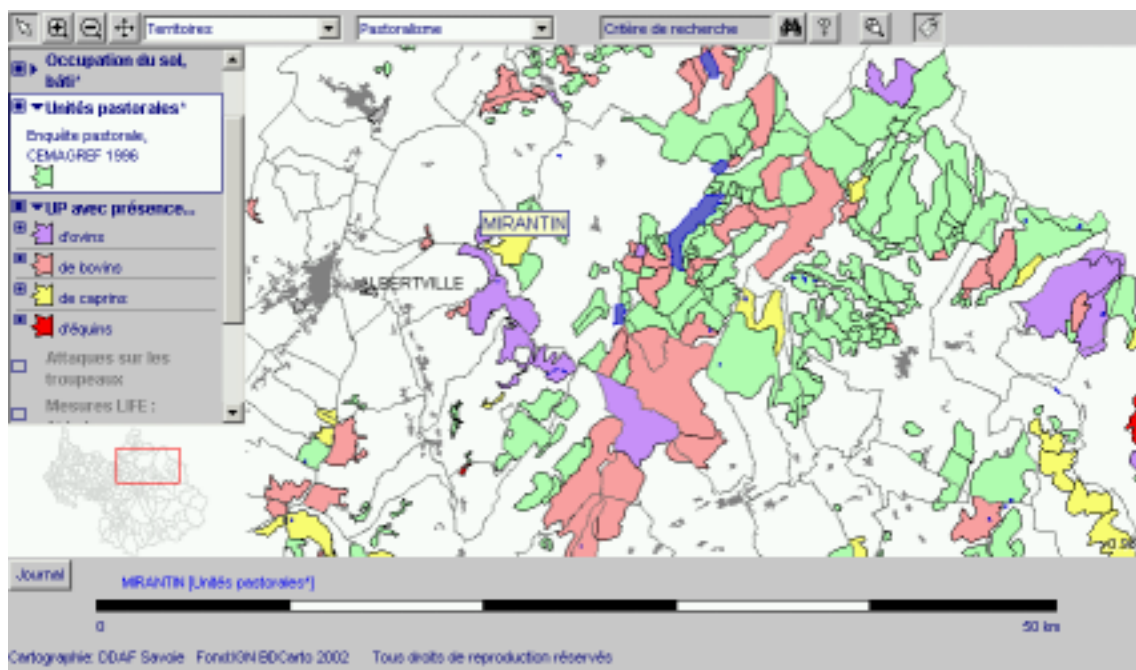


ALOV Map, installation et paramétrage

Configuration client-serveur,
connexion à une base de données MS Access ou MySQL



Hélène BUISSART

le 28 septembre 2004

© ENSG

Stage Pluridisciplinaire du 1^{er} juin au 10 septembre 2004

Diffusion Web : Internet Intranet Polytechnicum Intranet ENSG

Situation du document :

Etude réalisée au cours du stage pluridisciplinaire de fin de 2^{ème} année du cycle des ITGCE

Nombre de pages : 101 dont 37 d'annexes

Système hôte : Word 2000

MODIFICATIONS

EDITION	REVISION	DATE	PAGES MODIFIEES
1	0	09/08/04	Création
2	0	18/08/04	Création

Table des matières

<i>Table des matières</i>	3
<i>Liste des annexes</i>	5
<i>Liste des figures</i>	6
<i>Liste des tableaux</i>	7
<i>Glossaire et sigles utiles</i>	8
<i>Préambule</i>	9
<i>I – Installation d'ALOV Map en configuration Client-serveur avec Tomcat</i>	11
1 – Installation de la machine virtuelle Java (JRE)	11
2 – Installation de Tomcat 5.0	12
3 – Configuration d'ALOV Map	18
3.1 – Les fichiers utilisés par ALOV Map	18
3.2 – Accès à l'outil d'administration	19
3.3 – Connexion ALOV Map – SGBD	20
3.4 – Liaison d'ALOV Map à la base de données	23
<i>II – Paramétrage de l'Applet ALOV Map</i>	26
1 – La balise <applet>	26
2 – Les paramètres de l'applet ALOV Map	27
<i>III – Import et mise à jour des données</i>	28
1 – Import des données vecteur	28
1.1 – Export à partir de MapInfo	28
1.2 – Import dans ALOV Map	30
1.3 – Enregistrement du jeu de données	32
2 – Les données raster	35
2.1 – Paramètres nécessaires au calage sous ALOV Map	35
2.2 – Méthode de rasterisation et de calage à partir de MapInfo	36
3 – Mise à jour des données	37

IV – Structure de la base de données	39
1 – Les données d'administration et de paramétrage d'ALOV Map	39
1.1 – tm_datasets	40
1.2 – tm_instance	40
2 – Les données vectorielles	43
2.1 – Stockage de la géométrie des vecteurs	43
V – Paramétrage des données, le fichier "project"	45
1 – Présentation des éléments du fichier "project"	45
2 – Les éléments utilisés dans le fichier "project" et leurs attributs	46
3 – La sémiologie des couches	49
3.1 – Sémiologie par défaut	50
3.2 – Analyse par symboles proportionnels	51
3.2 – Analyse par valeurs individuelles ou par classes de valeurs	51
3.3 – Analyse par histogrammes ou diagrammes en secteurs	52
3.4 – Etiquettes	52
3.5 – Textures	53
3.6 – Formules	53
VI – Paramétrage de l'interface de l'applet	54
1 – Structure du fichier "layout" de mise en page	54
2 – Structure de l'interface	54
3 – Les éléments utilisés dans le fichier "layout" et leurs attributs	55
Annexes	59

Liste des annexes

<i>Annexe 1 – Notions de base sur le XML</i>	<i>60</i>
<i>Annexe 2 – Procédure de mise à jour d'un jeu de données</i>	<i>61</i>
<i>Annexe 3 – Code source du fichier de lancement de l'applet ALOV Map</i>	<i>Erreur ! Signet non défini.</i>
<i>Annexe 4 – Code source du fichier de mise en page de l'applet ALOV Map, fichier "layout"</i>	<i>Erreur ! Signet non défini.</i>
<i>Annexe 5 – Exemple de code source du fichier de paramétrage des données utilisées par l'applet ALOV Map, fichier "project"</i>	<i>Erreur ! Signet non défini.</i>

Liste des figures

Figure 1 – JRE, Création de la variable d'environnement	12
Figure 2 – Tomcat 5.0, accueil de l'utilitaire d'installation	12
Figure 3 – Tomcat 5.0, acceptation de la licence Apache	13
Figure 4 – Tomcat 5.0, configuration personnalisée de l'installation	14
Figure 5 – Tomcat 5.0, répertoire d'installation de Tomcat	14
Figure 6 – Tomcat 5.0, configuration des paramètres d'administration	15
Figure 7 – Tomcat 5.0, chemin d'accès à la machine virtuelle Java	16
Figure 8 – Tomcat 5.0, installation en cours	16
Figure 9 – Tomcat 5.0, lancement de Tomcat	17
Figure 10 – Tomcat 5.0, Variable d'environnement	18
Figure 11 – Interface de connexion à l'outil d'administration d'ALOV Map	20
Figure 12 – Liaison ODBC : choix du pilote	21
Figure 13 – Liaison ODBC : paramétrage de la connexion	22
Figure 14 – Configuration de la liaison entre ALOV et la base de données source	23
Figure 15 – Définition des pilotes JDBC associés aux SGBD	24
Figure 16 – MapInfo, lancement du Traducteur Universel	29
Figure 17 – MapInfo, paramétrage du Traducteur Universel	29
Figure 18 – Interface d'enregistrement des fichiers vecteur sous ALOV	30
Figure 19 – Fenêtre d'état après un import correct	31
Figure 20 – Enregistrement des données vecteur sous ALOV (1/3)	32
Figure 21 – Enregistrement des données vecteur sous ALOV (2/3)	34
Figure 22 – Enregistrement des données vecteur sous ALOV (3/3)	34
Figure 23 – Enregistrement des données rasters sous ALOV	35
Figure 24 – Recherche d'un jeu de données par ALOV	37
Figure 25 – Résultat de la recherche d'un jeu de données par ALOV	38
Figure 26 – Récupération de la sémiologie via MapInfo	50

Liste des tableaux

<i>Tableau 1 – Récapitulatif des éléments et attributs utilisés pour lancer une applet</i>	<i>26</i>
<i>Tableau 2 – Table tm_servers</i>	<i>39</i>
<i>Tableau 3 – Structure de la table tm_datasets</i>	<i>40</i>
<i>Tableau 4 - Structure de la table tm_instance</i>	<i>41</i>
<i>Tableau 5 – Extrait de la table tm_instance du DGEAF</i>	<i>41</i>
<i>Tableau 6 – Codages principaux des données dans tm_instance</i>	<i>42</i>
<i>Tableau 8 – Récapitulatif des éléments et attributs utilisés dans le fichier "project"</i>	<i>46</i>
<i>Tableau 9 – Définition des types de valeurs</i>	<i>49</i>
<i>Tableau 10 – Récapitulatif des éléments et attributs utilisés dans le fichier "layout"</i>	<i>55</i>

Glossaire et sigles utiles

API	Application Programming Interface : documentation de l'ensemble des classes Java de base disponibles sur la plateforme utilisée.
AWT	Abstract Window Toolkit : API standard pour la création des Interfaces Homme-Machine des programmes Java.
HTML	HyperText Markup Language : Langage utilisant les balises pour formater du texte sur le Web.
JDBC	Java DataBase Connectivity : API Java permettant de se connecter à des bases de données.
ODBC	Open DataBase Connectivity : Couche logicielle permettant à une application Windows d'accéder de façon transparente à une base de données SQL.
Raster	Image découpée en pixels, sans information de géoréférencement.
SGBD	Système de Gestion de Bases de Données.
SIG	Système d'Information Géographique : Base de données géographiques et apparentées, en général relative à une région déterminée plus ou moins étendue.
SQL	Structured Query Language : Langage d'interrogation des bases de données relationnelles.
XML	EXtensible Markup Language : Langage à balises défini par une syntaxe très stricte; il permet d'ajouter ses propres balises en fonction de ses besoins.

PREAMBULE

ALOV Map est un outil SIG diffusé gratuitement et développé par l'Université de Sidney. Ce projet fait désormais partie du projet TimeMap, <http://www.timemap.net>. ALOV diffuse sur son site Internet : <http://www.alov.org>, une aide de développement en anglais dont cette notice est en partie inspirée et dispose d'un forum d'utilisateurs très actif permettant de soumettre des questions aux programmeurs de l'outil.

ALOV Map s'appuie sur la technologie des applets et servlets Java et permet de diffuser des données cartographiques, aussi bien rasters que vecteurs en utilisant l'applet seule ou combinée à la servlet. Cette dernière permet alors d'assurer la gestion des données au sein d'un Système de Gestion de Bases de Données au moyen d'un "outil d'administration" : configuration dite "client/serveur". C'est cette seconde configuration, qui est conseillée par les développeurs d'ALOV Map pour obtenir des performances optimales de l'outil. Elle permet en effet de gérer les données géographiques et attributaires dans une base de données et d'améliorer considérablement par sa gestion de la géométrie les temps d'accès à l'information.

Cette base de données a été mise en œuvre sur deux SGBD différents : MS Access et MySQL. Pour configurer et paramétrer l'outil et les données, la servlet fournit une interface d'intégration et de conversion des données des formats vectoriels traditionnels (Shape d'ESRI ou MIF de MapInfo) vers le SGBD choisi. Il s'agit de "l'outil d'administration" géré par la servlet.

ALOV est paramétrable par le biais de fichiers XML. Ce langage à balises s'appuie sur une syntaxe très encadrée ne souffrant aucune exception : toute balise ouverte doit être fermée de manière à créer un fichier arborescent (voir en Annexe 1 pour une première approche de la syntaxe XML).

L'applet ALOV Map s'appuie sur 3 fichiers :

- le fichier de lancement de l'applet. Il s'agit d'un fichier HTML,
- le fichier de paramétrage de l'aspect visuel de l'applet et de ses fonctionnalités. Il s'agit d'un fichier XML appelé "*layout*" (pour mise en page),
- le fichier de paramétrage des données diffusées. Il s'agit d'un fichier XML appelé "*project*".

L'ensemble des informations contenues dans ce document correspondent à la configuration suivante, utilisée à la Direction Départementale de l'Agriculture et de la Forêt de la Savoie :

- système d'exploitation : Windows 2000,
- système de gestion de base de données : MS Access 2000, MySQL v. 3.23.49,
- serveur de servlets et serveur Web : Tomcat version 5.0,
- machine virtuelle Java : Java 2 Runtime Environment, Standard Edition version 1.4.2_04,
- ALOV Map : ALOV Map version 0.98h.

Cette notice expose chronologiquement les différentes étapes rencontrées par le gestionnaire de l'outil ALOV Map lors de sa mise en œuvre. Ainsi, nous aborderons successivement les questions d'installation et de lancement d'ALOV Map dans sa configuration Client-serveur, la gestion des données au sein des Systèmes de Gestion de Base de Données MS Access et MySQL ainsi que la structure des fichiers de paramétrage des données et de l'applet.

I – INSTALLATION D'ALOV MAP EN CONFIGURATION CLIENT-SERVEUR AVEC TOMCAT

1 – Installation de la machine virtuelle Java (JRE)

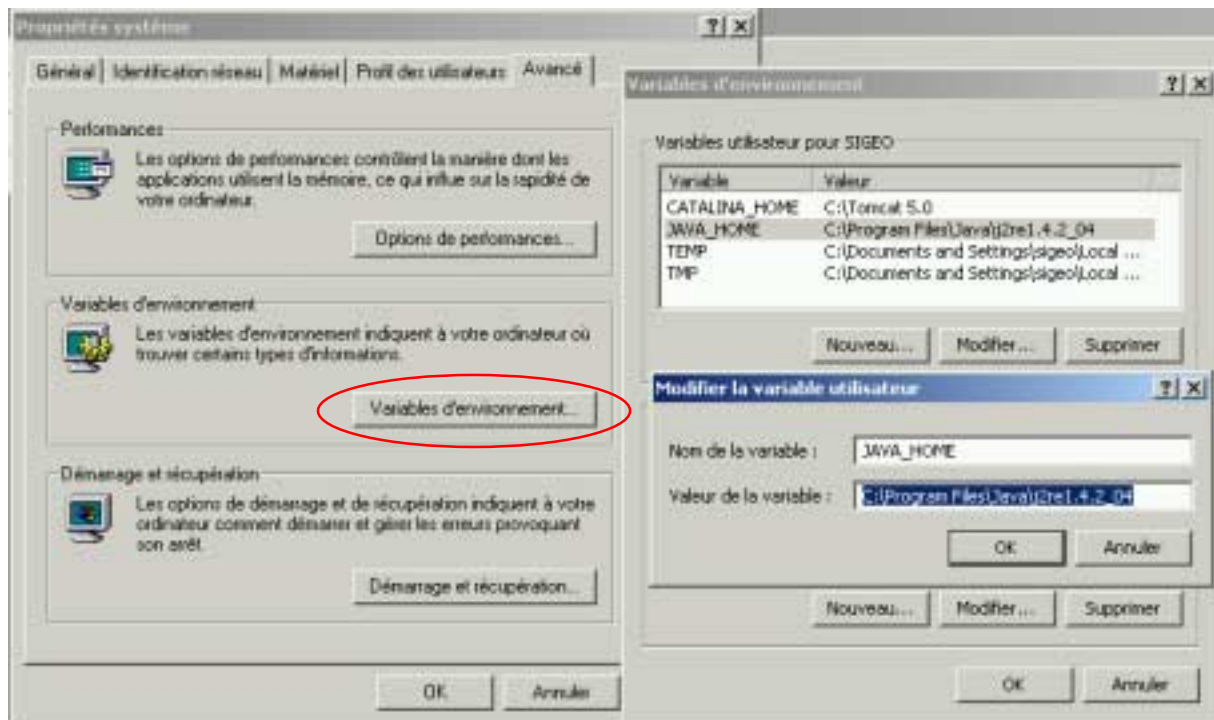
Les applets et servlets JAVA utilisent l'environnement d'exécution Java. Cet outil, le *Java Runtime Environment*, est diffusé gratuitement par *Sun Microsystems* et est intégré dans la plupart des navigateurs Web. La dernière version diffusée par *Sun* est disponible sur Internet à l'adresse suivante : <http://java.sun.com/downloads/index.html> ; il s'agit du **J2SE v 1.4.2_05 JRE** : *Java Platform, Standard Edition version 1.4.2_05 Java Runtime Environment*.

Pour un fonctionnement optimal de l'applet, il est nécessaire d'utiliser une version 1.4 ou supérieure. En effet, avec une version antérieure, les options de transparence et de texture notamment ne sont pas gérées et conduiront à l'emploi de la valeur jaune par défaut pour la représentation des couches.

L'installation de la machine virtuelle se fait en exécutant le fichier .exe disponible sur Internet. Celui-ci contient un assistant d'installation qui guide l'utilisateur.

A l'issue de l'installation, il est nécessaire de créer une nouvelle variable d'environnement pour le bon fonctionnement de Tomcat. Celle-ci, appelée JAVA_HOME, pointe vers le répertoire racine du JRE. Pour cela, dans la zone "Système" du "Panneau de configuration", dans l'onglet "Avancé", on choisit la rubrique "Variables d'environnement...", il s'agit ensuite de créer une nouvelle variable :

Figure 1 – JRE, Création de la variable d'environnement



2 – Installation de Tomcat 5.0

Tomcat est un serveur d'applications Java Open Source diffusé par "The Apache Software Foundation" sous licence "Apache Software License". Il s'agit d'un conteneur de servlets, c'est-à-dire d'un serveur pouvant exécuter des servlets Java à partir des requêtes Web qu'il reçoit et renvoyer les informations traitées au navigateur client.

Tomcat peut être utilisé comme serveur Web ou combiné à un autre serveur Web comme Apache, ce qui permet alors de disposer des fonctionnalités cumulées de ces deux serveurs.

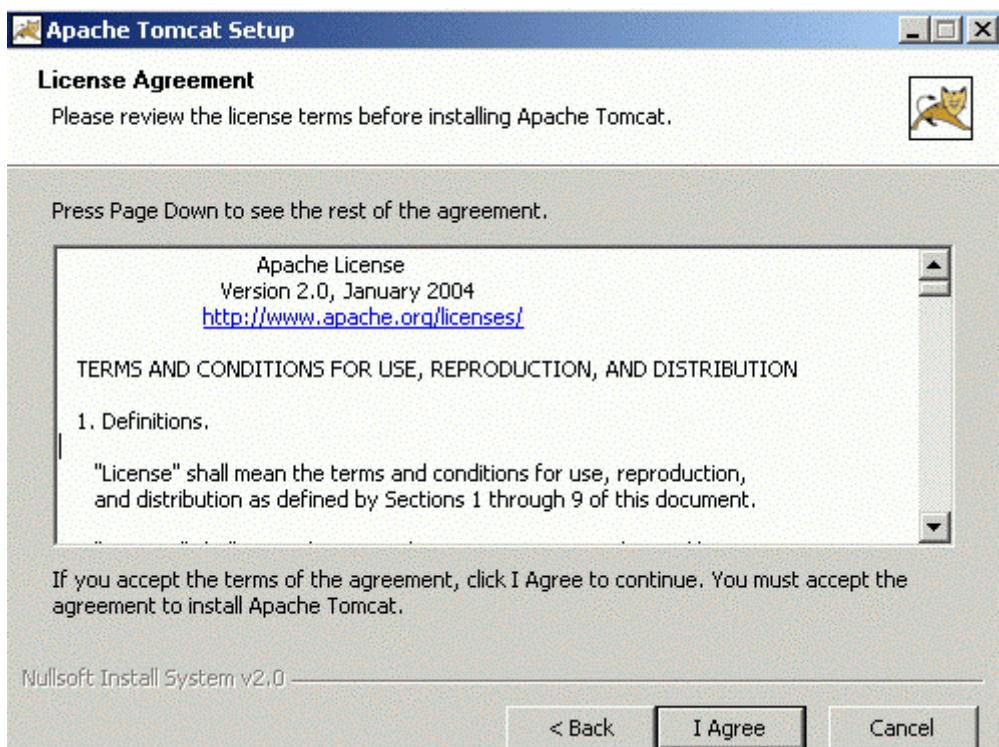
L'installation de Tomcat se fait en exécutant le fichier *jakarta-tomcat-5.0.25.exe*. Celui-ci contient un assistant d'installation qui guide l'utilisateur.

Figure 2 – Tomcat 5.0, accueil de l'utilitaire d'installation



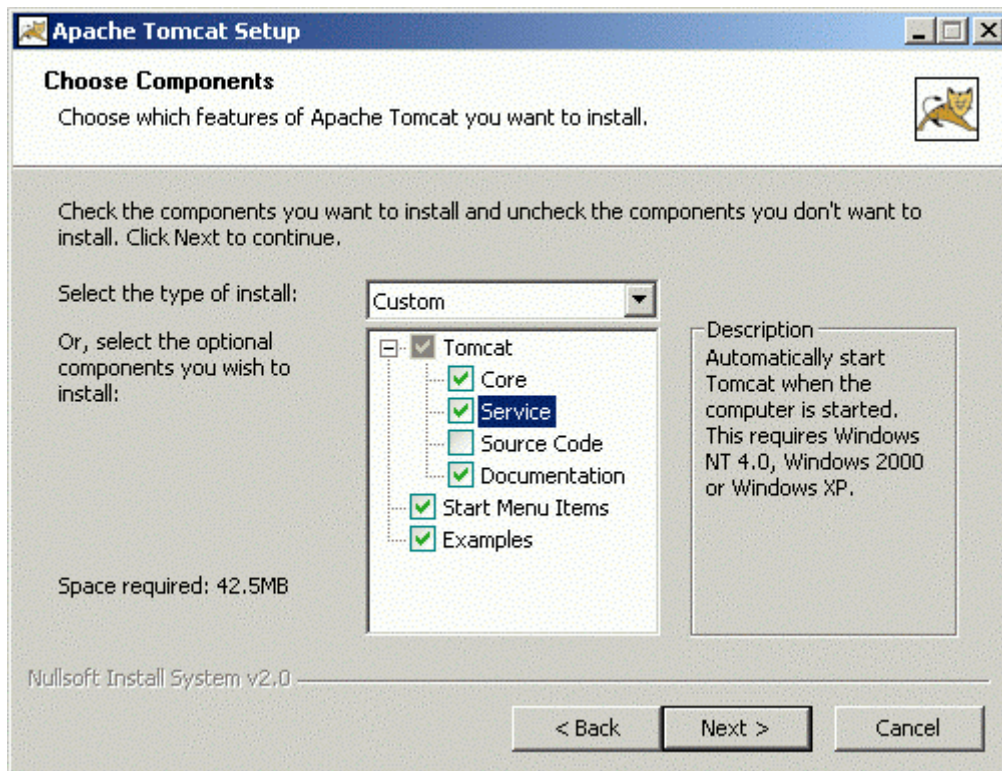
H. BUISSART, 2004, COPIE D'ECRAN : APACHE TOMCAT SETUP

Figure 3 – Tomcat 5.0, acceptation de la licence Apache



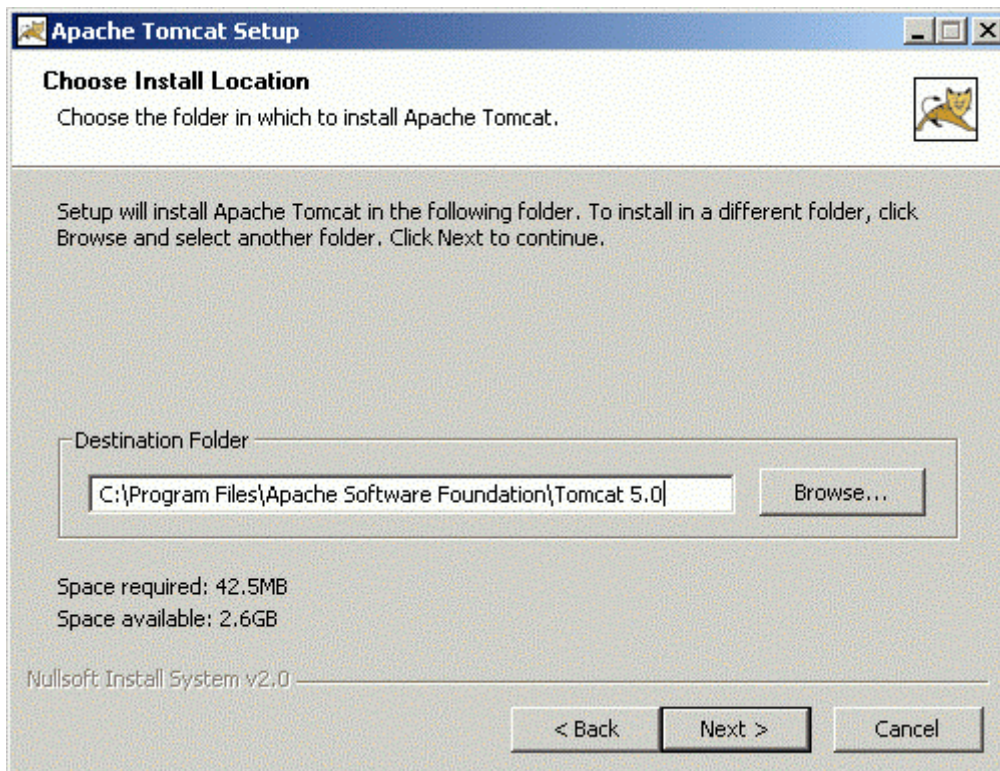
H. BUISSART, 2004, COPIE D'ECRAN : APACHE TOMCAT SETUP

Figure 4 – Tomcat 5.0, configuration personnalisée de l'installation



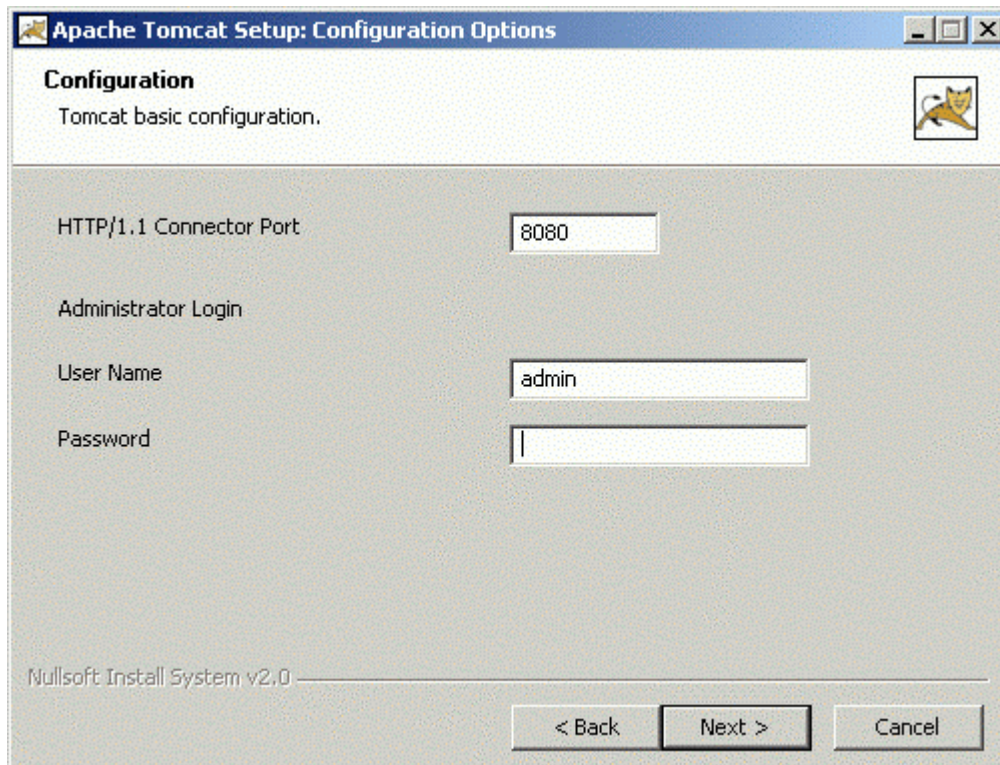
H. BUISSART, 2004, COPIE D'ECRAN : APACHE TOMCAT SETUP

Figure 5 – Tomcat 5.0, répertoire d'installation de Tomcat



H. BUISSART, 2004, COPIE D'ECRAN : APACHE TOMCAT SETUP

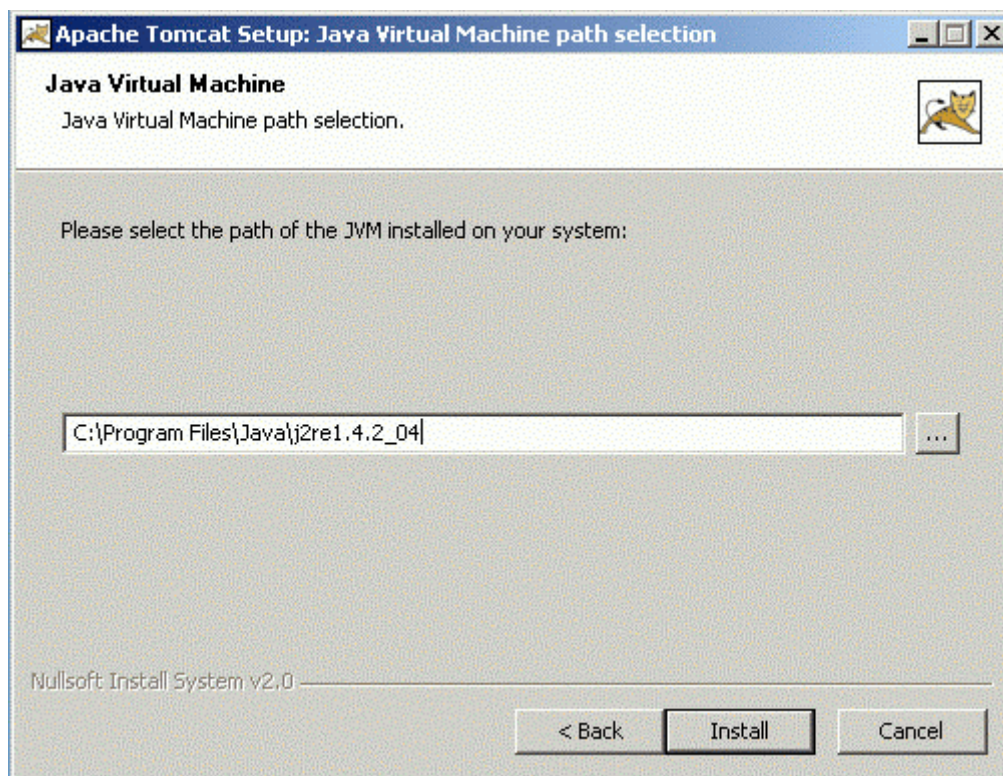
Figure 6 – Tomcat 5.0, configuration des paramètres d'administration



H. BUISSART, 2004, COPIE D'ECRAN : APACHE TOMCAT SETUP

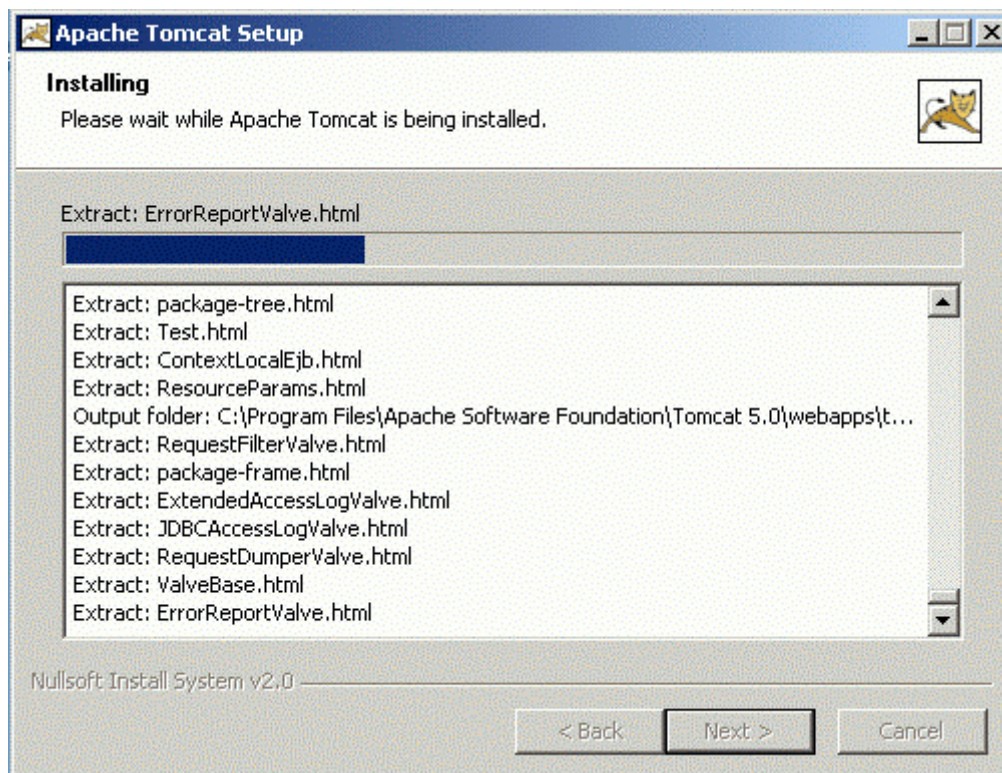
Tomcat s'appuie sur la machine virtuelle Java (JRE) installée précédemment.

Figure 7 – Tomcat 5.0, chemin d'accès à la machine virtuelle Java



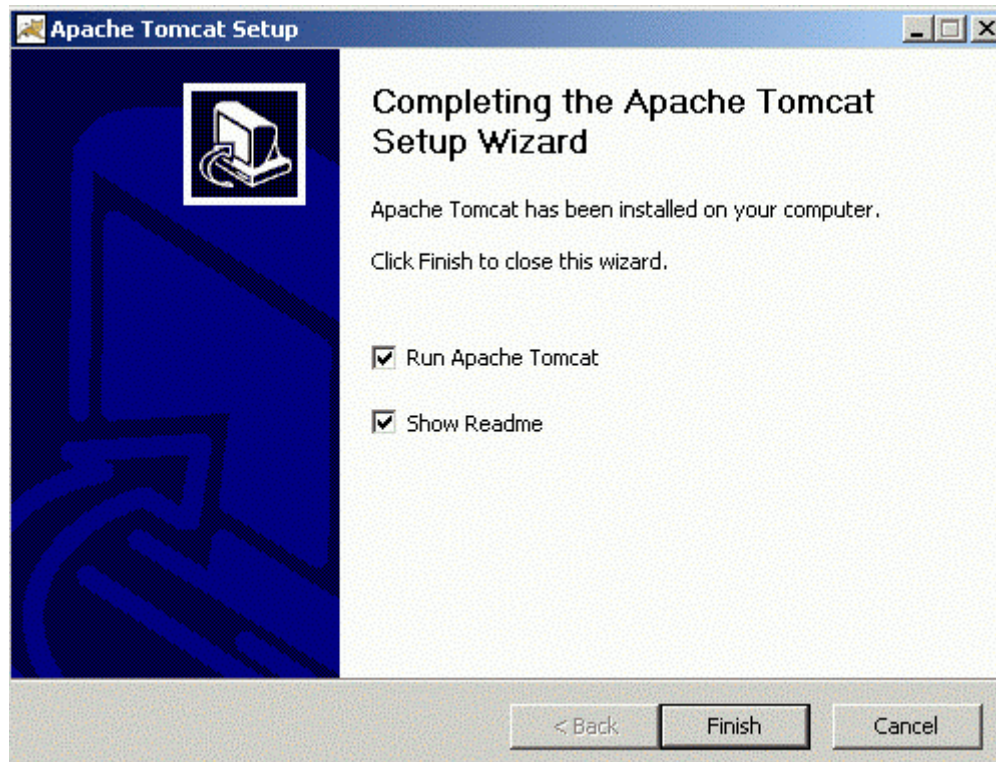
H. BUISSART, 2004, COPIE D'ECRAN : APACHE TOMCAT SETUP

Figure 8 – Tomcat 5.0, installation en cours



H. BUISSART, 2004, COPIE D'ECRAN : APACHE TOMCAT SETUP

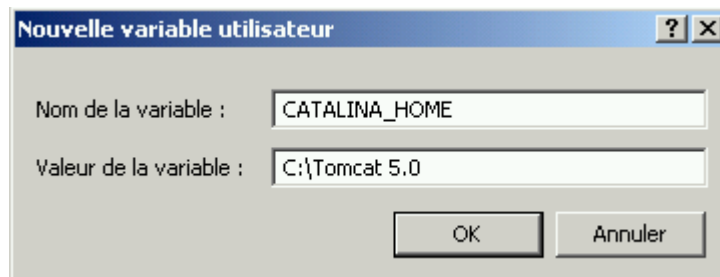
Figure 9 – Tomcat 5.0, lancement de Tomcat



H. BUISSART, 2004, COPIE D'ECRAN : APACHE TOMCAT SETUP

Il est nécessaire de créer une nouvelle variable d'environnement appelée CATALINA_HOME et pointant vers le répertoire racine de Tomcat. Pour cela, dans la zone "Système" du "Panneau de configuration", dans l'onglet "Avancé", on choisit la rubrique "Variables d'environnement...", il s'agit ensuite de créer cette nouvelle variable comme expliqué au paragraphe précédent :

Figure 10 – Tomcat 5.0, Variable d'environnement



H. BUISSART, 2004, COPIE D'ECRAN : VARIABLE UTILISATEUR WINDOWS

Les fichiers utilisés pour notre projet seront stockés dans le répertoire webapps de Tomcat au sein du dossier alovmap. Leur chemin est donc CATALINA_HOME\webapps\alovmap\.

3 – Configuration d'ALOV Map

3.1 – Les fichiers utilisés par ALOV Map

ALOV, en configuration client-serveur, s'appuie sur un ensemble de fichiers à intégrer au serveur Tomcat précédemment installé. La dernière version de l'outil peut être téléchargée sur le site d'ALOV Map sur Internet à l'adresse : <http://www.alov.org>, rubrique "Download".

Il est possible sur le site d'ALOV Map de télécharger la distribution complète contenant les archives Java, les fichiers de configuration ainsi que des exemples de données et de paramétrages ou de télécharger uniquement les dernières versions des archives Java. La structure des fichiers utilisée dans la distribution complète a été reprise pour le projet et est exposée ci-après.

Les fichiers JAVA sont distribués sous la forme d'archives JAR (Java Archive) dans lesquelles sont regroupées les fichiers .class utiles à l'application :

- `alov_applet.jar` : il est appelé par le fichier de lancement de l'applet (voir la partie II). Ce fichier est stocké au même niveau que le fichier de lancement de l'applet, sous `CATALINA_HOME\webapps\alovmap\`,
- `alov_servlet.jar` : il s'agit de l'application gérant l'outil d'administration d'ALOV Map. Ce fichier est stocké dans `CATALINA_HOME\webapps\alovmap\WEB-INF\lib`.

Un fichier XML est utilisé par ALOV pour regrouper les informations sur la connexion à l'outil d'administration et à la base de données. Il s'agit du fichier `mapserv.xml` stocké dans `CATALINA_HOME\webapps\alovmap\WEB-INF\mapserv-home`.

Voici l'exemple du fichier paramétrant la connexion à une base de données MS Access.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<mapserv>
  <clearinghouse type="db">
    <xml url="" />
    <database user="" password="" url="jdbc:odbc:alovbase"
driver="sun.jdbc.odbc.JdbcOdbcDriver" server="MS Access" />
  </clearinghouse>
  <logger>
    <destination type="console"
levels="information,warningExternal,warningInternal,dbFailure,serverFailure" />
  </logger>
  <master user="1" password="1" />
</mapserv>
```

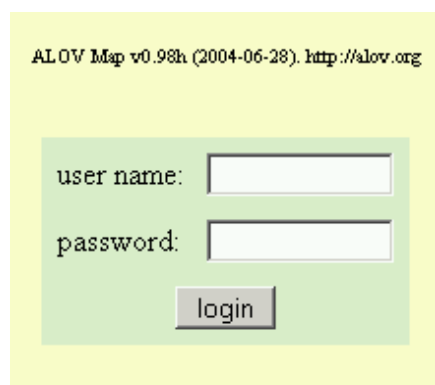
La balise `<clearinghouse>` est mise à jour automatiquement par l'outil d'administration d'ALOV Map. Les informations contenues dans la balise `<master>` sont le nom d'utilisateur et le mot de passe utilisés pour entrer dans l'outil d'administration.

3.2 – Accès à l'outil d'administration

Une fois la distribution d'ALOV chargée, il est nécessaire de structurer ces données dans notre site. Il convient donc de créer un répertoire que nous appellerons `alovmap` dans le dossier `webapps` situé sur la racine du serveur. Ce répertoire sera la racine de notre site : `CATALINA_HOME\webapps\alovmap\`.

La configuration d'ALOV se fait au moyen de l'outil d'administration des données proposé par ALOV. Ce dernier est accessible localement grâce à Tomcat à l'adresse suivante : <http://localhost:8080/alovmap/pump>.

Figure 11 – Interface de connexion à l'outil d'administration d'ALOV Map



Les principaux paramètres de configuration sont stockés dans *mapserv.xml*. Les paramètres de connexion à l'utilitaire lui-même sont contenus dans la balise `<master>`. Par défaut, le compte (*username*) et le mot de passe (*password*) d'accès à l'utilitaire sont fixés à "1". Leur modification doit être faite manuellement directement dans *mapserv.xml*.

3.3 – Connexion ALOV Map – SGBD

Une fois ALOV Map installé, il faut spécifier l'emplacement de la base de données utilisée et lui spécifier les données utiles à la connexion de la servlet avec cette base.

L'outil ALOV Map peut fonctionner avec la plupart des Systèmes de Gestion de Bases de Données (SGBD) du marché, et ce grâce à l'utilisation des connexions JDBC par la servlet d'ALOV. En effet, cette API Java permet à ALOV de se connecter au SGBD choisi dès lors qu'est fourni au serveur le pilote nécessaire à la connexion. Cette connexion n'est donc pas spécifique du SGBD choisi. Ainsi sa programmation est indépendante du SGBD et de la plate-forme utilisés.

ALOV Map a été mis en œuvre sur deux Systèmes de Gestion de Bases de Données :

- Microsoft Access 2000,
- MySQL.

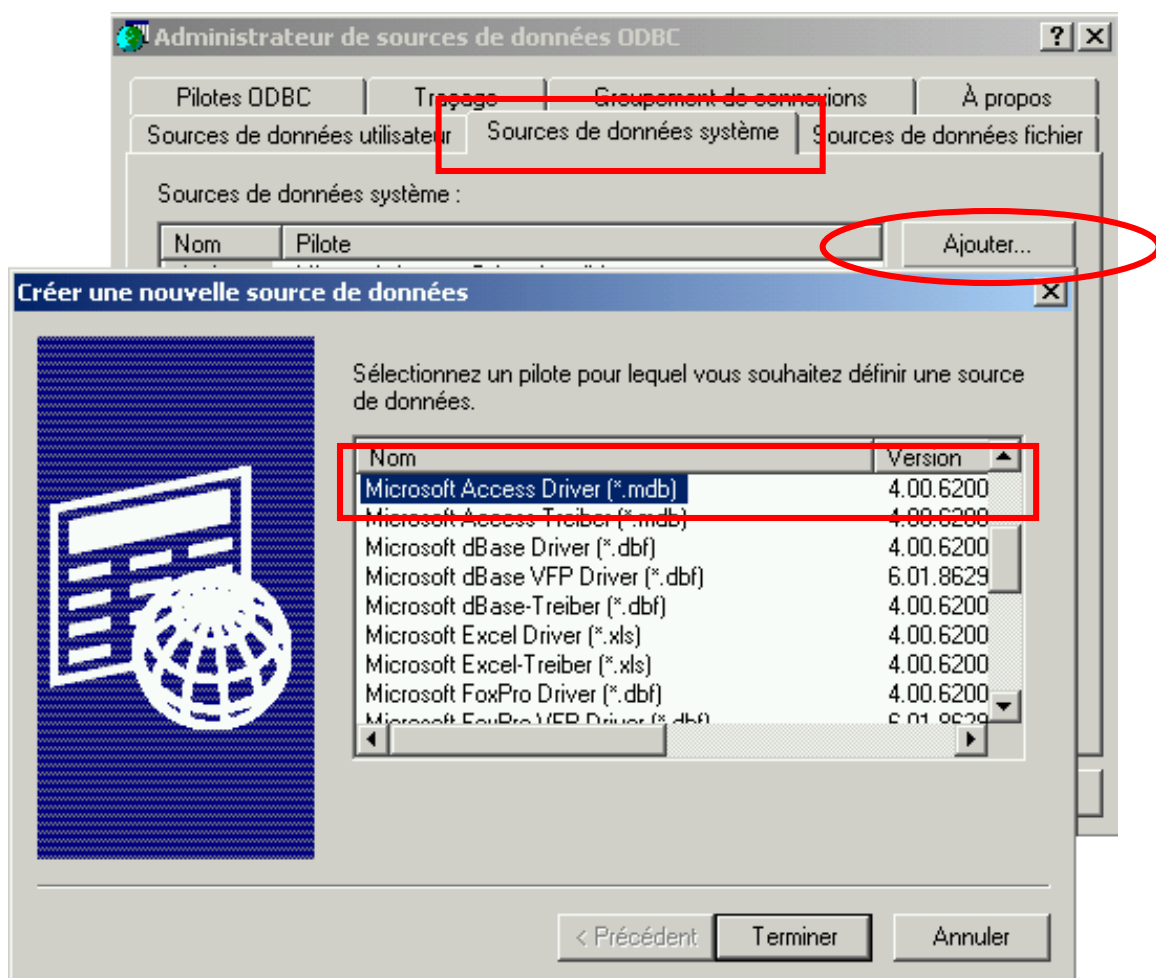
3.3.1 – Création de la liaison ODBC avec MS Access 2000

Pour lier la servlet Java d'ALOV Map à la base de données Access, il faut créer avec cette dernière une liaison ODBC. Celle-ci permettra via un pilote JDBC de type 1 de traduire les

appels JDBC de la servlet en appels ODBC qui seront exécutés par Access : il s'agit d'une "passerelle JDBC-ODBC".

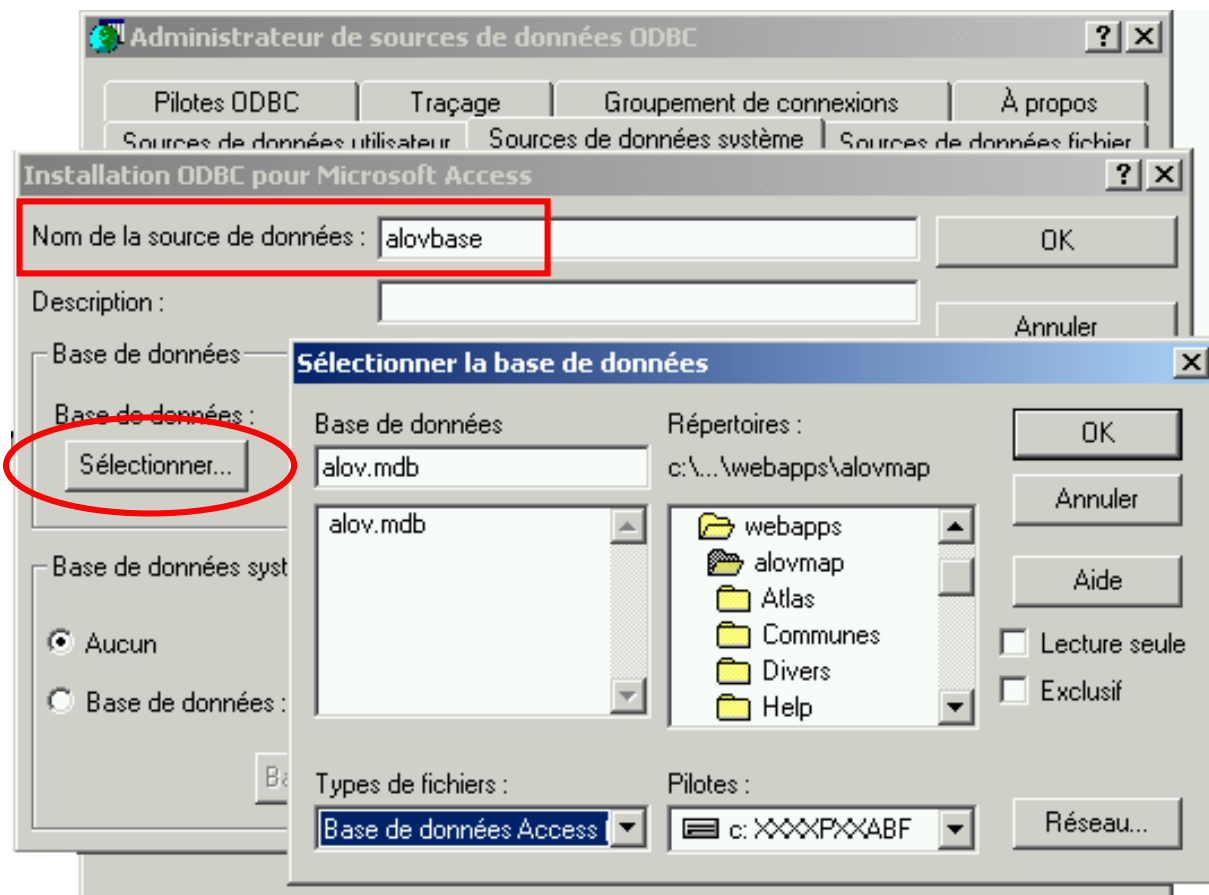
Sous Windows 2000, on ouvre l'outil *Sources de données (ODBC)* dans les *Outils d'administration* du *Panneau de Configuration*. On crée une "source de données système" en cliquant sur "Ajouter...". On peut alors choisir le pilote à utiliser pour la connexion avec le SGBD ; ici il s'agit de "Microsoft Access Driver (*.mdb) v.4.00.6200.00".

Figure 12 – Liaison ODBC : choix du pilote



Après avoir donné un nom à la source de données (DSN : Data Source Name) : alovbase, cliquer sur "Sélectionner" pour spécifier la base de données concernée par la connexion.

Figure 13 – Liaison ODBC : paramétrage de la connexion



La validation de ces choix conduit à la création du DSN. Le DSN, ici alovbase, sera utilisé par ALOV pour retrouver la base de données liée.

3.3.2 – Connexion ALOV Map – MySQL

La liaison à MySQL se fait au moyen du pilote MySQL Connector/J dont la dernière version est disponible gratuitement au téléchargement sur le site Internet de MySQL à l'adresse suivante : <http://www.mysql.com/products/connector/j/>. Ce pilote : *mysql-connector-java-3.0.14-production-bin.jar*, qui se présente sous la forme d'une archive JAVA doit être placé dans le répertoire *CATALINA_HOME\webapps\alovmap\WEB-INF\lib*. Il s'agit d'un pilote de type 4, écrit en Java et qui convertit les appels JDBC dans le protocole réseau utilisé par MySQL. La classe permettant le lancement du pilote est *org\gjt\mm\mysql\Driver.class*. La base de données utilisée est appelée alovmysql.

3.4 – Liaison d'ALOV Map à la base de données

Dans l'outil d'administration, l'onglet "Location" permet de définir les paramètres de la liaison avec le SGBD. Ces paramètres sont stockés dans le fichier *mapserv.xml* dans la balise <clearinghouse>.

Figure 14 – Configuration de la liaison entre ALOV et la base de données source

ALOV Map v0.98h (2004-06-28). <http://alov.org>

Setup

[Location](#) [JDBC drivers](#) [Build XML MasterBase](#) [New MasterBase](#)

Maintenance

[Search](#) | [Upload shapefile](#)

Register: [Vector Dataset](#) | [Raster Dataset](#) | [MrSID](#) | [WMS](#) | [WFS](#) | [Project](#)

Changes were applied successfully

Define the location of XML Master database:

XML file

Define the connection for Master database:

SQL server

JDBC driver class

Database URL

User name

Password

Type of Master database in use:

Type

Pour établir la connexion avec le SGBD, il faut renseigner la section "Define the connection for Master database" comme présenté sur la figure 14 :

- SQL server : il s'agit du SGBD utilisé ; il est possible d'utiliser un SGBD non proposé dans la liste par défaut sous réserve de disposer de son pilote JDBC et de renseigner la table *tm_servers* de la base de données connectée (voir au paragraphe IV-1).

- JDBC driver class : il s'agit de la classe Java servant à lancer le pilote JDBC. Les différents niveaux d'arborescence de l'archive sont séparés par des ".". La liste des pilotes JDBC disponibles est paramétrables dans l'onglet "*JDBC drivers*".

Figure 15 – Définition des pilotes JDBC associés aux SGBD

ALOV Map v0.98h (2004-06-28). <http://alov.org>

Setup

[Location](#) [JDBC drivers](#) [Build XML MasterBase](#) [New MasterBase](#)

Maintenance

[Search](#) | [Upload shapefile](#)

Register: [Vector Dataset](#) | [Raster Dataset](#) | [MrSID](#) | [WMS](#) | [WFS](#) | [Project](#)

JDBC drivers:

Interbase	<input type="text" value="interbase.interclient.Driver"/>
MS Access	<input type="text" value="sun.jdbc.odbc.JdbcOdbcDriver"/>
Sybase	<input type="text"/>
MySQL	<input type="text" value="org.gjt.mm.mysql.Driver"/>
Informix	<input type="text"/>
Hypersonic	<input type="text" value="org.hsqldb.jdbcDriver"/>
DB2	<input type="text"/>
Oracle	<input type="text"/>
MS SQL	<input type="text"/>

H. BUISSART, 2004, COPIE D'ECRAN : ALOV MAP V0.98H

- Database URL : il s'agit de l'URL permettant la connexion à la base
- User name et Password sont les comptes et mots de passe permettant d'accéder aux données. Il faut vérifier que le compte utilisé dispose des privilèges suffisants pour réaliser les opérations souhaitées.

Tous ces paramètres correspondent aux paramètres nécessaires à la connexion à une base de données avec JDBC utilisant les méthodes et objets du package java.sql.*.

Dans le cas d'utilisation d'une base de données, la section "*Type of Master database in use*" contient "*SQL database*".

3.4.1 – Connexion ALOVMap - MS Access 2000

Pour établir la connexion avec MS Access, on affectera aux paramètres les valeurs suivantes :

- SQL Server : MS Access,
- JDBC Driver Class : sun.jdbc.odbc.JdbcOdbcDriver,
- Database URL : jdbc:odbc:DSN, soit ici : jdbc:odbc:alovbase,

3.4.2 – Connexion ALOVMap – MySQL

Pour établir la connexion avec MySQL, on affectera aux paramètres les valeurs suivantes :

- SQL Server : MySQL,
- JDBC Driver Class : org.gjt.mm.mysql.Driver,
- Database URL : jdbc:mysql://localhost/alovbase où alovbase est le nom de la base de données MySQL à connecter,
- on définira les comptes et mots de passe le cas échéant.

II – PARAMETRAGE DE L'APPLET ALOV MAP

1 – La balise <applet>

L'applet est lancée au sein du fichier HTML qui la contient. Elle est intégrée dans la balise <body> de la page qui l'appelle et est définie dans une balise <applet> qui contient plusieurs attributs. Les paramètres de l'applet sont définis au sein de balises <param> dont on utilise deux attributs.

Tableau 1 – Récapitulatif des éléments et attributs utilisés pour lancer une applet

ATTRIBUT	TYPE	DESCRIPTION
Element : applet		
name	chaîne	Nom de l'applet
codebase	chaîne	Répertoire contenant l'applet. Si l'applet est dans le répertoire racine du site, le chemin est "."
archive	chaîne	Fichier compressé dans lequel aller chercher l'applet avec l'attribut code
code	chaîne	Chemin d'accès à la classe de l'applet lancée dans la page. Les différents niveaux du chemin d'accès sont séparés par des "."
width	entier, %	Largeur de l'applet dans la fenêtre du navigateur client
height	entier, %	Hauteur de l'applet dans la fenêtre du navigateur client
align	right left center	Position relative de l'applet par rapport à la fenêtre
Element: param		
name	chaîne	Nom du paramètre. Plusieurs paramètres pour l'applet ALOV Map : pid layout lang prepage
value	chaîne	Valeur du paramètre

2 – Les paramètres de l'applet ALOV Map

Plusieurs paramètres sont nécessaires au lancement d'ALOV Map et à sa personnalisation.

```
<param name=pid value="DGEAF.xml">
```

`pid` définit le chemin d'accès au fichier "*project*", décrit en détail au chapitre V. Il s'agit du seul paramètre réellement indispensable au fonctionnement de l'applet.

```
<param name=layout value="layout_fr.xml">
```

`layout` définit le chemin d'accès au fichier de mise en page "*layout*". Il permet notamment de personnaliser les messages en fonction du paramètre `lang`.

```
<param name=lang value="fr">
```

`lang` définit la langue à utiliser dans le fichier "*layout*".

```
<param name=prepage value="load_first.html">
```

`prepage` définit la page à ouvrir en pré-chargement, avant d'accéder aux pages liées.

III – IMPORT ET MISE A JOUR DES DONNEES

L'outil d'administration d'ALOV Map permet d'organiser les données utilisées dans une base de données. ALOV est capable de traiter plusieurs types de données:

- les vecteurs sous la forme de:
 - o couple de fichiers SHAPE et DBase IV,
 - o couple de fichiers MIF (Mapinfo Interchange Format) et DBase IV,
 - o tables dans une base de données SQL,
- les rasters : sous la forme de fichiers JPEG ou GIF.

Après étude de ces différents formats, nous utiliserons les fichiers SHP + DBF pour les données vecteurs et des fichiers GIF pour les données raster. Exceptionnellement, on pourra être amenés à traiter des données contenues dans des tables SQL.

1 – Import des données vecteur

L'outil d'administration d'ALOV permet l'import et l'enregistrement des données vecteurs dans une base SQL.

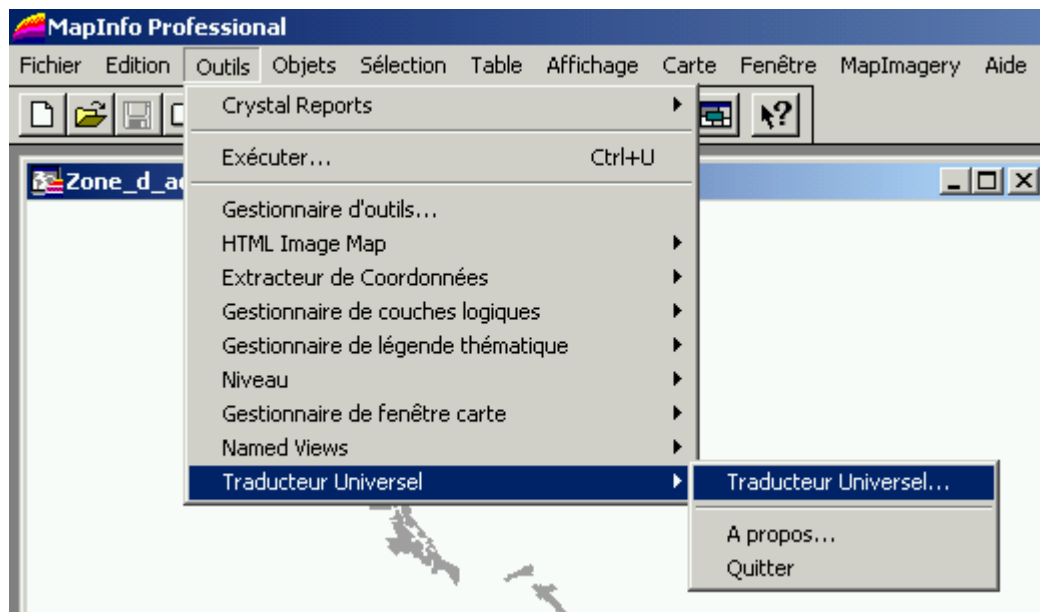
1.1 – Export à partir de MapInfo

Les données vecteurs utilisées à la Direction Départementale de l'Agriculture et de la Forêt de la Savoie sont initialement au format TAB, propriétaire de MapInfo. Il s'agit donc de les convertir en fichiers qu'ALOV est capable de traiter.

L'outil *Traducteur Universel* de MapInfo (MUT.MBX) permet de convertir différents formats de cartes et notamment des fichiers TAB en fichiers SHP.

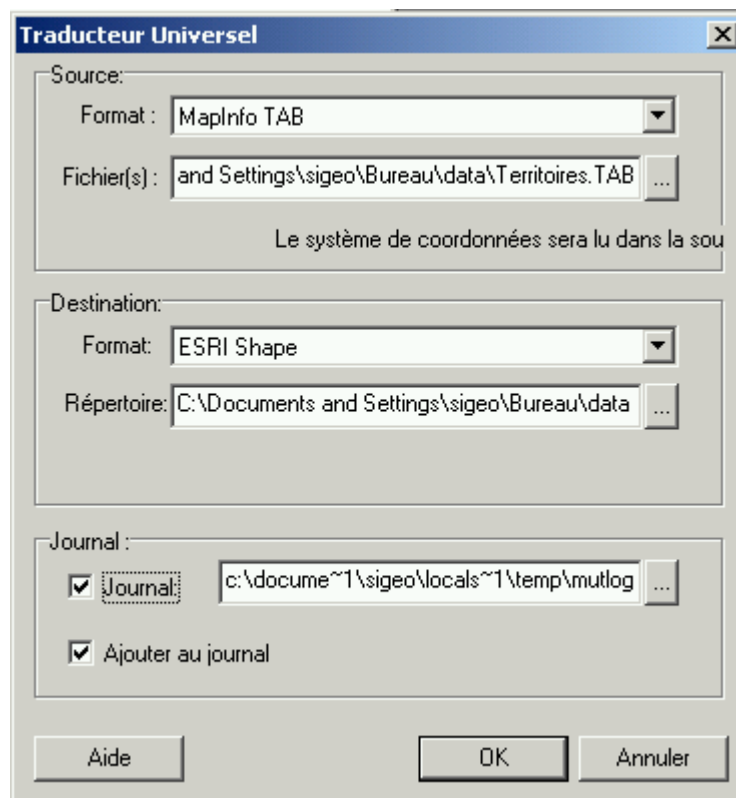
Il convient de s'assurer que la projection du fichier TAB est celle désirée avant d'effectuer la conversion. En effet, l'ensemble des fichiers chargés dans ALOV doivent avoir la même projection et les mêmes unités.

Figure 16 – MapInfo, lancement du Traducteur Universel



Il convient de spécifier via l'interface du Traducteur Universel le format de départ, les fichiers à convertir, le format d'arrivée ainsi que le répertoire de stockage des fichiers transformés.

Figure 17 – MapInfo, paramétrage du Traducteur Universel



Au cours de cette transformation, plusieurs fichiers accompagnent le fichier SHP (*.dbf, *.prj, *.shp, *.shx) : seuls les fichiers *.dbf et *.shp nous intéressent pour la suite.

1.2 – Import dans ALOV Map

Une fois la traduction effectuée, il est possible d'importer les nouveaux fichiers Shape et Dbf dans ALOV via l'outil d'administration à l'adresse <http://localhost:8080/alovmap/pump>.

Figure 18 – Interface d'enregistrement des fichiers vecteur sous ALOV

ALO V Map v0.98b (2004-05-20). <http://alov.org>

Setup

[Location](#) [JDBC drivers](#) [Build XML MasterBase](#) [New MasterBase](#)

Maintenance

[Search](#) | [Upload shapefile](#)

Register: [Vector Dataset](#) | [Raster Dataset](#) | [MrSID](#) | [WMS](#) | [WFS](#) | [Project](#)

Define the source of data:

Browse for Shape or MIF file and associated dBase file.

Shapefile/Mif

Dbf

OR, IF data are ALREADY on server define the absolute path to files

Shapefile/Mif

Dbf

Dbf encoding (leave empty to use default)

Destination SQL database connection:

Server type

Database URL

User name

Password

Database encoding (leave empty to use default)

Precision (for polylines and polygons)

Table info:

Table name

Unique key field (Optional)

List of fields to be uploaded (All fields are in process by default)

Définition de la source de données – Define the source of data :

Il convient de renseigner les chemins d'accès aux fichiers Shape et Dbf à l'aide des boutons *Browse...*

Connexion à la base de données SQL destinatrice – Destination SQL database connection :

Il faut aussi renseigner la base de données de destination. Il s'agit de la base pour laquelle nous avons créé la liaison précédemment (voir paragraphe I-3-4).

Server type est le SGBD utilisé, ici MS Access ou MySQL.

Database URL est le chemin d'accès à la base en empruntant la liaison créée. Il est de la forme : protocole:sous-protocole:connexion, ici jdbc:odbc:alovbase pour MS Access ou jdbc:mysql://localhost/alovbase pour MySQL.

User name, *Password* sont à renseigner si la base de données est protégée.

Database encoding n'est pas renseigné : on utilise l'encodage des caractères par défaut.

Precision peut être modifié pour les polygones et polygones.

Informations sur la table à créer – Table info :

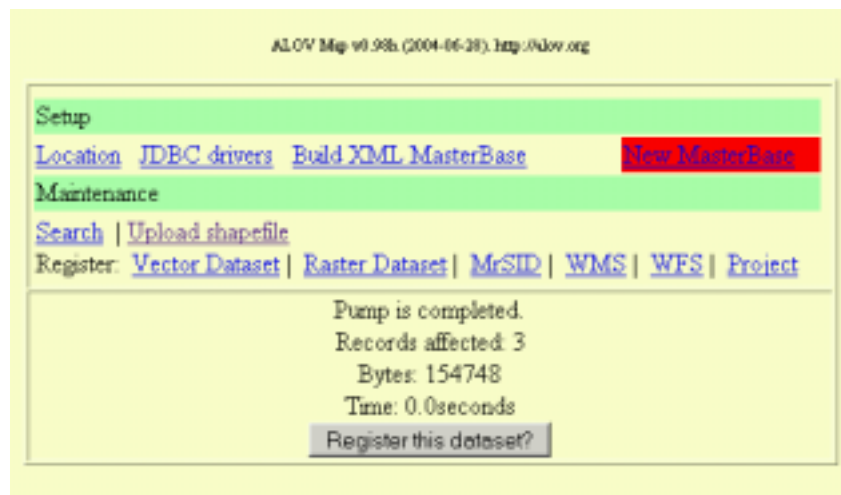
Il s'agit d'indiquer le nom de la table qui sera créée.

Si une clé primaire existe dans la table importée, il est possible de la garder en indiquant le nom du champ concerné dans *Unique key field*. Dans le cas contraire, un nouveau champ sera créé, intitulé NOMTABLE_ID.

List of fields to be uploaded est utilisé pour ne charger qu'une partie des champs de la table traitée. Par défaut, tous sont importés.

Lorsque le traitement est terminé, un résumé du transfert est affiché. Une table a été créée dans la base de données liée.

Figure 19 – Fenêtre d'état après un import correct



1.3 – Enregistrement du jeu de données

Une fois l'import réalisé, il s'agit d'enregistrer le jeu de données grâce à l'option "*Register this dataset?*". Certains champs du formulaire sont déjà renseignés à partir du traitement des données importées.

Figure 20 – Enregistrement des données vecteur sous ALOV (1/3)

Dataset description:	
Title *	NOM_TABLE
Description	
Themes	Administrative geography Archaeology Art Biography Built environment
Author	
E-mail	
Rights statement	
Authorization Password	
Documentation Web Page	
Spatial extent:	
Longitude (X) min	856239.94
Longitude (X) max	979490.06
Latitude (Y) min	2014163.0
Latitude (Y) max	2110165.0

Description du jeu de données – *Dataset description* :

Il s'agit de fournir des métadonnées sur la table importée. Celles-ci pourront servir pour rechercher la table, notamment les *Themes* représentatifs.

Emprise spatiale – *Spatial extent* :

Il s'agit du rectangle minimum englobant l'ensemble des données chargées. Ces coordonnées sont renseignées automatiquement si la table résulte d'une procédure d'import de vecteur.

Définition de la connexion à la base de données SQL ou emplacement du fichier Shape – *Define SQL database connection or Shapefile location* :

Ces champs reprennent les données renseignées lors de l'import du couple de fichiers.

Informations sur la table – *Table info* :

Ces données concernent la table créée dans la base de données avec en premier lieu son nom et sa clé primaire. Viennent ensuite les données spécifiques à ALOV.

Description field est le champ utilisé pour réaliser les requêtes attributaires et celui qui s'affiche dans les étiquettes.

List of fields to be downloaded est l'ensemble des champs à télécharger, séparés par une virgule. Pour pouvoir réaliser une analyse thématique sur un champ de la table, celui-ci doit avoir été préalablement téléchargé. On peut restreindre le nombre de champs visibles par l'utilisateur en renseignant *List of fields visible on client*.

Toute une partie de cette rubrique est consacrée à l'utilisation de données géométriques existantes dans une table SQL. Cet outil, facile à utiliser dans le cas d'objets ponctuels, est beaucoup plus complexe à mettre en œuvre pour des objets linéaires ou surfaciques. En effet, pour les objets ponctuels, il suffit d'indiquer les champs de la table représentant les coordonnées en X "*X coordinate field*" et en Y "*Y coordinate field*". Dans ce cas, nul besoin de réaliser toute l'opération de conversion des données précédemment décrite.

Type of objects permet de désigner le type d'objets chargés dans la table. Chaque table ne peut contenir qu'un seul type d'objet. Ils peuvent être ponctuels (*Points*), linéaires (*Lines*) ou surfaciques (*Polygons*). Il peut être intéressant de renseigner ce champ dans le cas par

exemple de polygones sur lesquels on voudrait appliquer une sémiologie considérée comme linéaire par ALOV (trait gras, pointillés...).

Web link masks permet de définir les liens auxquels il sera possible d'accéder via les objets de la table.

Figure 21 – Enregistrement des données vecteur sous ALOV (2/3)

Define SQL database connection or Shapefile location:	
Server type *	MS Access
Database URL or Shapefile URL *	jdbc:odbc:ALOVBASE
User name	<input type="text"/> Use readonly user+password
Password	<input type="password"/>
Table info:	
Table name *	NOM_TABLE
Encoding	<input type="text"/>
Unique key field *	NOM_TABLE_ID
Description field	<input type="text"/>
List of fields to be downloaded	NOM,NUMERO,
List of visible fields on client	<input type="text"/>
Filter (conditional expression) (Applicable for SQL datasets only)	<input type="text"/>
<p>If you utilized the upload routine to pump your shapefile or mif to SQL database you have to leave the next four fields empty. If you have table that contains coordinate fields define them. The point features may be in the main table. The lines and polygons coordinates must be in the separate table. This table has to relate with main table via field that name equals to unique key field of main table.</p>	
X coordinate field	<input type="text"/>
Y coordinate field	<input type="text"/>
Coordinate Table name	<input type="text"/>
Vertex order field	<input type="text"/>
Type of objects	<input type="text"/>
Web Link masks:	
Main link mask	<input type="text"/>
Search link mask	<input type="text"/>
Info link mask	<input type="text"/>
Default symbology:	

Une fois enregistré, le jeu de données se voit attribué un identifiant unique qui servira de référence pour le désigner dans le fichier "*project*".

Figure 22 – Enregistrement des données vecteur sous ALOV (3/3)

ALOV Map v0.98h (2004-06-28). <http://alov.org>

Setup

[Location](#) | [JDBC drivers](#) | [Build XML MasterBase](#) | [New MasterBase](#)

Maintenance

[Search](#) | [Upload shapefile](#)

Register: [Vector Dataset](#) | [Raster Dataset](#) | [MrSID](#) | [WMS](#) | [WFS](#) | [Project](#)

Dataset was registered. ID #125

[Map This Dataset](#)

2 – Les données raster

Outre les données "natives" rasters, il est possible et même préférable d'avoir recours aux rasters pour permettre une visualisation rapide de l'information à des niveaux de zoom faibles dès lors que les vecteurs correspondant sont trop lourds à charger et n'apportent aucune information à ces échelles. Il s'agira alors de créer un fichier raster à partir des vecteurs, grâce aux outils de traitement disponibles sur MapInfo.

2.1 – Paramètres nécessaires au calage sous ALOV Map

La description du jeu de données et de l'emprise spatiale est identique pour les rasters et les vecteurs.

Figure 23 – Enregistrement des données rasters sous ALOV

Define raster image file location and registration values:

Image file URL *

User name HTTP user+password

Password

Image registration values:

X-coordinate of the center of the upper left pixel

Y-coordinate of the center of the upper left pixel

Dimension of a pixel in map units in the x-direction

Dimension of a pixel in map units in the y-direction

Web Link masks:

URL

[Register](#)

Définition de l'emplacement du fichier image – Define raster image file location and registration values :

Il s'agit d'indiquer le chemin relatif permettant d'accéder à l'image raster traitée.

Géoréférencement de l'image – Image registration values :

Les données utilisées par ALOV pour caler les images rasters sont les coordonnées terrain en abscisse et en ordonnée du centre du 1^{er} pixel de l'image (en haut à gauche) ainsi que la taille terrain d'un pixel dans les deux directions, exprimées en unités de la carte (`mapunits` du fichier "*project*").

Une méthode de calage des rasters dans ALOV Map, exposée dans le livret intitulé "Etude de faisabilité : Utilisation de rasters dans ALOV Map", permet d'obtenir ces renseignements. La procédure à suivre est ici rappelée brièvement.

2.2 – Méthode de rasterisation et de calage à partir de MapInfo

Ces renseignements de coordonnées du pixel référence et de la largeur et hauteur d'un pixel est à renseigner au moment de l'enregistrement du jeu de données raster dans l'outil d'administration d'ALOV Map.

Sous MapInfo, cadrer la carte sur la zone que l'on cherche à exporter. Construire dans la couche dessin deux droites : une verticale (à gauche de l'export souhaité) et une horizontale (en haut de l'export souhaité), s'intersectant. Ces notions de verticalité et d'horizontalité sont prises d'un point de vue "géométrique" dans MapInfo et obtenues en maintenant la touche SHIFT enfoncée lors de la création de ces droites.

Exporter l'image au format TIF (Tagged Image File Format) par le menu Fichier>Exporter Fenêtre...à l'emplacement souhaité. Choisir une taille personnalisée pour l'export et spécifier la résolution (200 dpi). Garder les options par défaut et valider.

Le pixel d'intersection des droites donnera les coordonnées du pixel de référence une fois l'image recadrée. Ce recadrage se fait sous un utilitaire de traitement d'images classique. Un logiciel comme IrfanView suffit. Il s'agit de recadrer l'image en s'appuyant sur ce pixel référence puis en suivant les droites de calage (fonction Edit>Crop Selection (CTRL+Y) pour recadrer sur la zone sélectionnée sous Irfan View). Attention : ce recadrage (rectangulaire) ne suit pas scrupuleusement les droites de calage. La connaissance de la longueur de ces droites et de la taille en pixels de l'image recadrée donne une approximation de la taille

moyenne d'un pixel en hauteur et en largeur. Il convient ensuite de convertir l'image dans un format qu'ALOV peut lire : le format GIF est préférable.

3 – Mise à jour des données

Les mises à jour de données attributaires ne concernent que les données vecteurs ; en effet, une mise à jour de raster correspond à l'enregistrement d'une nouvelle image raster. Elles peuvent être faites directement dans le SGBD MS Access sans avoir à réimporter la géométrie des couches. Deux cas se présentent alors :

- la mise à jour de données attributaires dans une table : il s'agit de remplacer les données attributaires d'un ou plusieurs champs de la table par leurs nouvelles données. Pour cela, la nouvelle table attributaire doit être exportée vers le SGBD utilisé et une requête simple permet de remplacer les données obsolètes dans la base utilisée par ALOV par ces nouvelles données.
- l'import d'un nouveau champ de la table : après création d'un nouveau champ dans la table MS Access, on procède comme pour une mise à jour. Il faudra une fois cette mise à jour réalisée indiquer à ALOV qu'un nouveau champ existe et qu'on souhaite l'utiliser dans l'outil d'administration des données.

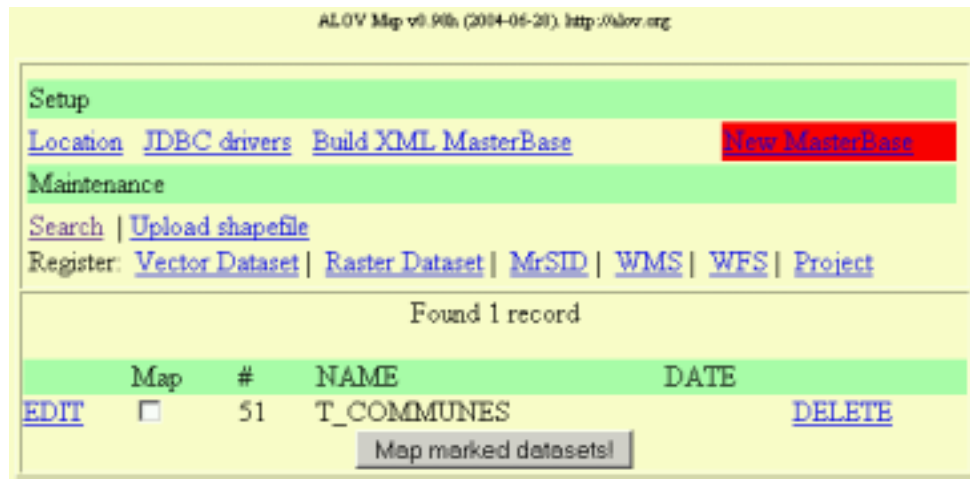
Pour mettre à jour les propriétés d'une table sous ALOV, il convient dans un premier temps de lancer une recherche sur le nom du jeu de données via l'outil "Search".

Figure 24 – Recherche d'un jeu de données par ALOV

The screenshot shows the ALOV Map v0.98b (2004-05-26) web interface. At the top, there is a navigation bar with links: [Location](#), [JDBC drivers](#), [Build XML MasterBase](#), and a red button labeled [New MasterBase](#). Below this is a 'Maintenance' section with links: [Search](#), [Upload shapefile](#), [Register: Vector Dataset](#), [Raster Dataset](#), [MrSID](#), [WMS](#), [WFS](#), and [Project](#). The main area is titled 'Define the search condition' and contains a form with the following fields: 'Type' with radio buttons for 'Datasets' (selected), 'Projects', and 'Both'; 'Title' with a text input containing 'commune'; 'Description' with an empty text input; 'Theme' with a dropdown menu showing 'Any'; 'Author' with an empty text input; and 'Order by' with a dropdown menu showing 'Dataset Id'. A 'Search' button is located at the bottom right of the form.

Une fois la table trouvée, il faut éditer ses propriétés avec l'outil "EDIT".

Figure 25 – Résultat de la recherche d'un jeu de données par ALOV



On retrouve alors l'interface présentée plus haut. Les champs à mettre à jour sont les champs donnant les informations sur la table (rubrique "Table Info") et en particulier "Description field", "List of fields to be downloaded" et "List of visible fields on client". Il suffit alors de rajouter dans les listes le champ de la table à ajouter en utilisant le séparateur virgule.

Pour consulter la procédure complète de mise à jour des données attributaires, voir en Annexe 2.

IV – STRUCTURE DE LA BASE DE DONNEES

1 – Les données d'administration et de paramétrage d'ALOV Map

Les données d'administration utilisées par ALOV sont stockées dans la base connectée, dans 5 tables dont le nom est préfixé par "tm" (tm_blobs, tm_datasets, tm_instance, tm_servers et tm_users). Nous étudierons en détail la structure des tables tm_datasets, tm_instance qui sont les plus importantes dans notre configuration.

En effet, **tm_blobs** et **tm_users** sont vides dans notre configuration. **tm_blobs** peut contenir les données du fichier "project" et **tm_users** regroupe les caractéristiques des utilisateurs ainsi que leurs paramètres de connexion et privilèges d'utilisation ; ici nous utilisons un fichier "project" séparé et un unique utilisateur par défaut.

La table **tm_servers** récapitule les pilotes à utiliser pour établir les connexions aux bases de données en fonction du SGBD utilisé, pour les principaux SGBD.

Tableau 2 – Table tm_servers

SRV_ID	SRV_NAME	SRV_JCLASS
1	MySQL	org.gjt.mm.mysql.Driver
2	Interbase	interbase.interclient.Driver
3	Sybase	NULL
4	MS SQL	NULL
5	Oracle	NULL
6	Informix	NULL
7	DB2	NULL
30	MS Access	sun.jdbc.odbc.JdbcOdbcDriver
31	Hypersonic	org.hsqldb.jdbcDriver

Pour utiliser un SGBD non spécifié, il suffit de le rajouter dans cette table en lui attribuant un nom dans SRV_NAME et de lui adjoindre la classe Java du pilote à utiliser.

La table **tm_counter** est utilisée par l'outil d'administration d'ALOV pour générer les identifiants et regroupe pour chacune des 4 autres tables "tm" le prochain identifiant à attribuer.

1.1 – tm_datasets

La table **tm_datasets** contient des informations relatives aux jeux de données et à leur enregistrement. Elles sont en partie reprises dans la table **tm_instance**.

Tableau 3 – Structure de la table tm_datasets

Champ	Type	Description
DS_ID	int(11)	Identifiant du dataset
DS_USR_ID	int(11)	Identifiant de l'utilisateur autorisé
DS_REGDATE	datetime	Date d'enregistrement
DS_UPDATED	datetime	Date de mise à jour
DS_AUTHRSE	varchar(20)	Autorisation (NULL)
DS_TITLE	varchar(250)	Titre du dataset
DS_STATUS	int(11)	Statut (6)
DS_XMIN	double	Bornes de la zone de visibilité de la couche entrées dans l'outil d'administration. Elles peuvent être différentes des coordonnées du RME de l'ensemble des objets de la table.
DS_YMIN	double	
DS_XMAX	double	
DS_YMAX	double	
DS_T_EARLY	int(11)	Utilisé dans le projet TimeMap uniquement
DS_T_LATE	int(11)	Utilisé dans le projet TimeMap uniquement
DS_SRV_ID	int(11)	9 pour les rasters, 30 pour les vecteurs
DS_DAYSBAD	int(11)	Utilisé dans le projet TimeMap uniquement

1.2 – tm_instance

Cette table stocke les paramétrages des jeux de données réalisés au moment de l'enregistrement sous la forme de d'enregistrements successifs dans cette table, et ce, pour les données vecteurs et rasters.

Tableau 4 - Structure de la table *tm_instance*

Champ	Type	Description
MD_ID	int(11)	N° de l'enregistrement
MD_DS_ID	int(11)	Identifiant du dataset
MD_EL_ID	int(11)	Code
MD_SCH_ID	int(11)	"0"
MD_STR_VAL	varchar(255)	Valeur

Voici l'exemple des données relatives aux datasets n°51 et 120, correspondant respectivement aux données vectorielles de la table "T_COMMUNE" et au raster "R_RELIEF":

Tableau 5 – Extrait de la table *tm_instance* du DGEAF

MD_ID	MD_DS_ID	MD_EL_ID	MD_SCH_ID	MD_STR_VAL
2804	51	4	0	856239.94
2805	51	5	0	979490.06
2806	51	6	0	2014163.0
2807	51	7	0	2110165.0
2808	51	8	0	T_COMMUNES
2809	51	24	0	IGN BDCarto 2002
2810	51	36	0	3
2811	51	43	0	T_COMMUNES
2812	51	49	0	INSEE
2813	51	51	0	NOM
2814	51	80	0	INSEE,NOM,INSEEARRDT,INSEECANT,STATUT,SUP,POP,DE NSITE,EVOLPOP,EXP00,TAUXBOIS,PARTPRO,EXPL8800,NCB 3,INDDEPRISE,UDE_MOY,SAU,PLURIAC,POPFAM,PARTSTH,P RODQUAL,SURFTOT,REMEMB,DATEFINALE,PRESAFP,EQUIP TDIST,SERVPROD,IS,URBA032004,LITS2003,RESSEC,PMPOA, PPR,

2815	51	91	0	Administrative geography
2816	51	133	0	51
2817	51	168	0	MS Access
2818	51	172	0	jdbc:odbc:ALOVBASE
7667	120	4	0	850000
7668	120	5	0	980000
7669	120	6	0	2001000
7670	120	7	0	2115000
7671	120	8	0	R_RELIEF
7672	120	133	0	120
7673	120	142	0	850000
7674	120	143	0	2114000
7675	120	144	0	88.851
7676	120	145	0	88.792
7677	120	168	0	MS Access
7678	120	172	0	jdbc:odbc:ALOVBASE

On remarque que de la longueur du champ MD_STR_VAL dépend la quantité d'informations pouvant être stockées. Ainsi dans le cas extrême de la table des communes où il y a beaucoup de champs à stocker, on a été contraint d'augmenter la taille de champ au maximum des possibilités du SGBD.

Tableau 6 – Codages principaux des données dans tm_instance

MD_EL_ID	Description
4	XMIN
5	XMAX
6	YMIN
7	YMAX
8	Titre du dataset
24	Source des données
36	Code relatif au type d'objet
43	Nom de la table
49	Clé primaire
51	Champ utilisé pour les étiquettes et les requêtes

80	Champs à télécharger
91	Thèmes de la table
133	Identifiant
168	SGBD utilisé
172	URL utilisée

2 – Les données vectorielles

Chaque couche vectorielle importée grâce à l'outil d'administration à partir du format Shape fait l'objet d'une table dans la base de données. Sont alors stockées dans une unique table, les données attributaires et géométriques relatives à une couche vectorielle, qu'elle soit ponctuelle, linéaire ou surfacique.

Les données rasters, quant à elles, ont leur lien vers le fichier image et leurs autres propriétés spécifiés dans la table tm_instance. (voir plus bas)

2.1 – Stockage de la géométrie des vecteurs

La géométrie des objets est stockée dans la base de données connectée. Chaque table contient la géométrie des objets de la couche à laquelle elle se rapporte.

Le codage de la géométrie est différent pour les objets de types ponctuels et pour les objets de types linéaires ou surfaciques.

Géométrie des objets ponctuels

La géométrie des objets ponctuels est stockée dans deux champs de type "réel double" de la table. Ceux-ci nommés X_MAPOBJ et Y_MAPOBJ contiennent respectivement l'abscisse et l'ordonnée des objets considérés.

Géométrie des objets linéaires ou surfaciques

Dans le cas plus complexe des objets non ponctuels, la définition dans des champs différents de chaque point constitutif de la géométrie de l'objet est impossible à moins de lier une table à chaque objet de la base. Le choix opéré par ALOV est donc de regrouper la géométrie dans un champ "binaire" appelé GEOBLOB (pour Geographical Binary Large Object). Quatre autres champs l'accompagnent : MBR_XMIN, MBR_YMIN, MBR_XMAX et

MBR_YMAX contenant les coordonnées du Rectangle Minimum Englobant (RME, ou Minimum Bounding Rectangle, MBR en anglais) de l'objet.

Ce stockage dans la table du rectangle minimum englobant permet au moment du chargement des données dans l'applet et grâce à une requête SQL très simple de ne charger que les objets dont le RME intersecte la zone visualisée dans l'applet. Cette technique offre un gain de temps considérable dès lors qu'il est combiné à des seuils de visualisation.

V – PARAMETRAGE DES DONNEES, LE FICHIER *"PROJECT"*

Les données utilisées pour la création de cartes via ALOV Map sont décrites dans le fichier *"project"* écrit en XML.

1 – Présentation des éléments du fichier *"project"*

La balise `<project>` est la balise racine du fichier *"project"*. En effet, elle englobe tout le reste du code de ce fichier.

L'élément `<keymap>` permet de définir la couleur du cadre décrivant la zone visible dans la carte.

L'élément `<map>` définit les thématiques utilisées dans le projet. Ces thématiques vont permettre de regrouper au sein d'une même "carte" différentes couches avec des analyses thématiques (renderer) spécifiques. Un identifiant unique permet de faire référence à chaque thématique.

L'élément `<domain>` décrit des vues prédéfinies. Elles permettent à l'utilisateur un centrage rapide sur des zones particulières de la carte.

L'élément `<layer>` décrit les couches qui pourront être affichées dans le projet.

L'élément `<dataset>` fait référence au jeu de données utilisé par sa balise-mère `<layer>` qui l'encapsule. A un layer correspond un unique dataset. Mais un même dataset peut être utilisé par plusieurs layers. Il peut être intéressant de considérer les `<layer>` comme les entrées des légendes et les `<dataset>` comme les couches de données. Prenons l'exemple de la couche Communes très utilisée dans le DGEAF ; il est indispensable pour faciliter le confort d'utilisation de considérer chaque analyse thématique comme un `<layer>` à part entière pour pouvoir les activer indépendamment les unes des autres, et ce bien qu'elles aient comme source le même `<dataset>`.

La sémiologie des `<layer>` est définie au moyen de l'élément `<symbol>` lorsqu'il s'agit d'une sémiologie à appliquer à tous les éléments de la couche et au moyen de l'élément `<renderer>` lorsqu'on souhaite réaliser une analyse thématique. Un `<layer>` peut contenir

plusieurs éléments `<render>` s'appliquant en fonction de l'attribut `map` ou des seuils de zooms (`zmin`, `zmax`) appliqués.

2 – Les éléments utilisés dans le fichier "*project*" et leurs attributs

*Tableau 7 – Récapitulatif des éléments et attributs utilisés dans le fichier "*project*"*

ATTRIBUT	TYPE	DESCRIPTION
Element : project		
name	chaîne	Nom du projet
mapunits	degrees meters km inches feet miles	Unité de stockage des coordonnées des données géographiques utilisées dans le projet
zoomunits	meters km inches feet miles	Unité utilisée pour établir les zooms
zmin	flottant	Zoom minimum autorisé
zmax	flottant	Zoom maximum autorisé. Par défaut, s'adapte à la largeur définie comme emprise du projet
backcouleur	couleur	Couleur de fond de la carte
Element: domain		
name	chaîne	Nom de la vue
full	bouléen	Full="yes" définit les bornes de visualisation du projet
startup	bouléen	Startup="yes" définit la vue chargée au démarrage du projet
xmin	flottant	Coordonnées de l'emprise de la vue
ymin	flottant	
xmax	flottant	
ymax	flottant	
movebeyond	bouléen	movebeyond="yes" autorise l'utilisateur à zoomer en dehors de la vue définie comme "full domain". Applicable à cette dernière uniquement
Element: map		
name	chaîne	Nom de la thématique
id	chaîne	Identifiant unique
startup	bouléen	Startup="yes" définit la thématique chargée au démarrage du projet
Element: layer		
name	chaîne	Nom du layer
imagelabel	chaîne	Texte dans la légende pour les rasters
cansearch	bouléen	Permet de faire une requête sur le champ défini comme "Description field"

ATTRIBUT	TYPE	DESCRIPTION
map	liste	Spécifie dans quelles thématiques (map) le layer est visible au moyen d'une liste d'identifiants des thématiques
zmin	flottant	Zoom minimum sur la carte en dessous duquel le layer est désactivé
zmax	flottant	Zoom maximum sur la carte au-dessus duquel le layer est désactivé
id	chaîne	Identifiant unique
dependent	liste	Liste des identifiants des layers dépendantes les unes des autres. La visibilité de l'ensemble des layers dépendantes suivra le comportement du layer racine.
order	entier	Ordre d'affichage
startup	bouléen	Définit le layer actif au démarrage du projet
visible	bouléen	Visibilité de la couche au chargement de la thématique
showlegend	bouléen	Représentation dans la légende
legendoutofrange	bouléen	Cache la légende lorsque le layer est hors de ses seuils de visibilité
hidelegend_zoom	bouléen	Supprime le layer de la légende lorsqu'il est hors de ses seuils de visibilité
WrapLayerTitles	bouléen	Autorise le passage à la ligne dans la légende
legendexpanded	bouléen	Développe la légende au chargement
Element: dataset		
id	chaîne	Identifiant du jeu de données dans la base de données
name	chaîne	Nom du jeu de données (facultatif)
type	shape image	Type de jeu de données. Indispensable dans le cas de données chargées à partir d'une base de données. Par défaut, type="shape"
direct	bouléen	Définit le mode de chargement du fichier. Utilisé pour les jeux de données basés sur des fichiers (shp, mif, dbf, jpg, gif) : directement à partir de l'applet ou via le serveur
full	bouléen	full="no" définit un chargement partiel des données vecteurs en fonction de l'emprise visible
Element: selsymbol. Sémiologie des entités sélectionnées		
size	entier	Taille du symbole
filled	bouléen	Rempli
fill	couleur	Couleur de remplissage
outlined	bouléen	Ligne de contour
outline	couleur	Couleur du contour
image	URL	URL de l'image symbole
Element: symbol. Sémiologie		
Etiquette		
size	entier	Taille de la police
style	0 1 2 3	Style de police :Normal-0; Gras-1; Italique-2; Gras italique-3

ATTRIBUT	TYPE	DESCRIPTION
Element: symbol. Sémiologie		
Etiquette		
font	Dialog DialogInput Monospaced Serif SansSerif Symbol	Type de police logique (voir liste) ou nom de police
Ponctuel		
size	entier	Taille du symbole en pixels
style	0 1 2 3	Type de symbole ponctuel :Cercle-0; Carré-1; Triangle-2; Croix-3.
image	URL	URL de l'image symbole
Linéaire		
size	1 2	Largeur de la ligne
step	entier	Distance séparant deux tirets
style	0 1 2 3 4 5	Type de ligne : Continue-0; Tirée-1; Pointillée-2; Tret,Point-3;Tret,Point,Point-4; Barrée-5
Tous		
filled	bouleen	Rempli, pour les polygones et les symboles
fill	couleur	Couleur de remplissage, de police
outlined	bouleen	Ligne de contour, halo pour les étiquettes
outline	couleur	Couleur du contour, du halo pour les étiquettes
label	chaîne	Texte dans la légende (par défaut, la valeur de l'attribut val)
val	chaîne	L'expression est évaluée à vrai ou faux pour les analyses thématiques
Element: paint. Définit les textures des symboles pour les surfaces		
type	texture	Spécifie qu'on crée une texture
image	URL	URL de l'image de base de la texture
Element: texture. Définit le motif de la texture		
xmin	flottant	Coordonnées définissant la fréquence de répétition de l'image
ymax	flottant	
ymin	flottant	
xmax	flottant	
Element: renderer. Définit les analyses thématiques pour personnaliser la sémiologie des couches		
zmin	flottant	Zoom minimum sur la carte en dessous duquel le renderer est désactivé
zmax	flottant	Zoom maximum sur la carte au-dessus duquel le renderer est désactivé
type	default label gradcolor gradmarker chart	5 types d'analyse possible - default (défaut), label (étiquette), gradcolor(plages de couleurs), gradmarker(symboles proportionnels), chart(histogramme)
label	chaîne	Texte dans la légende
map	liste	Liste des thématiques <map> auxquelles l'analyse s'applique. Par défaut, toutes : map="all"

ATTRIBUT	TYPE	DESCRIPTION
Element: renderer. Définit les analyses thématiques pour personnaliser la sémiologie des couches		
showlegend	bouléen	Représentation dans la légende
equal	bouléen	Type de comparaison. Equal="yes" indique que la valeur du champ doit être égale à l'attribut "val" : analyse par valeurs individuelles . Sinon, l'analyse utilisera la sémiologie de la première classe qui a dans l'attribut "val" moins que la valeur du champ : analyse par classes de valeurs.
drawdefault	bouléen	Utiliser l'analyse par défaut (renderer="default") si aucun symbole ne correspond à l'analyse
field	champ	Champ fixant la valeur pour l'analyse "gradcouleur"
labelfield	champ	Champ contenant la valeur du texte à afficher pour l'analyse "label"
charttype	pie bar	Type de représentation. Par défaut charttype="pie"
size	entier	Diamètre des secteurs ou hauteur des barres
sizefield	champ	Champ stockant le diamètre de chaque secteur ou la hauteur de chaque barre. Utilisé à la place de "size"
minsize	entier	Diamètre minimum pour les secteurs
maxsize	entier	Diamètre maximum pour les secteurs
Element: field. Définit le champs utilisé pour l'analyse de type "chart"		
name	champ	Nom du champ
fill	couleur	Couleur de remplissage
label	chaîne	Texte dans la légende

Tableau 8 – Définition des types de valeurs

TYPE DE VALEUR	DESCRIPTION
bouléen	"yes" "no"
chaîne	Chaîne de caractère écrite entre guillemets "aaa"
champ	Champ de la table correspondant au dataset
couleur	"RRR:VVV"BBB"
entier	Nombre entier
entier, %	Nombre entier entre guillemets "nnn" ou pourcentage "nnn%"
flottant	Nombre réel
liste	Items séparés par un espace
URL	Chemin d'accès à la donnée relatif à la racine du projet

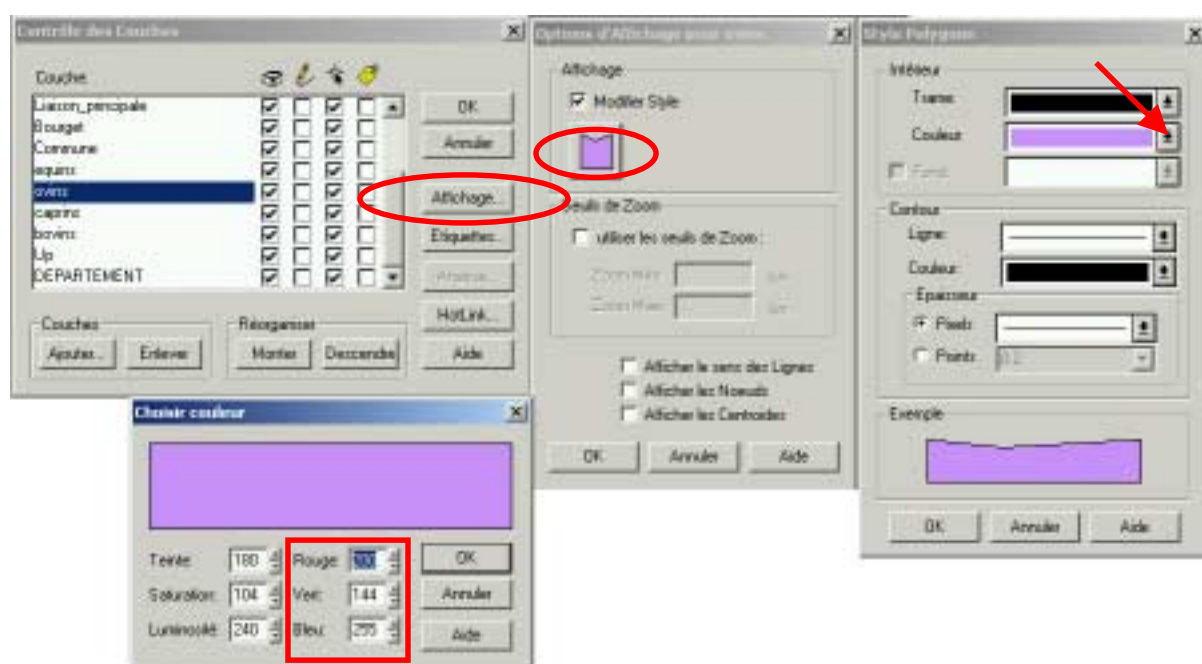
3 – La sémiologie des couches

Dans cette partie, le code en **rouge** correspond à la structure de base de la solution présentée ; le code en **vert** correspond aux parties de code facultatives.

Les codes couleur RVB peuvent être retrouvés au moyen d'une palette, soit dans un logiciel avancé de traitement d'images (Adobe Photoshop par exemple), soit dans MapInfo où il est possible de retrouver les codes couleur RVB utilisés pour la sémiologie des tables.

Dans MapInfo, on ouvre le contrôle des Couches et pour chaque couche, via le bouton "Affichage", on peut "Modifier le style". Dans la boîte de dialogue "Style", dans l'onglet "Couleur", en cliquant sur la case "...", on accède à la boîte "Choisir couleur". Celle-ci nous donne la couleur utilisée, codée en RVB.

Figure 26 – Récupération de la sémiologie via MapInfo



3.1 – Sémiologie par défaut

La sémiologie des couches peut être décrite à l'aide de la balise <symbol> dans le cas simple où tous les éléments d'une couche sont représentés avec le même code graphique. La balise <render type="default"> peut alors être omise sauf s'il y a d'autres analyses rattachées à cette couche et qu'il s'agit de l'analyse à appliquer par défaut.

```
<render type="default">
  <symbol fill="255:210:0" outlined="no"/>
</render>
```

Ce style remplit un objet surfacique avec une teinte orangée codée Rouge=255, Vert=210, Bleu=0, sans contour.

3.2 – Analyse par symboles proportionnels

Les analyses par symboles proportionnels font appel au type **gradmarker** de l'élément **renderer**.

3.1.1 – Utilisation de symboles existants

```
<renderer type="gradmarker" field="EXP00" equal="no" >
  <symbol val="0" style="0" size="5" fill="255:0:0" label="De 0 à 10"/>
  <symbol val="10" style="0" size="10" fill="255:0:0" label="De 10 à
25"/>
  <symbol val="25" style="0" size="16" fill="255:0:0" label="De 25 à
50"/>
  <symbol val="50" style="0" size="22" fill="255:0:0" label="Plus de
50"/>
</renderer>
```

L'exemple présenté montre une analyse par classes de valeurs (`equal="no"`) représentée avec des symboles proportionnels. Il incombe au programmeur de définir lui-même la taille de ses symboles (attribut `size` de l'élément `symbol`). On a utilisé ici les ponctuels correspondant à `style="0"`, i.e. des disques.

3.1.2 – Utilisation de symboles spécifiques

```
<renderer type="gradmarker" field="ANNEE" equal="yes">
  <symbol val="2001" image="/symbols/bergers2001.GIF"/>
  <symbol val="2000" image="/symbols/bergers2000.GIF"/>
</renderer>
```

Dans ce second exemple, on voit qu'il est possible d'utiliser des symboles autres que ceux proposés par défaut par ALOV. Il convient alors de les créer au format GIF transparent et de les appeler à l'aide de l'attribut `image`.

3.2 – Analyse par valeurs individuelles ou par classes de valeurs

Cette fonctionnalité se retrouve pour différents types de `renderer` : `gradmarker`, `label`, `gradcolor`. Suivant la valeur de l'attribut `equal` du `renderer`, il est possible de réaliser une analyse par valeurs individuelles (`equal="yes"`) ou par classes de valeurs (`equal="no"`).

```
<renderer type="gradcolor" field="PRODQUAL" equal="yes">
  <symbol val="0" fill="255:255:255" label="Aucun"/>
  <symbol val="1" fill="237:255:208" label="Secret statistique"/>
</renderer> → Analyse thématique par valeurs individuelles
```

L'analyse par valeurs individuelles est cas le plus simple. L'attribut `val` de l'élément `symbol` indique la valeur du champ `field` défini dans le `renderer` pour laquelle on souhaite que le symbole s'applique. Dans l'exemple présenté, on note que les valeurs 0 et 1 sont utilisées

avec un sens particulier. Il convient donc de les séparer et de traiter le cas d'égalité à ces valeurs dans l'analyse.

```
<renderer type="gradcolor" field="PRODQUAL" equal="no">  
  <symbol val="1" fill="160:224:192" label="moins de 40%" />  
  <symbol val="40" fill="80:160:128" label="de 40 à 60%" />  
  <symbol val="60" fill="48:128:96" label="plus de 60%" />  
</renderer> → Analyse thématique par classes de valeurs
```

Dans le cas des classes de valeurs, l'attribut `val` va contenir les bornes des classes utilisées. Dans notre exemple, la première ligne contient la borne inférieure et correspond à $1 \leq \text{PRODQUAL} < 40$. Il est possible de ne pas spécifier la borne inférieure, le premier symbole correspondra alors à $\text{PRODQUAL} < 40$. La seconde ligne correspond à $40 \leq \text{PRODQUAL} < 60$. La dernière classe enfin est constituée par les données telles que $\text{PRODQUAL} \geq 60$.

3.3 – Analyse par histogrammes ou diagrammes en secteurs

ALOV permet de réaliser des analyses thématiques avec des diagrammes en secteurs. Dans ce cas, on utilise un `renderer` de type `chart`. Par défaut, la représentation sera en secteurs `charttype="pie"`. Pour avoir une représentation avec des histogrammes, il faudra spécifier `charttype="bar"`. L'attribut `size` indique la taille des secteurs ou la taille de la plus grande barre représentée. Il s'agit ensuite d'indiquer les différents champs qui permettent l'analyse à l'aide de l'élément `field`. Son attribut `name` correspond au nom du champ dans la table du jeu de données représenté.

```
<renderer type="chart" charttype="pie" size="30">  
  <field name="CEREALES" fill="255:228:144" />  
  <field name="VIGNES" fill="255:0:0" />  
  <field name="FOURRAGE" fill="0:255:0" />  
  <field name="AUTRES" fill="208:104:0" />  
</renderer>
```

3.4 – Etiquettes

ALOV offre la possibilité d'attacher des étiquettes aux éléments représentés. Pour cela, on utilise le `renderer` de type `label`. Il s'agit alors de lui spécifier quel est le champ de la table du dataset représenté servant comme étiquette au moyen de l'attribut `labelfield`. Il est nécessaire d'indiquer la couleur du texte à afficher dans un élément `symbol` avec l'attribut `fill`.

```
<render type="label" labelfield="NOM" zmax="30" map="all">
  <symbol fill="0:0:0"/>
</render>
```

3.5 – Textures

Des textures peuvent être créées, essentiellement pour étoffer la sémiologie des couches surfaciques. Pour ce faire, on utilise les éléments `paint` et `texture`. La texture réalisée va consister en la juxtaposition à intervalles réguliers d'une image. L'image, dont le chemin d'accès relatif est défini dans l'attribut `image`, est déformée pour correspondre à la maille du quadrillage définie dans `texture` par `xmin`, `ymin`, `xmax`, `ymax`.

```
<symbol val="2">
  <paint type="texture">
    <texture xmin="0" ymin="0" xmax="10" ymax="10"
      image="symbols/hachure_vert.GIF"/>
  </paint>
</symbol>
```

3.6 – Formules

Il est possible d'utiliser des formules pour limiter l'application d'un symbole à certaines valeurs (les chefs-lieux de la Savoie dans notre exemple). On applique l'étiquetage uniquement aux éléments de la table dont le champ, spécifié par `labelfield` ou `field` suivant le type du `render`, renvoie la valeur booléenne vraie à l'assertion proposée.

```
<render type="label" labelfield="NOM">
  <symbol val="NOM LIKE 'CHAMBERY'" fill="0:0:0"/>
  <symbol val="NOM LIKE 'ALBERTVILLE'" fill="0:0:0"/>
  <symbol val="NOM LIKE 'SAINT-JEAN-DE-MAURIENNE'" fill="0:0:0"/>
</render>
```

VI – PARAMETRAGE DE L'INTERFACE DE L'APPLET

Lorsque le paramètre `layout` n'est pas spécifié au lancement de l'applet, ALOV charge une interface par défaut. L'utilisation d'un fichier spécifique de mise en page permet une personnalisation de l'applet au niveau des fonctions utilisées, des messages envoyés et de l'aspect visuel de l'applet.

1 – Structure du fichier "*layout*" de mise en page

La balise racine qui encapsule l'ensemble des paramètres de mise en page est la balise `<layout>`. Elle peut contenir certains attributs comme la couleur dont les balises enfants hériteront par défaut : `backcolor` et `forecolor` désignent respectivement les couleurs de fond et de fonte.

Les différents objets constituant l'interface d'ALOV sont regroupés dans des balises `<object>`. Ils peuvent être issus des composants AWT (Abstract Window Toolkit) de Java ou être spécifiques à ALOV.

De plus, on trouve une balise `<resources>` qui spécifie les messages utilisés par l'application en fonction du paramètre `lang` de l'applet.

2 – Structure de l'interface

L'interface utilisée pour le DGEAF s'articule autour de 4 blocs :

- la barre d'outils (type `toolbar`),
- la légende (type `legend`) et la carte de référence (type `keymap`),
- la carte (type `map`),
- la informations diverses : barre d'échelle (type `scalebar`), statut (type `btn_status`, `statuspanel`).

A ceci s'ajoute le paramétrage de la fenêtre de résultats des interrogations attributaires dérivée des classes Java d'ALOV `Map`, `FrameRes` et `LightGrid`.

3 – Les éléments utilisés dans le fichier "layout" et leurs attributs

Tableau 9 – Récapitulatif des éléments et attributs utilisés dans le fichier "layout"

Type d'objet	Attribut	Type d'attribut	Description
Tous	align		Alignement du composant en haut, en bas, à droite ou à gauche de l'applet ou du composant parent sont la position est forcée à cette valeur même si la taille du conteneur change. Quand le conteneur est redimensionné, un contrôle aligné est aussi redimensionné de sorte qu'il continue à occuper le haut, le bas, les bords droit ou gauche du conteneur. Valeurs possibles :
			Valeur par défaut : la position est définie par les valeurs de l'attribut bounds
		top	Le contrôle s'aligne en haut du conteneur et est redimensionné pour occuper toute la largeur. La hauteur du contrôle n'est pas modifiée
		bottom	Le contrôle s'aligne en bas du conteneur et est redimensionné pour occuper toute la largeur. La hauteur du contrôle n'est pas modifiée
		left	Le contrôle s'aligne à gauche du conteneur et est redimensionné pour occuper toute la hauteur. La largeur du contrôle n'est pas modifiée
		right	Le contrôle s'aligne à droite du conteneur et est redimensionné pour occuper toute la hauteur. La largeur du contrôle n'est pas modifiée
		client	Le contrôle est redimensionné pour s'adapter à la surface client de son conteneur. Si un autre contrôle occupe déjà une partie de cette surface, le contrôle est redimensionné pour s'adapter à la taille de l'espace restant.
	backcolor	Couleur	Couleur de fond du composant. Hérite par défaut de celle du composant parent
	bounds		Emplacement du composant sous la forme d'une liste de 4 nombres : x_haut_gauche, y_haut_gauche, largeur, hauteur

Type d'objet	Attribut	Type d'attribut	Description
Tous	class	chaîne	Chemin d'accès à la classe JAVA définissant un objet spécial
	name	Chaîne	Nom du composant utilisé. C'est l'attribut name qui détermine le comportement d'un composant
	font		Police du composant sous la forme d'une liste de 3 valeurs : "name, style, size" Valeur par défaut : "SansSerif,0,11"
		Chaîne	name - Nom de police ou nom logique. Noms logiques possibles : Dialog, DialogInput, Monospaced, Serif, SansSerif, Symbol.
		Entier	style - Style appliqué à la police. 0 - Normal, 1 - Gras , 2 - Italique 3 - Gras/Italique
		Entier	size -Taille des polices en points
	forecolor	Couleur	Couleur de police du composant. Hérite par défaut de celle du composant parent
	type	legend statuspanel toolbar imagebutton keymap btn_status map panel button label textfield choice image	Types de composants possibles: Composants ALOV : respectivement, légende, barre de statut, barre d'outils, bouton (représenté par une icône), carte de référence, journal, carte Composants AWT : respectivement, conteneur, bouton, texte, champ texte, menu déroulant, image
imagebutton	group	Entier	Comportement du boutons quand il est cliqué. Les boutons ayant le même attribut group (différent de 0) fonctionnent comme un groupe : quand l'un des boutons est cliqué, il reste actif jusqu'à ce qu'un autre bouton du même groupe soit activé.
image	networkactive	Bouléen	Apparition de l'image pendant les périodes de connexion au réseau
legend	disabled_clr	Couleur	Couleur d'une couche inactive
	err_color	Couleur	Couleur d'une couche ayant généré une erreur
	net_color	Couleur	Couleur surlignant la couche en cours de chargement
	select_clr	Couleur	Couleur de la couche active
	separator	Bouléen	Items de la légende séparés par une barre horizontale
map	starttool	Entier	Outil actif au démarrage 1 - Centrage

Type d'objet	Attribut	Type d'attribut	Description
			2 - Zoom Avant 3 - Zoom Arrière 4 - Déplacement , Valeur par défaut 5 - Outil de sélection 6 - Lien hypertexte 7 - Rectangle de recherche 8 - Dessine un rectangle
	tips	Bouleen	Affichage d'étiquettes dans la barre de status (statuspanel). Différent des étiquettes s'affichant à côté de la souris (classe Tooltip)
textfield	size	Entier	Nombre de colonnes du champ texte. Une colonne correspond à la largeur moyenne d'un caractère.
	value	Chaîne	Valeur initiale du champ texte
toolbar	equal	Bouleen	Type de recherche : equal="yes" : on recherche le terme exact dans le champ de la table. equal="no", la recherche peut ne comporter qu'une partie du nom cherché
	helpdoc	URL	Chemin du document d'aide, chargé par clic sur le bouton btn_help
	name		L'attribut nom détermine le comportement et les fonctions associées au composant déterminé
		btn_weblink btn_zoomout btn_zoomin btn_zoomfull btn_pan btn_select	Boutons agissant sur le composant map respectivement accès aux liens, zoom arrière, zoom avant, vue globale, déplacement, sélection d'entités
		btn_help	Lien vers la page d'aide
		btn_getdata	Ouverture la fenêtre des données attributaires sur la sélection
		btn_tips	Activer/Désactiver toutes les étiquettes (tips de l'élément map et classe Tooltip)
		btn_hints	Activer/Désactiver les étiquettes de la classe Tooltip uniquement
		btn_status	Lien vers le journal de l'applet
toolbar	name	tf_search	Champ texte pour entrer le critère de recherche
		btn_search	Lancer la recherche sur les données attributaires
		lst_domains	Liste des "domains", vues prédéfinies du projet
		lst_themes	Liste des thématiques du projet

Type d'objet	Attribut	Type d'attribut	Description
scalebar	size	Entier	Hauteur en pixels de la barre d'échelle (classe ScaleBar)
statuspanel	err_color	Couleur	Couleur des messages d'erreur
	net_color	Couleur	Couleur des messages réseau
	progress_color	Couleur	Couleur de la barre d'avancement du chargement
	select_clr	Couleur	Couleur des messages venant de la carte
	tip_color	Couleur	Couleur des étiquettes
	zoom	Bouleen	Affichage du zoom et des coordonnées du curseur dans statuspanel

ANNEXES

Annexe 1 – Notions de base sur le XML

Les fichiers de paramétrage d'ALOV sont écrits en XML (eXtensible Markup Language) version 1.0. La première ligne de tous ces fichiers doit y faire référence comme suit :

```
<?xml version="1.0" encoding="ISO-8859-1" lang="fr" ?>
```

Le XML est un langage à balise au même titre que le HTML mais sa structure est beaucoup plus stricte.

Le fondement du XML est que toute balise ouverte doit être fermée. De plus, un fichier, outre la 1^{ère} ligne présentée ci-dessus, contient une unique balise racine à l'intérieur de laquelle sont encapsulées toutes les autres.

Une balise peut contenir des attributs. Dans ce cas, ils sont insérés dans la "balise ouvrante". On appelle "balise ouvrante" la balise entre chevrons *<balise>* et "balise fermante" la balise dont le premier chevron est immédiatement suivi d'un slash *</balise>*. Il est à noter que dans un souci de rapidité cette écriture peut être simplifiée dans le cas où la balise ne contient pas d'éléments "enfants". On peut alors observer la syntaxe *<balise/>*.

```
<balise attribut1="valeur1" attribut2="valeur2"> → balise ouvrante  
</balise> → balise fermante
```

Une balise peut contenir d'autres éléments-balises qui lui sont propres qui peuvent eux-mêmes posséder des attributs. On arrive ainsi à une structure arborescente du fichier.

```
<?xml version="1.0" encoding="ISO-8859-1" lang="fr" ?>  
<balise_racine attribut1="valeur1" attribut2="valeur2">  
  <balise_enfant1 attribut3="valeur3"> → balise enfant  
    <balise_enfant1_1 attribut4="valeur4"/> → balise fermée  
  </balise_enfant1>  
  <balise_enfant2 attribut5="valeur5"/>  
</balise_racine>
```

Annexe 2 – Procédure de mise à jour d'un jeu de données

- 1 - **Importer les nouvelles données dans la table Access** correspondant au jeu de données dans le champ existant ou dans un nouveau champ.

Attention : la clé primaire primaire utilisée peut être celle créée par ALOV (NOM_TABLE_ID) au moment de l'import de la couche ; celle-ci n'existe pas au départ dans les données utilisées sous MapInfo.

- 2 - **Mettre à jour les spécifications de la table dans l'outil d'administration d'ALOV** dans le cas où un nouveau champ est créé dans la table Access.

(Voir paragraphe III-3) Pour cela, lancer une recherche sur le jeu de données à actualiser. Une fois sélectionné, ajouter le nouveau champ à la liste des champs à télécharger. Attention : la présence du nom du champ dans cette liste est indispensable pour son utilisation dans une analyse thématique.

- 3 - **Adapter la sémiologie relative à ce champ dans project.xml**

(Voir paragraphe V-3) Trois cas de figure :

- a. Le champ était déjà la source d'une analyse thématique : réutiliser l'ancienne sémiologie ou la mettre à jour
- b. Il s'agit d'un nouveau champ sur lequel on souhaite créer une analyse servant d'entrée de légende : créer un nouveau layer s'appuyant sur cette table et lui spécifier une analyse thématique pour ce champ
- c. On souhaite compléter les analyses proposées par une couche existante avec une nouvelle analyse thématique sur ce champ : compléter les analyses existantes dans la couche relative à la table dont il est issu