Gestion de parc : solution libre VS commerciale

■ Philippe HOUE, Philippe.Houe@emn.fr
Ecole des Mines de Nantes

Il existe de nombreux logiciels permettant de gérer un parc informatique. Ils ne répondent pas toujours aux réels besoins des administrateurs du parc, peuvent être coûteux à l'acquisition ou en maintenance.
Notre idée a donc été de développer notre application de gestion et pour cela nous avons décidé de comparer deux solutions utilisant pour l'une des produits propriétaires d'Oracle et pour l'autre des produits du domaine public. Cette étude s'est faite en plusieurs étapes par l'encadrement de projets étudiants.
Voici, à travers notre article, la démarche suivie et les conclusions issues de cette étude.

■ Contexte

Dans le cadre de la gestion de parc informatique, un centre de calcul souhaite souvent mettre en place un outil simple de gestion tout en se dotant d'une base de données sur ses interventions permettant de retrouver les solutions ou d'avoir un suivi de ses équipements informatiques.

De nombreuses solutions commerciales existent et selon le degré de complexité (et donc selon le coût) vont proposer d'une gestion simple (inventaire) jusqu'au modèle s'apparentant plus à une Gestion Technique de Bâtiment, où l'informatique est assimilée à du matériel équipant les bureaux.

Dans les faits, un centre de calcul doit recourir à plusieurs outils pour gérer son parc matériel et la multiplication de ces outils entraîne malheureusement une dispersion de l'information. Cette dispersion s'accroît avec le nombre d'intervenants.

Dans ce contexte nous avons cherché à créer nous mêmes le produit qui répondrait le mieux à nos besoins et nous avons étudié plusieurs aspects de notre gestion de parc.

Dans le cadre de la gestion de parc imprimantes, nous avons décidé de comparer des solutions utilisant des bases de données commerciale et publique et qui permettraient de centraliser l'information.

■ Objectifs

Le produit devra répondre à plusieurs objectifs : caractériser l'équipement informatique (fiche technique), caractériser les interventions de maintenance au niveau matériel ou logiciel (fiche intervention), permettre de suivre le parcours du matériel (acquisition, exploitation, stockage) et organiser les informations en une base de données accessible en réseau par plusieurs utilisateurs avec sécurité d'accès (gestion de profils).

Avant d'aborder les outils logiciels devant permettre la réalisation du produit, une méthodologie réfléchie impose les étapes suivantes :

- prendre connaissance de la gestion actuelle du parc,
- lister les fonctions attendues des utilisateurs,
- définir les modèles conceptuel /physique de la base de donnée et créer la base de données,
- définir les requêtes utiles pour répondre aux besoins utilisateurs.

Le choix d'un SGDBR va entraîner quelques différences notoires dans le schéma de la base puisque nous ne disposerons pas des mêmes types de données ou des mêmes objets chez Oracle ou MySQL.

■ Réalisation du schéma

Le schéma est constitué des tables et des objets de la base de données.

Afin de le réaliser, le développeur étudie les types de données dont il dispose, crée un schéma entité/relation (dans le cas d'une approche selon la méthode Merise), impose des contraintes d'intégrité à ces données et pensent aux objets du SGDBR qui l'aideront à générer une application propre.

Types de données

Les tables sont constituées de colonnes caractérisées par un type de données.

Les types de données dépendent du SGDBR. Le site www.mysql.com présente des tableaux comparatifs complets entre divers SGDBR courants selon les normes ANSI SQL 92 et ODBC 3.0, notamment entre MySQL 3.23.2 et Oracle 8.0.3.0.0 qui répondent aux normes précédentes. Ces tableaux présentent les types de données disponibles et les fonctions intégrées au moteur du serveur SQL. Même vis-à-vis de l'impressionnant Oracle, MySQL semble bien positionné et donner un large choix dans la conception du schéma.

Contraintes d'intégrité

Des contraintes appliquées à des colonnes dans les tables garantissent l'intégrité des données, donc celle de la table dans sa globalité. Les contraintes courantes sont : NOT NULL (pas de valeur nulle), UNIQUE, PRIMARY KEY (pk : NOT NULL + UNIQUE), FOREIGN KEY (fk : PK issue d'une autre table).

Les FK ne sont pas supportées par MySQL. Le manuel de MySQL traite de cette absence et répond aux cas particuliers d'utilisation des FK : intégrité référentielle et jointures entre tables.

Ceci n'est pas une différence négligeable. L'existence des FK permet d'éclater une table en plusieurs quand les colonnes sont indépendantes et quand des valeurs reviennent souvent.

Cependant, MySQL supporte la syntaxe de création des FK. Ceci est autorisé uniquement dans le but de conserver les mêmes scripts de création de base issus des outils propriétaires car en fait rien ne se passe.

Modèles conceptuel et physique

Le modèle conceptuel présente les tables et les relations voulues entre elles par le développeur.

Le modèle physique présente les tables issues du modèle conceptuel et qui traduisent les relations définies précédemment.

Pour réaliser les modèles conceptuel et physique, à moins d'être un expert en Merise et en SQL, le mieux est encore d'utiliser un outil d'aide à la conception. Les outils courants sur le marché sont : Power AMC Designer de PowerSoft, Designer 2000 d'Oracle. Ces outils permettent en outre de générer un modèle physique issu d'un modèle conceptuel tout en tenant compte du SGDBR, puis de générer les scripts SQL de création/suppression de la base de données.

Ainsi dans la liste des SGDBR disponibles pour la génération des scripts de création de bases, on trouvera, bien sûr, Oracle et sinon on choisira la norme SQL ANSI 92 pour MySQL.

Objets de la base de donnée Oracle

En dehors des tables, Oracle met à disposition un très grand nombre d'objets dont nous présentons les principaux dans le cadre de notre développement.

- Séquence : Une séquence est une suite d'entiers uniques en ordre croissant ou décroissant et non obligatoirement consécutifs. L'intérêt des séquences réside dans la génération des valeurs uniques pour une clé primaire. La séquence est incrémentée ou décrémentée par une routine interne au SGDBR et permet un gain de temps.
- Index : Cet objet permet d'accélérer l'extraction de ligne par le biais d'un pointeur. En l'absence d'index, lors d'une requête sql, la table entière est balayée. L'index donne un accès direct et rapide aux données.
  Sous oracle, les index sont automatiquement créés sur les PRIMARY KEY.
  Nous pouvons également en créer sur les colonnes qui seront appelées le plus souvent dans les critères de recherche.
- Procédures et fonctions : Oracle permet au développeur de définir de nouvelles fonctions.
  Il est souvent judicieux de regrouper des commandes SQL dans un bloc PL/SQL puis de déclarer ce bloc comme une procédure. Il est conseillé au développeur de stocker ses propres procédures dans la base, même s'il est possible de faire autrement. Une procédure stockée peut être référencée à partir de n'importe quelle application connectée à Oracle.

- Package : Cet autre fonctionnalité est apportée encore par le PL/SQL et permet de rassembler des procédures et fonctions en paquet pour faciliter la programmation.
- Trigger : Le trigger est un programme PL/SQL exécuté lorsque certains événements se produisent sur une table. Il est équivalent à un démon unix.

Objets de la base de données MySQL

MySQL présente une autre approche et ne bénéficie pas d'un moteur PL/SQL.

- Séquence : MySQL ne propose pas d'objet séquence mais fournit les mécanismes de remplacement. Une routine applicative est nécessaire pour remplir cette fonction ou bien nous pouvons nous appuyer sur une table et utiliser la fonction last_insert_id().
- Index : Ils sont identiques à ceux définis dans Oracle et présentent les mêmes fonctionnalités.
- Procédure : Première pierre d'achoppement, les "stored procedures" ne sont pas supportées par MySQL. Des procédures et des fonctions « natives » peuvent cependant être ajoutées à MySQL. Vous connaissez le C++ et avez déjà entendu parler de lex et yacc. Alors récupérez les sources de MySQL et n'hésitez pas à rentrer dans le code tel que MySQL Reference Manual le décrit pour y insérer vos nouvelles fonctions. Elles seront disponibles alors pour tout utilisateur dans le moteur de la base.
- Trigger : C'est la deuxième pierre d'achoppement pour le développeur sous MySQL. Très vite, celui-ci s'inquiète de l'existence de cet outil. La réponse est négative mais des mécanismes de remplacement sont proposés. Ils sont basés sur la fonction update_time : ce n'est donc plus un mécanisme interne au moteur du SGDBR qui veille comme un démon mais l'application elle-même qui doit être écrite pour tenir compte des événements.
  PostgreSQL, le concurrent de MySQL au niveau logiciel libre, propose par exemple l'objet trigger.

■ Solutions logicielles

Dans les deux cas, le client retenu est un browser web. En effet l'application s'adressera à des utilisateurs aux profils différents (consultation, saisie ou administration), sur des plates-formes également différentes et à partir de postes non dédiés.

L'utilisation d'un browser standard comme Netscape 4.5 nous garantit le suivi de la norme HTML 3.2.

Solution Oracle

Présentation d'Oracle Web Application Server (owas)

C'est un serveur Web particulier qui écoute sur plusieurs ports tcp précis par l'intermédiaire de programmes démons : les listeners. Le serveur Web répond aux requêtes reçues sur ces ports en fonction de ses paramètres de configuration. Les appels sont adressés à des composants internes appelés Web Request Broker Cartridges, capables de gérer les petites comme les larges requêtes avec les performances accrues apportées par le multiprocessing.

Les WRB Cartridges peuvent être de types différents : PL*SQL, Java, liveHTML, VRML. La possibilité est offerte d'en créer de nouveaux Cartridges (vendor, Personal).

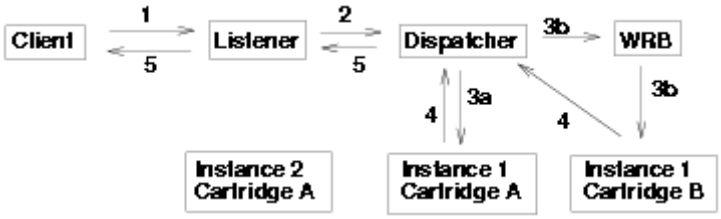
Au niveau d'OWAS, nous devons configurer un DataBase Access Descriptor (DAD) comme méthode de connexion à la base Oracle, puis un nouveau Cartridge PL*SQL pour préciser le schéma à utiliser dans Oracle et enfin un listener pour indiquer un port de communication privilégié du serveur Web.

Mécanismes de la solution Oracle

Oracle contient un moteur PL*SQL. C'est donc ce langage qui est choisi pour le développement. Les procédures et les packages sont inclus dans le schéma de l'utilisateur Oracle.

OWAS met à disposition un schéma supplémentaire chargé dans Oracle (WWW_USER) constitué de fonctions et de procédures qui sont rendues accessibles à n'importe quel utilisateur par l'intermédiaire de « grant ». Ce schéma apporte des fonctionnalités HTML dans deux packages WWW_USER.HTF et WWW_USER.HTP.

Voici au travers du chemin d'une requête, une vue simplifiée du mécanisme interne à OWAS :



1- le client adresse une requête sur le port d'un listener
2- le listener la passe au Dispatcher.
3- le Dispatcher a pour rôle de diriger la requête vers le bon Cartridge :
    3a- si une instance du Cartridge est disponible, la requête lui est transmise
    3b- si aucune instance n'est disponible, WRB en ouvre une nouvelle.
4- le Cartridge répond au Dispatcher
5- et celui-ci transmet le résultat de la requête au client.

Pour illustration, le service s'appelle de la façon suivante : http://us7.eleve.emn.fr :8880/

Une Procédure s'appelle de la façon suivante : http://us7.eleve.emn.fr :8880/owp_printer/plsql/nom-package.nom-procedure

Le Dispatcher ou le WRB d'OWAS adresse cette requête au Cartridge PL*SQL défini par owp_printer. Ce cartridge adresse la procédure du package nommé au schéma qui lui est associé par le DAD.

Passage des paramètres

Le passage des paramètres saisis par un utilisateur dans un formulaire HTML peut se faire par deux méthodes.

- Par le méthode GET, les paramètres sont envoyés à la procédure appelée en étant ajoutés à la fin de l'URL: http://us7.eleve.emn.fr:8880/owp_printer/plsql/
- nom-package.nom-procedure ?nom_param1=valeur_param1&nom_param2=valeur_param2...
- Par la méthode POST, les paramètres sont envoyés à la procédure appelée sans être ajoutés dans l'URL : http://us7.eleve.emn.fr:8880/owp_printer/plsql/nom-package.nom-procedure

Dans la mesure où le nombre de paramètres à passer à une procédure peut être élevé (jusqu'à 30 dans l'ajout d'un modèle d'imprimante), nous avons préféré travailler avec la méthode POST.

Au niveau de la procédure, les valeurs contenues dans les tables sont récupérées au moyen de l'objet CURSOR. Les paramètres passés par POST peuvent alors être comparés aux valeurs d'un curseur pour vérifier leur unicité dans les tables.

Solution MySQL

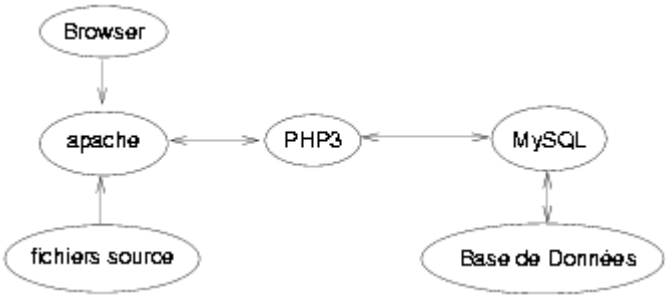
Pourquoi PHP3 ?

PHP3 est un langage de script côté serveur similaire à Perl et s'intégrant dans des pages html. Il permet de générer dynamiquement des pages web. Il nous intéresse car il peut être utilisé comme un programme séparé (CGI) ou comme module d'extension dans Apache.

Le code en PHP3 aura plusieurs tâches à effectuer : générer dynamiquement le code HTML, interroger la base de données pour stocker les données transmises via des formulaires ou les récupérer.

Mécanisme de la solution MySQL

L'architecture de l'application est la suivante :



La procédure de dialogue est décrite par les étapes suivantes :

- Le client demande accès à une page du serveur Apache.
- Le serveur détecte l'extension.php3 ou .phtml et fait traiter la page par le module PHP3.
- Le script est exécuté par le module et si besoin, il adresse des requêtes SQL à MySQL via une connexion TCP/IP ou bien utilise les fonctions offerte par les librairies C distribuées avec MySQL (c API).
- MySQL répond au module après une consultation possible d'une ou plusieurs tables dans le schéma de la Base de données (avec authentification des droits de l'utilisateur courant).

Passage des paramètres

Nous recourrons aux deux méthodes qui s'appuient sur une routine interne (sqlUpdate) : la méthode GET pour passer les valeurs saisies en petit nombre et la méthode POST dans l'utilisation de formulaire.

Nous utiliserons pour cela les fonctions C distribuées avec MySQL dans la librairy mysqlclient (C API) : mysql_insert_id() et mysql_query().

Sécurité et droits d'accès

Nous sommes dans une architecture trois tiers (un client, un serveur web et un serveur de base de données) et les problèmes de sécurité deviennent complexes. Nous supposerons donc que la sécurité est déjà assurée (ou suivie) au niveau système d'exploitation et au niveau réseau.

Solution Oracle

La sécurité se fait au niveau d'Oracle et d'Oracle Web Application Server.

Au niveau du SGDBR, un schéma principal existe. L'administrateur octroie des privilèges system à cet utilisateur principal. D'autres profils utilisateurs peuvent être créés et des privilèges objets accordés en fonction de ces profils par l'utilisateur principal. Ces privilèges Objets d'Oracle sont les suivants : alter, delete, execute, index, insert, references, select et update.

Avec OWAS, les possibilités sont multiples car chaque module DAD, Cartridge et Listener peut être sécurisé. L'idée principale consiste à gérer les profils utilisateurs en fonction des ports des listeners.

Solution MySQL

Concernant l'accès à la base de données, MySQL dispose d'un système d'authentification intégré permettant d'identifier l'utilisateur par son nom et l'hôte depuis lequel il se connecte. Les privilèges fonctionnent sur des droits d'actions équivalents à certains privilèges sytem et objets d'Oracle (pas tous en fait). Les droits peuvent être fixés sur un utilisateur, une base de données, une table ou une colonne dans une table.

Apache offre un autre moyen de contrôle sur l'accès aux pages de gestion par des filtres sur l'hôte et en utilisant un fichier de mots de passe. Une boîte de dialogue s'ouvre lors de l'accès à une page protégée.

Nous l'avons dit, PHP3 peut fonctionner en mode CGI ou en mode intégré au serveur Apache.

En mode CGI, l'interprétation des fichiers se fait en appelant le programme par la commande system() et la sortie standard du programme est la page HTML à afficher. Dans ce mode de fonctionnement, le programme bénéficie des droits d'exécution d'Apache et il peut accéder à l'ensemble des fichiers que ces droits lui permettent.

En s'insérant dans Apache sous le forme d'un module, PHP peut avoir accès aux variables et commandes internes. La gestion de la sécurité de PHP est alors la même que celle du serveur web. La possibilité nous est offerte de bénéficier des transferts sécurisés des logins ou d'installer un module de sécurisation des transferts de pages web.

Le contrôle d'accès à la Base se fait entièrement via MySQL. Le dialogue avec MySQL se fait soit par socket unix sur un hôte local, soit par socket TCP/IP sur un numéro de port précis sur un hôte distant.

■ Les conclusions

L'expérience acquise autour de la réalisation de l'application précédente nous a permis de redéfinir correctement notre cahier des charges pour la gestion d'un parc dans sa globalité.

L'interface graphique

Le HTML est simple mais le revers de cette simplicité se retrouve dans l'interface graphique. Il faut aller vers le Javascript pour rendre la page plus conviviale. Une solution purement HTML conduit à un nombre impressionnant de scripts et de pages et l'utilisateur s'y perd (ne parlons pas du développeur). Le javascript nous ouvre la possibilité de faire le contrôle de la nature des données avant envoi sur le serveur.

Par exemple, pour générer les formulaires de saisie des fournisseurs, des imprimantes, de leur nom, de leur modèle et des interventions nous sommes arrivés à 2500 lignes de code en HTML pure et à 700 lignes en utilisant le javascript pour un résultat plus convivial.

Développer des formulaires dynamiques reste laborieux. Une solution utilisant MS-Access peut s'avérer plus rapide à développer.

Les langages de développement

Dans le cadre d'une solution Oracle, le PL*SQL nous est presque imposé. Il permet de définir des structures de données nécessaires à la programmation relative à une base de données. Mais dans les versions que nous avons utilisées, il ne nous a pas permis de programmer une interface graphique sans utiliser de manière intensive le HTML. Il ne nous a pas permis non plus d'effectuer une programmation orientée objet, c'est-à-dire facilement réutilisable. Dans les évolutions des produits Oracle, ces problèmes devraient disparaître : Oracle 8i est complètement orienté Internet, Oracle Application Server 4.0.8 doit intégrer Java et Oracle.

Le langage PHP3 est très inspiré du Perl, mais nous restons loin de langages comme Java, le C ou le C++. C'est un langage très orienté manipulation de chaînes de caractères et connexion à des bases de données. La possibilité de l'insérer dans Apache est très intéressante du point de vue sécurité. Il offre également la possibilité de se connecter à de nombreux SGBD. Par contre, il est dépourvu de la gestion de structures de données et la notion de classe est incomplète et frustrante.

L'utilisation de Java semble plus indiquée pour une solution web. Non seulement nous avons un langage riche et intégrant un grand nombre de composants graphiques pour définir l'interface, mais en plus, nous avons toujours le contrôle de validité comme avec le javascript et surtout, toutes les variables du programme sont gardées dans la mémoire du client (àà mettre en opposition avec la perte d'information due au principe des connexions : lorsque le script se termine, les variables définies sont perdues).

Les avantages de Java sont :

- le développement de composants réutilisables en utilisant les mécanismes de l'objet : définition de classes de connexion ou de déconnexion à la base de données, modules d'extraction de données...,
- une bien plus grande variété de structures de données : les concepts orientés objets permettent de créer tout type de structure de données adapté au problème à résoudre,
- la présence d'une vaste librairie graphique : Swing de Java 1.2, l'AWT (Abstract Window Toolkit) de Java 1.1, Java 2D...,
- la présence d'une librairie dédiée aux bases de données : JDBC (avec des drivers JDBC pour la majorité des bases de données du marché et bien sûr Oracle8 et MySQL),
- la connaissance du langage par un grand nombre de programmeurs (il y a beaucoup plus de gens connaissant le Java que le PL/SQL),
- la possibilité de créer des applets,
- la plus grande facilité de maintenance de l'application : le code source est composé d'un seul langage (Java) et non de deux langages (HTML et PL/SQL) et les composants réutilisables sont facilement modifiables,

Nous avons donc retenu Java pour nos développements futurs.

Le SGDBR

Oracle reste le SGDBR de référence pour les professionnels mais MySQL nous est apparu comme une alternative très séduisante.

Pour des schémas très complexes, nous aurions tendance à choisir Oracle qui a l'avantage de gérer dans son moteur interne des objets ou des notions qui nous sont apparus essentiels dans notre développement : séquence, trigger, foreign key. Cependant MySQL n'est pas sclérosé et des évolutions sont attendues pour intégrer de nouvelles fonctionnalités. La liste des TODO est longue, pointilleuse et prometteuse.

En dehors des objets utiles au développement, nous pouvons avoir d'autres attentes de la part du SGDBR. La différence fondamentale entre les deux SGDBR se situe en fait au niveau du transactionnel. Rappelons que les ordres SQL avec contrôle de transaction sont : INSERT, UPDATE et DELETE. Des mécanismes (commit, SAVEPOINT et ROLLBACK) servent à valider/contrôler ces ordres dits de manipulation de données. Oracle accepte les opérations COMMIT et ROLLBACK, MySQL ne les accepte pas. Dans notre cas, ce n'était pas problématique : nous avons peu de saisies à effectuer simultanément. Ces opérations de contrôle du transactionnel entraînent d'ailleurs une gestion gourmande de la mémoire dans les SGDBR qui l'implémentent et cela explique

la rapidité de MySQL vis-à-vis de ceux-ci. Cependant notons que MySQL propose une solution par l'intermédiaire de verrouillage de table.

Dans le cadre de la maintenance, les mécanismes offerts chez Oracle se retrouvent chez MySQL, à savoir : sauvegardes, export/import de schéma et même la réplication.

Dans notre cas, nous administrons déjà un grand nombre d'applications sous Oracle et il est, bien sûr, plus judicieux de conserver une homogénéité dans nos produits. Nous finaliserons donc notre application sous Oracle. Mais pour ne pas perdre les compétences acquises, nous mènerons encore des projets autour de MySQL.

■ Conclusion

MySQL répond plus simplement à nos besoins mais si vous possédez déjà les produits Oracle, séparément ou en bundle (iDP), autant profiter de l'ensemble. Gardons cependant à l'esprit que le domaine public nous offre visiblement un grand nombre d'alternatives.

La question devient alors plus une histoire de coût et de temps à consacrer à la découverte de ces produits.

Ce sujet concret nous a permis de comparer deux logiciels différents dans leur philosophie. Nous serions tentés de dire encore une fois que la communauté informatique est capable en coopérant de se doter d'outils puissants et simples à la fois pour répondre à ses attentes.

■ Références

Christophe Calandreau et Laurent Boulard : Rapport sur le développement du modèle de gestion de Parc Imprimantes.

Samad Bernichi, Fabrice Retière et Laurent Conin : Etude du modèle de gestion de Parc Informatique.

Vous trouverez les références bibliographiques, manuels, documentations et liens à l'adresse ftp.emn.fr:/pub/sic/jres99/references.doc ainsi que d'autres fichiers relatifs à cet article.