

Plateforme De Passage D’ordre Boursier

Ahmed Ayadi, Nathan Kruck, Nolan Potier, Yann Bondue

  2013/2014

**Encadré Par Monsieur Philippe Salvan**

Table of Contents

[Chapitre I Contraintes exprimées 3](#_Toc379319909)

[A. Besoins techniques 3](#_Toc379319910)

[1. Acteurs : 3](#_Toc379319911)

[B. Besoins métier 3](#_Toc379319912)

[C. Contraintes structurantes 3](#_Toc379319913)

[1. Architecture centralisée répartie 3](#_Toc379319914)

[2. Qualité de service 3](#_Toc379319915)

[Chapitre II Contraintes existantes 4](#_Toc379319916)

[A. Contraintes organisationnelles 4](#_Toc379319917)

[B. Contraintes liées à l’environnement technique 4](#_Toc379319918)

[Chapitre III Architecture applicative 5](#_Toc379319919)

[1. Agence centrale (Paris) : 6](#_Toc379319920)

[a. Exigences fonctionnelles: 6](#_Toc379319921)

[b. Exigences techniques: 6](#_Toc379319922)

[2. Agence régionales : 6](#_Toc379319923)

[3. Agence Locales : 6](#_Toc379319924)

[A. Cartographie des flux 7](#_Toc379319925)

[1. Description détaillée des blocs applicatifs 8](#_Toc379319926)

[a. Stock information : 8](#_Toc379319927)

[i. Description fonctionnelle: 8](#_Toc379319928)

[ii. Description applicative et technique : 8](#_Toc379319929)

[b. Transaction System: 9](#_Toc379319930)

[i. Description fonctionnelle : 9](#_Toc379319931)

[ii. Description applicative et technique : 9](#_Toc379319932)

[c. Account Information : 10](#_Toc379319933)

[i. Description fonctionnelle: 10](#_Toc379319934)

[ii. Description applicative et technique: 10](#_Toc379319935)

[d. Web application: 11](#_Toc379319936)

[i. Description fonctionnelle: 11](#_Toc379319937)

[ii. Description applicative et technique : 11](#_Toc379319938)

[e. Application Trader: 12](#_Toc379319939)

[i. Description fonctionnelle: 12](#_Toc379319940)

[ii. Description applicative et technique : 12](#_Toc379319941)

[f. Application Courtier: 13](#_Toc379319942)

[i. Description fonctionnelle: 13](#_Toc379319943)

[ii. Description applicative et technique : 13](#_Toc379319944)

[g. Gros Client: 14](#_Toc379319945)

[i. Description fonctionnelle : 14](#_Toc379319946)

[ii. Description applicative et technique : 14](#_Toc379319947)

[h. Agence Régionale : 15](#_Toc379319948)

[i. Description fonctionnelle : 15](#_Toc379319949)

[ii. Description applicative et technique : 15](#_Toc379319950)

[i. Agence Locale: 16](#_Toc379319951)

[i. Description fonctionnelle : 16](#_Toc379319952)

[ii. Description applicative et technique : 16](#_Toc379319953)

[2. Matrice des flux 17](#_Toc379319954)

[a. Cinématique représentative de l’utilisation du system 17](#_Toc379319955)

[b. Vision Dynamique 18](#_Toc379319956)

1. Contraintes exprimées
2. Besoins techniques

# Acteurs :

* Courtiers :
* Clients individuels: Connexion avec des PC.
* Clients mobiles : Clients légers avec connexion très rapide : accès direct aux cours.
* “Gros clients” : Contrainte critique ; délai d'exécution des transactions.
* Clients traditionnels : pas de contraintes techniques, leurs transactions sont assurées à travers les agences régionales.
* Les courtiers de la banque vont assurer les transactions des clients traditionnels.
* Agents “à la corbeille” : accès direct à la centrale.

1. Besoins métier

Fournir des outils d’aide à la décision.

1. Contraintes structurantes

# Architecture centralisée répartie

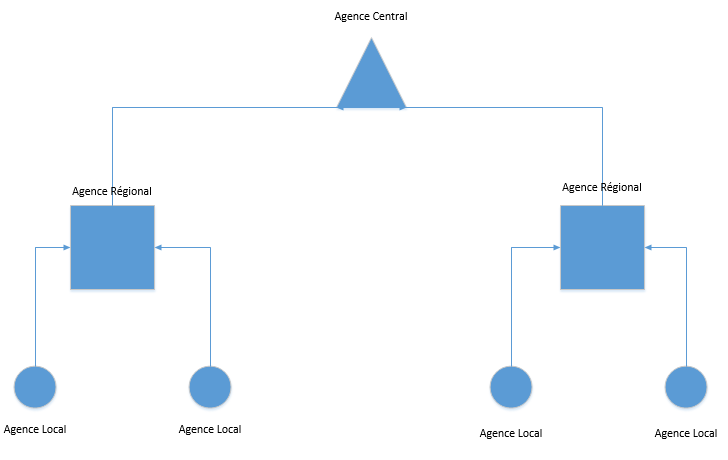
Architecture répartie dans les agences régionales et locales qui est centralisée pour l’aspect Business Intelligence dans l’agence centrale.

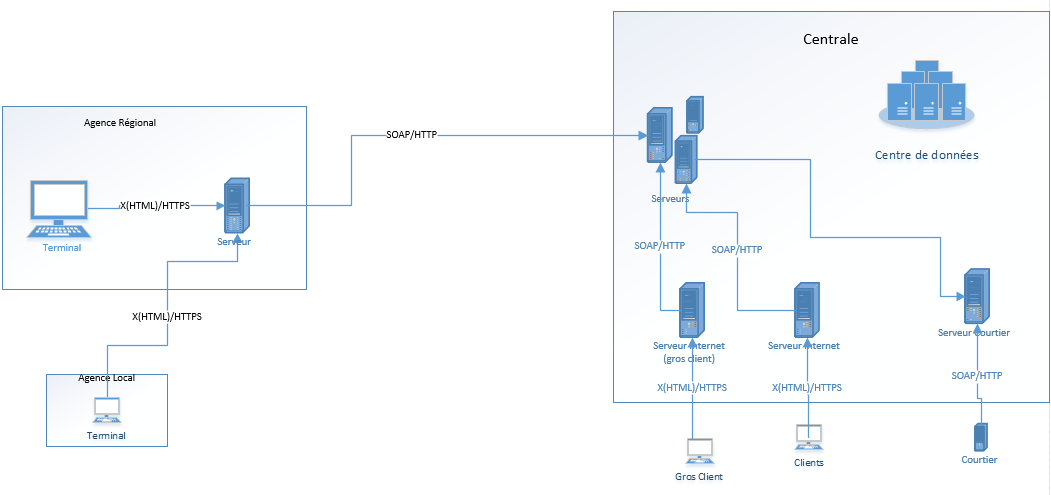
* Agence centrale (Paris) :
  + Reliée directement à la place de marché
  + Obtenir les actualités du marché (carnets d’ordre)
  + Doit recevoir tous les ordres des agences régionales
  + Passe les ordres sur le marché
  + Mettre à jour le carnet d’ordres
  + Stocker les historiques d’ordres
* Agences régionales :
  + Reliées à l’agence centrale
  + Transmettent les ordres à cette dernière.
* Agences locales :
  + Reliées à l’agence régionale la plus proche. (Définir les régions)
  + Transmettent les ordres à cette dernière.

# Qualité de service

Le service de passage d’ordres doit être assuré à tout moment lors des heures d’ouverture de la place financière.

1. Contraintes existantes
2. Contraintes organisationnelles
3. Contraintes liées à l’environnement technique
4. Architecture applicative





**3 blocs applicatifs :**

Architecture répartie dans les agences régionales et locales qui est centralisée pour l’aspect Business Intelligence dans l’agence centrale.

# Agence centrale (Paris) :

## Exigences fonctionnelles:

* Reliée directement à la place de marché
* Obtenir les actualités du marché (carnets d’ordre)
* Doit recevoir tous les ordres des agences régionales
* Passe les ordres sur le marché
* Mettre à jour le carnet d’ordres
* Stocker les historiques d’ordres
* Le Carnet d'ordres
  + Le Carnet d'ordres propose les 5 meilleures limites à l'achat et à la vente sur un titre ainsi que le nombre d'actions et d'ordres positionnés à chaque niveau. Dans un respect réglementaire, Boursorama diffuse sur son site visiteur/membre le carnet d'ordre décalé de 15 minutes.
  + Boursorama vous propose également un indicateur permettant d'identifier de façon plus facile la tendance du carnet d'ordre (achat ou vente) en affichant où se situe l'équilibre entre l'ensemble des titres à la vente et à l'achat. Ainsi, si la ligne verte de l'indicateur est plus longue que la ligne rouge, cela signifie que la quantité de titres à l'achat est supérieure à celle à la vente.
* Bid & Ask
  + Le Bid and Ask (ou fourchette de cotation) représente la fourchette de prix achat-vente fournie en continu par les émetteurs. Les émetteurs partenaires fournissent un prix à l'achat et à la vente de façon continue et pendant les mêmes horaires d'ouverture que sur le marché réglementé de NYSE Euronext.

## Exigences techniques:

Haute disponibilité, supervision

Développement spécifique.

Matériel : IBM CICS Transaction Server

Système : Linux / Z

# Agence régionales :

Reliée à l’agence centrale

Transmets les ordres à cette dernière.

# Agence Locales :

* + \*\*\* Reliée à l’agence régionale la plus proche. (Définir les régions)
  + Transmets les ordres à cette dernière.

1. Cartographie des flux

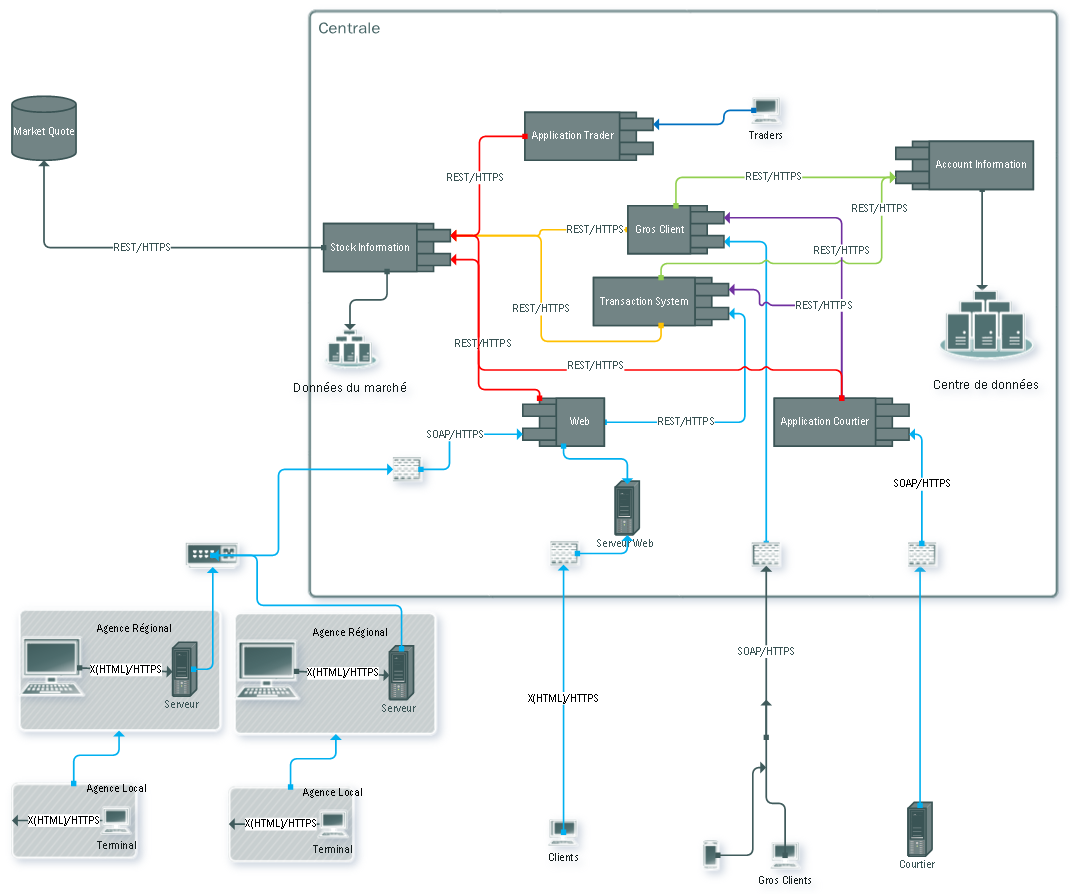


Figure 1 Architecture Applicative

# Description détaillée des blocs applicatifs

## Stock information :

Le bloc Stock information est une brique cruciale de l'architecture. Il s'agit de la composante qui interroge le marché pour obtenir son état en direct et en informer le reste du système, en vue du stockage des informations et de leur utilisation dans notre contexte applicatif.

### Description fonctionnelle:

|  |  |
| --- | --- |
| Objectifs | Obtenir les informations du marché et passer des ordres de la manière la plus rapide possible, afin d'être aussi proche du direct que possible |
| Classification | Critique, la gestion des informations du marché constitue la base même de notre architecture |
| Entrées / Sorties | Entrées :   * Flux d'ordres provenant du marché * Flux d'ordres provenant du bloc Transaction Information   Sorties :   * Flux de passage d'ordres provenant du bloc Transaction Information vers le marché * Flux de stockage de données vers une base de données du marché * Flux d'envoi d'informations d'ordres vers les agents à la corbeille |
| Fonctions | * Passer un Ordre * Récupérer les Ordres |
| Classes concepts gérées | * Lecteur et parseur d'information * Transmetteur d'information |

### Description applicative et technique :

|  |  |
| --- | --- |
| Développement spécifique ou progiciel | Développement spécifique |
| Volume traité | Millions de transactions/jour |
| Disponibilité | Très haute disponibilité (> 99.9%)  (La fermeture de la bourse permet d'assurer une maintenance quotidienne) |
| Fiabilité, niveau de service, performances attendues | Nécessité d'être aussi proche du marché que possible, et de permettre des transactions en temps réel |
| Matériel | IBM CICS Transaction Server (Système permettant d'effectuer des transactions avec une très grande économie de moyen, très utilisé dans la finance) |
| Système d'exploitation | z/OS (Système accompagnant le serveur) |
| SGBDR ou SGF | Oracle Database (Meilleur support de grands volumes de transaction, et est porté sur notre système d'exploitation) |
| Middleware | Terminal Bloomberg |
| Types de sites concernés | Centrale (Paris) |
| Sécurité | Haute sécurité, le passage d'ordres requiert une authentification |

## Transaction System:

Le bloc Transaction System reçoit les données de transaction avant qu'elles ne soient passées afin d'en identifier la source et de s'assurer qu'elle est autorisée à la passer.

### Description fonctionnelle :

|  |  |
| --- | --- |
| Objectifs | Faire le lien entre les données de client et les données de transaction pour contrôler leur authenticité avant de les transmettre au service de passage d'ordre |
| Classification | Critique, il ne faut pas permettre à des utilisateurs non-autorisés à passer des ordres |
| Entrées / Sorties | Entrées :   * Flux de transactions provenant des clients * Données d'autorisation associées à un client provenant du bloc Account Information   Sorties :   * Flux de passage d'ordres provenant de nos clients vers Stock Information * Demandes d'authentification vers le bloc Account Information |
| Fonctions | * Contrôler la validité des données * Transmettre les ordres |
| Classes concepts gérées | * Lecteur et contrôleur d'information * Transmetteur d'information |

### Description applicative et technique :

|  |  |
| --- | --- |
| Développement spécifique ou progiciel | Développement spécifique |
| Volume traité | Millions de transactions/jour |
| Disponibilité | Très haute disponibilité (> 99.9%)  (La fermeture de la bourse permet d'assurer une maintenance quotidienne) |
| Fiabilité, niveau de service, performances attendues | Doit pouvoir gérer un grand nombre de connexions simultanées |
| Matériel | IBM CICS Transaction Server |
| Système d'exploitation | z/OS |
| SGBDR ou SGF | Microsoft SQL server |
| Middleware | N/A |
| Types de sites concernés | Centrale (Paris) |
| Sécurité | Haute sécurité, les applications qui font appel à lui doivent être clairement identifiées |

## Account Information :

Le bloc Account Information gère les comptes utilisateurs qui utilisent notre architecture logicielle, elle permet principalement l'enregistrement de clients.

### Description fonctionnelle:

|  |  |
| --- | --- |
| Objectifs | Assurer la persistance des informations clientes et permettre de vérifier leur authenticité |
| Classification | Critique, la gestion d'utilisateurs est cruciale |
| Entrées / Sorties | Entrées :   * Flux d'informations clientes à vérifier provenant de Transaction System * Données clientes provenant serveur web pour la création des comptes   Sorties :   * Validations/Rejets d'informations d'identification |
| Fonctions | * Contrôler la validité des données clientes |
| Classes concepts gérées | * CRUD |

### Description applicative et technique:

|  |  |
| --- | --- |
| Développement spécifique ou progiciel | Développement spécifique (partie base de données assurée par Oracle) |
| Volume traité | Milliers de transactions/jour |
| Disponibilité | Très haute disponibilité (> 99%)  (La fermeture de la bourse permet d'assurer une maintenance quotidienne) |
| Fiabilité, niveau de service, performances attendues | Gérer un grand nombre d'authentification (milliers/jour), stocker un grand nombre (1 compte par client) d'informations client |
| Matériel | Oracle SPARC T4-1 (serveur dédié de base de données hautes performances) |
| Système d'exploitation | Oracle Solaris (OS de gestion de base de données couplé avec SPARC) |
| SGBDR ou SGF | SGBDR Oracle |
| Middleware | N/A |
| Types de sites concernés | Centrale (Paris) |
| Sécurité | Très haute sécurité, la base de données contient des informations sensibles et confidentielles |

## Web application:

Le bloc Web application représente le site de notre agence et permet aux clients non-professionnels de créer un compte et de passer leurs ordres personnels depuis la plate-forme.

### Description fonctionnelle:

|  |  |
| --- | --- |
| Objectifs | Proposer les web services pour les différentes applications clientes légères |
| Classification | Important, touche le plus grand nombre de clients à petite échelle |
| Entrées / Sorties | Entrées :   * Flux d'informations clientes de créations de compte et de passage d'ordres * Reçoit les données du marché via Stock Information   Sorties :   * Transmission des informations à vérifier vers Transaction System * Transmission des informations clientes vers Account Information * - Transmission de confirmations et d'infirmations aux utilisateurs |
| Fonctions | * Créer un compte utilisateur * Passer un ordre * Récupérer les informations en base de données * - Transmettre des informations aux clients |
| Classes concepts gérées | * Interfaces Web Services |

### Description applicative et technique :

|  |  |
| --- | --- |
| Développement spécifique ou progiciel | Développement spécifique |
| Volume traité | Milliers de transactions/jour |
| Disponibilité | Très haute disponibilité (> 99.99%)  (La fermeture de la bourse permet d'assurer un service dégradé de nuit) |
| Fiabilité, niveau de service, performances attendues | Assurer le passage d'ordres pour des clients individuels à petite échelle, avec un niveau d'exigence plus faible que les clients privilégiés |
| Matériel | Dell PowerEdge M520 (Serveur haute-performance pour applications web) |
| Système d'exploitation | Microsoft Windows Server 2012 R2 (Permet de déployer des web services) |
| SGBDR ou SGF | N/A |
| Middleware | N/A |
| Types de sites concernés | Centrale (Paris) |
| Sécurité | Les applications utilisant les web services doivent être clairement identifiées et autorisées |

## Application Trader:

Le bloc applicatif Trader permet aux agents à la corbeille d'agir directement au plus près du marché, en limitant tous les intermédiaires et en leur fournissant une connexion très haut débit.

### Description fonctionnelle:

|  |  |
| --- | --- |
| Objectifs | Assurer un passage d'ordre en temps réel et être informé en temps réel des fluctuations du marché |
| Classification | Important, il s'agit de la composante qui permet à la banque d'assurer la plus grande partie de ses transactions à grande échelle |
| Entrées / Sorties | Entrées :   * Reçoit les données du marché via Stock Information * Reçoit les données de transactions de l'agent à la corbeille   Sorties :   * Transmission des ordres à Stock Information |
| Fonctions | * Passer un ordre * Recevoir les informations du marché |
| Classes concepts gérées | * Récepteur d'informations * Émetteur d'informations * CRUD |

### Description applicative et technique :

|  |  |
| --- | --- |
| Développement spécifique ou progiciel | Développement spécifique |
| Volume traité | Millions de transactions/jour |
| Disponibilité | Très haute disponibilité (> 99.999%)  (La fermeture de la bourse permet d'assurer une maintenance de nuit) |
| Fiabilité, niveau de service, performances attendues | Nécessité d'être aussi proche du marché que possible, et de permettre des transactions en temps réel |
| Matériel | Dell PowerEdge M420 (Idéal pour assurer un grand nombre de petites transactions rapides) |
| Système d'exploitation | Microsoft Windows Server 2012 R2 (Permet un déploiement rapide) |
| SGBDR ou SGF | N/A |
| Middleware | N/A |
| Types de sites concernés | Centrale (Paris) |
| Sécurité | Seuls des terminaux réservés peuvent accéder au serveur |

## Application Courtier:

Le bloc applicatif Courtier permet de proposer aux courtiers un service de passage d'ordre afin qu'ils puissent fournir un service équivalent à leur clientèle.

### Description fonctionnelle:

|  |  |
| --- | --- |
| Objectifs | Assurer un passage d'ordre pour les courtiers externes à notre réseau |
| Classification | Important, ce bloc permet à la banque de gérer tous ses clients externes |
| Entrées / Sorties | Entrées :   * Reçoit les données du marché via Stock Information * Reçoit les données de transactions du courtier   Sorties :   * Transmission des ordres à Stock Information |
| Fonctions | * Passer un ordre * - Recevoir les informations du marché |
| Classes concepts gérées | * Récepteur d'informations * Émetteur d'informations |

### Description applicative et technique :

|  |  |
| --- | --- |
| Développement spécifique ou progiciel | Développement spécifique |
| Volume traité | Milliers de transactions/jour |
| Disponibilité | Très haute disponibilité (> 99%)  (La fermeture de la bourse permet d'assurer une maintenance de nuit) |
| Fiabilité, niveau de service, performances attendues | Garantir le passage des ordres en un temps minimal |
| Matériel | Dell PowerEdge M520 (Serveur haute-performance pour applications web) |
| Système d'exploitation | Microsoft Windows Server 2012 R2 (Permet de déployer des web services) |
| SGBDR ou SGF | N/A |
| Middleware | N/A |
| Types de sites concernés | Centrale (Paris) |
| Sécurité | Seuls des courtiers identifiés peuvent accéder au serveur |

## Gros Client:

Le bloc applicatif Gros Client est un software déployé auprès des agences de clients qui leur donne un accès privilégié pour passer des ordres à grande échelle.

### Description fonctionnelle :

|  |  |
| --- | --- |
| Objectifs | Assurer un passage d'ordre de grands volumes et montants en un temps minimal |
| Classification | Important, ce bloc permet à la banque de gérer tous ses clients principaux |
| Entrées / Sorties | Entrées :   * Reçoit les données du marché via Stock Information * Reçoit les données de transaction du client * Reçoit les vérifications d'authentification de Account Information   Sorties :   * Transmission des ordres à Stock Information * Émet des données d'authentification vers Account Information |
| Fonctions | * Passer un ordre * Recevoir les informations du marché |
| Classes concepts gérées | * Récepteur d'informations * Émetteur d'informations |

### Description applicative et technique :

|  |  |
| --- | --- |
| Développement spécifique ou progiciel | Développement spécifique |
| Volume traité | Milliers de transactions/jour |
| Disponibilité | Très haute disponibilité (> 99.9%)  (La fermeture de la bourse permet d'assurer une maintenance de nuit) |
| Fiabilité, niveau de service, performances attendues | Garantir le passage des ordres en un temps minimal |
| Matériel | Dell PowerEdge M520 (Serveur haute-performance pour applications web) |
| Système d'exploitation | Microsoft Windows Server 2012 R2 (Permet de déployer des web services) |
| SGBDR ou SGF | N/A |
| Middleware | N/A |
| Types de sites concernés | Centrale (Paris) |
| Sécurité | Seuls des clients identifiés peuvent accéder au serveur |

## Agence Régionale :

Le bloc applicatif des agences régionales consiste en une application légère reliée à un serveur local, ce serveur communique ensuite avec les web services du bloc applicatif Web.

### Description fonctionnelle :

|  |  |
| --- | --- |
| Objectifs | Assurer un passage d'ordre pour les agences régionales au travers d'une application dédiée |
| Classification | Important, ce bloc permet aux agences de la banque de fournir le service à ses clients |
| Entrées / Sorties | Entrées :   * Reçoit les données du marché via Web * Reçoit les données de transaction de l'agence   Sorties :   * Transmission des ordres à Web |
| Fonctions | * Passer un ordre * Recevoir les informations du marché |
| Classes concepts gérées | * Récepteur d'informations * Émetteur d'informations |

### Description applicative et technique :

|  |  |
| --- | --- |
| Développement spécifique ou progiciel | Développement spécifique |
| Volume traité | Milliers de transactions/jour |
| Disponibilité | Haute disponibilité (> 90%)  (La fermeture de la bourse permet d'assurer une maintenance de nuit) |
| Fiabilité, niveau de service, performances attendues | Garantir le passage des ordres sans véritables contraintes temporelles |
| Matériel | Dell PowerEdge T110 II (Serveur idéal pour un petit nombre de transactions et de connexions) |
| Système d'exploitation | Microsoft Windows Server 2012 R2 (Permet un déploiement simple et rapide) |
| SGBDR ou SGF | N/A |
| Middleware | N/A |
| Types de sites concernés | Régional (Province) |
| Sécurité | Seul le personnel de l'agence à accès au serveur. |

## Agence Locale:

Le bloc applicatif des agences locales consiste en un terminal dédié aux fonctionnalités limitées et en communication avec le serveur de l'agence régionale afférente.

### Description fonctionnelle :

|  |  |
| --- | --- |
| Objectifs | Assurer un passage d'ordre pour les agences locales au travers d'une application dédiée |
| Classification | Utile, ce bloc permet aux agences de la banque de fournir le service à ses clients les moins avertis |
| Entrées / Sorties | Entrées :   * Reçoit les données du marché via l'agence régionale afférente * Reçoit les données de transaction de l'opérateur   Sorties :   * Transmission des ordres à l'agence régionale |
| Fonctions | * Passer un ordre * - Recevoir les informations du marché |
| Classes concepts gérées | * Récepteur d'informations * Émetteur d'informations |

### Description applicative et technique :

|  |  |
| --- | --- |
| Développement spécifique ou progiciel | Développement spécifique |
| Volume traité | Milliers de transactions/jour |
| Disponibilité | Haute disponibilité (> 90%)  (La fermeture de la bourse permet d'assurer une maintenance de nuit) |
| Fiabilité, niveau de service, performances attendues | Garantir le passage des ordres sans aucunes contraintes temporelles |
| Matériel | Dell OptiFlex 7010 (Ordinateur fourni au personnel des agences) |
| Système d'exploitation | Microsoft Windows 8.1 (Suffisant pour l'usage réalisé) |
| SGBDR ou SGF | N/A |
| Middleware | N/A |
| Types de sites concernés | Local |
| Sécurité | Seul le personnel à accès à l'ordinateur |

# Matrice des flux

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| # | Source | Destination | Support | TP/Batch | Message | Transport | Description |
| 1 | Navigateur client de l’agence régional | Serveur agence régional | Ethernet | TP | (x)HTML | HTTPS | Connexion au portail de l’agence régional |
| 2 | Navigateur client de l’agence local | Serveur agence régional | Internet | TP | (x)HTML | HTTPS | Connexion au portail de l’agence régional |
| 3 | Serveur agence régional | Serveur web services agence central | Internet | Batch | SOAP | HTTPS | Utilisation des web services de l’agence centrale |
| 4 | Serveur web services agence centrale | Transaction System | Ethernet | Batch | REST | HTTPS | Gere le passage des ordres |
| 5 | Transaction System | Stock Information | Ethernet | Batch | REST | HTTPS | Traitement des données du marché |
| 6 | Transaction System | Account Information | Ethernet | Batch | REST | HTTPS | Traitement des informations client |
| 7 | Account Information | Data Base Acces | Ethernet | Batch | REST | HTTPS | Accès à la base de données |
| 8 | Stock Information | Market Quote | Internet | Batch | REST | HTTPS | Passe les ordres sur le marché |

## Cinématique représentative de l’utilisation du system

**UC : Afficher les flux des prix**

**User** : Client Ordinaire

**Précondition** : Client Authentifié

|  |  |
| --- | --- |
| Etape | Description |
| 1 | Le client à travers son navigateur demande l’affichage du flux des prix d’une valeur boursière choisie |
| 2 | La requête est transmise vers l’application web des clients ordinaires |
| 3 | L’application web fait un appel en SOAP/HTTPS au block stock information system |
| 4 | Le block **Stock Information System** vérifie si le prix de la valeur boursière est le plus récent (peut inclure une requête REST/HTTPS pour mettre à jours le carnet d’ordre) et répond en REST/HTTPS au block **Web.** |
| 5 | Le block **Web** répond au client en html. |
| 6 | 3 jusqu’à 5 se répètent jusqu’à le client affiche autre chose. |

## Vision Dynamique



**UC : Afficher les flux des prix pour un tradeur**

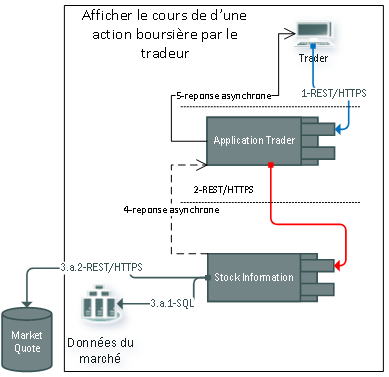
****

Figure 2 Vision Dynamique du System: Tradeur affiche le cours d'une action

1. Architecture logicielle :

Après la modélisation des besoins des utilisateurs du système, on a pu déceler les exigences. Ça nous a permis de :

* Mettre le focus sur le comportement extérieur du système, indépendamment de sa conception interne.
* Décrire les besoins des utilisateurs et des parties prenantes « avec beaucoup moins d'ambiguïté ».
* Combler les lacunes et les incohérences dans les exigences.
* Planifiez l'ordre dans lequel les caractéristiques seront élaborées.
* Utilisez les modèles de base pour les tests du système, ce qui rend une relation claire entre les critères et les exigences. Lorsque les conditions changent, cette relation permet de mettre à jour les essais correctement. Cela permet de s'assurer que le système répond aux nouvelles exigences.

1. Stock information :

Le logiciel implémenté va servir de parseur des données du marché. Il devra pouvoir rendre persistantes ces données et fournir des services de lecture et écriture sur ces données. Il doit aussi passer des ordres sur le marché.

Le programme étant exécuté sur le système d’exploitation z/OS et nécessitant de très bonne performance en terme de rapidité sera implémenté avec le langage C++.

# Couches :

|  |  |
| --- | --- |
| Présentation | N/A : aucune interaction avec l’utilisateur |
| Contrôleur | Doit contrôler l’appel de services, et doit contrôlé que le carnet d’ordre présent dans la base de données est à jours. |
| Services | Web services REST   * Récupérer un carnet d’ordre * Passer un ordre   **Performance :** Très performant  **Security :** Contraintes sécurité fortes  **Robustesse :** Très robuste même avec un comportement dégradé du system |
| Domaine | **Objets métiers :**   * Carnet d’ordres (liste d’ordres) ; Utilisation de la libraire STL * Ordre Trader Ordre * Gros Client * Ordre   **Fourniture des moyens d’accès à l’information :**  Oracle Call Interface (OCI) est une API fournie par Oracle permettant aux développeurs de créer des applications en utilisant des appels C++ bas niveau afin d'accéder à des bases de données Oracle. |
| Persistance | Librairie OCILIB encapsule OCI afin de fournir une interface beaucoup plus simple à coder et facilite la maintenance. |

# Tiers :

Modèle 2-tiers.

1 serveur Logique : IBM CICS z/OS : application C++

1 serveur Données : Oracle

1. Transaction System :

Le logiciel implémenté va servir de contrôleur afin d’autoriser les ordres avant que ceux-ci ne soit passé au Stock Information.

# Couches :

|  |  |
| --- | --- |
| Présentation | N/A : aucune interaction avec l’utilisateur |
| Contrôleur | Doit contrôler l’appel de services, Contrôler la validité des données et appelant les services de bloc Account Information |
| Services | Web services REST   * Passer un ordre (ordre, client)   **Performance :** Très performant  **Security :** Contraintes sécurité fortes  **Robustesse :** Très robuste même avec un comportement dégradé du system ; doit assurer des contraintes de transactions sur les ordres qu’il traite. |
| Domaine | **Objets métiers :**   * Carnet d’ordres (liste d’ordres)   Utilisation de la libraire STL   * Ordre * Client   **Fourniture des moyens d’accès à l’information :**  Les accès à l’information son géré par le bloc Account Information |
| Persistance | La persistance des données est gérée par le bloc Account Information |

# Tiers :

Modèle 1-tiers.

1 serveur Logique : IBM CICS z/OS : application C++

1. Account information :

Le logiciel implémenté va servir à persister toutes les données relatives aux clients.

Le programme étant exécuté sur le système d’exploitation z/OS et nécessitant de très bonne performance en terme de rapidité sera implémenté avec le langage C++.

# Couches :

|  |  |
| --- | --- |
| Présentation | N/A : aucune interaction avec l’utilisateur |
| Contrôleur | Doit contrôler l’appel de services, et doit permettre de créer, rechercher, modifier et supprimer des composants d’objets métiers dans le respect des propriétés transactionnelles. |
| Services | Web services REST   * Créer un client * Supprimer un client * Modifier un client * Ajouter un ordre * Supprimer un ordre * Modifier un ordre   **Performance :** Très performant  **Security :** Contraintes sécurité fortes  **Robustesse :** Très robuste même avec un comportement dégradé du system |
| Domaine | **Objets métiers :**   * Carnet d’ordres (liste d’ordres) ; Utilisation de la libraire STL * Ordre * Client * Gros Client   **Fourniture des moyens d’accès à l’information :**  Oracle Call Interface (OCI) est une API fournie par Oracle permettant aux développeurs de créer des applications en utilisant des appels C++ bas niveau afin d'accéder à des bases de données Oracle. |
| Persistance | Librairie OCILIB encapsule OCI afin de fournir une interface beaucoup plus simple à coder et facilite la maintenance. |

# Tiers :

Modèle 1-tiers.

1 serveur Données : Oracle SPARC T4-1

1. Application Trader :

Le logiciel permet aux agents à la corbeille de passer leurs ordres directement au plus près du marché, en limitant toute autre traitement. Il va servir de contrôleur d’autorisation de passage d’ordres avant que ceux-ci ne soit passé au Stock Information.

# Couche :

|  |  |
| --- | --- |
| Présentation | Doit fournir une ergonomie acceptable par les agents à la Corbeil. |
| Contrôleur | Doit contrôler l’appel des services, et doit permettre de créer, rechercher, modifier et supprimer des composants d’objets métiers dans le respect des propriétés transactionnelles. |
| Services | Web services REST   * Passer un ordre (ordre, Trader)   **Performance :** Très performant  **Security :** Contraintes sécurité fortes  **Robustesse :** Très robuste même avec un comportement dégradé du system ; doit assurer des contraintes de transactions sur les ordres qu’il traite. |
| Domaine | **Objets métiers :**   * Carnet d’ordres (liste d’ordres) ; Utilisation de la libraire STL * Ordre Trader * Trader   **Fourniture des moyens d’accès à l’information :**  Oracle Call Interface (OCI) est une API fournie par Oracle permettant aux développeurs de créer des applications en utilisant des appels C++ bas niveau afin d'accéder à des bases de données Oracle. |
| Persistance | Librairie OCILIB encapsule OCI afin de fournir une interface beaucoup plus simple à coder et facilite la maintenance. |

# Tiers :

Modèle 2-tiers.

1 serveur Logique : Windows Server 2012 R2

1 serveur Données : Microsoft SQL Server 2012

Décrire de façon détaillée (fonctionnelle, applicative et technique) chacun des blocs applicatifs (interne, externe, filiale ou partenaire). De plus il faut :

–faire apparaître les blocs applicatifs concernant l’infrastructure (annuaire, SSO, supervision, ...) et

–pour chaque bloc applicatif, il faut préciser les aspects liés aux contraintes de sécurité, aux performances, à la continuité de service et à la géographie

Description fonctionnelle

– Objectifs

– Classification (Critique, Important, Utile)

– Entrées / Sorties

– Fonctions

– Classes concepts gérées (matrice entités gisement de données et référentiel) / Type gestion (création, modification, suppression, lecture)...

Description applicative et technique

– Développement spécifique ou progiciel

– Volume traité (volumétries statiques et dynamiques)

– Disponibilité (contraintes de disponibilité, plage horaire TP, fenêtre d’exécution des batchs)

– Fiabilité, niveau de service, performances attendues

– Matériel

– Système d’exploitation

– SGBDR ou SGF (Système de Gestion de Fichiers)

– Middleware

– Types de sites concernés (notion de localisation et de géographie)

– Sécurité (identification, authentification, gestion des habilitations, intégrité, confidentialité et non répudiation (données et échanges)...

“[Trading applications trade-offs : Databases and low-latency](http://stackoverflow.com/questions/7281391/trading-applications-trade-offs-databases-and-low-latency)

Caching: Cache the rules in your application, so you don't need to hit the database for every trade

Threading: Put the code that stores the transaction data in the database into another thread. Like this, you can route the trade to the exchange immediately and safe the data in the database parallel to that.”

www.ksvali.com/wp-content/uploads/2009/08/fixed\_income\_trading\_system\_architecture.pdf

<http://www.cisco.com/web/strategy/docs/finance/TradingFlrArch110707.pdf>