Capture des besoins techniques

Objectifs du chapitre

Ce chapitre traite du rôle d'UML lors de l'étape de capture des besoins techniques. Cette activité est généralement peu formalisée, soit par manque d'une notation et d'un processus approprié, soit parce que les architectes techniques recourent rarement à une approche formelle. Nous allons précisément étudier comment le concept de cas d'utilisation peut être étendu pour répondre à ce besoin, et de quelle manière le processus en Y répond particulièrement bien à la spécification technique d'un système client/serveur tel que SIVEx.

Dans ce chapitre, nous allons aborder :

- la construction d'un modèle d'analyse technique avec UML,
- les avantages d'une organisation en couches logicielles,
- l'emploi des cas d'utilisation pour décrire les comportements techniques du système,
- la description des cas d'utilisation techniques.

Quand intervient la capture des besoins techniques ?

La capture des besoins techniques couvre, par complémentarité avec celle des besoins fonctionnels, toutes les contraintes qui ne traitent ni de la description du métier des utilisateurs, ni de la description applicative. Le modèle de spécification logicielle concerne donc les contraintes techniques telles que nous avons pu les évoquer au chapitre 4. La spécification technique est une activité de la branche droite du Y ; elle est primordiale pour la conception d'architecture.

Cette étape a lieu lorsque les architectes ont obtenu suffisamment d'informations sur les prérequis techniques. Ils doivent a priori connaître au moins le matériel, à savoir les machines et réseaux, les progiciels à intégrer, et les outils retenus pour le développement. En cours d'élaboration, viendront s'ajouter les contraintes non fonctionnelles identifiées dans les cas d'utilisation de la branche gauche. Le niveau d'abstraction à atteindre est l'analyse technique. Le modèle s'y exprime suivant les deux points de vue que sont la spécification logicielle et la structure du matériel à exploiter. Cette étape se termine lorsque le niveau de description des cas d'utilisation techniques a permis l'identification des problèmes à résoudre. À ce moment-là pourra débuter l'étape de conception générique, qui consiste à construire une solution d'architecture technique.

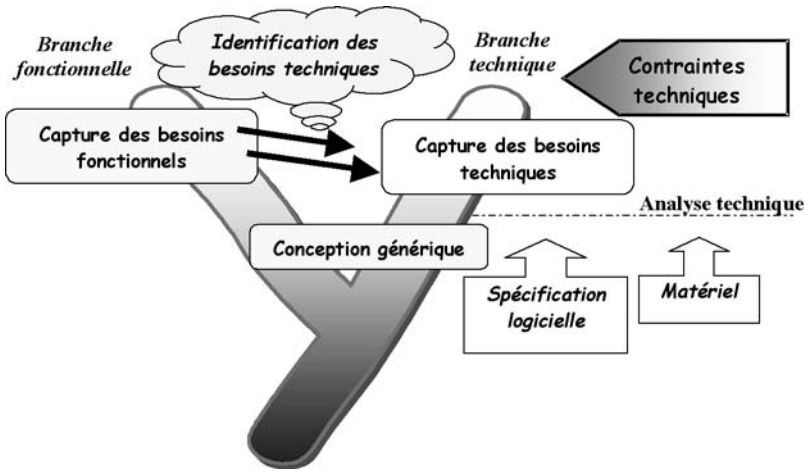


Figure 5-1 : Situation de la capture des besoins techniques dans 2TUP

Éléments mis en jeu

- Diagramme de déploiement, nœuds et connexions du réseau, architecture à 3 niveaux,
- Diagramme de composants, composants d'exploitation, architecture 3-tiers,
- Diagramme de cas d'utilisation, cas d'utilisation technique, description d'un cas d'utilisation technique, organisation en couches logicielles, architecture en 5 couches,
- Spécification logicielle détaillée.

Spécification technique du point de vue matériel

Les prérequis techniques ont été exprimés dans l'étude préliminaire, lors de l'expression des besoins opérationnels et de celle des choix stratégiques de développement au chapitre 3. Ces choix impliquent des contraintes relatives à la configuration du réseau matériel. Elles sont de nature géographique, organisationnelle, et technique. Elles concernent les performances d'accès aux données, la sécurité du système, l'interopérabilité, l'intégration des applications, la volumétrie et le mode d'utilisation du système.

Définition

STYLE D'ARCHITECTURE EN NIVEAUX

Le style d'architecture en niveaux spécifie le nombre de niveaux géographiques et organisationnels où vont se situer les environnements d'exécution du système [Orfali 94].

- L'architecture à deux niveaux met en œuvre un environnement de travail de niveau départemental et local. Un tel système répond généralement à la demande d'un métier particulier dans l'entreprise. Par exemple, le département des ressources humaines dispose d'un système informatique indépendant et localisé au sein de la société.
- L'architecture à trois niveaux met en œuvre le système informatique d'une entreprise. Nous y trouvons les niveaux suivants : central, départemental et local. Une telle architecture couvre les différents métiers de l'entreprise. L'étude de cas SIVEx en constitue un exemple.
- La contrainte géographique conditionne également l'architecture en niveaux. Le système de ressources humaines, réparti sur plusieurs agences ou départements, devient de fait un système à trois niveaux. La conjonction des contraintes géographique et organisationnelle conduit donc à des systèmes complexes dotés d'une architecture multiniveaux.

Les contraintes techniques amènent également à diversifier le nombre et le type des machines :

- soit pour des raisons de performances : c'est le cas d'un montage en cluster,
- soit pour des raisons de sécurité : c'est le cas de l'ajout de ponts, de routeurs ou d'un serveur dédié au Web (typiquement dans la mise en œuvre d'une zone démilitarisée DMZ),
- soit pour des raisons d'interopérabilité : lorsque les logiciels sont conçus sur des plates-formes différentes,
- soit pour des raisons de disponibilité : on privilégiera les postes de travail utilisés 24h/24, aux terminaux utilisés pour des requêtes très ponctuelles.


Conseil

STRUCTUREZ VOS SPÉCIFICATIONS D'EXPLOITATION TECHNIQUE AUTOUR DU MODÈLE DE CONFIGURATION MATÉRIELLE

Les spécifications qui concernent l'exploitation technique d'un réseau ont toutes une relation directe soit avec une connexion, soit avec une machine particulière du modèle de configuration matérielle. Du fait de leur existence, les machines imposent des contraintes de performances ou d'intégration matérielle. La nature des connexions permet également de spécifier des contraintes liées au besoin de communication et de bande passante. L'intégration de l'application dans le système d'information existant impose de nouvelles contraintes liées aux machines dédiées à une fonction particulière du système informatique.

Nous vous conseillons donc d'intégrer vos spécifications dans le dictionnaire du modèle de configuration matérielle.

ÉTUDE DE CAS : LA CONFIGURATION MATÉRIELLE DE SIVEX

La configuration géographique du système SIVEx impose le développement d'une solution client/serveur à trois niveaux : un niveau central pour les informations partagées entre agences et consolidées pour le siège, un niveau départemental pour chaque agence et un niveau local pour les applications à déployer sur les postes de travail. La configuration matérielle est schématisée par un diagramme de déploiement UML à la figure 5-2.

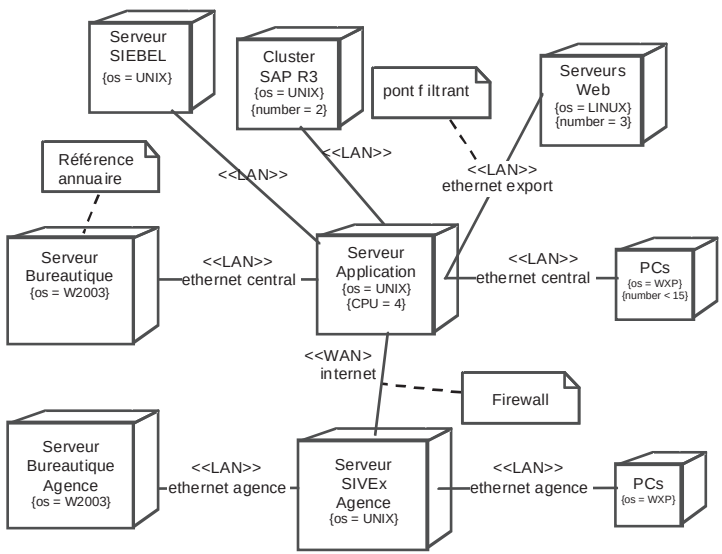


Figure 5-2. : Configuration matérielle du système SIVEx

Le point de vue matériel met en évidence les contraintes d'exploitation technique suivantes :

- pour fonctionner avec les progiciels SAP R3 et SIEBEL, SIVEx doit nécessairement intégrer une technologie d'EAI ;
- la communication WAN prise en charge par Internet est subordonnée à l'usage de *firewalls*, ce qui limite potentiellement le jeu de protocoles de communication possibles entre clients et serveurs ;
- l'export des informations sur Internet impose d'isoler le serveur Web du réseau *ethernet* central ;
- l'annuaire centralisé de référence doit correspondre à celui qui existe sur le serveur de bureautique central.

Spécification d'architecture et influence sur le modèle de déploiement

L'expression des prérequis techniques implique également le choix d'un style d'architecture client/serveur. Ce choix conditionne la façon dont seront organisés et déployés les composants d'exploitation du système.

COMPOSANT D'EXPLOITATION

Un composant d'exploitation est une partie du système logiciel qui doit être connue, installée, déclarée et manipulée par les exploitants du système. Un composant d'exploitation doit être interchangeable entre différentes versions et peut être arrêté ou démarré séparément. Il assume des fonctions bien identifiées dans le système, de sorte qu'en cas de dysfonctionnement, le composant incriminé est facilement repérable.

STYLE D'ARCHITECTURE EN TIERS

Le style d'architecture en tiers (*tier* signifie « partie » en anglais) spécifie l'organisation des composants d'exploitation mis en œuvre pour réaliser le système. Chaque partie indique une responsabilité technique à laquelle souscrivent les différents composants d'exploitation d'un système.

On distingue donc plusieurs types de composants en fonction de la responsabilité technique qu'ils jouent dans le système. Un système client/serveur fait référence à au moins deux types de composants, qui sont les systèmes de base de données en serveur, et les applications qui en exploitent les données en client.

Le style d'architecture 2-tiers correspond à la configuration la plus simple d'un système client/serveur. Dans ce cas, il incombe aux clients de gérer l'interface utilisateur et les processus d'exploitation. Les serveurs ont pour

responsabilité de traiter le stockage des données. Ce type d'architecture est parfaitement bien adapté aux systèmes départementaux, dans la mesure où les concepts et les processus manipulés n'existent qu'une seule fois au sein d'un département de l'entreprise.

Dans le cadre des architectures d'entreprise, certains concepts et processus sont communs à plusieurs domaines d'activité. Cette caractéristique implique une synchronisation souvent complexe des données entre différents départements de l'entreprise. Le concept d'objet métier consiste à centraliser cette gestion afin d'en maîtriser la complexité. L'objet métier est à la fois un modèle d'analyse qui colle à la réalité du problème de l'entreprise, mais également un modèle de composant d'exploitation qui s'insère dans le déploiement du système d'entreprise [Eeles 98]. L'intégration des objets métier sous la forme de composants métier fait passer l'architecture client/serveur du 2-tiers au 3-tiers, car elle implique un nouveau type de composants d'exploitation qui s'insère entre les clients et les serveurs de données.


Définition

COMPOSANT MÉTIER

Un composant métier est un composant d'exploitation dont la fonction est de distribuer les services d'un ou de plusieurs objets métier de l'entreprise. L'intégration de composants métier implique le recours à un style d'architecture 3-tiers.

Le style d'architecture 3-tiers facilite la réutilisation au sein d'un système, puisque les composants métier correspondent à des concepts communs à différents métiers de l'entreprise. Dans le cas de SIVEx, les classes Commande et Mission sont de bons composants métiers candidats, puisqu'ils concourent tous deux au métier du répartiteur, du chauffeur, du réceptionniste, voire même du comptable.

Comme le style d'architecture 3-tiers définit un moyen logiciel intermédiaire entre les applications clientes et les serveurs de base de données, il fournit au système les moyens techniques qui lui permettent de garantir des temps de réponse constants, quel que soit le nombre d'utilisateurs connectés.

ÉTUDE DE CAS : SPÉCIFICATION DU STYLE D'ARCHITECTURE 3-TIERS

La spécification d'une architecture à composants métier 3-tiers implique des contraintes sur le modèle d'exploitation. Une solution client/serveur 3-tiers entraîne en effet la répartition des composants d'exploitation suivant les responsabilités :

- le stockage des données sera réparti entre plusieurs instances de bases de données en central ou en agence. On a par ailleurs retenu un moteur de base de données relationnel ;

- la distribution des services métier est réalisée sur plusieurs composants métier dont le déploiement est à préciser. La technologie Java-RMI est arrêtée pour sa réalisation ;
- la présentation et la gestion des applications correspondent à différents composants d'exploitation répartis en central ou en agence. Ces applications seront développées en Java et déployées sur les postes clients. Le serveur Web est une application particulière qui a pour rôle de rendre accessible la situation des commandes aux différents clients de SIVEx. Elle sera réalisée avec la technologie Java-applet ;
- l'intégration des données et des processus complète l'architecture afin de permettre l'utilisation des progiciels du commerce et de faciliter leur cohabitation avec des développements spécifiques. La réalisation de cette fonction technique s'appuie sur un outil EAI composé de serveurs de messages d'échanges (*Message hub* ou *broker*) et de connecteurs.

La définition d'un composant, au sens UML du terme, n'est ni une classe, ni une technologie, mais une partie de logiciel qui peut être interchangeable, au sens où l'emploie l'industrie. Dans notre cas, il s'agit de ne montrer que les composants d'exploitation, par opposition aux composants de type « librairie » ou « package java » qui ne sont pas visibles de l'exploitant.

On ne peut formaliser, à ce niveau d'étude, qu'une typologie de déploiement, où seuls les différents types de composants d'exploitation du système SIVEx sont apparents. Ce modèle précise les dépendances entre types de composants et définit les stéréotypes qui seront employés pour la suite du projet. Le recours à la propriété UML « number » permet de spécifier le nombre d'instances de composants d'exploitation qui peuvent exister pour chacun des types.

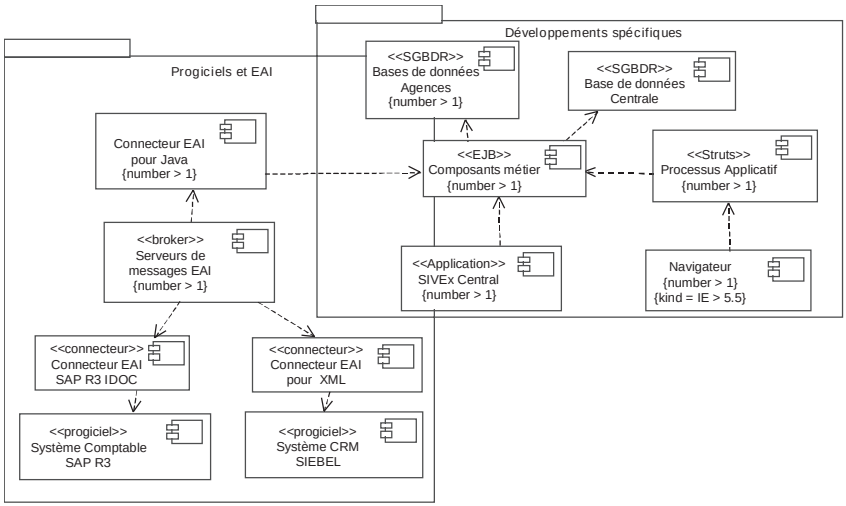
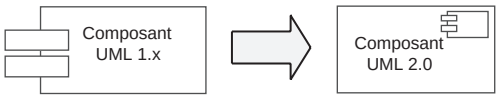


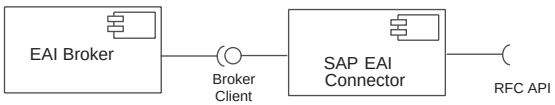
Figure 5-3 : Spécification d'organisation du modèle de déploiement SIVEx

ÉVOLUTION DU DIAGRAMME DE COMPOSANT EN UML 2

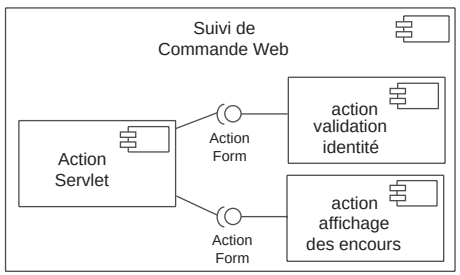
Suite au schéma de l'étude de cas précédente, il convient de faire différentes remarques quant à l'évolution de notation apparue avec UML 2.0. Il s'agit en premier lieu du changement de notation d'un composant.



Ensuite, UML 2.0 permet d'affiner la relation de dépendance entre composants en précisant les interfaces fournies et requises. Dans notre exemple, un connecteur EAI dispose d'une interface de scrutation de messages (*Broker Client*) en provenance du broker, tel qu'illustré dans la figure ci-dessous. Inversement un composant EAI Broker discute avec des composants présentant ce type d'interface. De même, le connecteur SAP fonctionne avec un composant présentant une interface RFC (technologie propre à SAP).



Enfin UML 2.0 permet d'exprimer la composition d'un composant. Dans notre exemple, un composant Struts (voir http://jakarta.apache.org/struts/index.html) est composé d'une ou plusieurs actions et d'une servlet qui centralise les requêtes de l'utilisateur, tel qu'illustré dans la figure ci-dessous.



Élaboration du modèle de spécification logicielle

Une fois que les spécifications techniques et d'architecture sont exprimées, on peut s'intéresser aux fonctionnalités propres du système technique en procédant à une spécification logicielle. Dans ce cadre, on propose d'utiliser les cas d'utilisation de manière différente que pour la spécification fonctionnelle. C'est pourquoi nous avons introduit le concept d'exploitant et de cas d'utilisation technique.

EXPLOITANT

L'exploitant est un acteur au sens d'UML, si ce n'est qu'il ne bénéficie que des fonctionnalités techniques du système.

Tout système informatique possède au minimum un exploitant qui est « l'utilisateur du système ». Il s'agit ici de l'utilisateur dans son sens le plus général, indépendamment des fonctions ou du métier qu'il réalise au travers de l'application. Dans ce cadre, tout utilisateur se connecte au système ou consulte l'aide en ligne. Ce sont les fonctionnalités purement techniques dont il bénéficie en tant qu'exploitant.

CAS D'UTILISATION TECHNIQUE

Un cas d'utilisation technique est destiné à l'exploitant. C'est une séquence d'actions produisant une valeur ajoutée opérationnelle ou purement technique.

Note : le concept de cas d'utilisation technique introduit dans ce livre est une proposition faite aux architectes logiciels et aux concepteurs de procéder à une phase d'analyse des comportements techniques du système. En effet, le comportement des logiciels développés obéit trop souvent au style des développeurs qui participent à sa construction, sans qu'une concertation préalable n'ait été établie sur : les styles d'architecture, les frameworks employés et le vocabulaire utilisé dans la documentation technique du produit. Cependant, la diffusion des standards techniques : J2EE, .Net, Apache/Struts, PHP, etc. tend à uniformiser les styles de conception et à amoindrir l'intérêt d'une capture des besoins techniques aussi formalisée que dans cet ouvrage.

Les cas d'utilisation techniques sont absolument distincts des cas d'utilisation de la branche gauche : ils ne produisent aucune valeur ajoutée fonctionnelle. La branche droite recouvre en effet tous les services techniques dont un utilisateur bénéficie, parfois même sans s'en rendre compte.

Un modèle de spécification logicielle est généralement construit en deux itérations. Le modèle initial consiste à recenser les besoins des différents exploi-

tants du système et à en extraire les cas d'utilisation techniques. Lors de la deuxième itération, le modèle de spécification est réorganisé en couches de responsabilités techniques de manière à affiner les exigences. Cette dernière technique sera approfondie ultérieurement dans le chapitre.

ÉTUDE DE CAS : IDENTIFICATION DES CAS D'UTILISATION TECHNIQUES

Les exploitants du système SIVEX sont :

- l'utilisateur, qui utilise une des applications du système SIVEx. La majorité des acteurs de la branche fonctionnelle sont donc des utilisateurs dans la dimension technique,
- l'ingénieur d'exploitation, qui est chargé de déployer et de dépanner le système.

Les cas d'utilisation techniques de SIVEx sont d'abord identifiés en considérant l'attente opérationnelle de chaque exploitant :

- l'utilisateur va travailler avec des entités sous la forme d'objets, ce qui implique la mise en œuvre des mécanismes de persistance et de gestion de cycle de vie des objets ;
- certains utilisateurs vont bénéficier d'applications du commerce afin d'accélérer le déploiement de SIVEx. Les objets qu'ils utilisent implicitement au travers de ces progiciels doivent être synchronisés avec ceux qui sont enregistrés dans les bases de données SIVEx. En conséquence, un mécanisme d'EAI doit être mis en œuvre pour permettre aux différents utilisateurs de partager les mêmes données quelle que soit l'application qu'ils utilisent ;
- plusieurs utilisateurs peuvent travailler en parallèle. L'intégrité est le mécanisme qui em-pêche la mise à jour simultanée d'une même entité par deux utilisateurs différents ;
- chaque utilisateur bénéficie également d'une gestion des charges au niveau du serveur. Ainsi, les temps de réponse du système ne s'en trouvent pas dégradés en fonction du nombre d'utilisateurs connectés ;
- l'utilisateur doit se connecter et être reconnu du système pour pouvoir y travailler. L'authentification est le mécanisme qui protège le système des intrusions externes ;
- chaque utilisateur doit disposer d'une aide contextuelle qui l'aide à exploiter le système de la manière la plus efficace ;
- le système doit être exploitable ; à ce titre, il faut qu'il soit en mesure de générer des traces et des alertes qui vont faciliter sa maintenance au sein du système informatique global de l'entreprise. C'est cette analyse technique du problème qui permet d'introduire l'ingénieur d'exploitation comme autre exploitant du système ;
- l'ingénieur d'exploitation ainsi que l'utilisateur sont soumis à des règles de sécurité. Dans un système client/serveur ces aspects recouvrent l'authentification, l'habilitation, le cryptage, la non-répudiation et l'audit.

L'ensemble des cas d'utilisation cités ici ne sont pas spécifiques à SIVEx. Leur position en branche droite en fait des problèmes récurrents aux systèmes client/serveur. Ces contraintes d'utilisation techniques donnent lieu à un premier modèle de spécification logicielle représenté par un diagramme de cas d'utilisation.

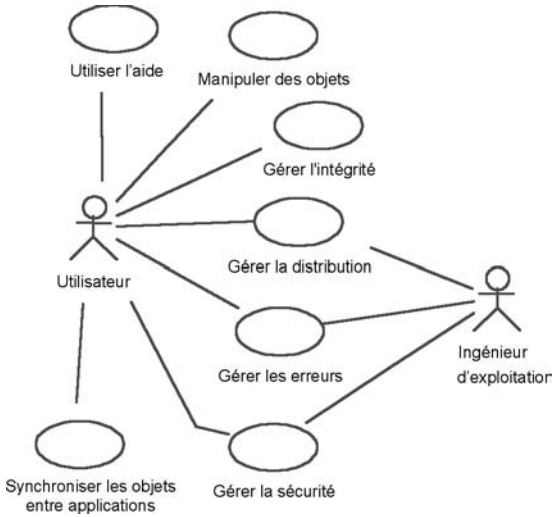


Figure 5-4 : Modèle de spécification logicielle de SIVEx

Dans le modèle initial de spécification logicielle, chaque cas d'utilisation est sommairement détaillé suivant la même technique que celle décrite au chapitre 4 comme l'illustre l'exemple 5-1. Les informations du modèle de spécification fonctionnelle servent à justifier les besoins qui vont être exprimés dans le cas d'utilisation technique.

Cas d'utilisation : manipuler des objets

Intention	l'utilisateur désire agir sur le cycle de vie d'un ou plusieurs objets.
Actions	créer, modifier, supprimer un objet ou un graphe d'objets.
Identification du besoin	gestion des commandes.
Exemple	le réceptionniste gère le cycle de vie d'une commande qu'il crée, modifie ou supprime. Il manipule également en une seule fois l'ensemble des colis décrits par la commande.

Tableau 5-1 : Définition initiale d'un cas d'utilisation technique

Organisation du modèle de spécification logicielle

À l'usage, le modèle de spécification logicielle obtenu est trop sommaire pour initier une spécification technique suffisamment détaillée. Tous les cas d'utilisation tels que « manipuler des objets » concernent différentes responsabilités de traitement qui vont de l'interface utilisateur à la base de données. Dans ce contexte, il est difficile de pouvoir préciser de manière détaillée les comporte-

ments techniques attendus, si l'on n'organise pas la spécification suivant les différentes responsabilités de traitement.

COUCHE LOGICIELLE

Une couche logicielle représente un ensemble de spécifications ou de réalisations qui respectivement expriment ou mettent en œuvre un ensemble de responsabilités techniques et homogènes pour un système logiciel.

Les couches s'empilent en niveaux pour couvrir des transformations logicielles successives, de sorte que la couche d'un niveau ne puisse utiliser que les services des couches des niveaux inférieurs.

Le recours aux couches logicielles va nous permettre d'affiner la spécification technique en divisant le problème en sous-parties spécialisées. Notre point de départ consiste à considérer le rôle et la description des cinq couches logicielles illustrées par la figure 5-5. Cette organisation correspond au style d'architecture en couches préconisé pour le développement d'une solution client/serveur [Rumbaugh 91].

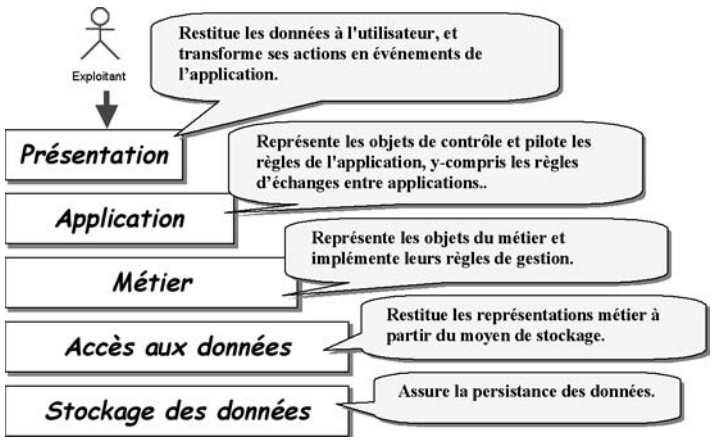


Figure 5-5. : Style d'architecture en 5 couches

Dans le modèle UML, les couches logicielles correspondent à des packages. Pour préciser leur spécificité, nous avons introduit le stéréotype « layer ». Ces packages contiennent des cas d'utilisation techniques qui ne sont plus forcément pilotés par un des exploitants du système. À chaque fonction observable pour l'exploitant, correspond en effet une cascade de responsabilités techniques qui se déploient sur les différentes couches logicielles. Chaque couche produisant des services pour les niveaux supérieurs contient en conséquence des cas d'utilisation pilotés par les couches exploitantes.

COMPLÉTEZ LES COUCHES PAR DES PARTITIONS

Le style d'architecture en 5 couches est une recommandation qui doit être appliquée au contexte du système en développement. À un même niveau, il est cependant possible de répartir les responsabilités techniques en partitions. Une partition correspond à plusieurs packages indépendants, mais au même niveau de responsabilités.

Une partition apparaît lorsqu'une même couche est concernée par différentes technologies ou lorsque le système communique avec d'autres systèmes par des mécanismes spécifiques.

ÉTUDE DE CAS : ORGANISATION EN COUCHES DU MODÈLE DE SPÉCIFICATION

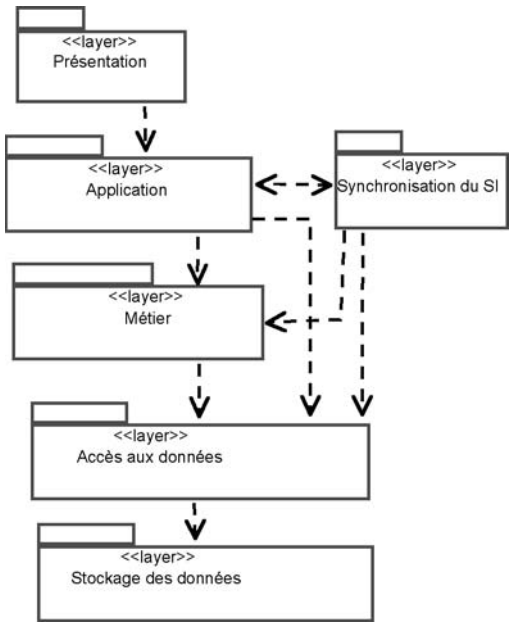


Figure 5-6. : Organisation du modèle de spécification logicielle (diagramme de packages)

Le style d'architecture en 5 couches structure le modèle de spécification logicielle. Les couches s'organisent suivant les dépendances qui s'établissent entre elles. Pour gérer la problématique des interfaces avec les systèmes ERP, CRM et plus largement avec les autres composantes du système d'information, l'architecte technique a ajouté une partition au niveau de l'application qui permet de synchroniser le système SIVEx avec le reste du système d'informations de l'entreprise.

Développement des couches logicielles

Dans le cadre de la responsabilité affectée à chaque couche, une identification poussée des cas d'utilisation techniques permet de poser de manière plus précise des problèmes à traiter. D'une part, les cas d'utilisation techniques peuvent se spécialiser suivant les couches sur lesquelles ils vont s'exécuter ; d'autre part, de nouveaux cas d'utilisation peuvent apparaître pour répondre à la particularité d'une des couches.

Le cas de SIVEx offre une illustration de ces différents cas : « Manipuler des objets » va donner lieu à plusieurs cas d'utilisation qui vont s'enchaîner depuis la couche de présentation jusqu'à la couche d'accès aux données. Par ailleurs, pour manipuler des objets, il est nécessaire de gérer des transactions avec la base de données relationnelle. Il s'agit donc d'un nouveau cas d'utilisation spécifique pour la couche de stockage des données.

On utilise des relations de dépendances entre cas d'utilisation pour formaliser les échanges des couches clientes aux couches fournisseuses. Parce qu'elles traduisent une délégation d'un cas d'utilisation vers l'autre, nous avons introduit le stéréotype « delegate ».

ÉTUDE DE CAS : DÉVELOPPEMENT DE LA COUCHE « ACCÈS AUX DONNÉES »

Les cas d'utilisation d'une couche logicielle s'identifient par rapport aux services attendus par les couches exploitantes. Dans ce cadre, chaque dépendance « delegate » représente une relation de client à fournisseur entre couches.

Pour obtenir une spécification technique détaillée du système, l'architecte technique a choisi de recourir aux délégations suivantes :

- rechercher un objet au niveau de la présentation nécessite de s'appuyer directement sur l'exploitation du schéma de données d'une classe au niveau de la couche d'accès aux données ;
- exécuter un service au niveau de la couche métier repose sur l'exploitation de requêtes spécifiques au niveau de la couche d'accès aux données, qui utilise systématiquement la gestion des transactions.

Notez bien qu'il s'agit ici d'un découpage qui permet à l'architecte de spécifier ses besoins techniques avant de les concevoir. Ce n'est donc en rien l'amorce d'une conception qui serait ici purement fonctionnelle. Par ailleurs, d'autres découpages permettent d'obtenir une qualité de spécification équivalente.

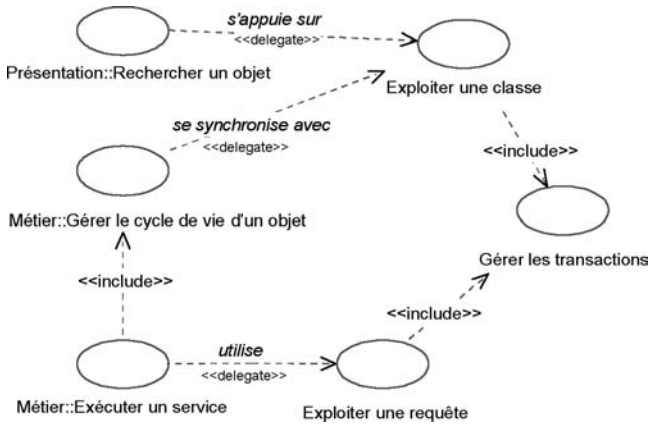
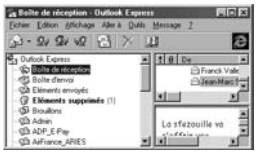
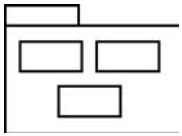


Figure 5-7 : Structure de la couche d'accès aux données

Définition des concepts techniques

Rappelez-vous qu'un bon modèle est un modèle qui colle à la réalité. Le domaine technique possède également sa réalité qui correspond d'une part à l'ensemble des outils mis en œuvre et d'autre part à l'aptitude de l'exploitant à s'interfacer avec le système, manipuler des objets, partager et stocker l'information.

Chaque couche comporte un vocabulaire spécifique correspondant à la description du point de vue technique qu'elle représente. Le tableau ci-après fournit un aperçu des différents concepts utilisés au niveau de chacune des couches. La définition des termes employés sera précisée dans les explications ultérieures.

Couche logicielle	Représentation	Concept manipulé
Présentation		Panneau Menu Bouton d'action ….
Application		Modèle Commande de l'application Liste d'objets …

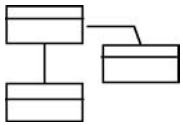
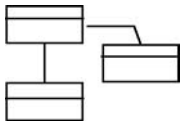
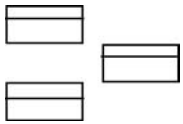
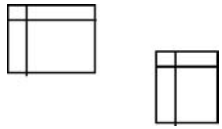
Couche logicielle	Représentation	Concept manipulé
Synchronisation du SI		Formats pivots Transformations Transcodage Références croisées …
Métier		Objet métier Objet composite Données réduites d'identification …
Accès aux données		Métaclasse T-uplet …
Stockage des données		Table Clé étrangère …

Tableau 5-2 : Exemples de concepts manipulés suivant les couches

Dans ce contexte, il paraît évident que chaque couche recourt à des concepts distincts pour lesquels il convient de spécifier une terminologie précise au sein d'un dictionnaire des termes techniques.


Conseil

ÉTABLISSEZ UN DICTIONNAIRE DE TERMES TECHNIQUES

Lorsque plusieurs architectes et concepteurs participent à la spécification technique, il existe un risque important d'incohérence entre les termes techniques, alors qu'une terminologie précise est indispensable à l'élaboration d'une architecture technique. Il est donc impossible de développer une spécification logicielle détaillée, sans recourir à la définition des termes techniques qu'un projet doit utiliser.

En ce qui concerne la gestion des commandes dans SIVEx, nous trouverons en principe :

- au niveau de la couche présentation : un panneau de présentation des commandes déjà prises dans une agence, un panneau de liste de commandes, un panneau d'édition d'une commande, des panneaux de détail pour les descriptions de colis, les adresses, les sites, les conditions tarifaires, etc ;
- au niveau de la couche application : un objet composite construit sur les classes d'analyse Commande, Colis et Site nécessaire à l'édition complète

d'une commande et de ses liens, une liste de commandes classées par agences de distribution et sites, des libellés caractéristiques qui permettent de sélectionner de manière univoque un objet dans une liste et que l'on compose généralement à partir d'une liste réduite de données de l'objet ;

- au niveau de la synchronisation du SI : des échanges entre applications réalisés sur la base de formats pivots, des transformations aux formats attendus par les applications, des transformations de codes qui demandent généralement la mise en œuvre de tables de correspondance et des références croisées qui permettent d'assurer l'unicité des objets référencés par des clés différentes dans chacun des progiciels concernés ;
- au niveau métier : un objet métier distribuant la commande et ses colis, un objet métier permettant d'accéder aux sites, un objet métier permettant d'accéder aux agences ;
- au niveau de l'accès aux données : une classe technique qui gère l'accès aux données pour chaque classe d'analyse Commande, Colis, Agence, Site, etc. Puisque cette classe représente les données d'une classe d'analyse, nous l'avons qualifiée de métaclasse. Une classe t-uplet représente l'ensemble des valeurs d'une ligne de résultats en réponse à une requête de sélection de données ;
- au niveau du stockage des données : des tables pour stocker les données de chaque classe et des clés étrangères pour y accéder.

Nous voyons ainsi comment les classes d'analyse vont se multiplier en différentes entités suivant les couches de conception. Si l'on considère de plus le besoin de regrouper les classes d'analyse Commande, Colis, Agence et Site dans un même panneau de présentation, on comprend mieux dans quelle mesure les besoins spécifiques à une couche nécessitent de manipuler des composites ne correspondant pas forcément aux seules classes du modèle d'analyse.

On est donc loin de la vision naïve qui consiste à croire qu'une classe d'analyse produit systématiquement une classe de conception à périmètre identique dans chacune des couches. C'est même rarement le cas tant les besoins de présentation, de distribution et de stockage agrègent souvent les classes en paquets rendus insécables pour des raisons de performance. Chaque couche doit donc gérer séparément des concepts qui vont servir à décrire les cas d'utilisation techniques, dans la mesure où le périmètre de responsabilités est forcément différent d'une couche à l'autre.

Description d'un cas d'utilisation technique

La description d'un cas d'utilisation technique est analogue à celle des cas d'utilisation de la spécification fonctionnelle. Dans ce cadre, on utilise un premier niveau de description, composé d'une fiche textuelle et d'un diagramme d'activité. Un second niveau de description objet complète éventuellement la spécification. On utilise alors un diagramme de classes et un diagramme d'interaction.

Les concepts utilisés pour décrire les cas d'utilisation appartiennent à la couche logicielle considérée et font l'objet d'une définition dans le dictionnaire des termes techniques. À titre d'illustration, sachez que l'on distingue le concept « Objet » suivant son appartenance à la couche « *Accès aux données* » ou à la couche « Métier ».

ÉTUDE DE CAS : DESCRIPTION D'UN CAS D'UTILISATION TECHNIQUE

Voici la description détaillée du cas d'utilisation « Accès aux données::Exploiter une classe ». Par analogie avec ce qui est présenté au chapitre 4, on retrouve ici les rubriques de pré- et post-conditions ainsi que la description des enchaînements.

Couche logicielle : Accès aux données.

Titre du cas d'utilisation : Exploiter une classe.

But : Le métier nécessite de charger, de répertorier et de sauvegarder une ou plusieurs instances d'une même classe.

Résumé : Répertorier plusieurs instances, charger ou sauvegarder une instance.

Exploitants et/ou couches exploitantes :

- la couche application lors d'une recherche d'objets ;
- la couche métier lors de l'exploitation des données de plusieurs instances particulières.

Tableau 5-3 : En-tête du cas d'utilisation technique « Accès aux données::Exploiter une classe »

Préconditions :

Néant.

Enchaînements :

Ce cas d'utilisation survient lorsque la couche de présentation souhaite rechercher ou présenter les données représentatives d'une liste d'objets, ou lorsque la couche métier désire synchroniser les objets avec leur système de persistance en SGBDR.

Enchaînement (a) renseigner le critère

Lorsque l'action de chargement, de suppression ou de modification concerne plusieurs instances, les demandes s'accompagnent d'un critère qui est exploité par la métaclasse pour produire la requête nécessaire.

Chaque critère s'accompagne lui même d'un t-uplet précisant les valeurs de référence ou de seuil à utiliser pour la requête.

Enchaînement (b) charger des objets

La couche métier demande le chargement d'instances à la métaclasse correspondante suivant un critère spécifié.

Enchaînement (c) supprimer des objets

La couche métier demande la suppression d'instances à la métaclasse correspondante suivant un critère obligatoirement spécifié.

Enchaînement (d) modifier des objets

La couche métier demande la modification d'instances à la métaclasse correspondante suivant un critère impérativement spécifié. Cet ordre s'accompagne obligatoirement d'un t-uplet fixant les valeurs d'entrée de la modification demandée.

Si le critère porte sur un identifiant nul, la demande sera interprétée comme une création ; l'identifiant du nouvel objet sera retourné. Dans tous les autres cas, il s'agira d'une mise à jour.

Enchaînement (e) charger une liste de libellés

La couche de présentation demande une liste de données – les données réduites – afin de construire les libellés permettant à l'utilisateur d'identifier un objet unique pour chaque item présenté. Chaque groupe de données est accompagné de l'identifiant de l'objet correspondant. Ce mécanisme permet à la couche présentation de synchroniser les sélections de l'utilisateur avec la couche métier.

Si le volume d'informations en retour dépasse le seuil de limitation réseau (par exemple 1 Ko en cas de ligne bas débit), le nombre de données réduites renvoyées est limité. Un indicateur de limitation est également retourné. La couche de présentation peut alors demander à recevoir la suite de la liste.

Enchaînement (f) produire et exploiter la requête

La métaclasse a pour rôle de produire et d'exploiter la requête nécessaire avec le cas d'utilisation « Gérer les transactions ». Dans tous les cas, un code d'erreur est retourné en cas d'échec de synchronisation avec le SGBDR. Dans le cas d'un chargement, les données sont renvoyées sous la forme d'une liste de t-uplets.

Exceptions :

Néant.

Post-conditions :

Les seuils de limitation réseau ne sont pas dépassés pour les requêtes qui concernent la couche de présentation.

Tableau 5-4 : Corps du cas d'utilisation technique « Accès aux données::Exploiter une classe »

Au même titre que ce qui a été réalisé pour les cas d'utilisation de la branche gauche, un diagramme d'activité, un diagramme de classes et un diagramme d'interaction peuvent être développés pour récapituler les spécifications du cas d'utilisation.

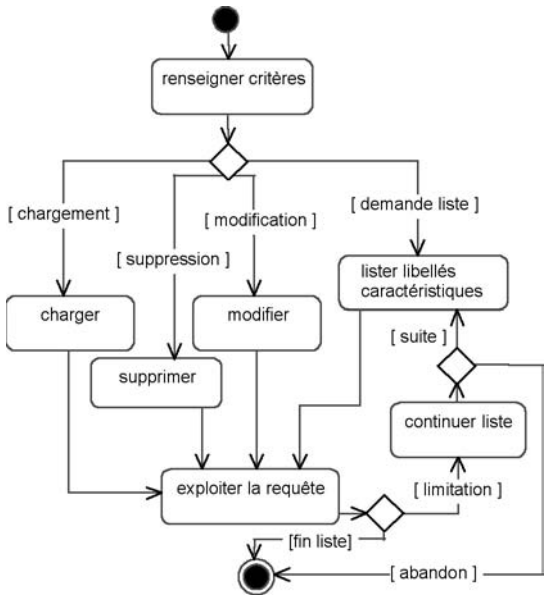


Figure 5-8 : Diagramme d'activité du cas d'utilisation technique « Accès aux données::Exploiter une classe »

Le diagramme des classes participantes aide à la définition des concepts techniques utilisés pour décrire le cas d'utilisation technique. Nous vous rappellons que chaque classe doit être définie dans le dictionnaire des termes techniques.

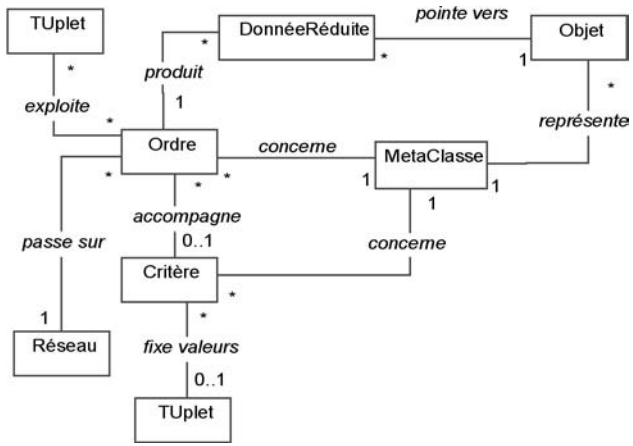


Figure 5-9. : Classes du cas d'utilisation technique « Accès aux données::Exploiter une classe »

Un diagramme de communication concourt à préciser la place et la responsabilité des concepts les uns par rapport aux autres dans le cas d'utilisation technique.

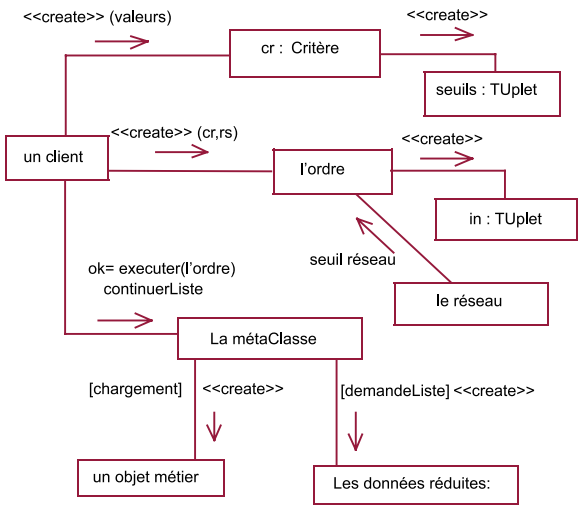


Figure 5-10. : Diagramme de communication du cas d'utilisation technique « Accès aux données::Exploiter une classe »

Ne pas faire

NE CODEZ PAS AVANT DE CONCEVOIR

Il résulte de la description d'un cas d'utilisation technique un modèle naïf, qui ne fait que refléter l'idée de l'architecte lorsqu'il spécifie les problèmes techniques. Les diagrammes UML développés ne sont donc qu'une façon d'exprimer un problème, ils ne constituent pas une solution.

Si nous implémentions les classes telles qu'elles sont identifiées dans les cas d'utilisation techniques, nous passerions à côté d'un véritable travail de conception :

- il n'y aurait pas de recherche de composants existants à réutiliser ;
- aucune recherche d'optimisation ne serait réalisée ;
- les concepts seraient mélangés et on obtiendrait des classes concentrant trop de responsabilités, c'est-à-dire un code objet lourd à maintenir. Ce serait le syndrome des classes obèses.

Par ailleurs, le modèle de spécification technique non encore bien défini reste nettement plus facile à corriger qu'un modèle de conception.

Phases de réalisation en capture des besoins techniques

La capture des besoins techniques est une étape de prise en compte des contraintes techniques et logicielles. Elle doit être suffisamment détaillée pour permettre d'aborder la conception générique du système.

Le processus de construction mis en œuvre dans l'étape est le suivant :

1. Capture des spécifications techniques liées à la configuration matérielle :
- identifier les contraintes techniques liées aux machines, aux connexions et aux déploiements existants ;
- produire le diagramme de configuration matérielle ;
- identifier les contraintes d'organisation spécifiées par les choix d'architecture.

2. Capture initiale des spécifications logicielles :
- identifier les besoins logiciels du point de vue des exploitants ;
- élaborer la description sommaire des cas d'utilisation techniques.

3. Spécification logicielle détaillée :
- identifier un découpage en couches logicielles ;
- identifier les cas d'utilisation techniques pour chaque couche ;
- élaborer la description détaillée des cas d'utilisation techniques.

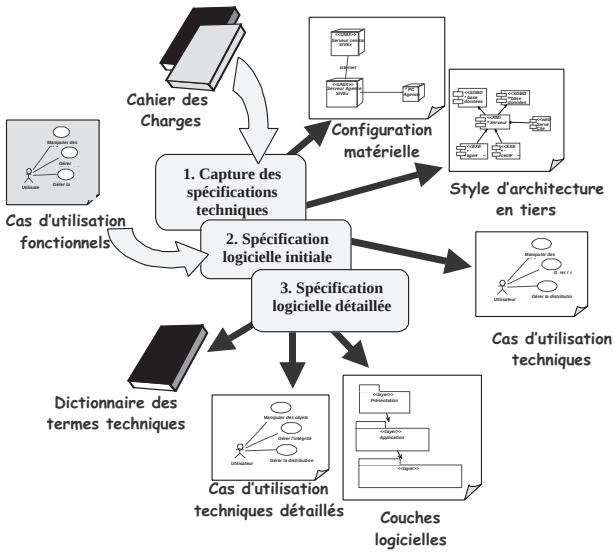


Figure 5-11 : Construction de l'étape de capture des besoins techniques