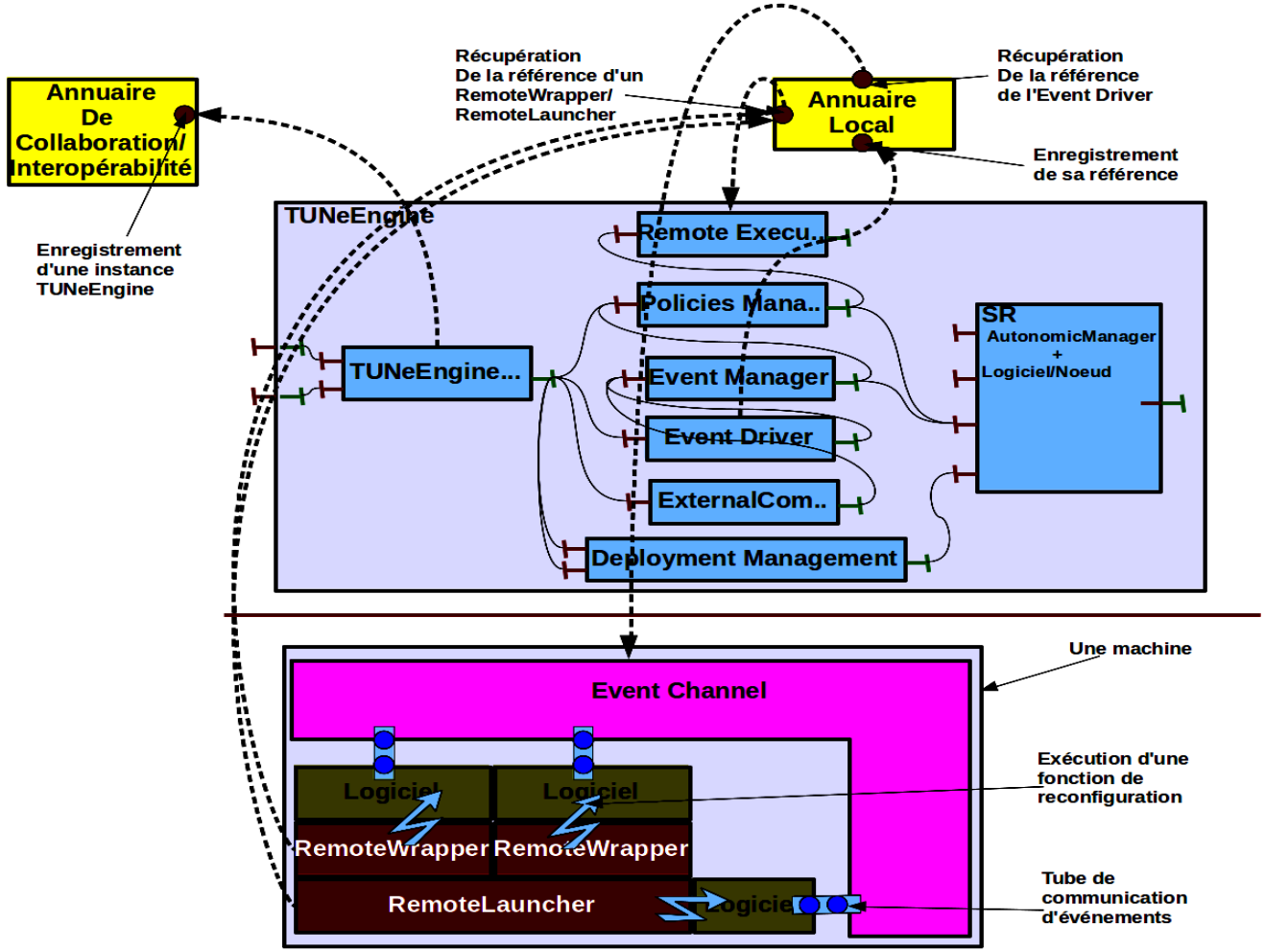
TUNeEngine : Architecture générale



TUNeEngine : Architecture détaillée



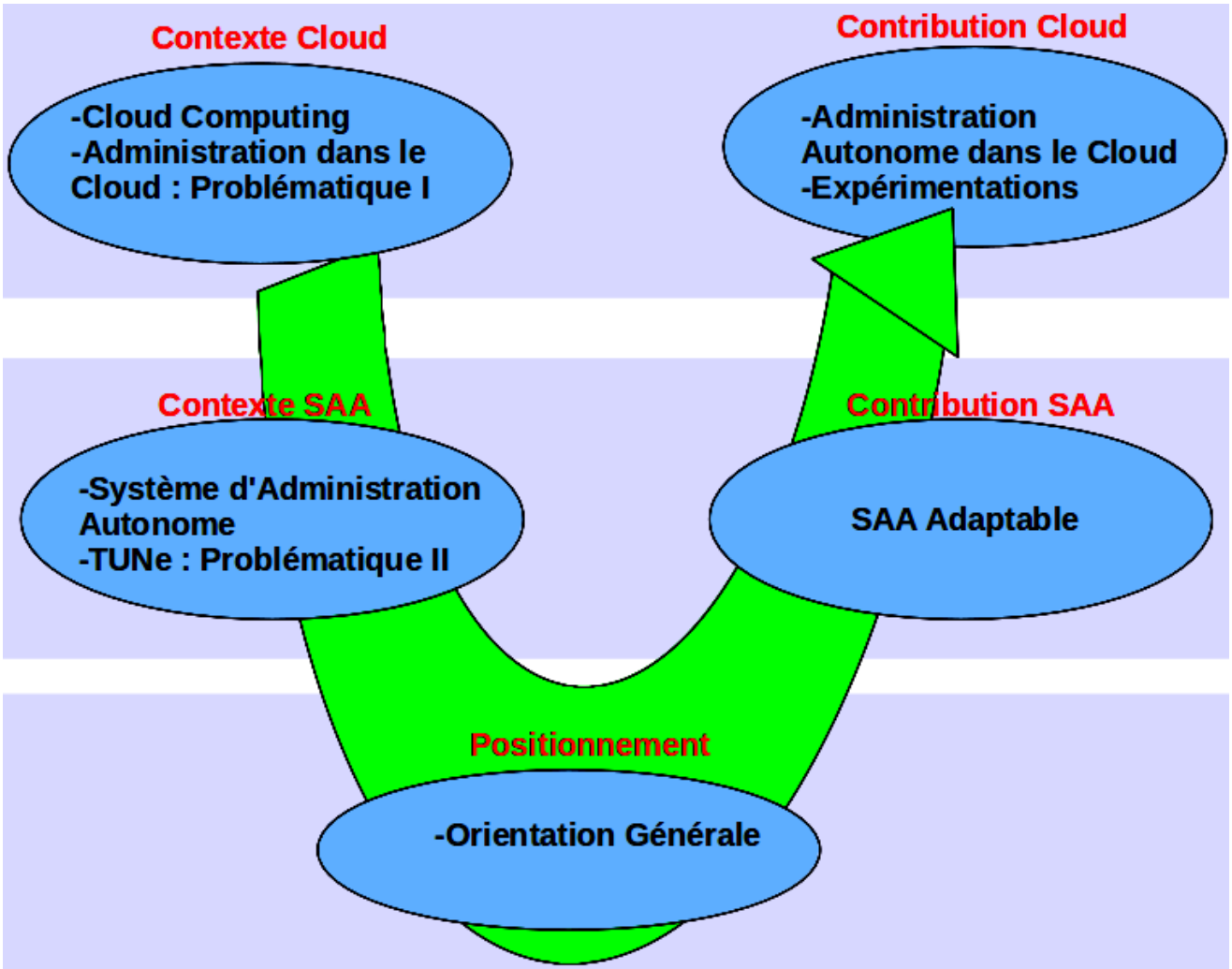
TUNeEngine : Implantation basée sur Fractal



Plan de validation

- 1re Validation : TUNeEngine pour une administration cluster
  - Version équivalent des services de TUNe
- 2e Validation en deux phases
  - TUNeEngine pour l'administration d'IaaS : CloudEngine
    - Consolidation et placement de VM
  - TUNeEngine pour l'administration d'applications J2EE dans le cloud
    - Scalabilité au niveau application
  - Gestion simultanée de ressources dans les deux niveaux

Étape 4

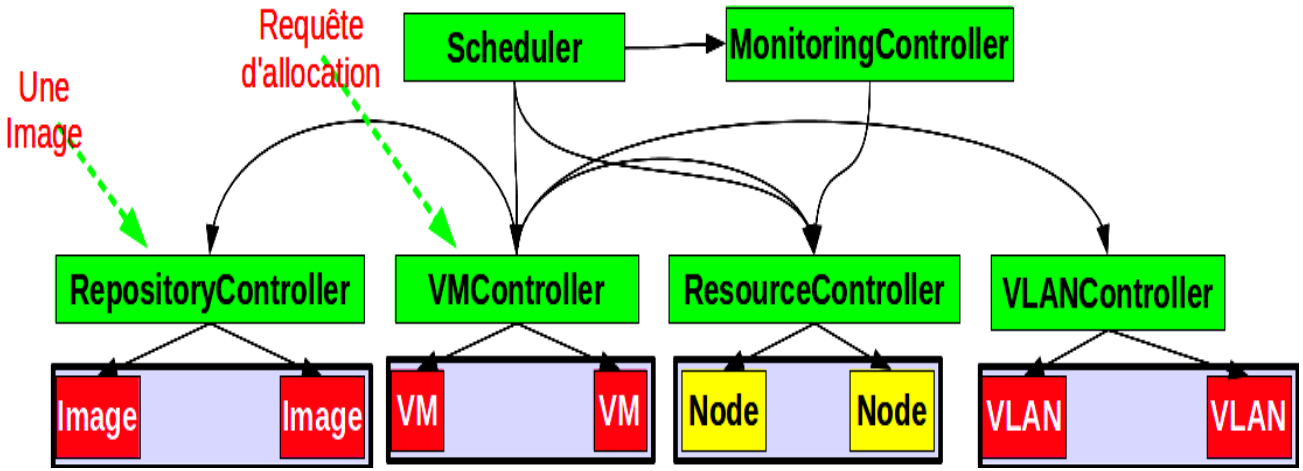

Contexte Cloud
-Cloud Computing
-Administration dans le Cloud : Problématique I
Contribution Cloud
-Administration Autonome dans le Cloud
-Expérimentations
Contexte SAA
-Système d'Administration Autonome
-TUNe : Problématique II
Contribution SAA
SAA Adaptable
Positionnement
-Orientation Générale

2e Validation : CloudEngine

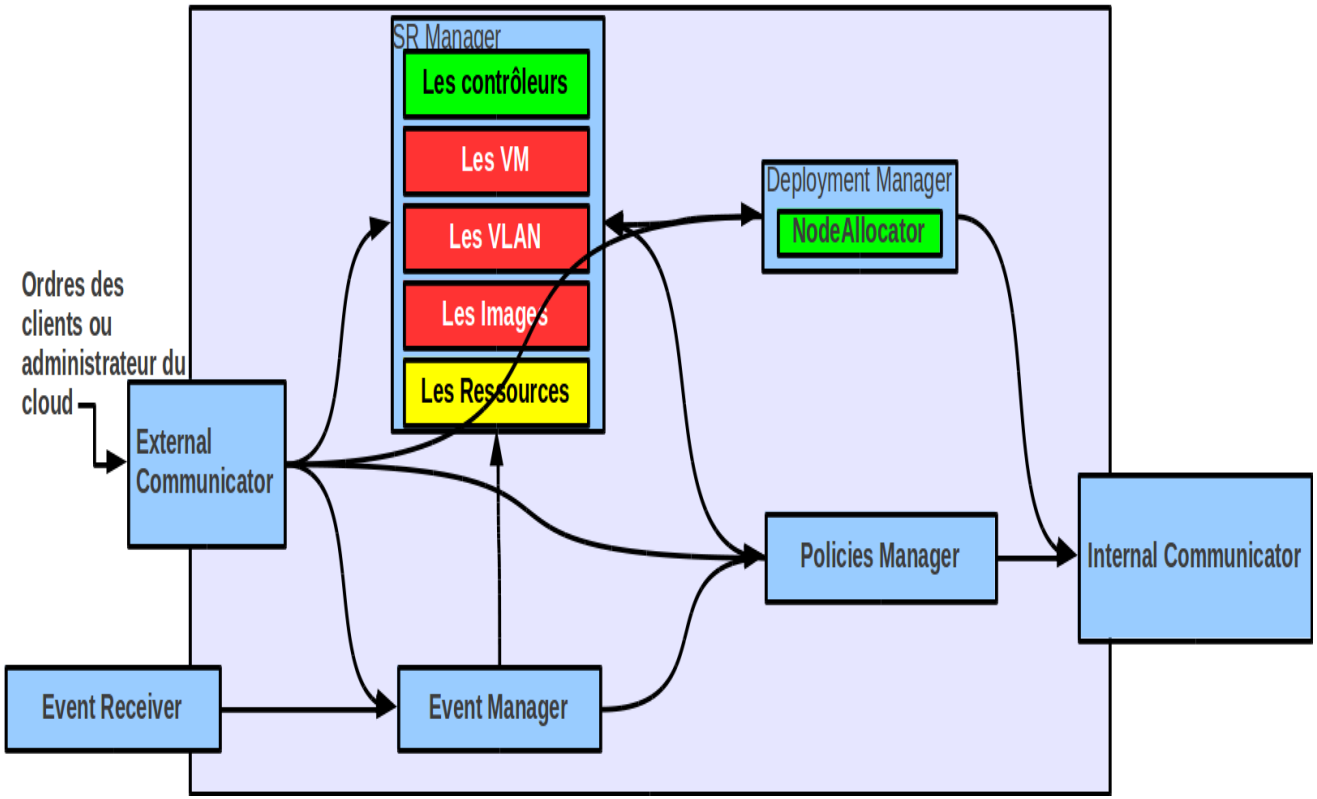
- CloudEngine est réalisé en 3 phases :
  - Construction des briques logicielles fournissant les services de base d'un IaaS
  - Intégration de ces briques à TUNeEngine
  - Personnalisation des composants de TUNeEngine pour l'auto-administration de CloudEngine

2e Validation : CloudEngine (Services de base)

- CloudEngine : Fournit les services de base d'un IaaS
  - Interfaçage avec des hyperviseurs
  - Gestion des VM, images et VLAN
  - Optimisation de l'utilisation des ressources



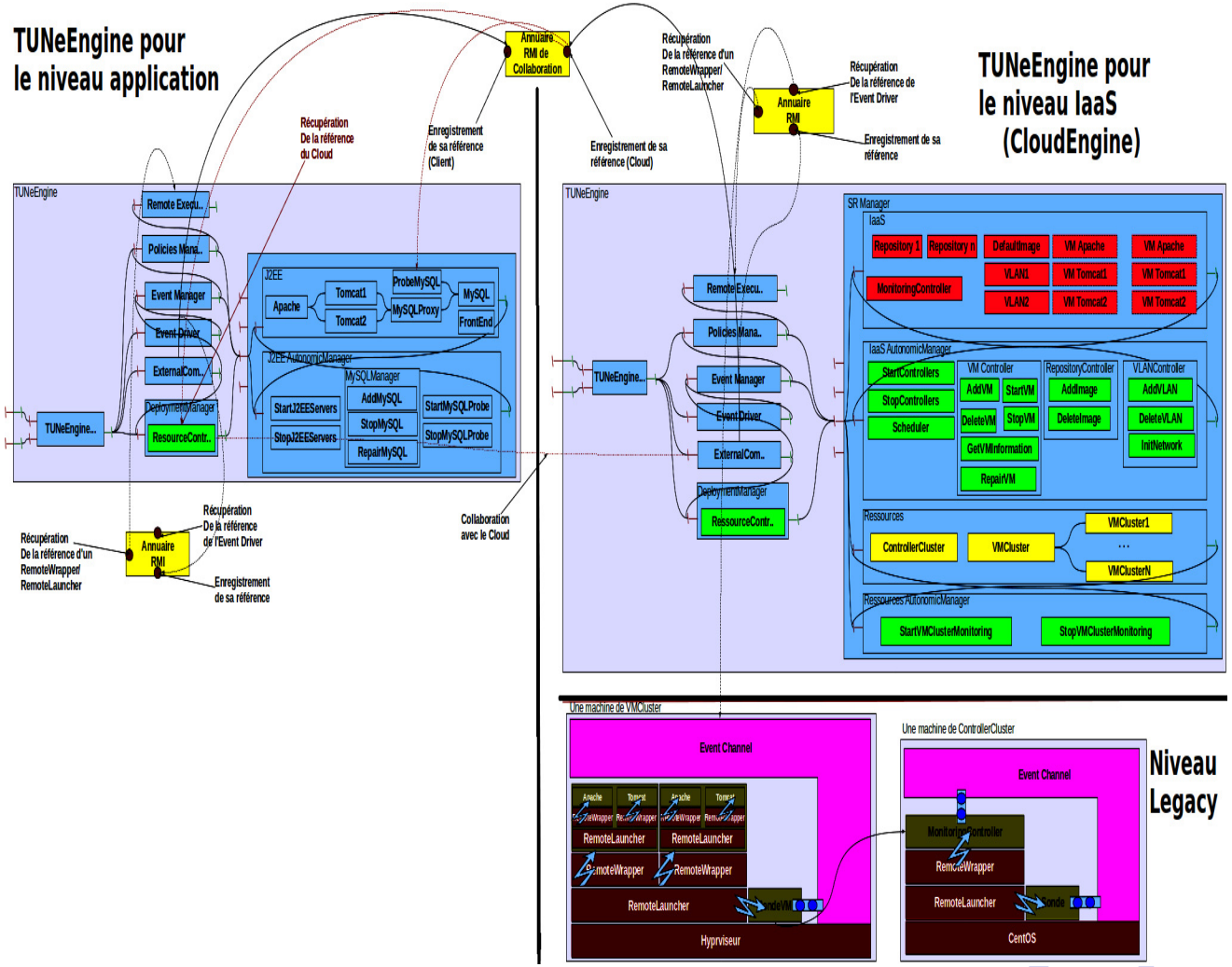
2e Validation : CloudEngine (Intégration à l'architecture de TUNeEngine)



2e Validation : 1) CloudEngine (Personnalisation de TUNeEngine)

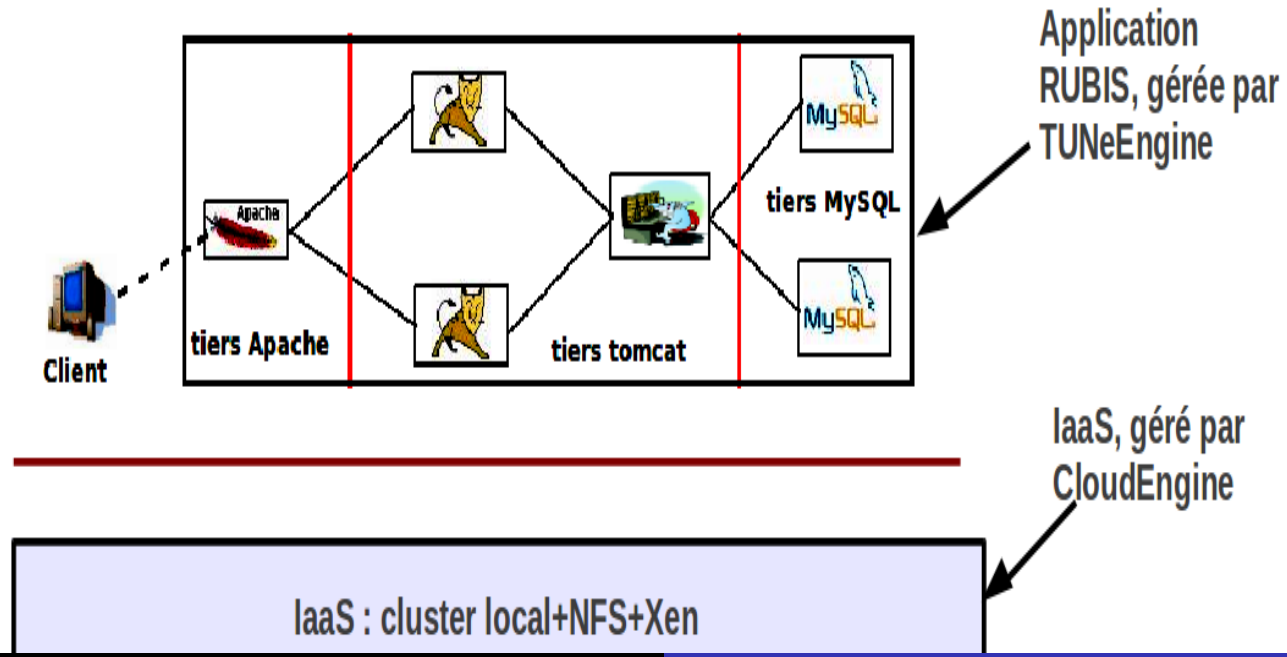
	Besoins d'administration	Spécialisation de TUNeEngine par rapport à TUNe
Administration des applications	-Administration des logiciels -Ajout dynamique d'instances de logiciels -Allocation de VM	-NodeAllocator -Collaborator
Administration de l'IaaS	-Administration de VM -Ajout dynamique de nouveaux éléments -Collaboration & Interopérabilité	-Wrapper -Deployer -Collaborator -EventManager

2e Validation : 1) CloudEngine (Finalité en Fractal)

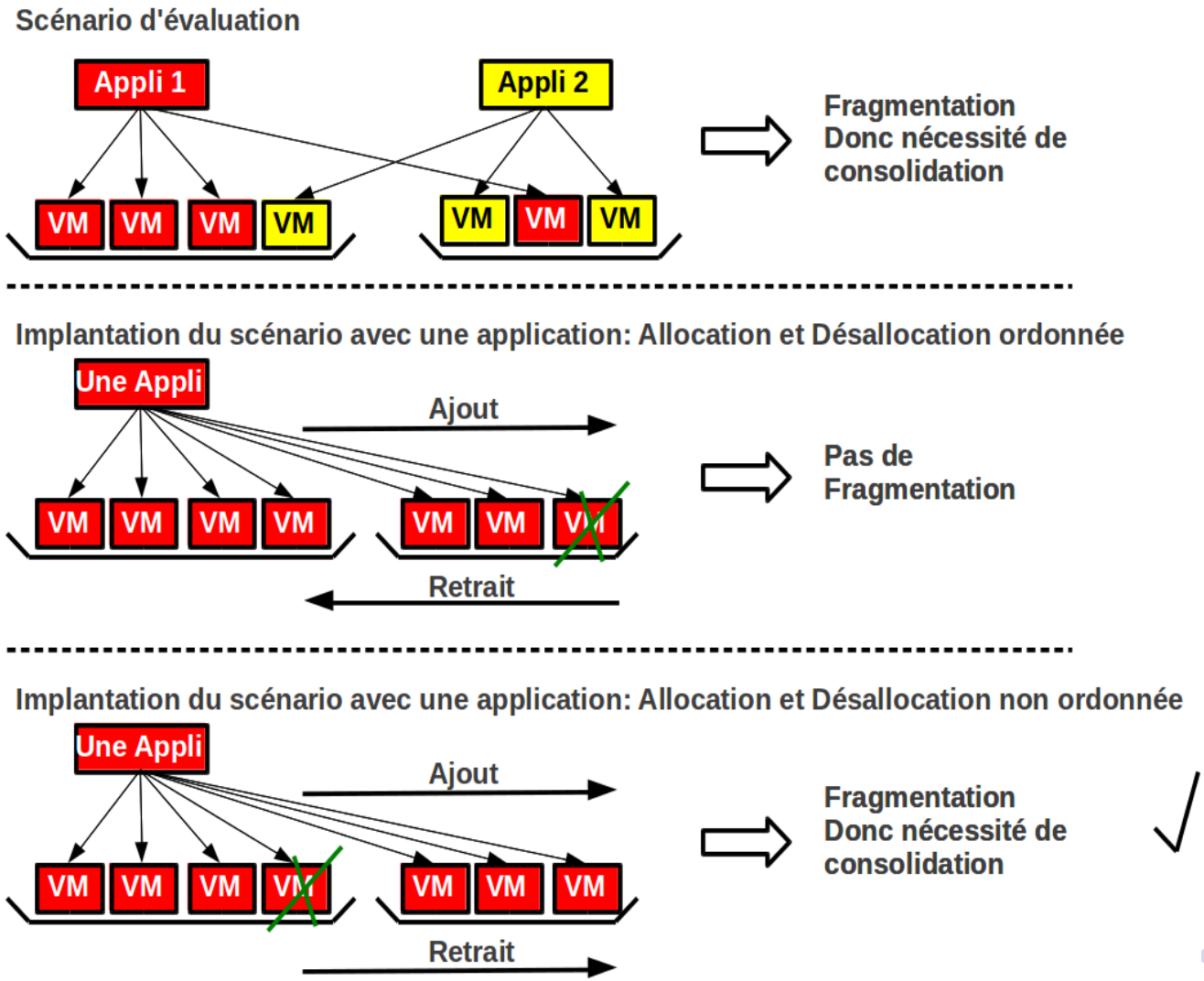


Contexte

- IaaS : Environnement local, NFS et Système de virtualisation Xen
- Application Cliente : RUBIS (site de commerce en ligne)
- Émulateur de clients web RUBIS
- Benchmark pyramidale : montée, plateau, descente



Scénario de validation

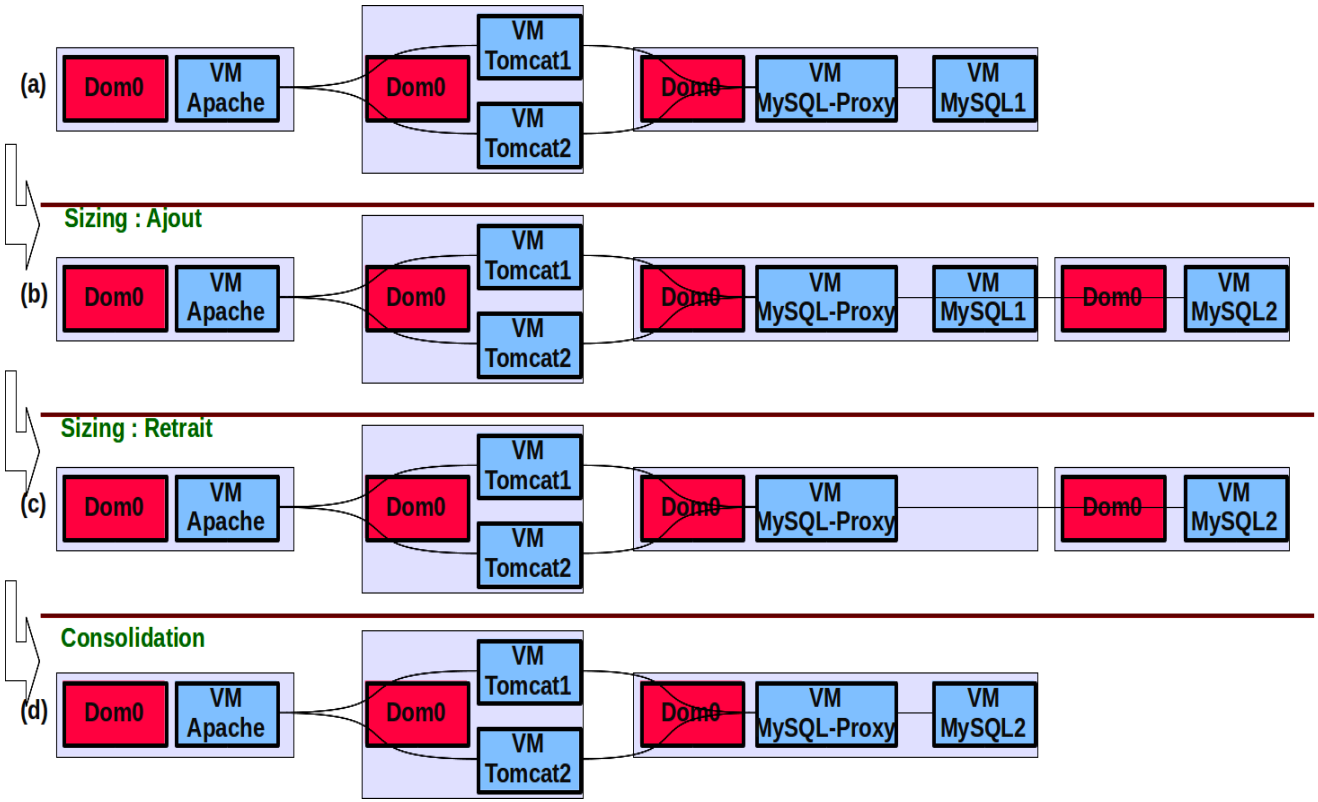


Contexte de la validation

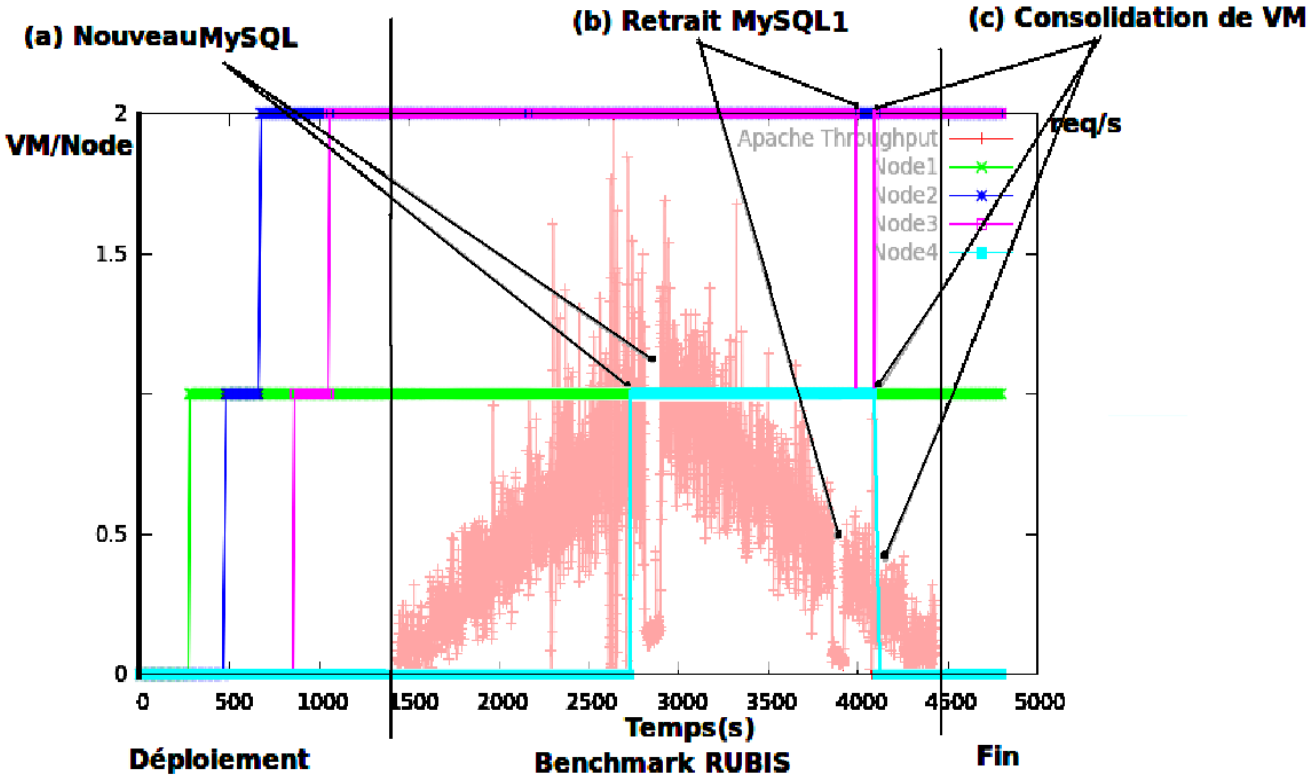
- Scalabilité
  - Scalabilité non ordonnée
  - Observation du tiers MySQL
  - Décision selon la charge CPU
  - Décision basée sur des seuils min et max
  - VM de taille fixe
- Placement et Consolidation des VM
  - Placement lors de l'allocation d'une VM
  - Placement par regroupement
  - Consolidation pendant l'exécution
  - Consolidation selon le quota de ressources (CPU et mémoire) réservé

Scénario

- Récapitulatif du scénario



Résultats


(a) NouveauMySQL
(b) Retrait MySQL1
(c) Consolidation de VM
VM/Node
req/s
Apache Throughput
Node1
Node2
Node3
Node4
2
1.5
0.5
0
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000
Temps(s)
Déploiement
Benchmark RUBIS
Fin

Plusieurs autres cas d'utilisation

- Nous en présentons 2

Tolérance aux pannes
Gestion exclusive : soit par le client (repair), soit par l'IaaS (checkpointing)
Gestion collaborative : partage de responsabilité entre IaaS (détection et redémarage de VM) et Client (redéploiement)
Consolidation et Scalabilité
VM de taille variable, Scalabilité ordonnée ==> Utilisation efficace des VM ==> L'augmentation de la taille des VM (migration potentielle) ==> Trou dans l'IaaS ==> Nécessite Consolidation pour tasser les trous ==> basée sur les la taille des VM (quotas CPU-Mémoire)

Conclusion : Problématiques

- Administration d'une plateforme de cloud
- Environnement Complexe
  - Hétérogène, Virtualisé, Évolutif
  - Deux niveaux d'administration (IaaS et Application)
  - Deux types d'utilisateurs (Fournisseur et Clients)
- Solution par l'administration autonome
  - Unique SAA pour les deux niveaux
  - Adaptable, Flexible, Collaboratif et Interopérable

Validations

- Une version analogue à TUNe pour la gestion d'applications sur cluster
- Administration d'un cloud
  - Au niveau des applications Clientes
  - Au niveau de l'IaaS (CloudEngine)
- Validation par cas d'utilisation dans le cloud
  - Tolérance aux pannes
  - Gestion de ressources

Travaux futurs à court terme

- Application de TUNeEngine dans les clouds existants
- Extension de CloudEngine
  - Construction d'images via OVF
  - Politiques fines de consolidation (Entropy)
  - Coordination des boucles de contrôle
  - Gestion des utilisateurs

Travaux futurs à long terme

- Support de langages d'administration
  - yTUNe : Éditeur de DSL pour l'administration
  - Construction des DSL propres au cloud
- Adaptabilité au niveau système
  - Cas des hyperviseurs

FIN

- Merci pour votre attention.