Thomas **Petillon**

les Cahiers du Programmeur

ASP.NET

ISBN : 2-212-11210-6

EYROLLES

L'étude de cas « Les Savons du Soleil »

1

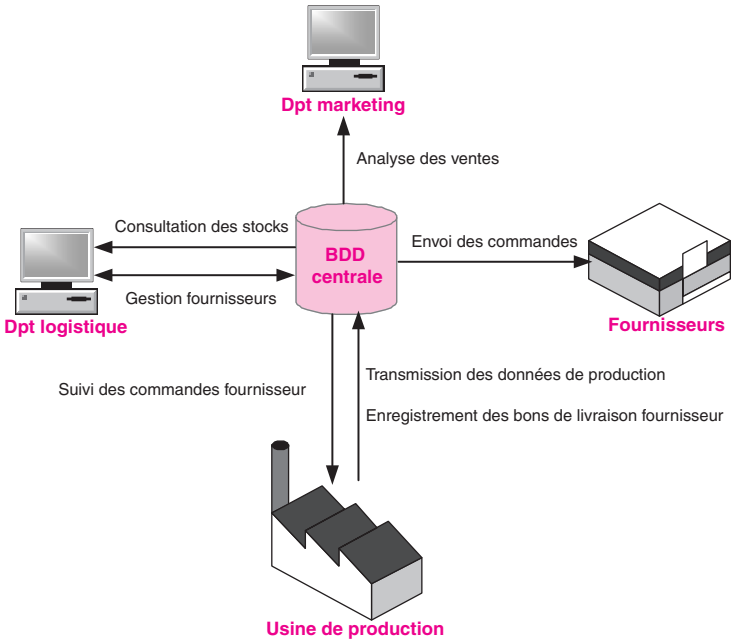
ASP.NET

Base de données | Intranet | Échanges XML | Services Web | ASP.NET | MSDE | Web Matrix

SOMMAIRE

MOTS-CLÉS

- Base de données centralisée
- Intranet
- Échanges XML
- Services Web
- ASP.NET
- MSDE
- Web Matrix



Dans ce chapitre, on présente en détail l'étude de cas qui servira de fil conducteur au reste de cet ouvrage : après avoir exposé les difficultés de circulation d'informations entre les différents sites de notre société exemple, on explique comment la mise en place une base de données centralisée régulièrement mise à jour depuis l'usine et dotée d'une interface de gestion adéquate peut améliorer la situation. Enfin, on détaille et on justifie les choix techniques retenus pour la réalisation de l'étude de cas : ASP.NET, MSDE et Web Matrix.

Une PME géographiquement dispersée

La société des Savons du Soleil est une PME spécialisée dans la vente par correspondance de cosmétiques à base de produits naturels : savons au miel, shampoing au tilleul, bain douche à la lavande…

Les activités de la société sont réparties entre trois entités géographiquement séparées :

- *Marseille* : l'usine de production assure la fabrication et le stockage des produits, ainsi que le traitement des commandes (prise de commande, préparation et envoi des colis) ;
- *Lyon* : le département logistique assure le suivi des stocks et les commandes de matières premières auprès des fournisseurs ;
- *Paris* : le département marketing travaille sur les actions publicitaires, le développement de nouveaux produits et l'analyse des ventes.

Des échanges d'informations fastidieux entre les sites

Les trois sites doivent échanger régulièrement des informations :

- le département logistique a besoin d'être renseigné sur l'évolution des stocks ;
- le département marketing doit analyser les chiffres de vente ;
- l'usine veut suivre les commandes passées par le département logistique.

Malheureusement, la remontée des informations depuis l'usine est complexe à gérer (figure 1–1), chaque unité possédant son propre système de gestion des données :

- l'usine de production utilise un logiciel de gestion commerciale dans lequel sont enregistrées toutes les commandes clients. À chaque fin de semaine, les informations relatives aux nouvelles commandes (coordonnées des clients, détail des produits commandés) sont extraites et transmises aux autres unités ;
- le département logistique suit l'évolution des stocks grâce à un tableur, dont le contenu est actualisé manuellement chaque semaine à partir des données fournies par l'usine ;
- le département marketing effectue ses analyses des ventes grâce à une base de données relationnelle, actualisée chaque semaine, elle aussi, à partir des informations de l'usine.

Quant à la gestion des commandes fournisseur – qui implique trois acteurs distincts – elle est, elle aussi, fastidieuse (figure 1–2) :

- le département logistique passe commande par télécopie auprès du fournisseur ;

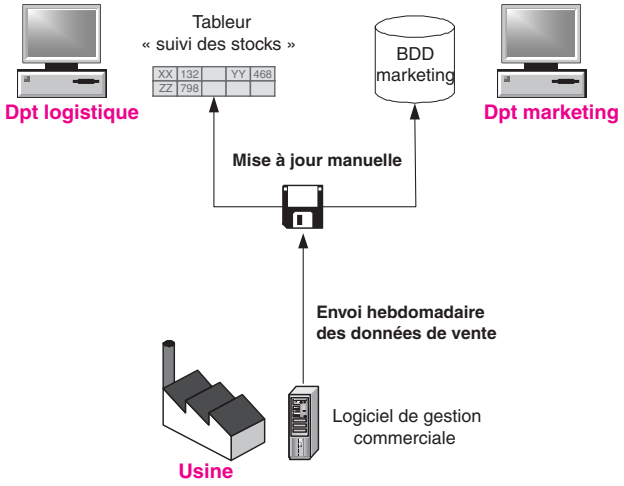


Figure 1–1 Transfert des données commerciales (avant)

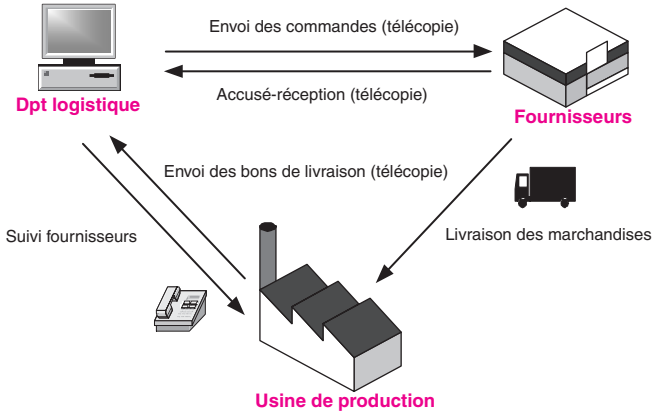


Figure 1–2 Gestion des commandes fournisseur (avant)

- le fournisseur accuse réception de la commande, par télécopie également ;
- le bon de livraison remis par le fournisseur lors de la livraison de la marchandise est envoyé par télécopie, depuis l'usine, au département logistique.

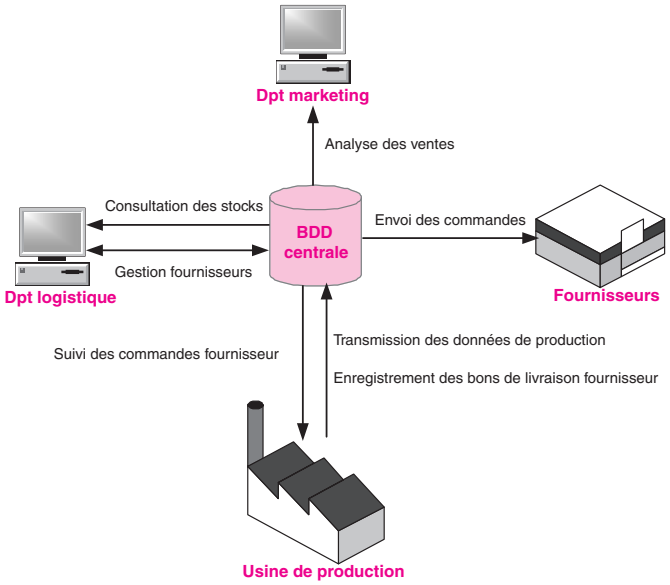
Inconvénients liés à la situation actuelle

La situation actuelle présente un certain nombre d'inconvénients :

- les départements marketing et logistique n'ont pas accès aux informations « en temps réel », ce qui n'est pas dramatique pour le premier, mais nettement plus gênant pour le second ;
- les opérations de mise à jour des données à partir des informations envoyées par l'usine sont longues et fastidieuses ; de plus, elles contraignent le département marketing à employer à mi-temps un administrateur de base de données ;
- le département logistique ne dispose que de moyens limités d'analyse et souhaiterait disposer d'une base de données de suivi des stocks ;
- la gestion des commandes fournisseur est entièrement manuelle et pénible ;
- l'usine de production est contrainte de téléphoner au département logistique chaque fois qu'elle désire obtenir des informations sur les commandes fournisseur en cours.

En résumé, le fait d'avoir des sources d'information décentralisées est pénalisant pour l'activité de l'entreprise.

Le projet de nouvelle infrastructure technique



Pour augmenter sa productivité au quotidien, la société des Savons du Soleil a donc décidé de mettre en place une nouvelle infrastructure technique :

- mise en place d'une base de données centralisée ;
- accès sécurisé à cette base pour toutes les unités via une interface Web permettant d'effectuer le suivi des stocks et des commandes fournisseur, ainsi que l'analyse des ventes ;
- mise à jour automatique et quotidienne de la base, à partir d'informations extraites des fichiers de gestion commerciale utilisés à l'usine ;
- nouveau mécanisme d'échange avec les fournisseurs, fondé sur XML.

Dans les sections qui suivent, nous détaillons les différentes composantes du projet :

- structure de la base de données ;
- interface de gestion de la base ;
- transferts de données automatisés (échanges fournisseur, mise à jour des données depuis l'usine).

Structure de la base de données

Pour simplifier la présentation de l'étude de cas – dont la base n'est pas le sujet principal – on se concentrera uniquement sur le modèle de données. Bien entendu, dans le contexte d'une application réelle, il faudrait aussi aborder d'autres sujets comme les règles d'intégrité référentielles (index, contraintes, déclencheurs), l'optimisation des performances, la gestion des transactions, la sécurité (authentification, rôles), la disponibilité de la base ou encore son administration.

Mise en place d'une base de données centralisée

L'intérêt de mettre en place une base centrale est de disposer d'un référentiel de données unique partagé entre toutes les unités : on évitera ainsi les pertes de temps dues à la mise à jour manuelle des données de suivi des stocks et d'analyse des ventes, et on donnera à tous l'accès à une information à jour. Cette base, accessible via Internet, sera hébergée chez un prestataire externe.

Les données à stocker dans la base peuvent être réparties en deux sous-ensembles :

- données utilisées pour l'analyse des ventes et le suivi des stocks ;
- données utilisées pour la gestion des commandes fournisseur.

DANS UN CAS RÉEL
Axes d'analyse plus nombreux

Dans un cas réel, les axes d'analyse des ventes seraient beaucoup plus nombreux (par produits, par types de client, etc.). De même, le suivi de la logistique serait probablement plus complexe (gestion des stocks de produits intermédiaires...).

Données nécessaires à l'analyse des ventes et au suivi des stocks

Pour éviter une trop grande complexité, on se limitera aux hypothèses suivantes :

- le département marketing souhaite pouvoir consulter la répartition des ventes par famille de produits ou par région, pour un mois donné ;
- le département logistique souhaite pouvoir suivre l'évolution des stocks de produits finis.

Pour effectuer ces opérations, il faut au minimum stocker dans la base :

- la liste des produits ;
- la liste des familles de produits ;

- la liste des mouvements de stocks (arrivée ou sortie du stock) ;
- la liste des ventes mensuelles par famille de produits et par région.

Tableau 1–1 Données nécessaires à l'analyse des ventes et au suivi des stocks

Entité	Caractéristiques, relations
Produit	• Désignation et description du produit • Un produit est rattaché à une famille de produits
Famille de produits	• Nom de la famille de produits • À une famille de produits sont rattachés de 1 à n produits
Mouvement de stocks	• Date, type de mouvement, produit concerné et quantité (positive ou négative) • Un mouvement est rattaché à un produit
Vente	• Ventes par famille de produits et par région • Une donnée de vente est rattachée à une famille de produits et à une région

Le modèle physique correspondant, qui ne comporte aucune difficulté particulière, est présenté figure 1-3.

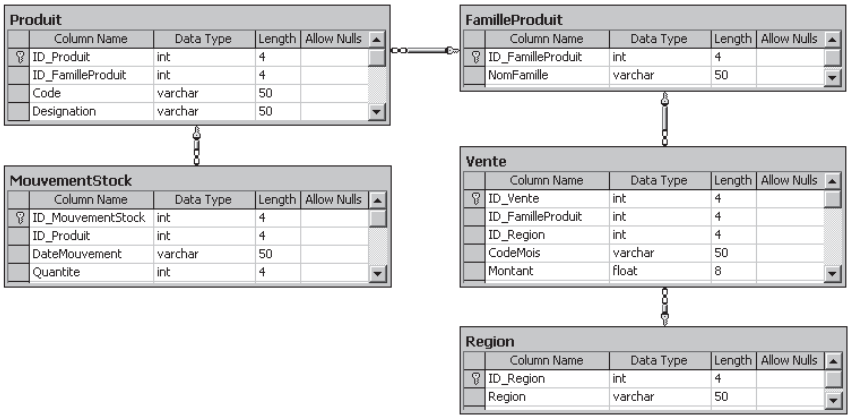


Figure 1–3 Modèle physique de la base : analyse des ventes et suivi des stocks

Notations

Par convention, toutes les clés commencent par « ID_ » comme « IDentifiant ».

Données nécessaires à la gestion des commandes fournisseur

Pour assurer le suivi des commandes fournisseur, le département logistique souhaite pouvoir garder trace des références fournisseur commandées (type et quantité), de la date de livraison prévue et du statut de la livraison (commande livrée ou non).

Par conséquent, il faut stocker dans la base :

- la liste des fournisseurs ;
- la liste des références fournisseur (produits proposés par ces fournisseurs) ;
- la liste des commandes passées auprès de ces fournisseurs (avec les lignes de commandes associées).

Tableau 1–2 Données nécessaires au suivi des commandes fournisseur

Entité	Caractéristiques, relations
Fournisseur	• Code fournisseur, nom et adresse • À un fournisseur sont associées de 1 à n références fournisseur
Commande	• Date création, date livraison prévue, date livraison effective • Une commande est rattachée à un fournisseur • 1 à n lignes de commandes lui sont associées
Ligne de commande	• Référence fournisseur, quantité commandée, quantité livrée • Rattachée à une commande
Référence fournisseur	• Référence et désignation du produit • Associée à un fournisseur

Le modèle physique correspondant est présenté figure 1-4.

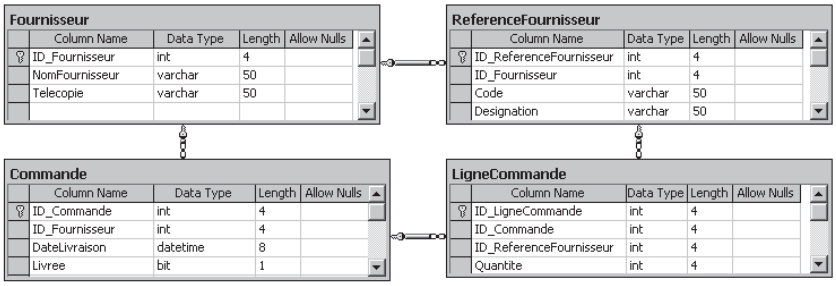


Figure 1–4 Modèle physique de la base : gestion des commandes fournisseur

Interface Web pour la consultation/mise à jour de la base

Les données de la base pourront être consultées et éventuellement mises à jour par les différentes unités de la société, en fonction de leurs besoins. Pour cela, les utilisateurs disposeront d'une interface de gestion permettant :

- le suivi des stocks ;
- la gestion de commandes fournisseur ;
- l'analyse des ventes.

Comme la base est hébergée sur un site externe, et que les utilisateurs se répartissent en trois endroits géographiquement distincts, ces opérations vont s'effectuer par l'intermédiaire d'une interface Web, dont l'accès devra être sécurisé (*intranet*).

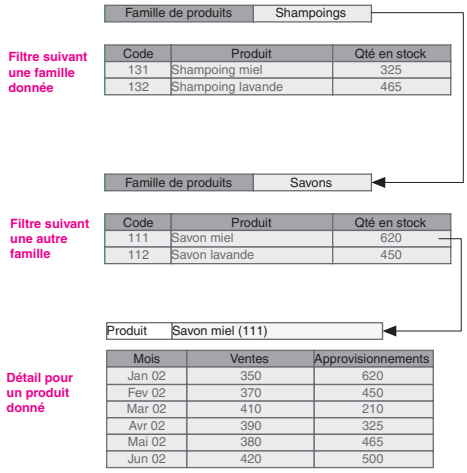


Figure 1–5 Interface de suivi des stocks

Module de suivi des stocks

Le module de suivi des stocks doit proposer :

- l'affichage de la liste des produits avec les quantités en stock correspondantes, filtrée par famille de produits ;
- l'accès à la fiche de détail d'un produit (historique de la variation du stock).

Toutes ces opérations sont présentées figure 1-5.

Module de gestion des commandes fournisseur

Le module de gestion des commandes fournisseur doit permettre :

- la création d'une commande fournisseur ;
- l'affichage de la liste des commandes fournisseur ;
- l'enregistrement d'un bon de livraison fournisseur.

La création d'une commande fournisseur se déroule en plusieurs étapes (voir figure 1-6) :

1. Sélection du fournisseur.
2. Sélection des références, des quantités commandées et de la date de livraison souhaitée.
3. Enregistrement de la commande dans la base.
4. Transmission de la commande au fournisseur (sous forme de fichier XML).

La page de consultation des commandes fournisseur doit présenter une liste récapitulative des commandes : en sélectionnant une commande, on doit accéder au détail correspondant (références et quantité commandées, date de livraison prévue). Enfin, lors de la réception d'une commande à l'usine, il doit être possible de modifier le statut de la commande : « livrée » au lieu d' « en attente de livraison » (voir figure 1-7).

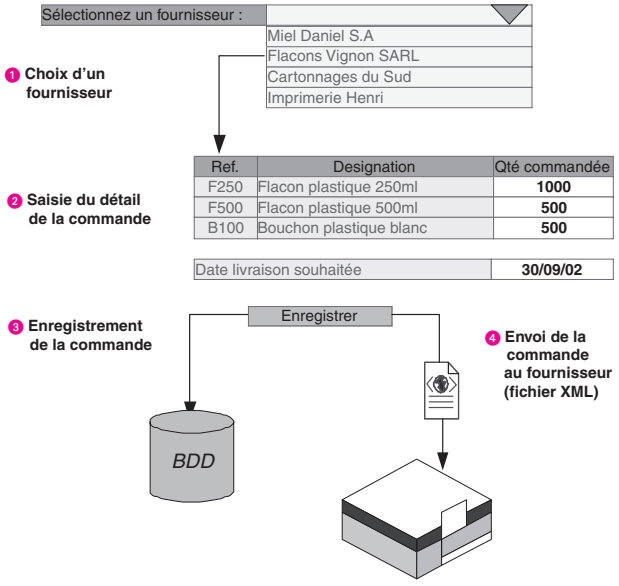


Figure 1–6 Création d'une commande fournisseur

Figure 1–7 Consultation des commandes fournisseur

Module d'analyse des ventes

Ce module doit permettre la consultation du chiffre d'affaires réalisé par famille de produits ou par région sur une période donnée, avec visualisation des résultats sous forme graphique.

La cinématique correspondante est présentée figure 1-8.

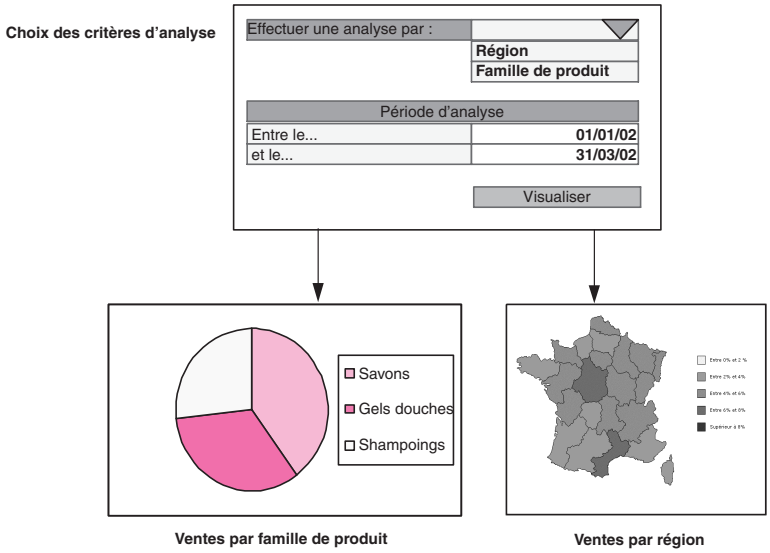


Figure 1–8 Interface d'analyse des ventes

Authentification

Cette base contient des données confidentielles qui ne doivent être consultées et modifiées que par les personnes qui y sont habilitées. Il est donc indispensable de prévoir un mécanisme d'authentification : les utilisateurs devront fournir un identifiant et un mot de passe valides pour pouvoir se connecter à l'application ;

DANS UN CAS RÉEL Sécurisation

Dans le cadre d'une application réelle, il faudrait passer en revue tous les aspects relatifs à la sécurité : protection du serveur Web par un pare-feu et un logiciel anti-virus, chiffrement des communications (SSL), authentification forte des utilisateurs, etc.

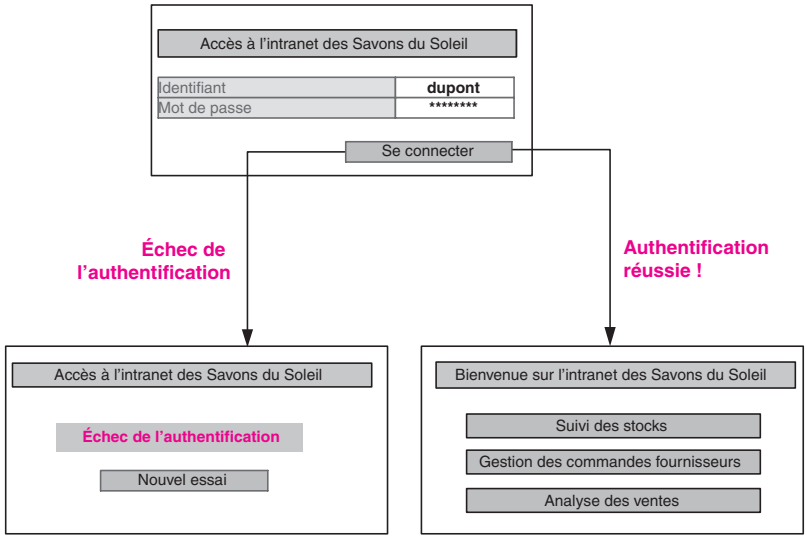


Figure 1–9 Authentification des utilisateurs

de plus, l'accès aux différents modules dépendra des droits de l'utilisateur (gestion des autorisations). Le mécanisme requis pour l'authentification est présenté figure 1-9.

Mise à jour automatique des données de la base

Le suivi des stocks et l'analyse des ventes n'auront de réel intérêt que si les données commerciales sont mises à jour régulièrement depuis l'usine : c'est pourquoi il est prévu de mettre en place un mécanisme de remontée d'informations depuis le logiciel de gestion commerciale vers la base centralisée.

Ce mécanisme reposera sur l'implémentation d'un service Web permettant la mise à jour de la base, qui sera appelé depuis l'usine de production (voir figure 1-10) ; dans un souci de simplification, on se limitera à la mise à jour des données de stocks.

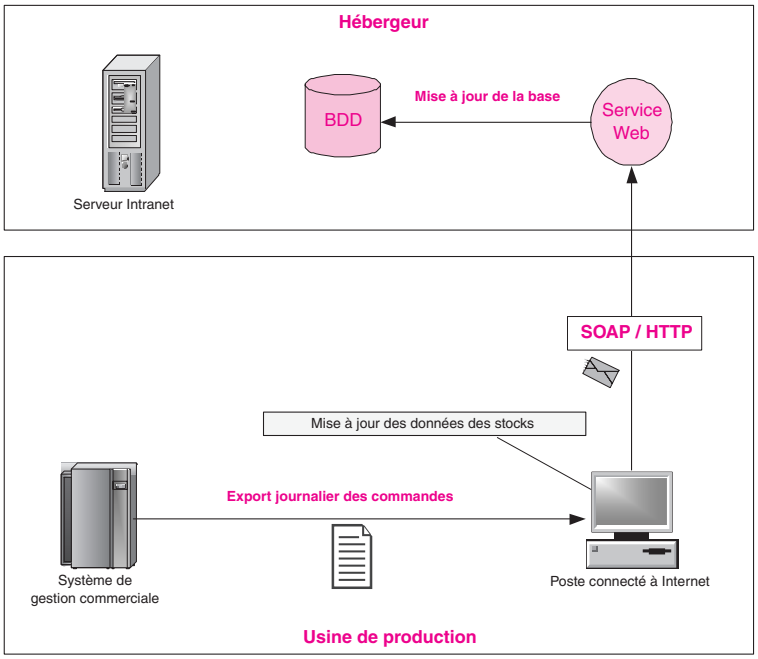


Figure 1–10 Mise à jour des données de stock via un service Web

Le cahier des charges étant établi, nous allons maintenant choisir les outils techniques qui vont permettre sa réalisation.

Utiliser un ERP ?

Pour répondre de manière exhaustive aux besoins actuels et futurs de la société, il pourrait être envisagé de mettre en place un progiciel de gestion intégrée (ERP) : néanmoins, cette option représenterait un coût et un délai de mise en place jugés trop importants pour la société de notre étude de cas.

Choix d'architecture technique : ASP.NET, MSDE et Web Matrix

Les utilisateurs étant situés dans des lieux distincts, équipés d'infrastructures techniques différentes, il est naturel de retenir une architecture de type intranet, reposant un serveur HTTP qui génère des pages dynamiques à partir d'informations contenues dans une base de données centralisée.

Ce type d'architecture nécessite de se déterminer sur les points suivants :

- choix de la technologie de développement Web ;
- choix de la base de données ;
- choix de l'environnement de développement.

Choix de la technologie de développement Web

La technologie de développement constitue le moteur d'une application Web : c'est elle qui détermine le mécanisme de génération dynamique des pages Web ainsi les systèmes d'exploitation, bases de données, langages et outils de développement utilisables.

À l'heure actuelle, les choix possibles sont les suivants (par ordre alphabétique) :

Technologie	Description succincte
ASP	Technologie de génération de sites Web dynamiques introduite par Microsoft en 1996, qui repose sur l'interprétation de scripts et l'activation d'objets ActiveX/COM côté serveur ; fonctionne uniquement sous Windows ; généralement utilisée avec Microsoft SQL Server.
ASP.NET	Nouvelle technologie de développement d'applications et de services Web, introduite par Microsoft en 2001, qui repose sur des pages compilées, des contrôles prédéfinis, un mécanisme de gestion événementielle et une infrastructure technique intégrant des fonctionnalités de sécurité, optimisation et gestion des erreurs ; fonctionne uniquement sous Windows 2000 et XP équipé de l'extension .NET Framework ; généralement utilisée avec Microsoft SQL Server.
ColdFusion	Nom désignant à la fois une technologie de type serveur d'application et un outil de développement, qui repose sur l'utilisation d'un langage spécifique (CFML : Coldfusion Markup Language) permettant d'activer des objets sur le serveur lors de l'interprétation d'une page Web ; fonctionne sous Unix, Linux, Windows, avec la majorité des serveurs HTTP et des bases de données ; historiquement développé par Allaire et récemment acquis par Macromedia (ColdFusion MX).
JSP	Technologie de génération de sites Web dynamiques développée par Sun, qui repose sur l'utilisation de pages compilées contenant du Java et l'activation d'objets serveurs (JavaBeans) ; fonctionne sur toutes les plates-formes équipées d'une machine virtuelle Java.
PHP	Langage de script open source créé en 1994, actuellement dans sa version 4, dont la syntaxe est inspirée de C / Perl et Java, permettant la génération de sites Web dynamiques ; fonctionne sur la majorité des plates-formes Unix, Linux et Windows et est compatible avec un grand nombre de bases de données ; couramment utilisé au sein du trio Linux/Apache/MySQL.

Aujourd'hui, toutes ces technologies ont atteint un niveau de maturité assez avancé et, sur le papier, toutes permettraient de répondre aux besoins exprimés dans notre cahier des charges.

Néanmoins, ASP.NET se distingue de ses concurrents par son architecture novatrice et son nouveau modèle de programmation qui permet la réalisation d'applications Web avec les mêmes méthodes de conception (découpage en objets métier et composants réutilisables), le même degré de robustesse et de puissance (orientation objet, typage fort des variables, compilation) et le même degré d'organisation du code (séparation entre code et contenu graphique, organisation en gestionnaires d'événements) qu'une application client-serveur classique. Pour en savoir plus, nous allons détailler ces différentes caractéristiques de la technologie, en commençant par répondre à une question simple : qu'est-ce qu'ASP.NET ?

Qu'est-ce qu'ASP.NET ?

En pratique, ASP.NET est une extension logicielle qui permet l'exécution d'applications Web (sites Web dynamiques, services Web). Disponible en téléchargement gratuit sur le site www.asp.net et destinée à être installée sur une machine équipée de Windows 2000/XP et d'un serveur HTTP, elle se compose des éléments techniques suivants :

- une extension du serveur HTTP, implémentée sous la forme de filtre ISAPI nommé `aspnet_filter.dll` (voir figure 1-11), qui permet de traiter spécifiquement les requêtes vers les pages comportant une extension particulière (notamment les fichiers `.aspx`, qui correspondent aux pages ASP.NET standards, voir figure 1-12) ;
- un processus principal nommé `aspnet_wp.exe` (pour : ASP.NET worker process) qui tourne en tâche de fond et traite les requêtes correspondant à des pages ASP.NET (figure 1-13), en s'appuyant lui-même sur trois éléments :

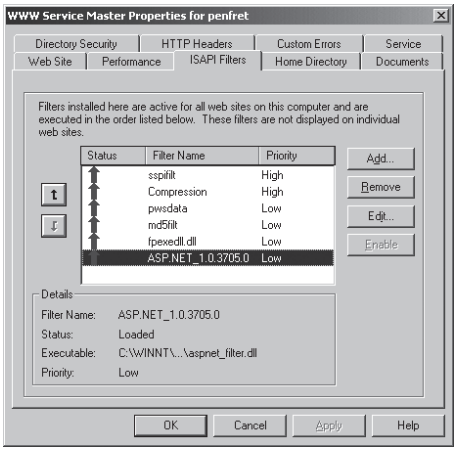


Figure 1–11
Le filtre ISAPI `aspnet_filter.dll`

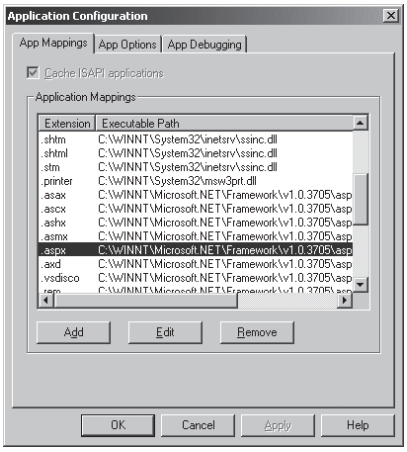


Figure 1–12 Configuration des extensions pour ASP.NET

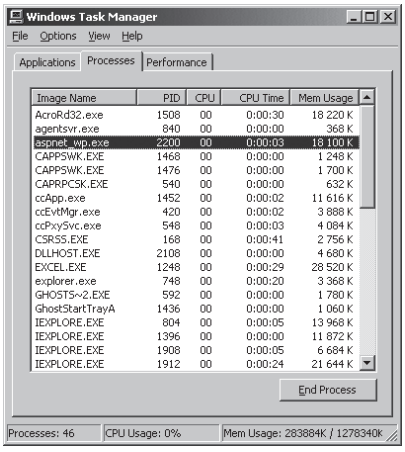


Figure 1–13
Le processus `aspnet_wp.exe`

ASP.NET est indissociable du framework .NET

De par son architecture, ASP.NET est indissociable du *framework .NET*, l'environnement d'exécution des applications .NET constitué du Common Language Runtime (CLR), de la bibliothèque de classes .NET et des compilateurs .NET

Anecdote

Le nom initial de la technologie ASP.NET était ASP+.

Extension .aspx

Les pages ASP.NET portent l'extension `.aspx` (pour : ASP eXtended).

La compilation n'a lieu que lors de la première requête

La compilation d'une page ASP.NET n'a lieu que lors du la première requête effectuée vers cette page ; l'assemblage produit est alors gardé en cache pour les requêtes successives (à moins que le fichier source ne soit modifié). Ceci garantit une performance optimale lors de l'exécution de l'application.

- les compilateurs .NET, qui permettent de compiler le code source des pages ASP.NET vers un format objet intermédiaire dit *Intermediate Language* (ou IL) ;
- une bibliothèque de classes utilitaires qui offre des fonctionnalités allant de l'accès aux données à la génération de documents XML, en passant par la réalisation de services Web ou de génération dynamique d'images, pour ne citer qu'une très brève étendue de ses possibilités ;
- une machine virtuelle nommée *Common Language Runtime* (CLR) prenant en charge l'exécution du code de la page (compilé en IL) et l'édition de liens avec les classes de la bibliothèque auxquelles le code de la page fait référence.

La figure 1-14 illustre le mécanisme d'appel d'une page ASP.NET :

- Lorsque le serveur HTTP reçoit une requête à destination d'une page portant l'extension `.aspx`, celle-ci est automatiquement traitée par le filtre ISAPI (`aspnet_filter.dll`) et transmise au processus principal ASP.NET (`aspnet_wp.exe`).
- La page demandée est alors compilée sous la forme d'un module objet en langage MSIL (Microsoft Intermediate Language) par le compilateur correspondant au langage utilisé dans la page (C#, VB.NET et JScript.NET sont les trois langages utilisables en standard) puis transmises à Common Language Runtime (CLR) qui réalise l'édition de lien avec les modules objet des classes utilisées de la bibliothèque .NET et la production d'un assemblage (composant binaire exécutable) dont l'exécution conduit à la production d'un contenu HTML, transmis en retour au navigateur.

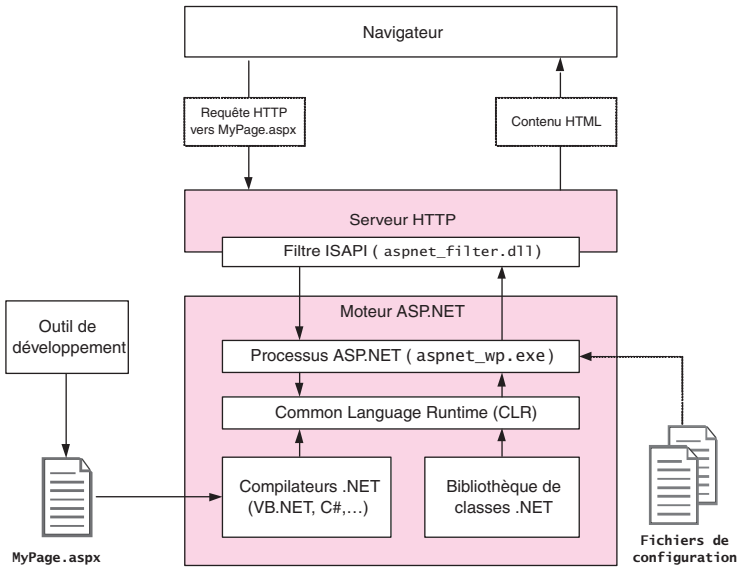


Figure 1–14 Traitement d'une requête vers une page ASP.NET

Notons que la configuration du processus principal ASP.NET (langue, sécurité, gestion des erreurs) est entièrement paramétrable à l'aide de fichiers texte au format XML (`machine.config` et `web.config`), dont les changements sont détectés et pris en compte dynamiquement, sans nécessiter de redémarrage du serveur HTTP.

Nous avons décrit succinctement l'infrastructure d'exécution des pages ASP.NET (nous aurons l'occasion de présenter plus précisément le mécanisme de compilation et la notion d'assemblage dans le chapitre 3 et les possibilités de configuration dans le chapitre 9) : détaillons maintenant les nouvelles possibilités offertes au développeur pour la réalisation de pages Web.

Parallèle avec Java

Le langage MSIL est en quelque sorte équivalent au *byte code* Java, tandis que le Common Language Runtime est proche conceptuellement de la machine virtuelle Java. De plus, le mécanisme de compilation des pages à la volée est proche de celui implémenté par l'architecture JSP.

Les nouveautés apportées par ASP.NET

Contrairement à une page ASP classique qui contient un entremêlement de contenu HTML et de script, les pages ASP.NET sont séparées en deux parties bien distinctes :

- la partie graphique constituée de balises et texte HTML (qui sera restitué tel quel), de contrôles serveur (éléments graphiques paramétrables qui seront, à l'exécution de la page, remplacés par un contenu HTML) et de contrôles utilisateur (fragments de pages HTML encapsulés au sein d'objets graphiques indépendants) ;
- la partie code qui spécifie le paramètrage des contrôles serveur et implémente la cinématique de la page en général, organisée au sein de gestionnaires d'événements et codée dans un des langages objet compatibles .NET : C# (proche de C++ et Java), VB.NET (proche de Visual Basic) ou JScript.NET (proche de JScript)

Formulaires Web (Web Forms)

Par analogie avec les formulaires Visual Basic, eux-mêmes constitués d'un assemblage de contrôle graphique dont on implémente la cinématique dans une partie code associée, les pages ASP.NET sont également appelées Formulaires Web (*Web Forms*).

Langages utilisables dans une page ASP.NET

À l'heure actuelle, le développeur a le choix entre trois langages par défaut pour développer une page ASP.NET :

- **VB.NET** : évolution majeure de Visual Basic auquel ont été principalement ajouté des fonctionnalités objet (constructeurs et destructeurs, surcharge de méthodes, héritage, etc.) ;
- **C#** : langage objet dont la syntaxe et les fonctionnalités sont proches de C++ et Java ;
- **JScript.NET** : évolution de JScript prenant en charge, entre autres, le typage des variables et les fonctionnalités objet (classe, héritage, etc.).

Ainsi, le développeur peut choisir le langage le moins éloigné de son champ de connaissance actuel et concentrer son attention sur les nouveautés de l'architecture ASP.NET plutôt que sur l'apprentissage d'un nouveau langage.

Autres langages

En théorie, une application .NET peut être développée avec tout langage conforme à la Common Language Specification (CLS) émise par Microsoft. Bien que le support d'une trentaine de langages ait été annoncé (dont Python, Perl, Eiffel, etc.), la mise en pratique peut s'avérer fastidieuse : nécessité de télécharger et d'installer un compilateur spécifique, absence d'exemples de code dans la documentation... C'est pourquoi les applications ASP.NET sont généralement implémentées avec VB.NET ou C#.

Les exemples de ce livre seront présentés dans les deux langages VB.NET et C#.

Code disponible en ligne

Dans certains cas, une version unique (VB.NET ou C#) sera proposée, dans un souci d'économie de place ; la version alternative sera alors consultable sur le site de l'étude de cas :

▸ www.savonsdusoleil.com

Aide-mémoire

Un aide-mémoire résumant les principales syntaxes des langages VB.NET et C# est fourni en annexe.

Cette architecture présente un certain nombre d'avantages :

- la séparation entre partie graphique et code augmente la lisibilité du code et facilite par conséquent sa maintenance ; d'autre part, elle permet, le cas échéant, de répartir plus facilement le travail entre des graphistes qui travaillent sur le contenu HTML et les développeurs qui se consacrent sur la logique applicative ;
- l'utilisation de langages orientés objet rend les développements plus robustes (typage fort, gestion des exceptions) et plus simples à réutiliser (héritage, polymorphisme) ;
- l'organisation du code en gestionnaires d'événements (« exécuter tel code lorsque tel événement survient »), rendu possible par un mécanisme capable de réaliser des allers-retours (*round-trip*) entre le navigateur et le serveur HTTP en conservant les valeurs contenues dans les contrôles, facilite l'implémentation de pages interactives (par exemple : mise à jour des produits disponibles lorsque l'utilisateur sélectionne une famille de produits).

Nous développerons tous ces sujets dans le chapitre 3, où nous étudierons l'architecture d'une page ASP.NET (séparation entre contenu graphique et code, technique du *code behind*, notion de contrôle serveur) ; la notion de gestionnaire d'événement sera, quant à elle, décrite en détail à la fin du chapitre 4.

Par ailleurs, les pages ASP.NET bénéficient de la richesse de la bibliothèque .NET qui comporte plusieurs milliers de classes utilitaires permettant entre autres :

- la consultation et la mise à jour de données contenues dans une base, notamment à l'aide de contrôles serveur liés aux données comme `DataGrid`, `DataList` et `Repeater` (voir chapitres 4 et 5) ;
- la création et la manipulation de documents XML ainsi que la réalisation de transformations XSL (voir chapitre 6) ;
- l'envoi de fichiers par messagerie (voir chapitre 6) ;
- la génération dynamique d'images (voir chapitre 7) ;
- l'implémentation et l'appel de services Web (voir chapitre 8).

Compatibilité ascendante avec ASP

Toute application ASP peut s'exécuter dans un environnement ASP.NET.

Choix de la base de données

Dans une application ASP.NET comme dans la grande majorité des sites Web dynamiques, la base de données est une composante fondamentale de l'architecture. L'offre logicielle en la matière est vaste ; parmi les produits les plus connus disponibles sous Windows, citons : Microsoft SQL Server, Oracle, PostgreSQL, Informix, 4D, Microsoft Access…

À l'heure actuelle, la bibliothèque .NET propose les types d'accès suivants :

- accès natif à Microsoft SQL Server (et à sa version allégée : MSDE) ;
- accès natif à Oracle (fournisseur disponible en téléchargement séparé) ;
- pilote OLE-DB ;
- pilote ODBC (fournisseur disponible en téléchargement séparé).

Pour notre étude de cas, nous nous proposons d'utiliser la base Microsoft SQL Server Desktop Engine (MSDE) en raison de sa gratuité.

Pourquoi pas Microsoft Access ?

Il est vrai que Microsoft Access a toujours constitué une alternative facile à utiliser et peu coûteuse à SQL Server. Néanmoins, le fait que le moteur Jet ne soit pas géré nativement par .NET, conjugué à la gratuité de MSDE plaide en défaveur du choix de cet outil pour notre étude de cas.

MSDE : une version allégée et gratuite de SQL Server 2000

Microsoft SQL Server Desktop Engine (MSDE) est une version allégée et gratuite de Microsoft SQL Server, doté d'un support complet de Transact SQL (y compris la gestion des transactions, des déclencheurs et de la sécurité). En revanche, elle n'offre qu'un support partiel de la réplication, ne dispose pas d'outils graphiques d'administration (comme SQL Enterprise Manager ou Analyseur de requête) et son usage reste limité à un nombre réduit d'utilisateurs simultanés.

Cette base constitue néanmoins une solution bien adaptée pour un poste de développement, du fait de sa faible consommation de ressources, d'une part, mais également de sa compatibilité totale avec SQL Server 2000, qui facilite un éventuel déploiement vers un serveur de production doté de ce SGBD. L'installation de MSDE sera décrite dans le chapitre suivant.

À propos de la similarité des moteurs SQL

La version précédente du moteur MSDE (1.0) était différente de SQL Server 7.0 ; désormais, MSDE 2000 et SQL Server 2000 sont dotés du même moteur.

Si vous disposez de SQL Server

Bien entendu, les lecteurs qui le souhaitent pourront utiliser SQL Server au lieu de MSDE : les exemples donnés dans cet ouvrage seront parfaitement applicables à l'un comme à l'autre.

Choix de l'environnement de développement

Dans l'absolu, il est possible de développer une application ASP.NET avec un simple éditeur de texte : fastidieuse, cette approche a néanmoins l'avantage indéniable de permettre au développeur de maîtriser tout le code qu'il écrit, et certains en sont partisans !

À l'opposé, il existe des environnements de développement qui se proposent d'automatiser les tâches laborieuses à l'aide d'assistants, de faciliter le confort de travail du développeur (coloration syntaxique, complétion automatique du code saisi, compilateurs intégrés, aide en ligne), citons :

- **Visual Studio.NET** : très complet, il gère de nombreux types d'applications (ASP.NET, WinForms, MFC, ATL), plusieurs langages (C, C++, VB.NET, C#), est doté de fonctionnalités puissantes (grand nombre d'assistants, aide contextuelle, technologie Intellisense pour compléter le code) et est adapté au travail en groupe (gestion de projets complexes, intégration avec des outils de contrôles de source) ;
- **Web Matrix** : environnement léger dédié au développement d'applications ASP.NET ; moins riche en fonctionnalités que Visual Studio.NET (pas d'aide contextuelle, pas de complétion automatique du code, pas de notion de projet), il n'en est pas moins doté d'un grand nombre d'assistants (modèles de pages, ajout de contrôles, gestion de la base de données) et présente l'indéniable avantage d'être gratuit ;
- **Delphi 7 Studio** : la nouvelle version du célèbre environnement de développement édité par Borland fournit désormais un support pour le développement d'application ASP.NET avec le langage Delphi.

Dans le cadre d'une grosse équipe de développement travaillant sur des projets importants, l'emploi de Visual Studio.NET ou de Delphi serait probablement nécessaire ; pour notre étude de cas, limitée à une application ASP.NET, réalisée a priori par un développeur seul, Web Matrix répond amplement aux besoins.

Utiliser quand même VS.NET ou Delphi

Bien entendu, les lecteurs qui le souhaitent pourront tout de même tirer parti de cet ouvrage en utilisant Visual Studio.NET ou Delphi au lieu de Web Matrix : en effet, l'accent sera plus porté sur la technologie ASP.NET proprement dite que sur les outils ou langages utilisés pour la mettre en œuvre.

Web Matrix : version 0.5 (en anglais)

À l'heure actuelle, Web Matrix n'est disponible qu'en version 0.5 (Technology Preview), en anglais. Si quelques fonctionnalités font encore défaut, cette version est globalement satisfaisante en terme de stabilité (et les quelques commandes de menu en anglais ne devraient pas dérouter le lecteur). Notons que dans sa version finale, le produit sera toujours gratuit.

- www.asp.net/webmatrix

Code in-line vs code behind

Une des différences fondamentales entre Web Matrix et Visual Studio.NET réside dans la gestion de la séparation entre code et contenu HTML : Web Matrix place le code et le contenu HTML d'une page dans le même fichier, tout en offrant une séparation visuelle entre les deux à l'aide d'onglets (*code in-line*), tandis que Visual Studio.NET place le code associé à une page dans un fichier séparé du contenu HTML (*code behind*). Nous aurons l'occasion de détailler ces deux techniques dans le chapitre 3.

Web Matrix : un environnement de développement efficace et gratuit

Web Matrix est un environnement léger dédié au développement ASP.NET. Disponible en téléchargement gratuit sur www.asp.net/webmatrix (son installation sera détaillée dans le chapitre suivant), il démontre la volonté de Microsoft de fédérer une communauté de développeurs autour d'ASP.NET : outil gratuit, nombreux exemples de sources, accès à des forums depuis l'interface.

Le tableau ci-dessous résume les principales différences entre Visual Studio.Net et Web Matrix :

	Visual Studio.NET	Web Matrix
Assistants graphiques	X	X
Gestion base de données	X	X
Complétion du code (Intellisense)	X	
Client FTP intégré		X
Serveur HTTP intégré		X
Gestion de projets	X	
Contenu du code source	X	
Compilation	Intégrée	Séparée (ligne de commande)
Séparation HTML/code	Fichiers séparés (code behind)	Même fichier (code in-line)
Version actuelle	7.0	0.5
Coût licence	Variable selon le type de licence	Gratuit

Bien qu'il soit moins riche en fonctionnalités que Visual Studio.NET ou Delphi, Web Matrix s'avère tout à fait adapté au développement ASP.NET : les nombreux assistants et modèles, le client FTP intégré et le serveur HTTP personnel font de cet outil gratuit un compagnon fidèle et efficace.

En résumé

Dans ce premier chapitre, nous avons posé les bases de notre étude de cas : après avoir présenté les activités des « Savons du Soleil » et les problématiques auxquelles est confrontée cette société, nous avons décrit en détail le projet d'évolution du système d'information.

Puis, nous avons proposé une vue d'ensemble de la technologie ASP.NET et justifié le choix de la base de données MSDE et de l'environnement de développement Web Matrix pour la réalisation de notre étude de cas.

Dans le chapitre suivant, nous allons décrire l'installation de ces outils sur le poste de travail.