Systèmes et Algorithmiques Répartis

Master 1
Informatique des Organisations - MIDO

Joyce EL HADDAD
joyce.elhaddad@dauphine.fr

Informations Administratives

- Cours-TD intégré
  - *http://www.lamsade.dauphine.fr/~elhaddad/Courses/SAR/*

- Formation
  - Apprentissage : 14 séances de 3h
  - Initiale : 18h de cours (1h30 par semaine), 18h de TD (1h30 par semaine)

- Contrôle de connaissance
  - Apprentissage : Examen
  - Initiale : Examen

Plan

- Chapitre 1 : Généralités
- Chapitre 2 : La communication
- Chapitre 3 : Le temps
- Chapitre 4 : La concurrence

Références

- A. S. Tanenbaum et M. van Steen, Distributed Systems Principles and Paradigms, Pearson Prentice, Hall, 2007

- G. Tel, Introduction to Distributed Algorithms, Cambridge University Press, 2000

- M. Singhal et A. D. Kshemkalyani, Distributed Computing : Principles, Algorithms, and Systems, Cambridge university press, 2011

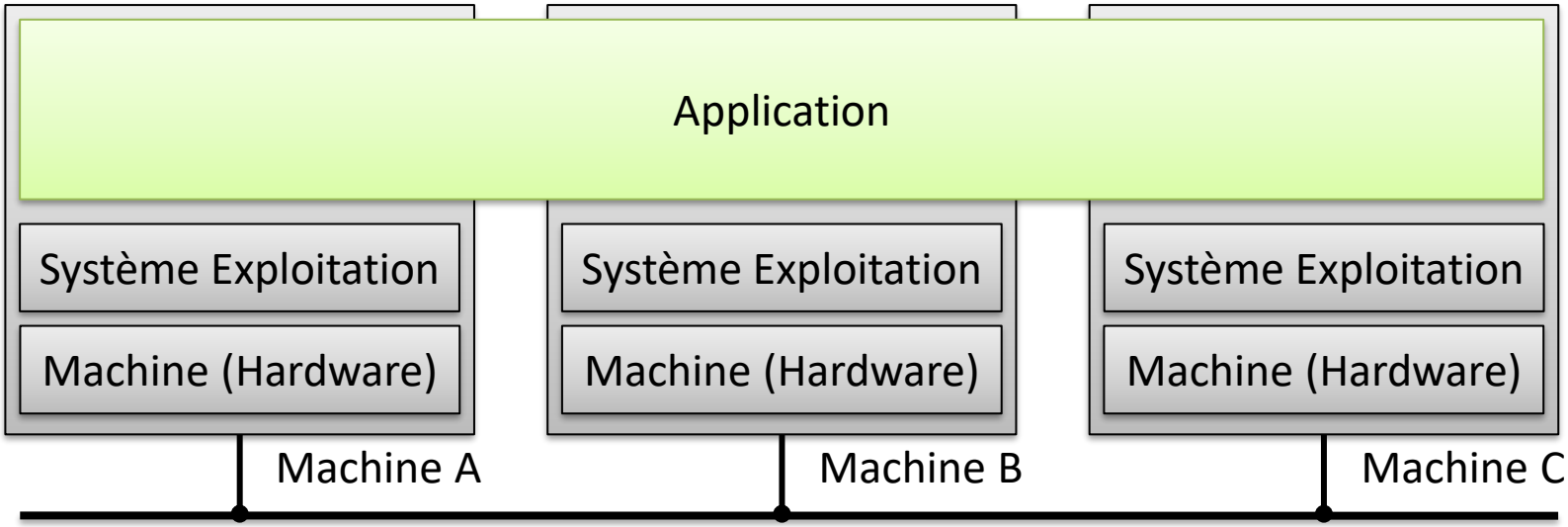
- S. Ghosh, Distributed Systems : an Algorithmic Approach, Chapman and Hall, 2011

Chapitre 1 : Généralités

- Introduction
- Modèle de répartition
- Déroulement de l'application
- Scénario, Évaluation, Vérification

Introduction

- *« Un système réparti est un ensemble d'ordinateurs indépendants qui apparaît à un utilisateur comme un système unique et cohérent. » [Tanenbaum 94]*



Un système distribué [Tanenbaum 1994]

Introduction

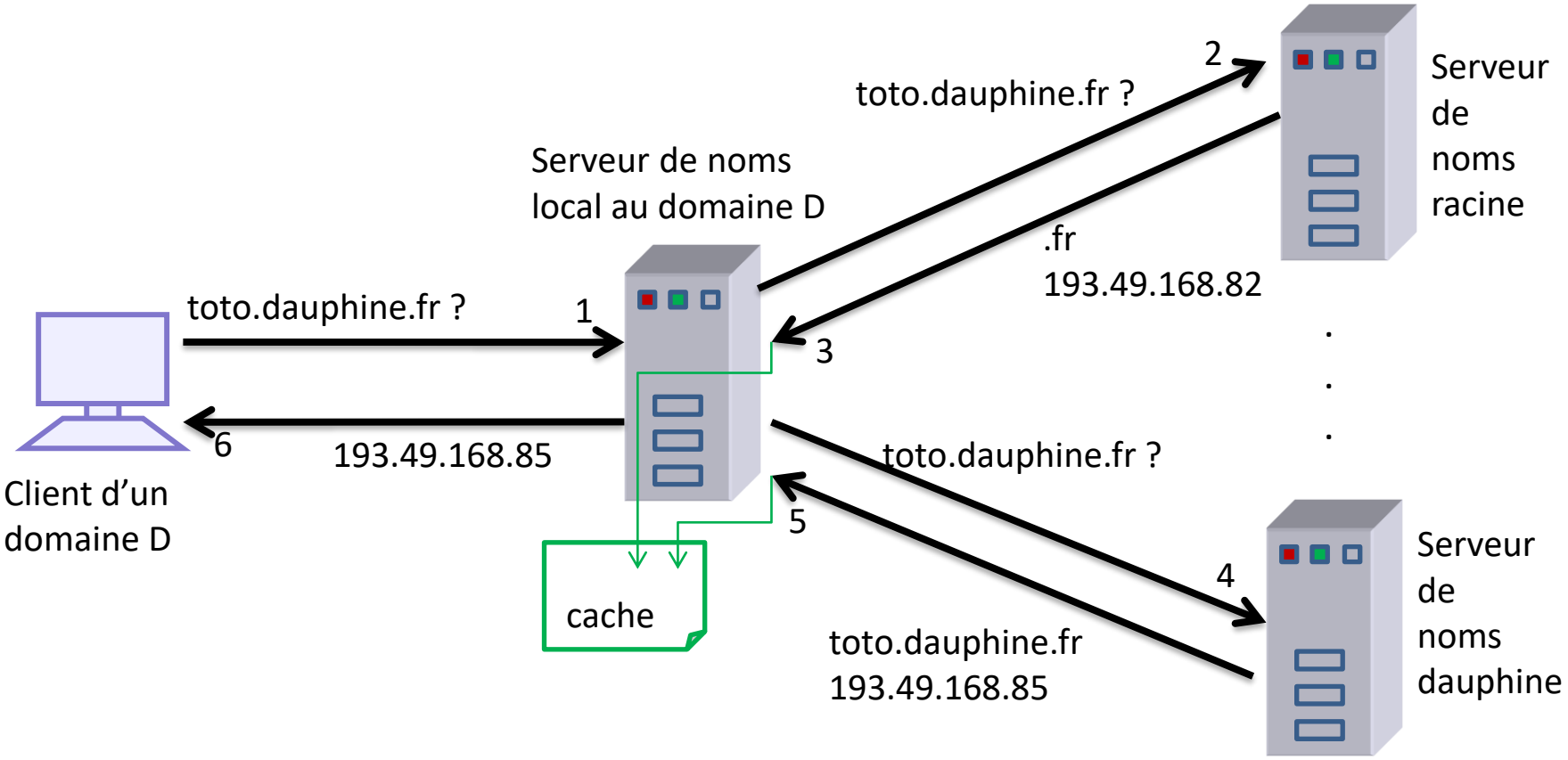
- Avantages d'un système réparti
  - Performance : puissance de calcul et de stockage importante
  - Fiabilité : résistance aux pannes logicielles et matérielles.
  - Souplesse : modularité
  - Evolutivité (scalability): passage à l'échelle aisé
  - …
- Exemples de systèmes répartis
  - DNS (Domain Name System)
  - Systèmes multi-couches
  - Peer-to-peer
  - Cluster, Grid, Cloud …

Introduction

- DNS (Domain Name System)
  - Trouver une adresse IP associée à un nom symbolique d'une machine.
  - Les noms symboliques sont organisés en domaines *Exemple : lamsade.dauphine.fr → fr (nom de domaine de 1er niveau), dauphine (nom de domaine de 2ème niveau) et lamsade (nom de la machine hôte)*
  - Une (grande) base de données répartie, réalisée par un ensemble de serveurs de noms qui communiquent entre eux

Introduction

- DNS (Domain Name System)



Modèle de répartition

- Tendance majeure des systèmes informatiques : la répartition des traitements entre des "entités" coopératives.

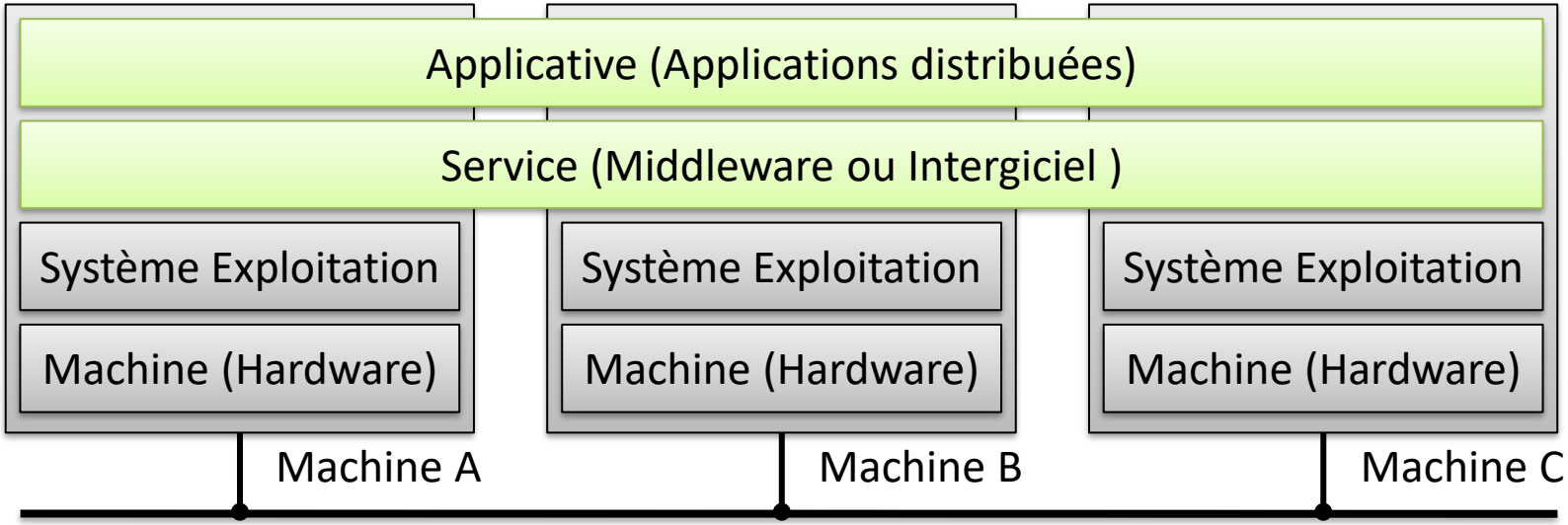
- Conception d'applications réparties se heurte à des difficultés algorithmiques spécifiques à l'activité concurrente des entités et à l'absence de mémoire partagée.

Modèle de répartition

- Le concepteur doit résoudre des problèmes liés à son application (messagerie, forum, etc.) mais aussi des problèmes liés à son modèle d'application comme :
  - La communication entre entités,
  - La portabilité et l'interopérabilité des codes,
  - L'exclusion mutuelle entre sections de code critiques,
  - …

Modèle de répartition

- Une manière naturelle d'aborder la conception consiste à construire l'application en couches.
  - Une couche applicative dont les traitements sont spécifiques aux fonctionnalités recherchées.
  - Une couche service qui résout des problèmes génériques et récurrents dans le domaine de la répartition.



Un système distribué [Tanenbaum 1994]

Modèle de répartition

- L'environnement réparti est composé :
  - Entités actives (machines, sites, stations, processus ou processeur) disposant d'une mémoire dédiée non accessible aux autres entités. Chaque station dispose d'une identité unique et numérique *(cette adresse pourrait être l'adresse MAC de sa carte réseau ou son adresse IP)*
  - Lignes de communication bipoints *(reliant deux stations)* et bidirectionnelles *(la transmission d'information est possible dans les deux directions)*. Chacune des directions est appelée un canal. **Ces lignes sont le seul moyen d'échange d'information entre les stations.**

Modèle de répartition

- Un graphe de communication non orienté est obtenu en considérant les stations comme des nœuds et les lignes de communications comme des arêtes

- Hypothèse n°1 :
  - Sauf mention du contraire, nous supposons que le graph de communication est une clique *(toute paire de nœuds est reliée par une arête)*

Modèle de répartition

- Chaque station est intéressante en tant que composante d'une application répartie.

- Une station est découpée en trois couches
  - Couche application : réalise les traitements spécifiques et fait appel à la couche service pour certaines fonctionnalités génériques.
  - Couche service : la réalisation de cette couche est l'objet de cette partie du cours et nous désignons par algorithme réparti l'ensemble des codes associés.
  - Couche réseau : pour l'échange des messages, la couche service se repose sur l'interface de la couche réseau dont la qualité de service dépend de l'environnement.

Modèle de répartition

- Couche réseau et le réseau
  - Hypothèse n°2 : sauf mention du contraire, un canal de communication reliant un site S_i à un autre site S_j a les propriétés suivantes :
    - Les messages ne sont pas altérés,
    - Les messages ne sont pas perdus,
    - Le canal est FIFO = *l'ordre temporel de réception des messages de S_j venant de S_i est identique à l'ordre d'émission des messages de S_i à destination de S_j*

Modèle de répartition

Couche réseau et le réseau

- Bien qu'un message soit certain de parvenir à son destinataire, deux situations sont à considérer :
  - Le délai de transit d'un message est quelconque, réseau asynchrone.
  - Le délai de transit d'un message est borné par un délai maximum, réseau synchrone.

- Hypothèse n°3 : sauf mention du contraire, nous supposons que le réseau est asynchrone.

Modèle de répartition

Couche réseau et le réseau

- La couche réseau fournit un mécanisme de transfert de message pour la couche service.

- Deux options sont possibles pour ce mécanisme
  - Communication synchrone : primitives synchrones (bloquantes).
  - Communication asynchrone : primitives asynchrones (non bloquantes).

Modèle de répartition

Couche réseau et le réseau

- Une émission est synchrone (resp. asynchrone) si le processus émetteur est bloqué (resp. n'est pas bloqué) jusqu'à la réception d'un acquittement de son message.

- Une réception est synchrone (resp. asynchrone) si le processus destinataire appelle explicitement (resp. n'appelle pas) une primitive pour recevoir un message et se bloque en attente de ce message.

- Il ne faut pas confondre le synchronisme de la couche réseau (communication) et celui du réseau. Toutes les combinaisons sont possibles.

Modèle de répartition

Couche réseau et le réseau

- Une communication synchrone conduit à des applications plus simples à vérifier et à concevoir, une communication asynchrone conduit à des applications plus flexibles où le concepteur peut choisir lui-même les points d'attente.

- Hypothèse n°4 : sauf mention du contraire, la communication est asynchrone.

Modèle de répartition

- Couche réseau et le réseau
  - Les primitives d'émission seront :
    - Envoyer_à (destinaire,message) : permet d'envoyer un message au destinataire indiqué et de poursuivre son exécution. Le destinataire est désigné par son identité et le message peut être structuré selon les besoins de la couche service.
    - Diffuser (message) : permet d'envoyer un message à tous les autres sites participant à l'application. L'utilisation de cette primitive n'est possible que si le graphe de communication est une clique.
  - Les primitives d'émission seront appelées dans l'algorithme réparti par la couche service.

Modèle de répartition

Couche réseau et le réseau

- Le trajet d'un message m :
  - A l'émission : la couche service de S_i appelle la primitive Envoyer_à (S_j,m), puis le message circule sur le canal.
  - A la réception : le message parvient à la couche réseau de S_j. Le traitement du message par S_j dépend du service mais puisque la réception est asynchrone, aucune primitive n'est appelé par la couche service.

- La couche réseau doit appeler une primitive "écrite" par le service qui fait partie de l'algorithme réparti. Cette primitive est une primitive de l'interface service/réseau.

Modèle de répartition

- Couche réseau et le réseau
  - Une primitive montante est une primitive écrite par la couche supérieure et appelée par la couche inférieure.
  - Une primitive descendante est une primitive écrite par la couche inférieur et appelée par la couche supérieure (comme Envoyer_à).
  - La primitive de réception sera :
    - Sur_réception_de (émetteur,message) : spécifie le traitement à effectuer sur réception d'un message. L'émetteur est désigné par son identité et le message peut être structuré selon les besoins de la couche service.

Modèle de répartition

Couche service

- Des variables qui lui sont propres. Une variable var d'une station S_p sera désignée dans l'algorithme par var_p pour insister sur le caractère local de la mémoire.
- Des primitives utilisées pour les traitements de l'algorithme.
- Des primitives descendantes de l'interface application-service et des primitives montantes (Sur_réception_de) de l'interface service-réseau.
- Occasionnellement, un processus interne au service lorsqu'une tâche devra être exécutée de manière concurrente au fonctionnement de l'application.

Modèle de répartition

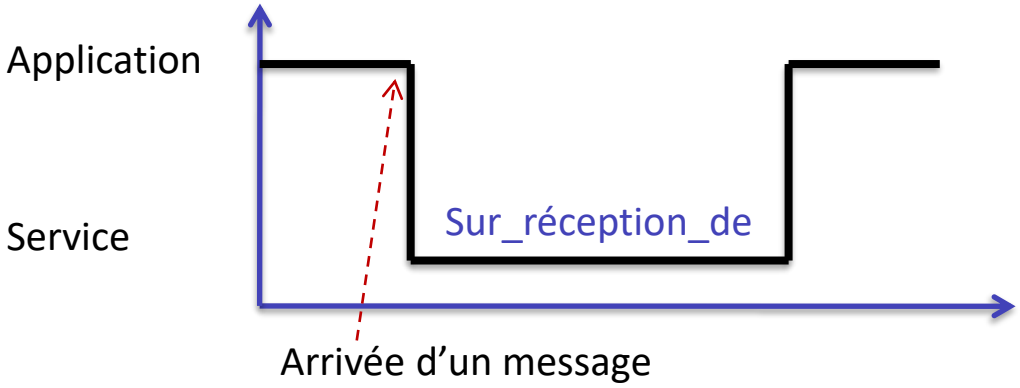
- Couche service
  - Quelques extensions :
    - Un processus d'application ou de service peut être suspendu jusqu'à ce qu'une condition soit remplie (elle peut l'être immédiatement). La primitive utilisée sera Attendre (Expression booléenne).
    - Un processus peut aussi être suspendu jusqu'à l'expiration d'un timeout. Dans ce cas, la primitive utilisée sera Attendre(). La primitive qui arme le temporisateur sera Armer (délai).
    - Des expressions booléennes avec les quantificateurs $\exists$ et $\forall$.

Modèle de répartition

- Couche application
  - Hypothèse n°5 : sauf mention du contraire, nous supposons que l'application est exécutée sur un site par un unique processus.
  - Le langage de programmation utilisé sera pseudo-C

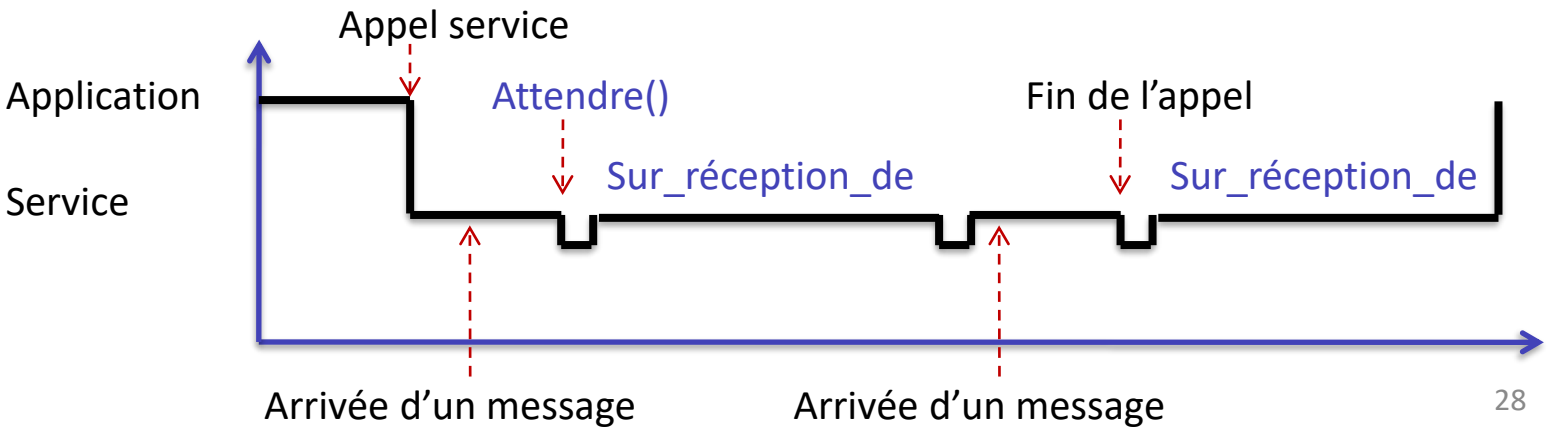
Déroulement de l'application

- Système d'exploitation capable de mettre en œuvre 4 règles pour l'interaction entre les différentes couches lors du déroulement de l'application :
  - R1 : si le processus applicatif exécute du code de la couche application, toute réception d'un message donne lieu à un déroutement et à l'exécution de la primitive Sur_réception_de correspondante. Le traitement applicatif se poursuit ensuite.



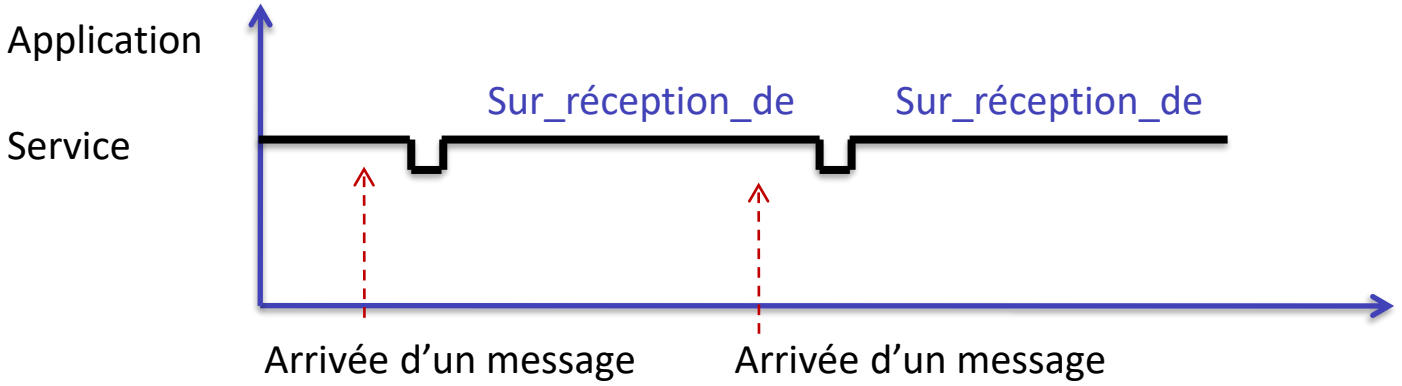
Déroulement de l'application

- Système d'exploitation capable de mettre en œuvre 4 règles pour l'interaction entre les différentes couches lors du déroulement de l'application :
  - R2 : si le processus exécute du code de la couche service, l'exécution de la primitive Sur_réception_de suite à une réception de message est retardée jusqu'à un blocage du processus (par la primitive Attendre) ou jusqu'au retour en couche application.



Déroulement de l'application

- Système d'exploitation capable de mettre en œuvre 4 règles pour l'interaction entre les différentes couches lors du déroulement de l'application :
  - R3 : si le système exécute une primitive Sur_réception_de suite à une réception de message, tout exécution de Sur_réception_de suite à une autre réception de message est retardée jusqu'à la fin de l'exécution de la première primitive.



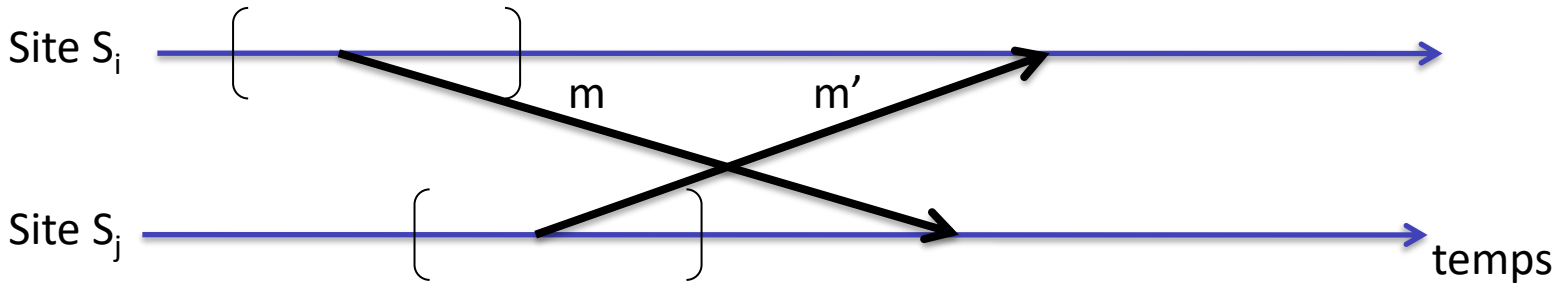
Déroulement de l’application

- Système d'exploitation capable de mettre en œuvre 4 règles pour l'interaction entre les différentes couches lors du déroulement de l'application :
  - R4 : le code des primitives Sur_réception_de ne comporte pas d'appel à la primitive Attendre.

Scénario, Evaluation, Vérification

Scénario d'exécution

- Illustration des algorithmes par un *chronogramme*
  - L'activité de chaque site est représentée sur un axe temporel orienté de gauche à droite (ou de haut en bas).
  - Les transferts de message sont dénotés par des segments orientés dont l'origine et l'extrémité sont respectivement associées à l'envoi et à la réception de message.



Scénario, Evaluation, Vérification

Evaluation

- Les mesures usuelles de complexité en algorithmique standard sont :
  - La taille de l'espace occupé par l'algorithme,
  - Le temps d'exécution (ou de manière équivalente le nombre d'instructions exécutées) de l'algorithme.

- En algorithmique répartie, sauf cas particulier, l'espace n'est pas un facteur déterminant et le temps d'exécution des primitives est négligeable devant le temps de transit des messages.

Scénario, Evaluation, Vérification

Evaluation

- Deux mesures de complexité pertinentes pour un algorithme réparti :
  - Le trafic sur le réseau : décompté par le nombre maximum de messages échangés en fonction du nombre de sites *(le plus souvent noté n)* de l'application.
  - Le temps d'exécution maximum : de l'algorithme en fonction du nombre de sites de l'application. Le temps d'exécution du code est égal à 0. Le temps maximum de transit de tous les messages.

Scénario, Evaluation, Vérification

Vérification

- Un algorithme réparti donne lieu à plusieurs exécutions possibles *(en raison du délai aléatoire de transit d'un message).*
- On considère comme représentation formelle du "comportement", l'ensemble des exécutions possibles.
- Les propriétés à vérifier sont généralement de deux natures : des propriétés de sûreté et des propriétés de vivacité.
- La vérification peut s'effectuer de manière manuelle en utilisant des techniques élémentaires comme l'induction *(vérifier que la propriété est vérifiée à l'état initial puis que, si elle est vérifiée en un état, elle est vérifiée dans tout état qui le suit).*

Références

- G. Tel, "Introduction to Distributed Algorithms", Second Edition Cambridge University Press, ISBN 0-521-79483-8, 2000
- A. Tanenbaum "Systèmes d'exploitation. Systèmes centralisés. Systèmes distribués", troisième Edition Dunod–Prentice Hall, ISBN 2-10-004554-7, 1994
- M. Raynal, "Synchronisation et état global dans les systèmes répartis", Collection Direction des Etudes et des Recherches d'EDF n°79, Hermès, ISSN 0399-4198, 1992
- M. Raynal, "Gestion de données réparties : problèmes et protocoles", Collection Direction des Etudes et des Recherches d'EDF n°82, Hermès, ISSN 0399-4198, 1992
- M. Raynal, "La communication et le temps dans les réseaux et les systèmes répartis", Collection Direction des Etudes et des Recherches d'EDF n°75, Hermès, ISSN 0399-4198, 1991