Présentation de l'architecture COM Compinent Object Model DCOM/ACTIVEX

Yves LALOUM

Conseil Audit de Systèmes d 'information CISA

ylaloum@advisehr.com

1.Introduction

· Les services de composant COM+ constituent un cadre général logiciel de COM conçues pour «gérer» des composants logiciels

Ces services sont intégrées à partir de Windows 2000 (NT5)

COM+ est le résultat d'une lente évolution qui débuta en 1990

· **1990 OLE/OLE 2** (Object Linking & embedding): Architecture de base de Windows. Permet la coopération Multi-Utilisateurs (documents composites)

· **1992COM** (Component Object Model) est livré avec Windows98 et Windows NT 4.0. Cœur de OLE2. Base de l'Internet aussi.

· **1996 DCOM** est la version pour les objets Distribués. IL Permet à un composant de se "déplacer" et de s'exécuter sur une autre machine.

Introduction

- **1996: ActiveX** version Internet d'accès aux objets COM/DCOM. Les contrôles ActiveX permettent de faciliter la distribution des composants à travers l'Internet, et d'intégrer DCOM sur le navigateur Web.
- **1997 MTS** (Microsoft Transaction Server):
  - C'est le modèle de serveurs d'objets COM dédié aux applications transactionnelles.
  - C'est en fait un environnement où tournent les composants (Runtime environment)
  - Il regarde les requêtes arriver aux composants et participe à leur exécution.
  - Il fournit : la sécurité, la gestion des transactions et celle de la répartition des charges.
- **1998 Début de COM+**

2. Modèle COM

- Le Component Object Model est un standard publié est conçu par Microsoft pour Microsoft
- Il implémente des « connexions » entre les différents composant logiciels : c’est un bus logiciel
- Il est indépendant du langage de programmation, du système d’exploitation et de la plate-forme

Caractéristiques des objets COM

- Pour utiliser COM dans un programme, il faut implémenter **des objets COM** Qui ont les caractéristiques suivantes :
- Encapsulation :
  - un objet COM est un paramétrage de données et /ou de fonctions enveloppées dans une entité identifiable unique.
  - Tout ce qui se trouve dans un objet COM est caché de l'extérieur de l'objet a l'exception des interfaces (a travers un pointeur vers un pointeur pointant sur une **table de pointeur de fonction**) !

Caractéristiques des objets COM

- · Les interfaces :
  - **IID (Interface Identifier) :** IID est un type de GUID (Globally Unique Identifier) entier de 16 octets qui permet d'identifié
    - Le nombre de fonctions dans la table
    - leur ordre
    - leurs signatures
  - Les trois premières méthodes d'une interface sont prédéfinies, en signature comme en signification
    - **QueryInterface():** Negotiation d'interface pour la découverte dynamique
    - **Addref()** et **Release()**: Comptage de référence pour la gestion de la durée de vie de l'objet

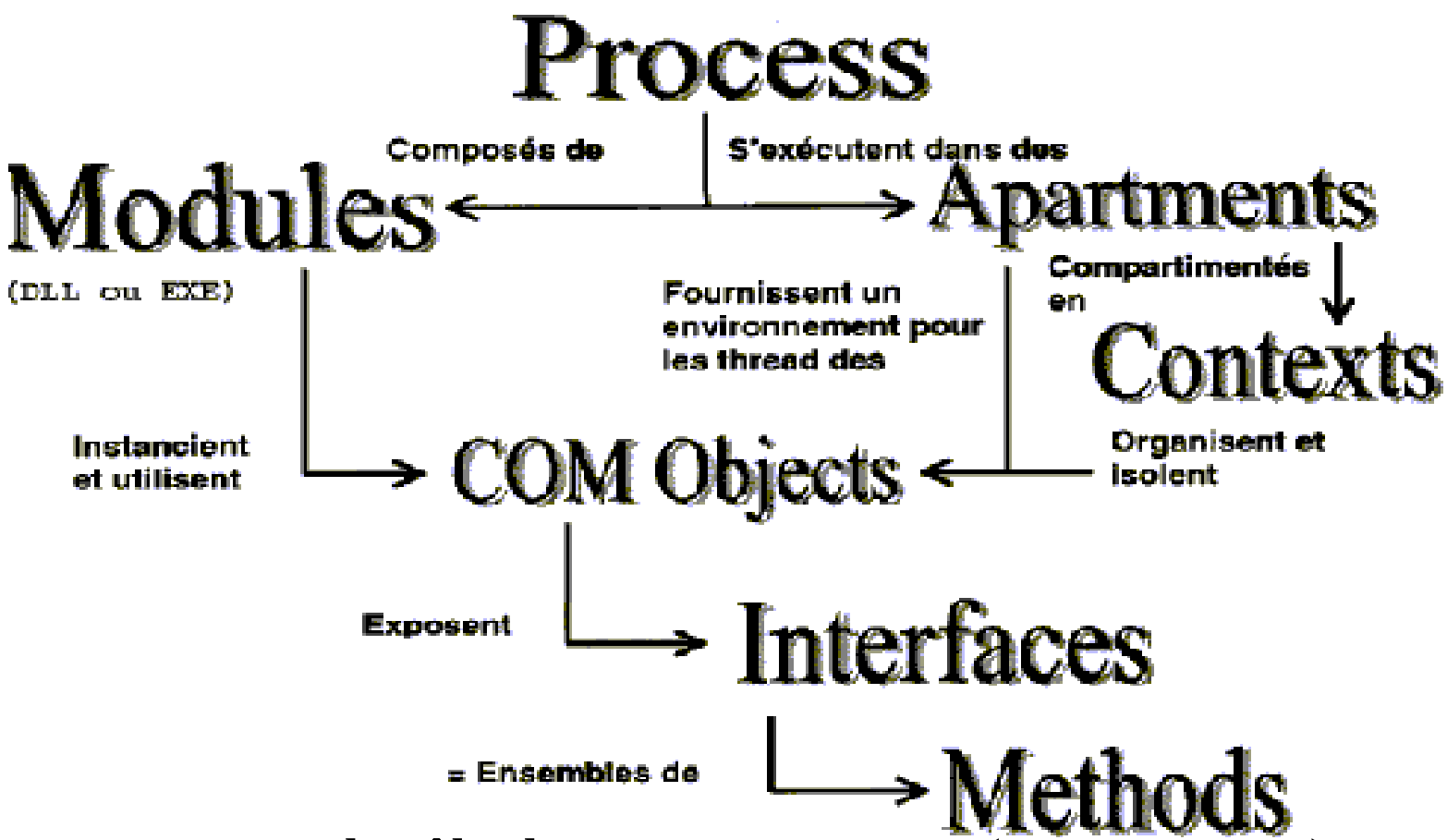
Support de COM par le système d'exploitation

- **COM implique l'existence de code au niveau de l'OS de la machine.**
- **Ce code forme le « middleware » , structuré sous la forme d'une bibliothèque (réalisée par des *DLL*s ou EXE).**
- **Cette bibliothèque comporte entre autres :**
  - **L'API de COM, formé par un petit nombre de fonctions accessibles par le client et le serveur.**
  - **Le Service Control Manager (ou SCM) : qui localise les serveurs des classes a l'aide des CLaSsID de la base registre.**
  - **L'appel de procédure distante par RPC: lorsque l'objet et le client ne sont pas au même endroit**

Support de COM par le système d'exploitation

- Ces bibliothèques fournissent un support pour :
  - Identification : identifier les programmes qui veulent implémenter un type d'objet (CLSID dans la base de registre)
  - Instanciations : Instancié l'objet voulu dans un module différent de celui du client
  - Marshalling : accéder indirectement aux interfaces d'un objet COM
- Les fonctions et les objets COM qui communiquent avec le système d'exploitation utilise des chaînes UNICODE

Hiérarchie de COM



Remarque : le rôle des « Apartments » (Appartement) est d'implémenter la sécurité des threads

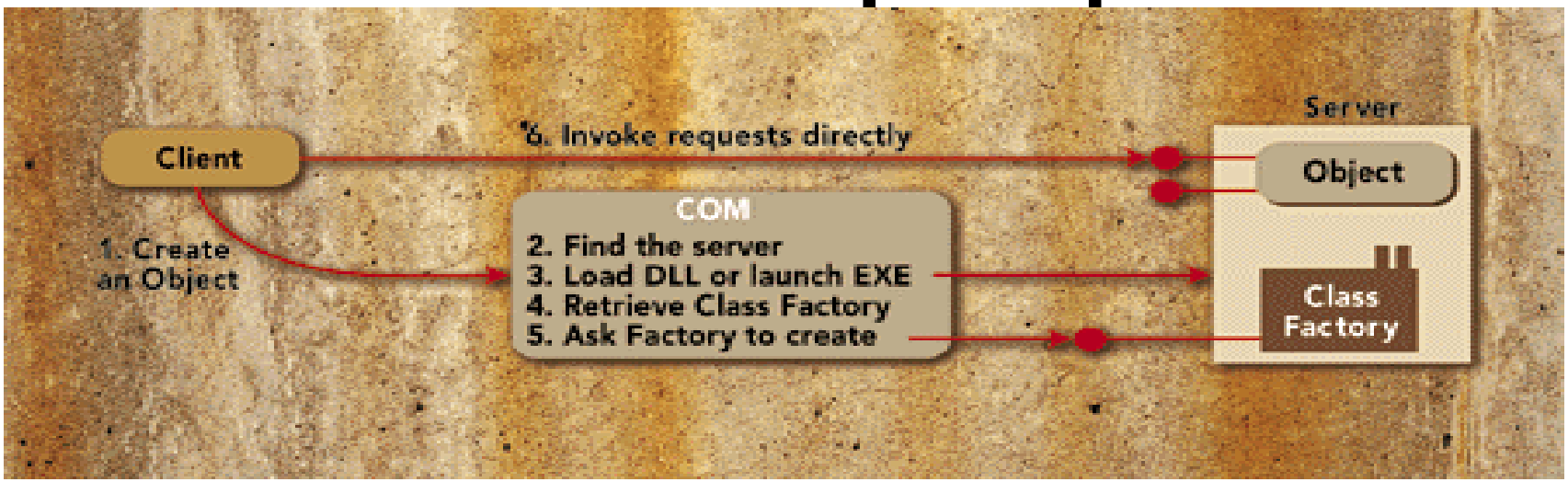
Caractéristiques des objets COM

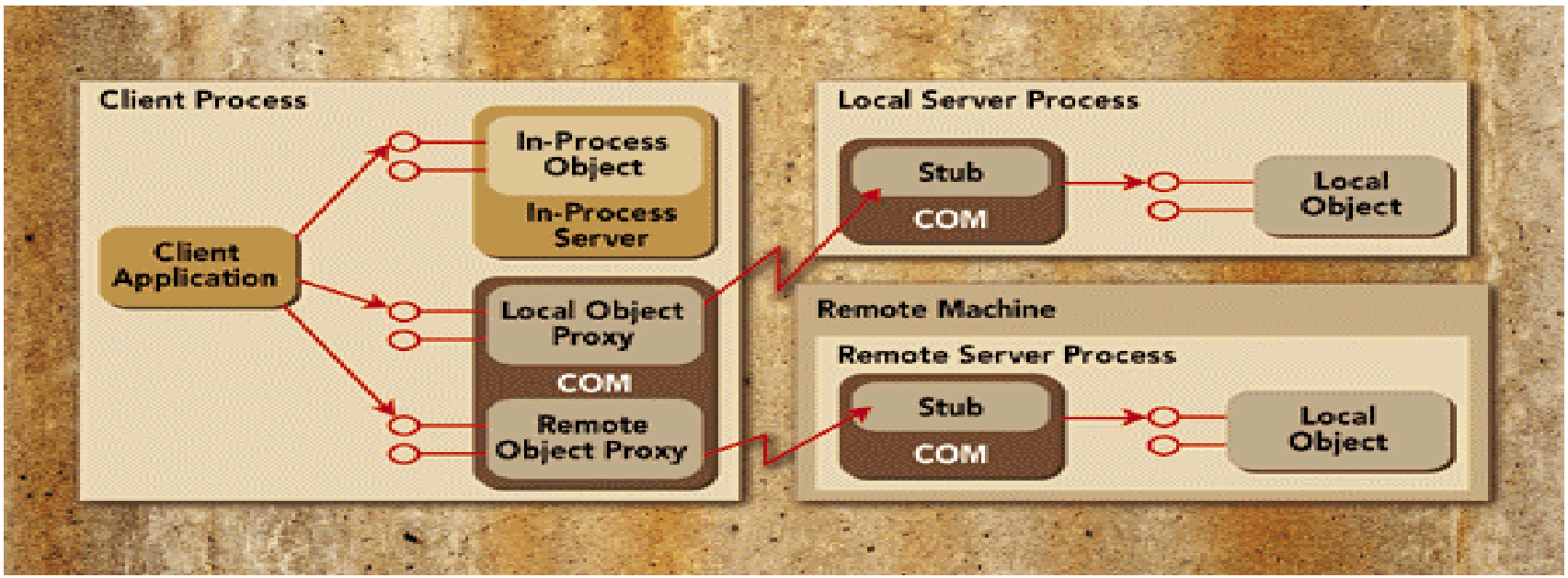
- La mère de toutes les interfaces « unknown » contient seulement ces trois méthodes et son IID est 0x00000000000000C000000000000046
- **Polymorphisme :** C'est la possibilité pour un objet de répondre correctement aux mêmes demandes en entrée que d'autre objets différents
- **Réutilisation ou héritage :** Uniquement au niveau de leurs interfaces
  - Mode inclusion : un objet COM peut en réutiliser un autre uniquement en tant que client à travers ses interfaces. L'objet réutilisé effectue une partie du travail
  - Mode agrégation : Un objet COM peut interroger un autre objet COM afin qu'il devienne une partie de lui-même. L'objet réutilisé doit savoir qu'il fait partie d'une équipe en déléguant ses méthodes Iunknown)

Quelques définitons

- Un **client** est n'importe quelle séquence de code, faisant appel au service d'un objet.
- Un **serveur** est un code associé à une classe d'objets identifiée par son CLSID unique.
  - Il doit implémenter un objet de class COM qui permet de créer des objets dynamiquement (Class Factory)
- MARSHALLING : Extensibilité de COM
  Définition : Assemblage (Marshalling) : Procédé consistant à prendre une collection de paramètres, à les arranger et à les coder en format externe pour constituer un message à émettre

Schémas de principe





Schémas de principe : commentaires

- Un objet peut être dans un process différent (en local ou sur une machine distante) .
- Dans ce cas :
    - le marshalling utilise un interface-spécifique proxy/stub (sous forme d'une DLL générer a partir d'un fichier IDL) et un mécanisme RPC.
- le "Microsoft Interface Definition Language" (MIDL) est un compilateur qui peut être utilisé a partir d'un fichier IDL pour générer automatiquement le code nécessaire aux transfère distant.
    - La DLL doit être implémentée côté serveur et côté client.

Stub et Proxy

- **Proxy** :
  - Il « représente » l'objet côté client
  - Il expose les interfaces que l'objet expose
  - Il sait comment communiquer avec le Stub
- **Stub:**
  - Il « représente » le client côté objet
  - Il expose les interfaces que le client utilise
  - Il sait comment communiquer avec le proxy
- **RPC**: **Remote Procedure Call (appel de procédure à distance) ou LRPC :Lightweight RPC (principe de RPC mais entre processus locaux)**

DCOM
Distributed Component Object Model

- C’est COM avec un support par le système d’exploitation pour :
  - Identification distante
  - Chargement distant des composants
  - Marshaling ou extensibilité distante
- Il est supporté par Windows NT4.0 et a partir de Windows 95 « patché »
- Il introduit une clé **AppID** pour les serveurs DCOM permettant d’identifier un module dans la base de registres de Windows.

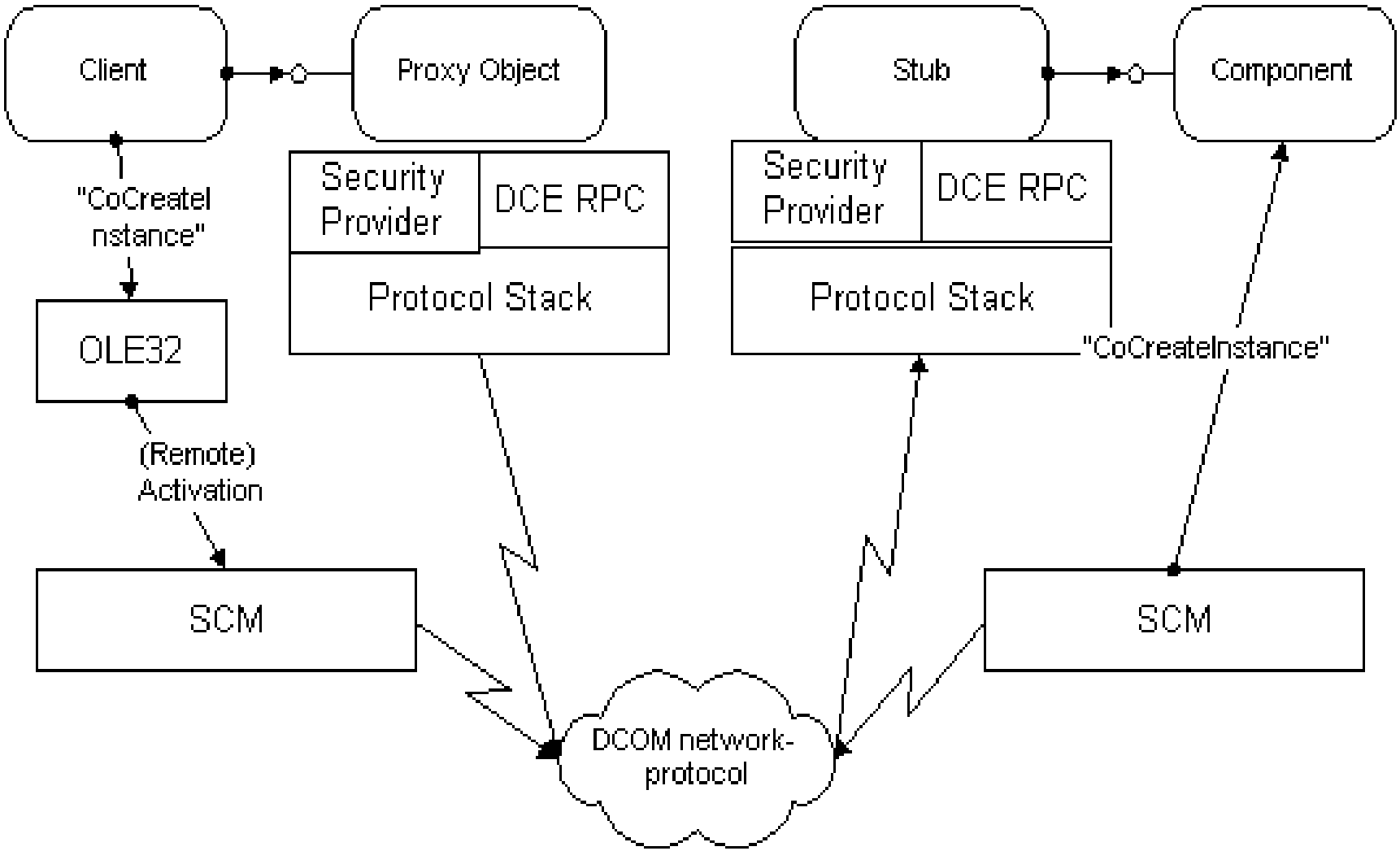
La Sécurité DCOM

- **La Sécurité :**
  - Authentification
  - Autorisation
  - Protection contre l'usurpation d'identité
  - La sécurité d'activation :
- Déclarer côté serveur elle spécifie :
  - les utilisateurs ou groupes qui peuvent lancer le serveur
  - Le compte sous lequel tourneras le serveur
  - La sécurité d'appel : Automatique ou Personnalisable pour chaque interface

Communications COM : « ORPC » Object Remote Procedure Call

- basé sur « Open Software Foundation » (OSF) « Distributed Computing Environement » (DCE) /RPC
- il rajoute aux paquets standard RPC une identification de l'objet appelé « OBJREF »
- « OBREF » est constitué de :
  - Interface Pointer Identifier (IPID): Utilisé coté serveur pour identifier une interface spécifique d'un objet.
  - Un « Object Exporter Indentifier » OXID :utilisé pour trouver une connexion au serveur qui abrite l'objet appelé .

ORPC : Architecture



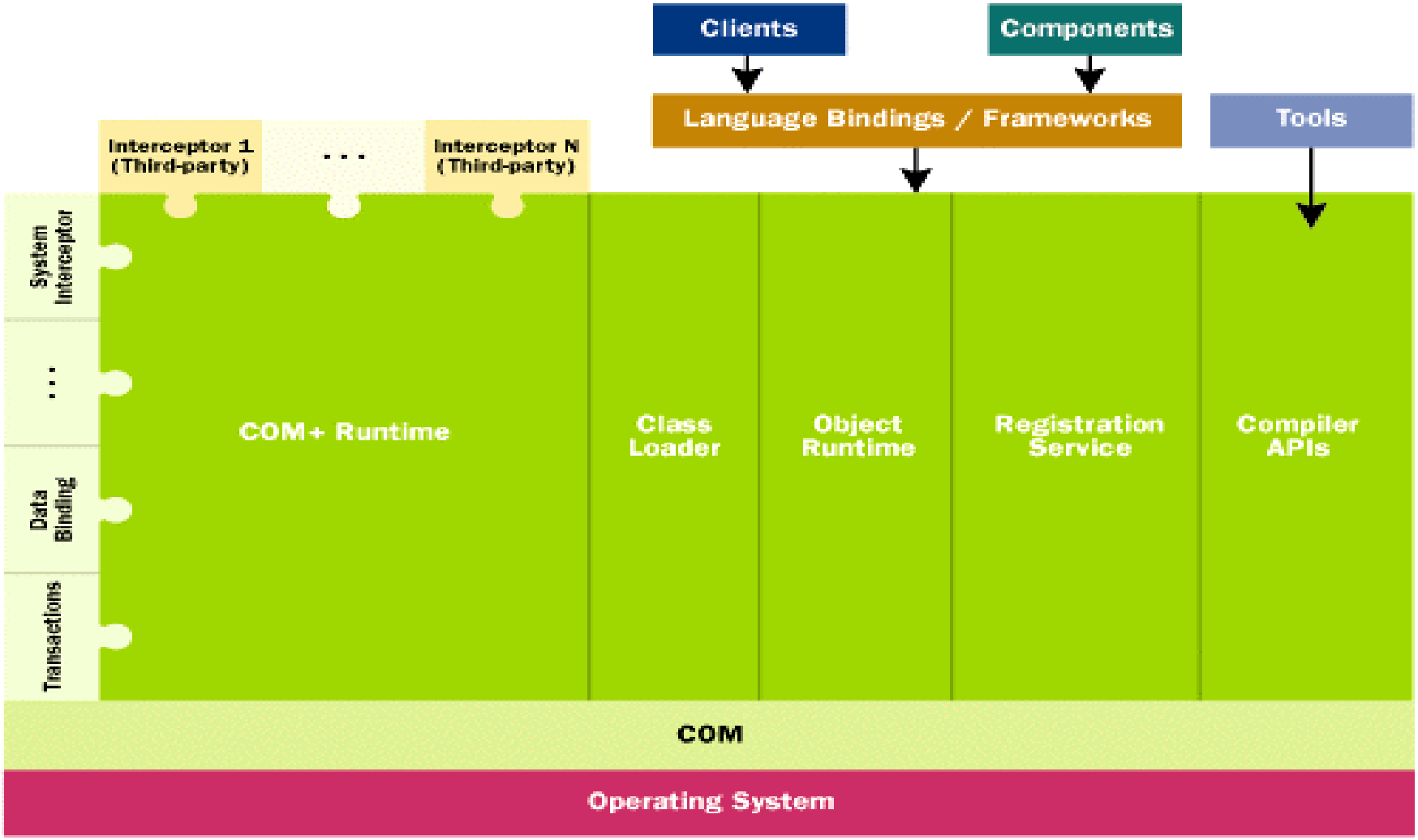
Les contrôles ACTIVEX

- **Les contrôles ACTIVEX** :
  - c’est l’application la plus sophistiquée de COM
  - Développée dans une optique Internet
  - c’est l‘équivalent Microsoft aux « applets Java » et aux «Java beans » dans le monde du Web.

Evolutions COM+

- Com+ est constitué de 3 parties :
  - COM Runtime
  - COM Services
  - DCOM

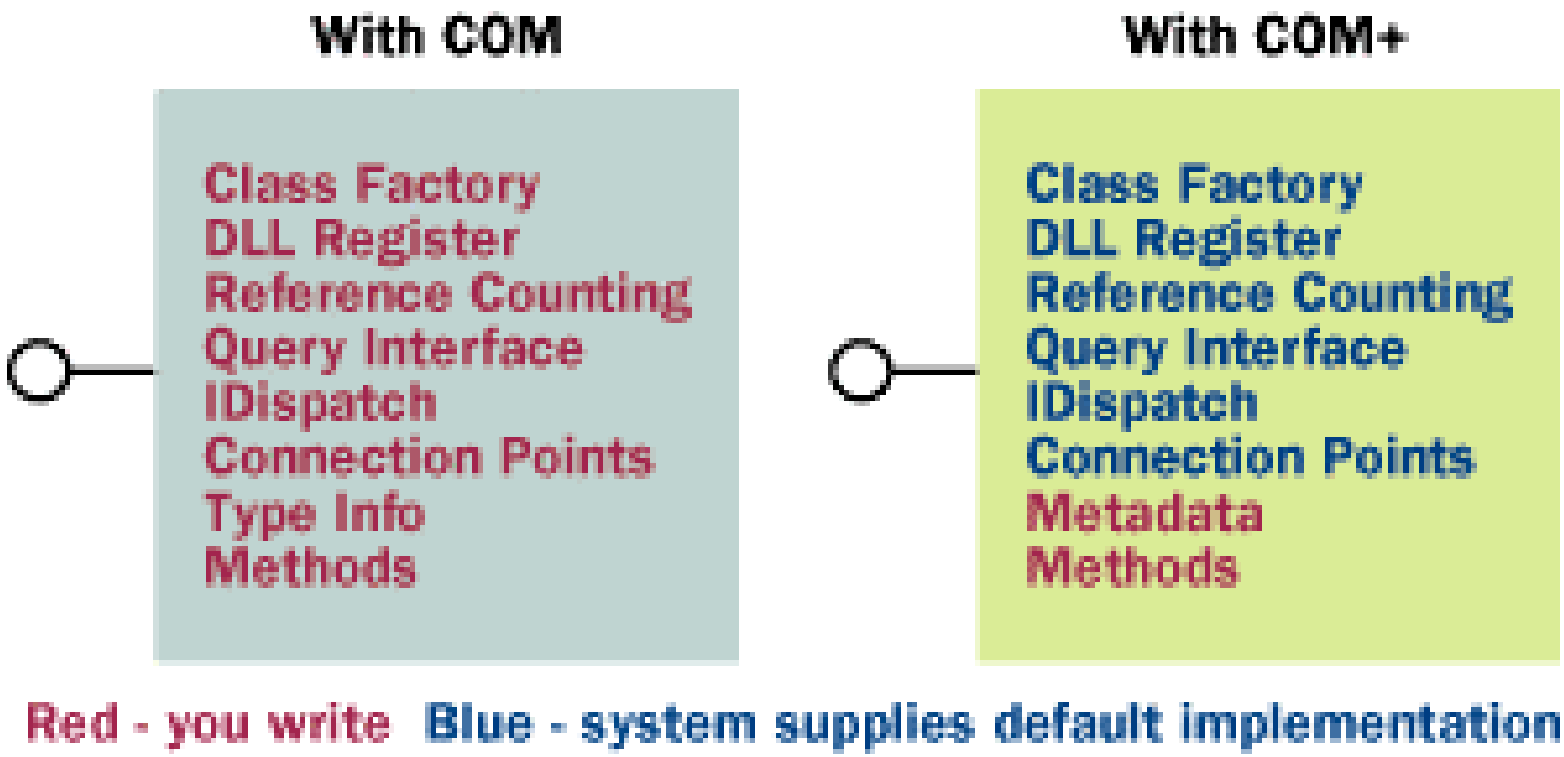
Architecture d'implémentation COM+



COM+ Runtime :

- COM+ implémente automatiquement la grande majorité des tâches nécessaire pour construire des objets COM au travers des « Meta data » codées par le programmeur réduisant de 30% l'effort de programmation
- C'est le concept de **« programmation basé attribut »**
- Principal innovation : Les « **Interceptors** » (les Intercepteurs) permettent de rerouter les accès des clients aux objets via les bibliothèques COM affectant ainsi la manière dont les objets interagissent entre eux.

Comparaison effort de programmation COM et COM+

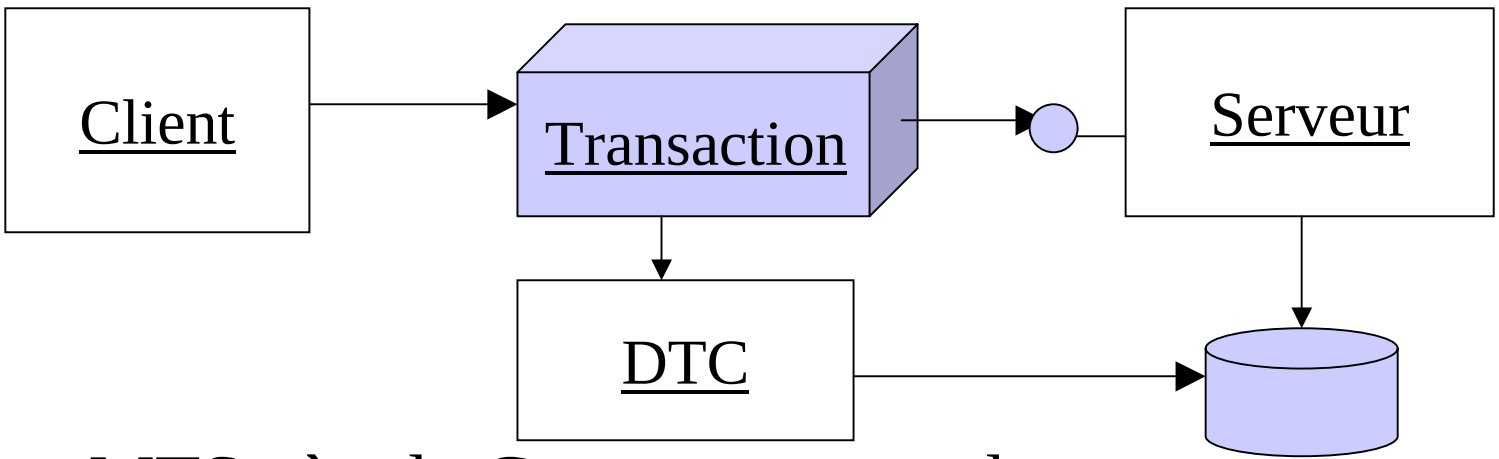


COM+ Services : Services de composants

- Les services de composants sont en fait des intercepteurs particuliers permettant d'intercepter et de rerouter les accès aux objets. Parmi les principaux :
  - **MTS** : Microsoft Transaction Server (moniteur transactionnel)
  - **Sécurité étendue**
  - **Load balancing** : Repartition de charges sur plusieurs serveurs

COM+ Services : Services de composants

- DTC (Distributed Transaction Coordinator) ou MTS : Microsoft Transaction Server



MTS gère la **Concurrence** et la **Sécurité** des transactions sur bases de données:

COM+ Services : Services de composants

- **Support d'environnements transactionnels hétérogènes :** Les composants COM peuvent participer à des transactions gérées par un environnement transactionnel (TP) autre que celui de COM+ par le support du protocole TIP (Transaction Internet Protocol).
- **Sécurité étendue :** La prise en charge de la sécurité est basée sur la notion de rôle ainsi que les permissions d'accès des processus. Dans le modèle de sécurité basé sur le rôle, l'accès aux différentes parties d'une application est accordé ou refusé en fonction du groupe logique auquel l'utilisateur appartient ou du rôle qui lui a été attribué (par exemple, administrateur, employé à temps complet, employé à temps partiel). COM+ implémente en plus de la sécurité basée sur le rôle, la sécurité au niveau des méthodes des interfaces.

COM+ Services : Services de composants

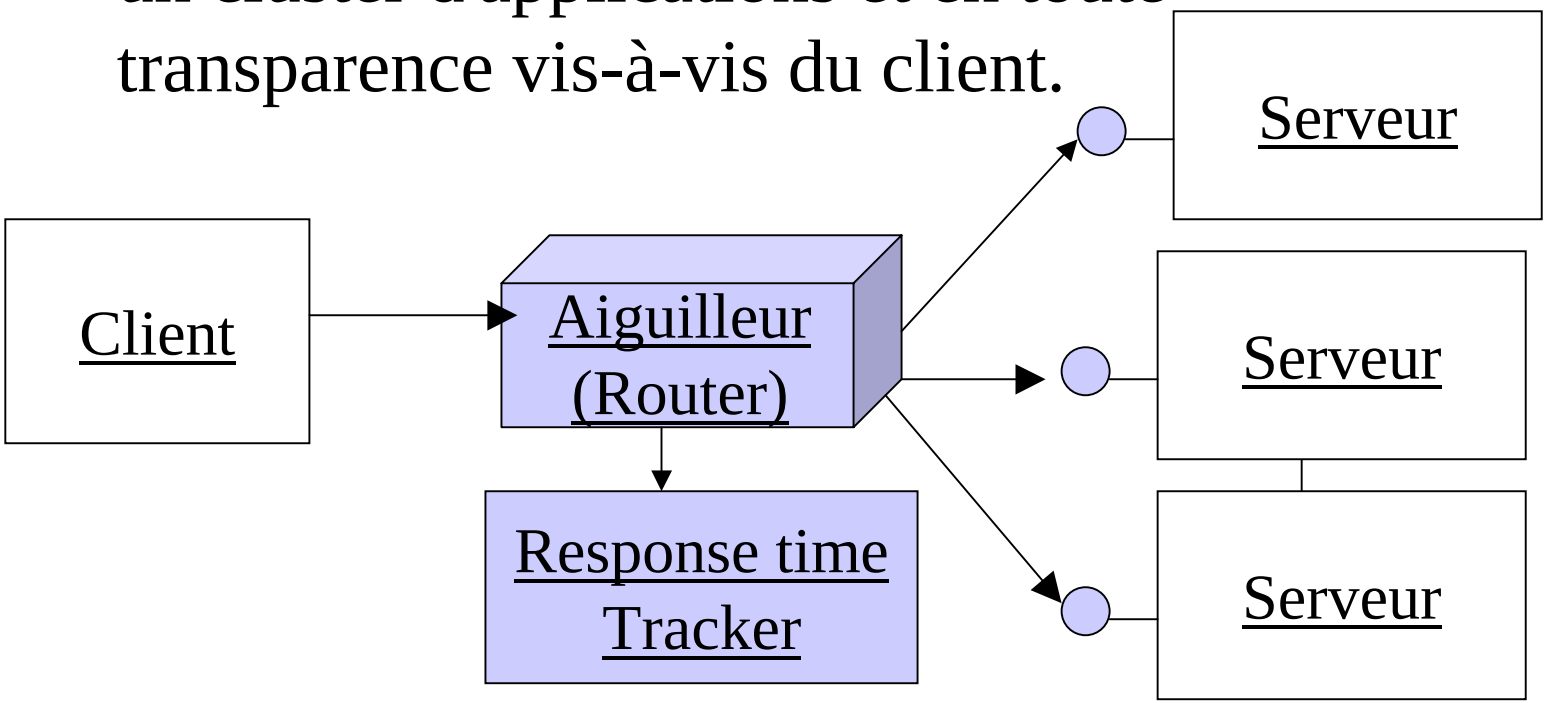
- **Administration centralisée :**L'Explorateur des services de composants, présente un modèle d'administration unifiée, qui facilite :
  - le déploiement,
  - la maintenance,
  - la surveillance d'applications à n niveaux, en éliminant le besoin d'utiliser de nombreux outils d'administration distincts.

COM+ Services : Services de composants

- **IMDB - In-Memory DataBase (base de données en mémoire) :**
  - Permet de gérer des informations permanentes et des informations temporaires de façon cohérente. In-Memory DataBase est un système de base de données entièrement transactionnel géré en mémoire et conçu pour fournir un accès extrêmement rapides aux données.
- **Object pooling** : Regroupement d'objets
  - Fournit aux applications des méthodes rapides et efficaces pour pouvoir ré-utiliser des objets COM dans le but d'améliorer les performances.

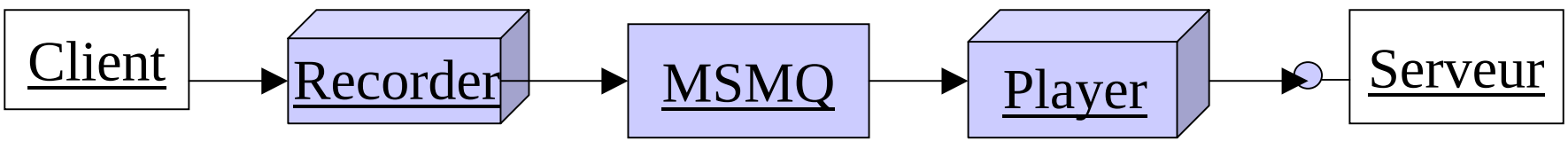
COM+ Services : Services de composants

- **Load balancing** : Répartition de charges
  - Permet aux applications basées sur le composant de répartir leur charge de travail sur un cluster d'applications et en toute transparence vis-à-vis du client.



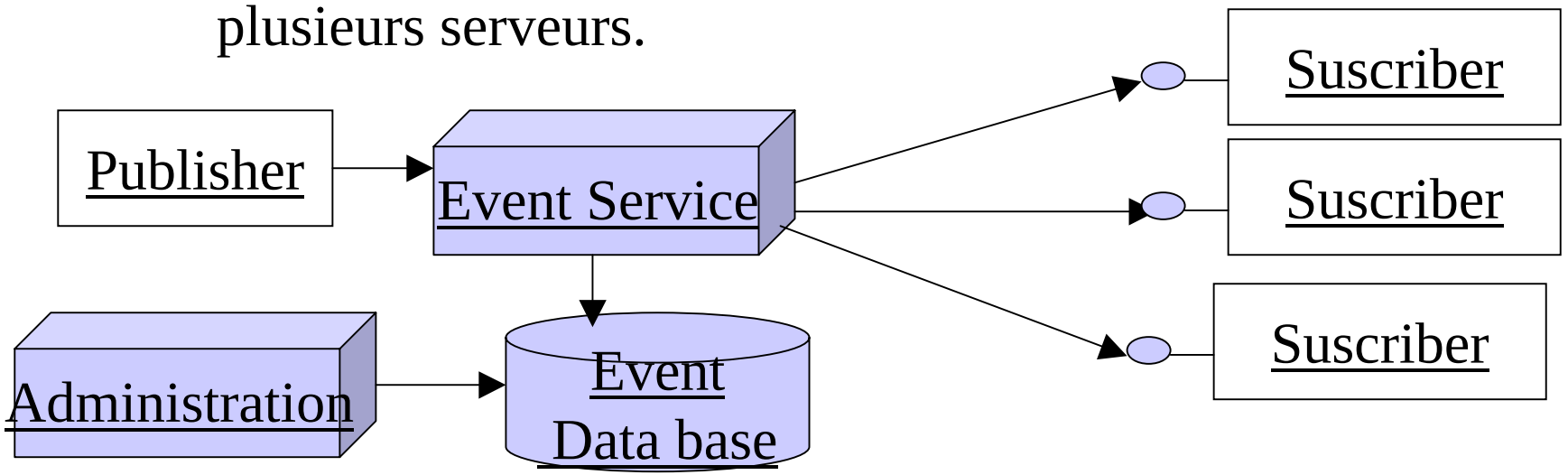
COM+ Services : Services de composants

- Asynchronous(Appels de Méthodes asynchrone) et Queued Components (composants en file d'attente) : basé sur MSMQ Microsoft Message Queuing Services
  - Assurent une exécution différée en mode asynchrone lorsque les composants travaillant en coopération sont déconnectés. Cette fonctionnalité vient s'ajouter au modèle de programmation client-serveur synchrone basé sur la session.



COM+ Services : Services de composants

- **COM+ Events** :
  - Notification d' événements : A utiliser lorsqu'un mécanisme de notification d'événements est souhaité. Le système d'événements de COM+ est un mécanisme de publication/abonnement qui permet à des clients de " s'abonner " à des événements " publiés " par plusieurs serveurs.



Produits supportant COM

- Sun Solaris (sparc)2.5
- Digital Unix 4.0 (Alpha)
- IBM MVS 5.2.2 (OS390)
- IBM OS/400
- IBM AIX
- HP/UX
- Digital Open VMS (sur Alpha)
- Siemens Nixdorf SINIX
- Linux 2.0 (Intel)
- SCO UnixWare

Ce qu'il faut retenir

- Comme pour Corba, les couches COM+ se situeny au niveau 5,6,7 du modèle OSI au dessus donc de la couche 4 (TCP/UDP)
- Association COM+/ACTIVEX (comme CORBA/JAVA)
- Des passerelles permettant de faire cohabiter les deux modèles (existe déjà sur le marché)

Ce qu'il faut retenir

- Point de convergence XML Exemple :
  - SOAP : Simple Object Access Protocol
    Protocole permettant l'invocation à distance de composants (objets et méthodes) à l'instar des protocoles DCOM (Microsoft) ou Corba (IIOP).
  - Soap utilise XML pour représenter ces invocations ce qui devrait permettre de faire interopérer des environnements Windows DNA, Corba ou Java. Soap véhicule sur le web (au format XML, via HTTP) les appels de fonctions (méthodes).
  - Soap est utilisable entre postes clients et serveurs, ce qui devrait permettre de l'utiliser depuis un poste client Internet (chez un particulier lambda) pour accéder à des services disponibles sur des serveurs web sans passer par des pages HTML.
  - Conçu à l'origine par Microsoft, DevelopMentor et Userland, Soap est soutenu également par IBM et Lotus. Depuis le ralliement de Sun, c'est devenu une note du W3C (World Wilde Web Consortium).