



Université du Québec  
**École de technologie supérieure**

RAPPORT TECHNIQUE  
PRÉSENTÉ À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE  
DANS LE CADRE DU COURS LOG792 PROJET DE FIN D'ÉTUDES EN GÉNIE  
LOGICIEL

**Les Emballages de la Beauce**  
**EB - Online**

**Auteur**

**Jean-René Fréchette**

**FREJ29058006**

DÉPARTEMENT DE GÉNIE LOGICIEL ET DES TI

**REMIS A**  
**LUCIE CARON**  
**PRÉPOSÉE AUX AFFAIRES ÉTUDIANTES**

**Professeur superviseur**

**Alain April**

MONTREAL, LE 13 AVRIL 2011  
HIVER 2011



## **REMERCIEMENTS**

J'aimerais remercier Martin Arts qui ma permis d'avoir ce projet et par ca par patience et ca disponibilité m'a permis de le développer

## **Les Emballages de la Beauce**

### **EB - Online**

#### **RÉSUMÉ**

Le projet EB Online est un projet d'application web transactionnel pour les Emballages de la Beauce inc. Le projet vise à créer selon les principes de l'ingénierie du logiciel une application web permettant à la compagnie de montrer les informations sur ces services, produit et pouvoir commander ces mêmes produits.

Ce projet se veut une première étape de la compagnie pour informatiser ces opérations.

## TABLE DES MATIÈRES

	Page
INTRODUCTION .....	1
CHAPITRE 1 CONTEXTE.....	2
1.1 Contexte du projet EB Online.....	2
CHAPITRE 2 ANNALYSE .....	3
2.1 Paramètre initial du projet.....	3
2.2 Problème et Besoins exprimés par le responsable .....	3
2.3 Explication du choix du scope .....	4
2.4 Vision et SRS.....	4
2.5 Risques.....	5
2.6 Difficultés .....	6
CHAPITRE 3 CONCEPTION.....	7
3.1 Architecture des systèmes.....	7
3.2 Diagramme de classe .....	8
3.3 Conception des pages web.....	10
CHAPITRE 4 IMPLÉMENTATION .....	11
4.1 Choix des technologies .....	11
4.1.1 Hibernate :.....	11
4.1.2 Tomcat 7 :.....	12
4.1.3 MySQL : .....	12
4.1.4 Struts 2 :.....	12
4.1.5 Apache 2.2 : .....	12
4.1.6 Java : .....	13
4.2 Autres Technologies possibles.....	13
4.2.1 ASP.NET/C# :.....	13
4.2.2 PHP :.....	13
4.3 Autres outils de développement.....	13
4.4 Autres considérations lors de l'implémentation .....	14
CHAPITRE 5 TEST .....	15
5.1 Test unitaire .....	15
5.2 Test Intégration .....	15
5.3 Test système.....	15
CHAPITRE 6 ÉVALUATION.....	16
6.1 Évaluation du projet EB Online.....	16
6.1.1 Avantage .....	16

6.1.1.1	Extensible.....	16
6.1.1.2	Par cher .....	16
6.1.1.3	Repose sur des technologies fiables.....	16
6.1.2	Limite.....	17
6.1.2.1	Utilisation de plusieurs technologies .....	17
6.1.2.2	Support.....	17
6.1.2.3	Ne réponds pas à tous les besoins.....	17
6.1.3	Amélioration possible .....	17
6.1.3.1	Optimisation :.....	17
6.1.3.2	Affichage graphique : .....	18
6.1.3.3	AJAX .....	18
CONCLUSION		19
7.1	Conclusion .....	19
RECOMMANDATIONS .....		20
8.1	Recommandations.....	20
LISTE DE RÉFÉRENCES .....		21
BIBLIOGRAPHIE.....		22
ANNEXE I VISION .....		23
ANNEXE II SRS .....		25
ANNEXE III ARCHITECTURE.....		26
ANNEXE IV DIAGRAMME DE CLASSE.....		28







## **LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES**

**SRS** - Software Requirements Specification

**AJAX** - Asynchronous Javascript and XML

**MVC** - Model-Vue-Contrôleur



## **INTRODUCTION**

Le présent rapport présente le contexte, l'analyse et les choix fait lors du projet EB Online. Il ce veut un résumer des questions et du raisonnement fait lors du projet EB Online. Dans un premier temps, un résumé du contexte de la compagnie Les Emballages de la Beauce inc. et des problèmes que celle-ci rencontre. Par la suite, la solution proposée sera présentée. Les choix faits pour concevoir une solution qui répond aux besoins seront en suite présentés. Suivra, les méthodes pour évalué cette solution. Puis finalement, la conclusion et les recommandations.

## CHAPITRE 1

### CONTEXTE

#### 1.1 Contexte du projet EB Online

Les Emballages de la Beauce (EB) inc. est une entreprise qui fournit du matériel d'emballage à plusieurs compagnies à Montréal et partout au Québec. La compagnie désire créer un site web pour faire la promotion de leurs produits et services. La compagnie désire aussi automatiser plusieurs de leur tâche afin de libérer du temps pour ces employés.

Il est à noter que la compagnie tente de diversifier ces activités. En ce moment, la compagnie a deux divisions :

**Distribution de produit d'emballage** : la compagnie offre différents produits pour l'emballage soit des boîtes imprimées, non imprimées, des papiers à bulles, etc.

**Service de récupération** : depuis peu la compagnie offre un service de récupération où elle offre à ces clients la possibilité de récupérer boîte et résidu de plastique. Elle revend ensuite les boîtes usées à d'autres compagnies.

D'ici quelques mois la compagnie veut pouvoir offrir des jeux-jouets pour les 3-7 ans dont la particularité est que les produits sont faits de matière recyclée ou sont facilement recyclables. Ce marché est très fort en Europe.

## CHAPITRE 2

### ANNALYSE

#### 2.1 Paramètre initial du projet

Lors de l'analyse initiale du projet, plusieurs choses sont ressorties de façon un peu chaotique. Le client désirait plusieurs choses :

-La création d'un site web

-Possibilité de recommander automatiquement des produits lorsque l'inventaire descend en dessous d'un seuil.

-Contacter les clients lorsqu'une commande est arrivée.

-Etc.

Pour bien servir le client, il était essentiel de bien identifier le problème et les besoins.

#### 2.2 Problème et Besoins exprimés par le responsable

Le principal problème du client qui a été identifié est celui-ci :

##### **Les marges de profits diminuent**

Pour régler ce problème il y a deux solutions : augmenté les revenus ou diminuer les couts

Une Analyse plus poussée a permis de bien identifier les besoins du client.

Deux besoins ont été identifiés :

1. La compagnie désire agrandir son marché et vise maintenant les particuliers.
2. Le logiciel actuel de gestion d'inventaire (prostock) ne répond plus à leurs attentes.

Seul le premier besoin a été analysé lors de ce projet

### **2.3 Explication du choix du scope**

Pour bien servir le client et puisque que le client n'avait pas d'idée fixe. Il a été convenu que le présent projet servirait à créer un site web transactionnel permettant :

- visionner les informations sur la compagnie
- visionner les services de la compagnie
- Visionner les produits
- commander des produits

Du à la limite de temps et à la nature du projet, il a été convenu que seulement ceux déjà inscrits pourraient commander soit les clients actuels et qu'il n'y aurait pas la possibilité de payer, et ce, pour raison de sécurité. Le temps alloué ne permettait pas de bien tester et surtout s assurer de la confidentialité des données.

Ce projet est considéré comme une première étape d'un projet plus gros.

Les recommandations présentent les prochaines étapes pour bien combler les besoins exprimés.

### **2.4 Vision et SRS**

Les Annexes I et II présentent le document de vision et le SRS du projet.

Ces deux documents représentent la base de raisonnement pour monter une bonne solution au client.

Les points importants des deux documents sont les utilisateurs et les use-cases car ils représentent l'interaction qu'auront les utilisateurs du système. Ces use-cases sont :

- UC1 Ajouté un item au panier
- UC2 Modifier un item au panier
- UC3 retirer un item au panier
- UC4 visionner le panier
- UC5 passer une commande
- UC6 visionner l'historique des commandes
- UC7 s'authentifier
- UC8 Ajouté un item à l'inventaire
- UC9 Modifier un item à l'inventaire
- UC10 retirer un item à l'inventaire
- UC11 visionner les commandes
- UC12 ajouter un client
- UC13 modifier un client
- UC14 retirer un client

Le SRS présente plus en détail ses use-cases.

## 2.5 Risques

Connaître les risques permet de mieux se préparer si elles surviennent. Les risques encourus lors du développement étaient :

**Le projet ne supporte pas la mission du client** : il avait un risque que le projet ne soit finalement pas une réussite, car il ne pouvait pas supporter les processus d'affaire du client.

**N'a pas le temps de compléter le projet** : le temps dans ce projet était un facteur clé

**Le client n'a pas d'expérience sur des projets similaires** : Le client n'avait aucune expérience dans des projets similaires. Ce qui comporte un risque, car il faut être proactif pour aller chercher les informations qu'il nous faut.

**Intégration avec les systèmes existants** : Il est encore possible que la solution ne soit pas viable avec la solution existante (prostock).

A part intégration avec les systèmes existants, aucun autre risque n'est apparu durant la conception de la solution. Intégration avec les systèmes existants reste néanmoins un risque important, car aucune analyse n'a été faite tant qu'à la viabilité d'intégration de la solution avec le système existant (prostock). Ce problème sera adressé dans l'étape 2 du projet (voir recommandations).

## **2.6 Difficultés**

La principale difficulté lors du développement du projet a été l'accès aux données techniques de la compagnie. À cause des conflits d'horaire et de l'accès au programme (prostock) difficile, il a été impossible de déterminer s'il était viable de lier la solution avec le système existant. Un effort particulier lors de la conception a été fait pour minimiser ce problème.

L'autre difficulté a été de visualiser le processus d'affaire du client. En ce moment, aucun produit n'est disponible pour la vente aux particuliers. Cette aspect rend difficile la prise de décision. C'est pourquoi seulement les clients actuels ont été considérés dans cette première version.



## CHAPITRE 3

### CONCEPTION

#### 3.1 Architecture des systèmes

L'annexe III présente l'architecture du système.

Les hypothèses qui ont été considérées lors de la conception du système sont :

-Il y a très peu d'information disponible sur le système prostock. Cependant, il est raisonnable de penser que des informations contenues dans le système seront transférables.

-De plus, selon le client des modifications ont été demandées pour modifier prostock sans succès il est donc fort possible que le système soit totalement à refaire.

-Il n'y a pas de limite budgétaire, cependant un effort a été fait pour que la solution ne soit pas dispendieuse.

Points importants de la conception :

**Trois machines** : La conception présente les systèmes sur trois machines différentes. Cet aspect n'est pas nécessaire au bon fonctionnement de l'application, mais fortement suggéré pour des raisons de sécurité des données et des systèmes. Ce genre de conception permet de mieux isoler les données, ce qui offre une meilleure protection en cas de panne.

Cet aspect permet aussi de séparer le système en deux parties soit les applications qui sont disponibles à l'externe de la compagnie et ceux disponibles à l'interne. Cette séparation est importante, conceptuellement parlant, car elle permet d'identifier les systèmes qui faut tester contre les risques d'intrusion.

**Architecture en couche de l'application web** : L'architecture 3 tiers : présentation, domaine et données est un modèle qui a fait ces preuves et permet de mieux concevoir des

applications, car il permet de diviser pour mieux régner surtout si l'application a tendance à changer dans le temps ce qui est une caractéristique du projet actuel.

Les Systèmes importants :

**-Application Web**

Système principal du projet. Site A et Site B représentent des vues du même site. Le client à exprimer le désir de séparer les différentes divisions de la compagnie en plusieurs sites qui afficheront les informations est produit propre à celle-ci. Site A et B représente cela.

**-Base de donné**

Système qui assure la persistance des objets du domaine d'affaires, Le fait qui soit détaché du système permet d'assurer une meilleure sécurité et une meilleure accessibilité pour les autres systèmes

**-Système autonome**

Système qui roule en parallèle et de façon autonome des autres systèmes. Représente des taches qui pourront être facilement réalisables (ex. production de rapport) lorsque la solution sera pleinement opérationnelle.

**-Prostock**

Système actuel de gestion de l'inventaire de la compagnie. Une solution optimale serait d'être capable de lier ce système au système présentement développé.

### 3.2 Diagramme de classe

L'annexe IV présente le diagramme de classe :

**Patron de conception :**

Les patrons de conception objet sont un moyen efficace de bien concevoir une application. Ils permettent de résoudre des problèmes communs avec des solutions éprouvées. Voici les patrons utilisés lors du développement

**MVC (Model-View-Contrôleur):** Est un patron qui permet de séparer les données de sont affichage et de sont contrôleur qui sert à modifier ces mêmes données .Il permet de facilités la conception en divisant le problème. Il permet de mieux tester, car les objets sont indépendants. Cet aspect est à la base de l'application.

**Factory :** Un objet doit souvent être créé d'une certaine manière à chaque fois. Un objet Factory permet de centraliser la création de ces objets. Cet aspect est utilisé lors de la création des utilisateurs.

**Singleton :** Le singleton permet qu'une seule instance d'un objet soit disponible dans le système. Le singleton est utilisé dans les factory.

Présentation des classes.

**Domaine d'affaire :** les objets qui représentent de processus d'affaires

- Client : Un client de la compagnie
- Commande : La commande du client
- Produit : Les produits offerts par la compagnie
- Item : Les items de la commande
- Facture : factures du client
- Categorie : Les catégories qui forme les sous-groupes de produits

**Données :** la persistante du système

- DataBaseObject : Objet qui représente une persistance dans la basse de données.

Toute la classe du domaine d'affaires hérite de cette classe.

-Utilisateur : Object qui représente un utilisateur du système. Il possède trois sous-classes qui ont des droits différents soit :

- Admin : administrateur, permet d'ajouté des produits dans le système
- UtilisateurInscrit : Visiteur qui c'est authentifié, permet de commander

-UtilisateurGenerique : Visiteur anonyme

-UtilisateurFactory : Factory permettant la création des utilisateurs

-DataManager : point d'entrée des accès à la persistance du système. Il est contextualisé avec les informations de l'utilisateur permettant d'offrir seulement l'information qui a droit.

-ProduitManager : spécialisé dans les requêtes de produit

-CommandeManager : Spécialisé dans les requêtes de commandes

-TaxeFactory : permet d'aller chercher les informations sur les taxes

**Contrôleur** : Contrôle du système

-BaseControleur : Offre les services communs aux autres contrôleurs. Tous les autres contrôleurs héritent de cette classe.

-CartControleur : Spécialisé dans le contrôle du panier.

-CommandeControleur : Spécialisé dans la prise de commande.

-LoginControleur : Spécialisé dans l'authentification des utilisateurs.

-ProduitControleur : Spécialisé dans les produits.

### 3.3 Conception des pages web

Voir SRS pour les maquettes proposées.

Les motivations lors de la conception des pages web ont été :

**Simplicité** : présenter un minimum d'information pour faciliter l'apprentissage de l'interface.

**Familier** : présenter une conception qui était familière à tous les utilisateurs du web (bar de navigation en haut par exemple) pour éviter de les perdre dans le site.

## CHAPITRE 4

### IMPLÉMENTATION

#### 4.1 Choix des technologies

Les choix technologiques est une étape importante. Leur choix doivent prendre en compte plusieurs facteurs et doivent permettre de faciliter la vie des développeurs tout en satisfaisant les besoins du client. Les facteurs considérés lors de la prise de décision sont :

**Prix** : le choix devra soit être gratuit ou très peu chers. Les projets Open-Source ont été explorés à cause de cette raison.

**Support** : La solution devra compter sur un bon support technique et documentation.

**Expérience** : Pour éviter des risques et ralentir le développement, le développeur devra avoir un peu d'expérience avec la technologie.

**Valeur ajoutée** : Pour qu'une technologie soit choisie, elle devra ajouter de la valeur au projet (ex : réduire le temps de développement)

##### 4.1.1 Hibernate :

Hibernate est une solution open-source de persistance objet. Il permet de sauvegarder et charger des objets dans une base de données(ou autre médium avec un peu de programmation) d'une façon simple et efficace. Hibernate a été choisi parce qu'il permettait d'instaurer une sauvegarde des objets du domaine d'affaire en minimum d'effort, mais surtout avec risque minimum d'erreur.

#### **4.1.2 Tomcat 7 :**

Tomcat est un serveur open-source applicative. Il permet de rouler des applications web java (servlet). Il a été choisi, car il est une solution gratuite, mais surtout fiable qui possède un très bon support communautaire.

#### **4.1.3 MySQL :**

MySQL est une base de donnée gratuite qui l'avantage d'être supporté par une compagnie il est ainsi possible de payer pour avoir une version commerciale et par conséquent du support si besoin. Il n'est pas à négliger que la base de données est très performante et que de nombreuses ressources sont disponibles.

#### **4.1.4 Struts 2 :**

Struts est un framework MVC pour application web java. Il permet d'émuler un fonctionne MVC pour un projet web. Struts 2 a été utilisé principalement parce qu'il est facile a comprendre et surtout il permet de générer des contrôleurs séparés du contexte HTTP (par rapport a struts 1) permettant des tests automatiques plus faciles et une conception plus versatile. Cet aspect pourrait être pratique lors du développement de l'étape 2 du projet (voir recommandation).

#### **4.1.5 Apache 2.2 :**

Apache est un serveur web grandement utilisé et fiable. À cause de ça grande compatibilité avec Tomcat, il sera utilisé comme proxy pour celui si.

#### **4.1.6 Java :**

Le langage java a été choisi surtout parce que le développeur avait beaucoup d'expérience avec celui-ci et qu'il est plateforme indépendante.

### **4.2 Autres Technologies possibles**

Voici une liste des technologies considérées, mais pas choisies pour le développement du projet :

#### **4.2.1 ASP.NET/C# :**

La technologie .NET été considéré initialement pour le développement sont principal avantage c'est qu'il contient plusieurs outils de développement efficace et performant en plus d'avoir un très bon support de la part de Microsoft. Cependant, le développeur avait moins d'expérience sur cette plateforme (1,5 ans) que sur Java (4-5 ans) et puisque qu'il était vraisemblable que le projet allait prendre du retard, Java a été favorisée. En plus, l'environnement de développement Java est gratuit comparativement au .NET ce qui est un avantage considérable.

#### **4.2.2 PHP :**

PHP est un autre langage très utilisé pour le développement web. Il a avantage d'être gratuits, performant et offre un bon support. Il n'a pas été considéré à cause du manque d'expérience du développeur sur cette plateforme (4 mois).

### **4.3 Autres outils de développement**

Les outils suivant ont été utilisé lors du développement :

Eclipse, argoUml, maven, JUnits, dbUnit et Selenium(voir TEST)

#### **4.4 Autres considérations lors de l'implémentation**

Puisque l'application était une application web une attention particulière a été faite pour valider les entrées du système et évité ainsi des comportements imprévus si certaines valeurs étaient entrées.



## CHAPITRE 5

### TEST

#### 5.1 Test unitaire

Les tests unitaires ont été effectués à l'aide de Junit. Les Objets du domaine d'affaires et une partie des contrôleurs ont été testés surtout la validation. Du a leur nature, les objets de la couche donnée n'ont pas été testés. Ils ont été testés durant la phase test d'intégration.

#### 5.2 Test Intégration

dbUnit a servis au test d'intégration. Il a permis de peupler la base de données avec des données de test et a permis de vérifier que l'intégrité des données.

#### 5.3 Test système

Selenium a servi pour tester le système en entier. À cause de son IDE dans firefox, il permet de créer des tests automatiques rapidement. Selenium a permis de facilité les tests de cas limite.

## CHAPITRE 6

### ÉVALUATION

#### 6.1 Évaluation du projet EB Online

Tout bon projet d'ingénierie ce doit d'être évaluer afin ce s'assurer qu'il répond bien aux attentes et besoin du client en plus d'offrir la meilleure solution possible. Cette évaluation est nécessaire aussi pour voir les améliorations possibles à apporter.

##### 6.1.1 Avantage

###### 6.1.1.1 Extensible

La solution actuelle du à ça séparation des données de l'application permet d'ajouté des systèmes qui améliorons les services rendus par celui-ci, car ils n'auront qu'a ce brancher à la base de données pour offrir de nouveau service (génération de rapports par exemple). L'architecture en couche permet le même résultat.

###### 6.1.1.2 Par cher

La solution offerte n'est pas dispendieuse puisqu'il repose sur des projets open-sources.

###### 6.1.1.3 Repose sur des technologies fiables

Les technologies utilisées sont pour la plus part en développement depuis des années. Ils ont fait leur preuve et ils sont fiables. Ce fait réduit les risques de défaillance du aux composantes formant la solution.

## **6.1.2 Limite**

### **6.1.2.1 Utilisation de plusieurs technologies**

La solution actuelle utilise plusieurs technologies et par conséquent de fichier de configuration. Une mauvaise configuration pourrait dégrader sérieusement les performances du système et du au nombre de fichier à modifier il pourrait être très difficile de déterminer ou la source du problème est.

### **6.1.2.2 Support**

La plupart des technologies utilisées sont open-source et par conséquent il n'y a pas de support direct d'une compagnie. Le client pourrait avoir de la difficulté si un problème survient.

### **6.1.2.3 Ne réponds pas à tous les besoins**

La solution actuellement disponible ne répond pas à tout les besoins exprimés initialement pas le client. Cette lacune devra être réglée ultérieurement (voir recommandations)

## **6.1.3 Amélioration possible**

### **6.1.3.1 Optimisation :**

La solution actuelle n'est pas optimisée. Des changements pourraient être apportés à la gestion de la mémoire et des ressources pour utiliser le maximum des machines.

### **6.1.3.2 Affichage graphique :**

L'affichage graphique a été conçu par l'ingénieur pour être efficace, mais pas pour être beau. Un infographiste pourrait être engagé pour régler ce problème.

### **6.1.3.3 AJAX**

Ajax pourrait être utilisé pour améliorer l'interactivité de la solution permettant une expérience utilisateur plus riche.

## CONCLUSION

### 7.1 Conclusion

En conclusion, la solution présentée malgré ses limites représente un bon départ pour doter la compagnie Les Emballages de la Beauce inc. d'une présence sur le web. À cause de la partition entre les données et l'application, il est raisonnable de penser que cette solution pourra être modifiée afin de bien répondre aux autres besoins de la compagnie et aux besoins futurs. Cette solution pourra aussi aider le client à mieux visualiser et cerner ses besoins futurs.

## **RECOMMANDATIONS**

### **8.1 Recommandations**

Le présent projet n'est qu'une première étape pour bien combler les besoins exprimés.

Il n'est pas recommandé d'utiliser la solution de commande en ligne telle quelle. Il a été conçu avec la mentalité qu'il servirait de point de référence pour les prochaines étapes.

Les prochaines étapes sont :

1. Modifier Prostock pour permettre de communiquer avec la nouvelle application web sinon vérifier si le client est intéressé à la création d'un nouveau système.
2. Ajouter un module de paiement électronique et de création de compte client dans l'application web.
3. Réévaluer les besoins et modifier s'il y a lieu. Avec la solution maintenant fonctionnelle, de nouveaux besoins pourraient apparaître.

A plus long terme : Plusieurs exigences techniques ont été supposées lors de l'analyse. Bien qu'elle soit réaliste et raisonnable, elle ne se base pas sur des données réelles. Dans le future, il faudrait analyser le trafic et reconsidérer si ces exigences sont bien réelles.

## LISTE DE RÉFÉRENCES

**Struts** : <http://struts.apache.org/>  
**Hibernate** : <http://www.hibernate.org/>  
**Tomcat** : <http://tomcat.apache.org/>  
**MySQL** : <http://www.mysql.com/>  
**APACHE** : <http://www.apache.org>  
**Selenium** : <http://seleniumhq.org/>  
**DBUNIT** : <http://www.dbunit.org/>  
**Junit** : <http://www.junit.org/>

## BIBLIOGRAPHIE

LEFFINGWELL, D. & D. WIDRIG, *Managing Software Requirements – A Use Case Approach*. 2<sup>nd</sup> edition, Addison-Wesley, 2003.

Abran, A., and Moore, J.W. (Exec. Eds.), Bourque, P. and Dupuis, R. (Eds.) *Guide to the Software Engineering Body of Knowledge*, 2004 Version

LARMAN, C., *Applying UML and patterns: an introduction to object-oriented analysis and design and the unified process*. 3<sup>rd</sup> Ed., Upper Saddle River (NJ), Prentice-Hall PTR, 2005.

BASS, L., P. CLEMENTS et R. KAZMAN, *Software Architecture in Practice*, 2<sup>nd</sup> edition, Reading (Mass.), Addison Wesley Longman, Inc., 2003.



## **ANNEXE I**

### **VISION**

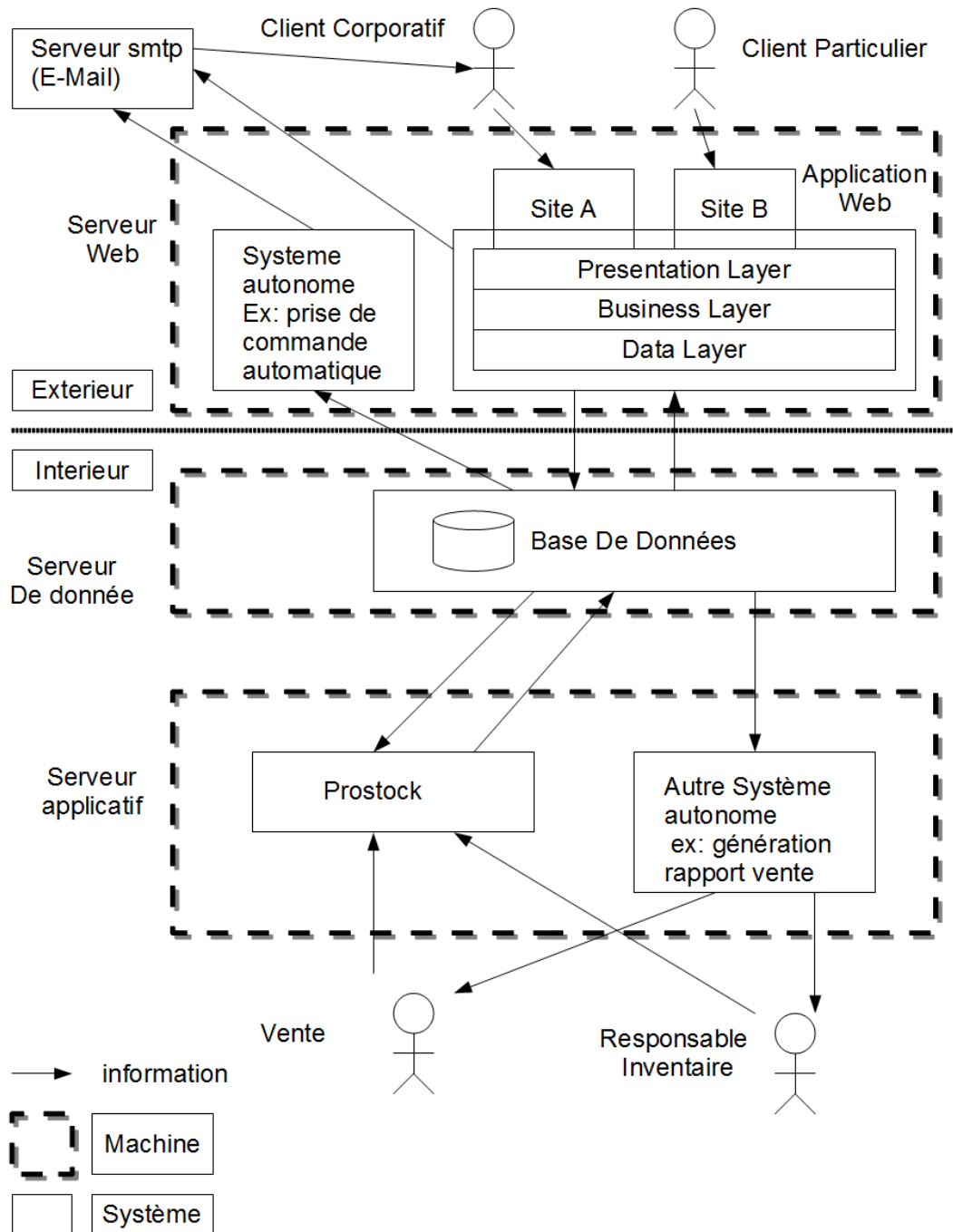


## **ANNEXE II**

**SRS**

**ANNEXE III**

**ARCHITECTURE**



**ANNEXE IV**

**DIAGRAMME DE CLASSE**

