

Projecte xulRecordset

<http://xulrecordset.sourceforge.net>

(Juny 2005)

Jordi Llonch

LAIGU Serveis d'Enginyeria, S.L. - Barcelona www.laigu.net

Correu: jordi@laigu.net

1. Resum

xulRecordset és un projecte que pretén la creació d'una llibreria que faciliti la creació d'una interfície d'usuari web vinculada a una base de dades.

La llibreria es basa en widgets *XUL* (tecnologia *Mozilla*) amb un comportament similar als recordsets, com en el cas de l'*ADORRecordset* en *Visual Basic*.

El projecte fa servir el llenguatge *PHP* en la banda del servidor i *XUL* derivat (emprant *XBL* – tecnologia *Mozilla*) i *Javascript* a la banda del client. La comunicació entre client i servidor està suportada per la llibreria *JPSpan* (*xmlHttpRequest*).

Es proveeix d'una aplicació agenda d'exemple per mostrar-ne el funcionament. L'exemple permet moure's al següent registre, al previ, al primer, al darrer, actualitzar, esborrar i afegir registres. L'exemple fa servir una base de dades *MySQL* com a sistema d'emmagatzematge.

El codi font s'ha realitzat sota llicència *LGPL* i es gestiona en el portal www.sourceforge.net.

2. Introducció

Aquest projecte parteix de la necessitat per part de l'autor de disposar d'un sistema de desenvolupament ràpid per a la creació d'un entorn web, independent de plataforma i que vagi més enllà de l'*HTML*. Fins al moment s'havia estat desenvolupant una aplicació de gestió emprant exclusivament *PHP/HTML+Javascript/MySQL*. Per a la gran majoria d'interfícies va ser suficient però es van trobar casos en el que era necessari disposar d'un entorn més complet, més ràpid i eficient, però sense deixar de banda el concepte d'aplicació web.

El projecte *Mozilla* vist des de la perspectiva d'entorn de desenvolupament va resultar ja des del primer moment una alternativa força atractiva. Tant pel concepte tecnològic (multiplataforma) com per la filosofia que implica (Programari Lliure).

L'article comença amb la descripció de la tecnologia *Mozilla* i dels conceptes bàsics sobre *XUL*, a continuació s'explica sota quins conceptes reposa el projecte *xulRecordset* i es presenta un exemple d'aplicació. Finalment es veuen les conclusions.

3. XUL (Mozilla)

El projecte *Mozilla* va ser iniciat el març del 1998 amb l'objectiu de desenvolupar el successor del navegador *Netscape Communicator 4.x*. Avui *Mozilla* és utilitzat pels desenvolupadors com a plataforma per a la creació d'aplicacions que poden ser instal·lades localment o executades remotament a través d'*Internet*.

La diferència entre l'objectiu del projecte original *Mozilla* de crear un navegador i el seu actual ús com a *framework* de desenvolupament multiplataforma no és tan estrany. Un navegador web no hauria de ser com un *framework* de desenvolupament d'aplicacions, però ho és. Els navegadors permeten a la gent fer servir

qualsevol tipus d'ordinador per accedir a aplicacions com el mail de *Google* per rebre i enviar correus electrònics, comprar llibres a *Amazon* o interactuar amb la casa de subhastes *eBay*.

Mozilla ha expandit la idea de fer servir un navegador només per accedir a aplicacions creades amb alguna de les tecnologies per crear portals web (*CSS* i *Javascript*). És així com s'ha convertit en un entorn per crear aplicacions multiplataforma.

És per això que es pot dir que *Mozilla* no és només un navegador web. És a més un *framework* per construir-hi aplicacions multiplataforma utilitzant estàndards com ara *Cascading Style Sheets (CSS)*, llenguatge *XML* com ara el *User-interface Language (XUL)*, *eXtensible Binding Language (XBL)*, i el *Resource Description Framework (RDF)*.

Gecko, el motor renderitzador de *Mozilla*, és emprat com a part del *framework*, així com altres tecnologies com *XPConnect* i *XPCOM* (*model de components de Mozilla*). També cal dir que el *framework* de desenvolupament de *Mozilla* fa servir llenguatges de programació com *Javascript*, *C++*, *C*, *Python* i *Interface Definition Language (IDL)*.

El *framework* de *Mozilla* s'utilitza per crear els navegadors de *Netscape* (*Netscape 6.x* i *7.x*), altres navegadors com *Galeon* i *Camino*, i clients xat com *ChatZilla* i *JabberZilla*. Els desenvolupadors a més fan servir *Mozilla* per crear eines de desenvolupament, navegadors perfeccionats, jocs i altres tipus d'afegits o *plug-ins* i aplicacions.

3.1. Extrema portabilitat

Potser la millor avantatge que té *Mozilla* pels desenvolupadors és que les aplicacions basades en *Mozilla* són multiplataforma, és a dir, que els programes funcionen d'igual forma en *Windows* com en *Unix* o en *Mac OS*. És possible tenir aplicacions funcionant en diferents plataformes perquè *Mozilla* actua com a capa intèrpret entre el sistema operatiu i les aplicacions.

El nombre de ports de *Mozilla* a diferents sistemes operatius permet fer-se a la idea del complet rang de plataformes on opera *Mozilla*. *Mozilla* funciona en *Windows*, *Macintosh* (*Classic Mac* i *Mac OS X*), i *Linux*, així com en la majoria de *Unix*, inclòs *Solaris*, *FreeBSD*, *HP-UX*, i *Irix*. El projecte de portatge està en el camí de fer funcionar *Mozilla* en *BeOS*, *OS/2*, *Open VMS*, *Amiga*, i altres sistemes operatius (<http://www.mozilla.org/ports>).

3.2. XUL

XUL (pronunciat “zuul”) va ser creat pel desenvolupament de la interfície del navegador *Mozilla* de forma fàcil i ràpida. És un llenguatge *XML* i per tant en *XUL* es disposa de totes les característiques d'aquest metallenguatge.

XUL és un altre llenguatge dissenyat específicament per a construir interfícies d'usuari portables.

Moltes aplicacions depenen de característiques específiques d'una plataforma i per tant aconseguir que aquestes aplicacions siguin multiplataforma és costós en temps i esforç. Això pot no ser important per algunes aplicacions però si es considera la possibilitat que l'usuari pugui utilitzar una aplicació en un dispositiu com una agenda de mà, pot ser molt útil desenvolupar en un entorn multiplataforma. Hi ha un cert nombre de solucions multiplataforma que han estat desenvolupades. *Java*, per exemple, té la portabilitat com una de les seves característiques destacades.

Amb *XUL*, una interfície pot ser implementada i modificada ràpida i fàcilment.

XUL té tots els avantatges dels altres llenguatges *XML*. Per exemple, altres llenguatges *XML* o *HTML* poden ser insertats amb ell. A més, *XUL* és fàcilment *localitzable*, és a dir que la interfície pot ser traduïda a altres llengües de forma senzilla (emprant *DTDs*). Les fulles d'estil (*CSS*) poden ser aplicades per modificar l'aparença de la interfície d'usuari (de forma similar als *skins* o temes). Per completar el *framework* podem encastar codi *Javascript* dins de *XUL* per a la programació de la interacció de l'usuari amb el client, obtenim així el que s'anomena *framework XPFE* (Figura 1).

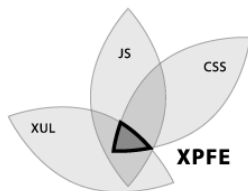


Figura 1. Framework XPFE

El següent que caldrà aclarir és quina mena d'interfícies d'usuari es poden fer amb *XUL*?. *XUL* ofereix la possibilitat de crear la gran majoria d'elements que avui en dia es poden trobar en les interfícies gràfiques modernes. És suficientment genèric per poder ser aplicat per les necessitats especials de certs dispositius i suficientment flexible perquè els desenvolupadors puguin crear sofisticades interfícies (Figura 2).

Alguns elements que es poden crear són:

- Controls d'entrada com els camps de text.
- Barra d'eines amb botons o contingut.
- Menús en una barra de menú o en *pop up*.
- Diàlegs tabulats.
- Arbres amb jerarquia o informació tabulada.
- Tecles ràpides.

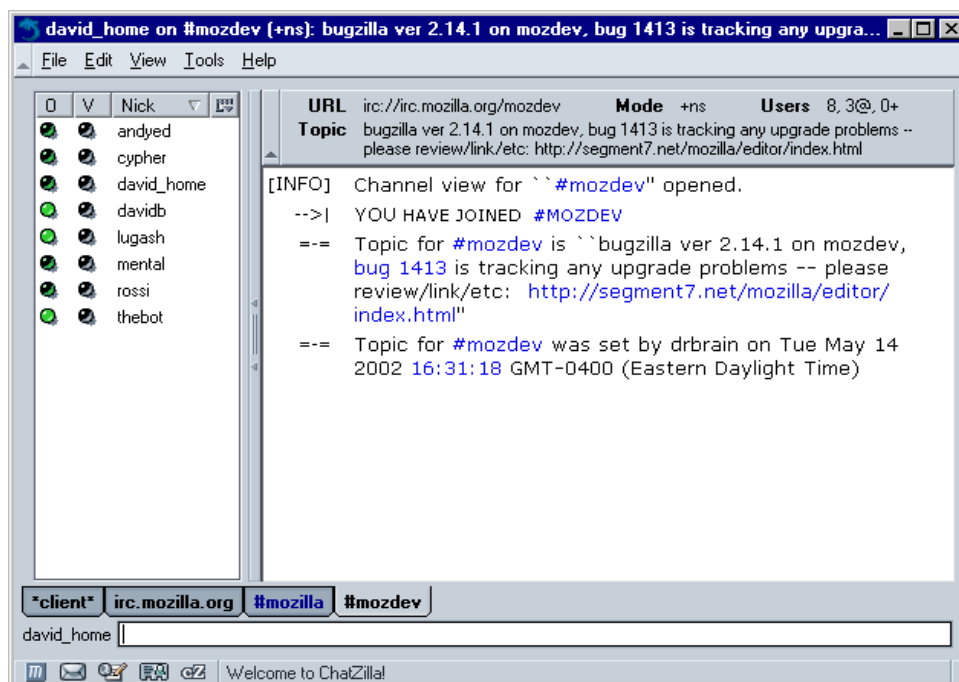


Figura 2. Exemple d'interfície XUL

3.3. XBL

El lector en aquest moment es deu haver adonat que *XUL* és l'eina bàsica per a la creació de la interfície de la nostra aplicació, però tot i així hi trobem limitacions. Doncs hi ha un nombre limitat d'elements o *widgets* que possiblement el nostra programa necessitarà transcendir. Si es troba la necessitat de reimplementar molts dels *widgets* del mateix grup en diferents aplicacions, o si es vol estendre la interfície de la nostra aplicació en algun sentit, es necessitarà fer ús de l'eina *eXtensible Binding Language (XBL)*.

XBL ofereix un camí per afegir nou contingut o comportaments a la nostra aplicació emprant els *XBL bindings*. *XBL* pot estendre, afegir, i reorganitzar les interfícies d'usuari. *XBL* pot ajudar a organitzar el codi *XUL* dispers en conjunts autocontinguts de *widgets* que fan la creació i manteniment de les nostres aplicacions *Mozilla* més senzilla.

Però que és *XBL*? *XBL* és un llenguatge d'etiquetes *XML* inventat específicament per crear *widgets*. *XBL* s'assembla al *XUL*, i pot contenir *XUL* o *HTML* i altres etiquetes. La flexibilitat i la interoperabilitat són els punts forts del *XBL*.

Si el *XUL* `textbox` es inadequat, per exemple, es pot fer servir *XBL* per crear i afegir un *widget* anomenat `datafield`, possiblement basat en `textbox`, que ens oferirà atributs especials i funcionalitats per validar l'entrada de dades contra la base de dades.

4. Projecte *xulRecordset*

Ja s'ha comentat la necessitat per part de l'autor de disposar d'un sistema de desenvolupament productiu basat en el concepte d'entorns web independents de plataforma. A més es desitjava fer ús al màxim dels estàndards oberts que avui en dia ens ofereix la comunitat. És així com neix la idea de crear el projecte *xulRecordset*.

Aquest projecte pretén la creació d'una llibreria de *widgets XUL* amb un comportament similar als *recordsets*, com en el cas de l'*ADOREcordset* en *Visual Basic*. Amb l'ajut d'aquest concepte es vol obtenir una llibreria que faciliti la tasca d'unir la lògica de l'aplicació amb la interfície d'usuari. Cal a més dir que fins al moment ja es disposava del desenvolupament de la lògica en forma de classes de dades en *PHP* i que una de les premisses va estar aconseguir el paradigma de la reutilització de codi.

Amb aquests objectius clars es va iniciar la recerca i anàlisi de possibles solucions. Cal dir que ja des del primer moment es tenia coneixement del projecte *Mozilla* com a plataforma de desenvolupament. Tot i així es van tenir en compte certes solucions basades per exemple en *Java* o en *Macromedia Flash® + XML*. Aquestes solucions, però, van ser descartades entre altres motius sobretot per no ser solucions realment obertes i/o lliures.

El *framework* de *Mozilla* ens ofería tot el que s'estava cercant. Disposàvem d'un entorn multiplataforma que qualsevol usuari sense gaires coneixements es pot instal·lar en el seu ordinador, i cobrint el 100% de les plataformes on la nostra aplicació va destinada. Amb *Mozilla* es podia crear un client remot per a la nostra aplicació amb una interfície d'usuari atractiva, però sobretot potent i de forma totalment transparent a l'usuari on la seva única tasca consisteix en navegar amb *Mozilla* o *Firefox*. Ja s'ha comentat en l'apartat anterior com mitjançant *XUL* i *XBL* es pot obtenir unes eines i un entorn per desenvolupar aplicacions. El projecte *xulRecordset* fa ús d'aquests llenguatges junt amb *Javascript* per a la creació d'un client remot. Emprant *XBL* s'ha creat una llibreria de *widgets* amb un comportament associat a un altre *widget* no visible i que s'encarrega del control de la resta. És aquest *widget* ocult anomenat també "*xulRecordset*", el que s'encarrega de gestionar les comunicacions amb el servidor i extreure i omplir les dades a la resta de *widgets*.

En aquest punt sorgeix la interessant qüestió de com resoldre la comunicació entre el nostre client remot i el servidor on recordem, opera un servidor d'aplicacions de *PHP*. Cercant la millor solució per a la comunicació, és a dir buscant com es podien unir aquestes tecnologies es va assolir l'objectiu de la reutilització de codi. Fent ús altre cop del que ens ofereix la comunitat es va trobar el projecte *JPSpan*. Aquest projecte crea una passarel·la entre classes escrites en *PHP* i el llenguatge *Javascript* que es troba en el client (Figura 3). A la pràctica el que s'aconsegueix és interactuar de forma transparent i des de la banda

del client amb objectes distribuïts, en el nostre cas amb els objectes de dades i la lògica de la nostra aplicació ja escrita en *PHP*. *JPSpan* fa ús de la tecnologia *AJAX* (objecte *XMLHttpRequest*) que des de *Mozilla* ofereix la possibilitat de obrir una connexió i interactuar amb un servidor mitjançant missatges *XML*. A més disposa de funcions de serialització de dades i objectes per a la plena interacció entre client i servidor. *JPSpan* és un híbrid de codi *PHP* i *Javascript* que d'una forma extremadament senzilla i sobretot transparent al desenvolupador converteix classes escrites en *PHP* en objectes distribuïts remotament.

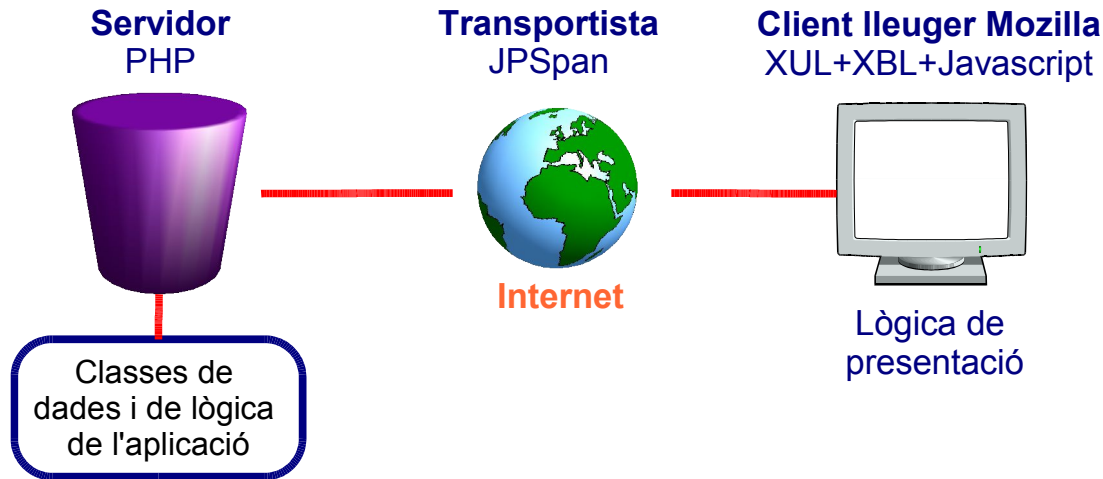


Figura 3. Esquema

En resum, el projecte *xulRecordset* ofereix una llibreria al desenvolupador que ens inclou les següents tecnologies:

- PHP, a la banda del servidor.
- JPSpan, a la banda del servidor i del client, és el *transportista*.
- XUL + XBL + Javascript, a la banda del client.

El desenvolupador ha de crear l'entorn client fent ús del *XUL* i els *widgets* del projecte *xulRecordset*. I per altre banda crear uns classe en *PHP* amb uns mètodes predefinits per a què el *widget* de control pugui interactuar correctament amb el servidor.

5. Exemple

Veiem tot seguit el funcionament de *xulRecordset* amb un exemple senzill en forma d'agenda de contactes (Figura 4).

Aquest exemple mostra les operacions típiques que facilita el projecte. Aquestes operacions són les habituals de manteniment de registres de dades com afegir, modificar i borrar dades, a més de les operacions de navegació d'anar al primer registre, anar al darrer i moure's endavant i endarrera.

La captura de pantalla mostra un formulari amb els següents camps i elements:

- Id:** 1
- Name:** Jordi
- Surname:** Llonch
- Address:** My address
- City:** Barcelona
- Country:** Spain (menú desplegable)
- Gender:** Male, Female
- Active:**
- Botons de navegació: |<, <, >, >|
- Botons d'operacions: Add, Update, Remove

Figura 4. Exemple simple de xulRecordset

L'exemple consta de 3 arxius:

agenda.xul – Arxiu on hi ha definida la interfície d'usuari, on es connecten els events i on es defineix la connexió al servidor. També s'hi inclou la llibreria *xulRecordset*. Aquest és l'arxiu que l'usuari carrega en el seu navegador.

```
...
<?xml-stylesheet href="chrome://global/skin/" style="text/css"?>
<?xml-stylesheet href="../../lib/xulrecordset.css" style="text/css"?>

<window id="xm:window0" xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul">

<script type="application/x-javascript" src="../../lib/xulrecordset.js"/>
<script type="application/x-javascript" src="xrs_server.php?client"/>
<script type="application/x-javascript" src="agenda.js"/>

<xulRecordset id="rs" xrsClass="xrsagenda"/>

<groupbox>
  <grid>
    <columns>
      <column/>
      <column/>
    </columns>

    <rows>
      <row>
        <label flex="1" value="Id:"/>
        <xrsInputField id="idContact" idRecordset="rs" readonly="true" size="5"/>
      </row>
    </rows>
  </grid>
</groupbox>
...
```

agenda.js – Arxiu on es defineixen les funcions que executaran els events que produeixi l'usuari.

```
...
function moveFirst()
{
  var rs = document.getElementById('rs');
  rs.moveFirst();
}
...
```

xrs_server.php – Classe en *PHP* que conté els mètodes bàsics d'operació de l'agenda.

```
...
class xrsAgenda extends xrsServer {
  var $db;

  function xrsAgenda () {
    // Init parent constructor
    parent::xrsServer("agenda");

    // Open a persistent database connection
    $this->db = DB::connect('mysql://'.DB_USER.':'.DB_PASS.'@'.DB_HOST.'/'.DB_NAME, 1);
    if (DB::isError($this->db)) {
      syslog (0, "[".__FILE__."": ".__LINE__."]: Error connecting to DB
      ".'mysql://'.DB_USER.':'.DB_PASS.'@'.DB_HOST.'/'.DB_NAME);
      return false;
    }

    return true;
  }

  function getCurrent() {
    $sql = "SELECT * FROM contacts LIMIT ".$this->getCurrentIndex().",1";
    if (DB::isError($result = $this->db->query($sql)) {
      syslog (0, "[".__FILE__."": ".__LINE__."]: ERROR: ".$result->getUserInfo($result));
      return;
    }
    else {
      $result->fetchInto($r, DB_FETCHMODE_ASSOC);
    }
  }
}
```

```

    $r["_xrs_current"] = $this->getCurrentIndex(); // Add current index to result
  }
  return $r;
}

function update($data) {
  if (!is_array($data)) return false;
  if (empty($data["idContact"])) return false;
  foreach ($data AS $k => $v) $kv[] = "$k='".addslashes($v)."'";
  $kv = implode(",", $kv);
  $sql = "UPDATE contacts SET $kv WHERE idContact={$data["idContact"]}";
  if (DB::isError($result = $this->db->query($sql))) {
    syslog (0, "[".__FILE__."": "__LINE__."]: ERROR: ".$result->getUserInfo($result));
    return false;
  }
  return true;
}
}
...

```

Es pot trobar el codi font complet i una demo funcional on-line a <http://xulrecordset.sourceforge.net>

6. Conclusions

En aquest article s'ha presentat el projecte *xulRecordset* com a plataforma de programació que permet simplificar el desenvolupament d'una interfície d'usuari flexible basat en el *framework* del projecte *Mozilla*. S'han presentat els antecedents que han motivat el projecte i s'ha descrit la solució adoptada fent destacar la fita aconseguida de la reutilització de codi. Finalment s'ha vist un exemple que mostra de quina forma els desenvolupadors poden fer ús del projecte.

En aquest moment s'està implementant aquesta solució en una aplicació real de gestió d'agències de viatges (Figura 5). En el futur es pretén disposar de més *widgets* fent ús de la propera versió de la plataforma *Mozilla*, la versió 2.0. A més es vol millorar el rendiment de la llibreria fent ús de memòria cau.

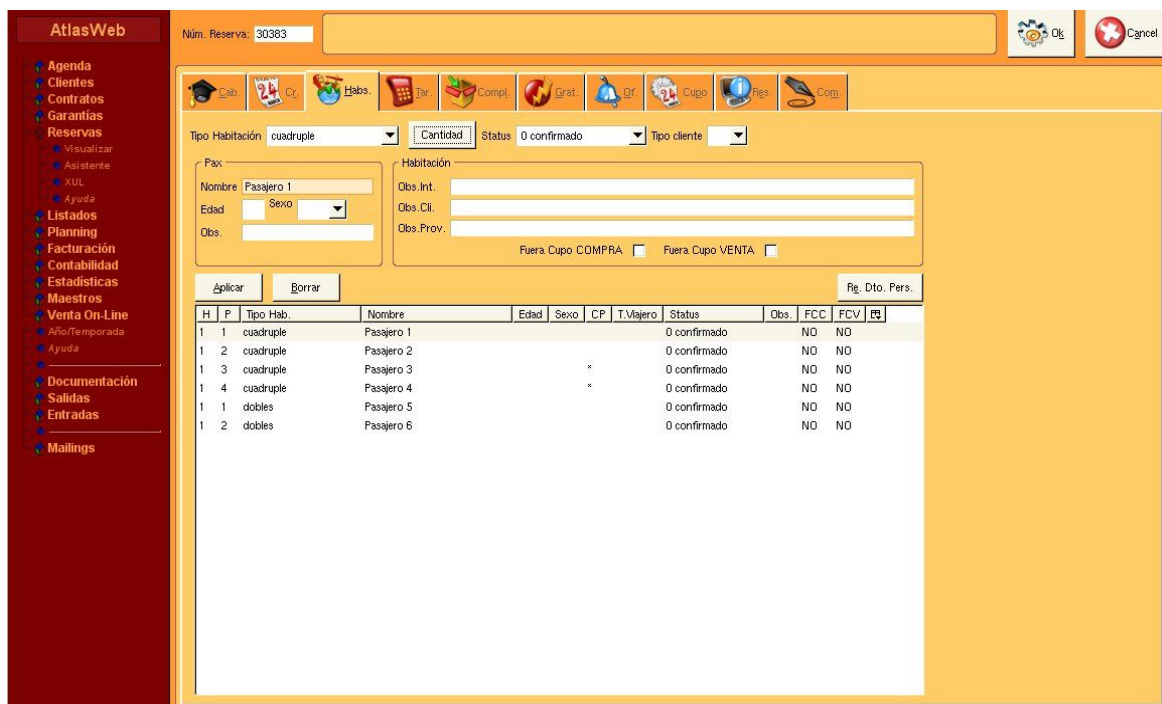


Figura 5. xulRecordset en aplicació real

7. Referències

- [1] Nigel McFarlane. Rapid Application Development with Mozilla.
- [2] David Boswell, Brian King, Ian Oeschger, Pete Collins, Eric Murphy. Creating Applications with Mozilla. <http://books.mozdev.org>
- [3] Neil Deakin. The XUL Tutorial. <http://www.ar-ent.net/dar/arlib32/out/html/man/xul/index.html>
- [4] Harry Fuecks, Jason E. Sweat. Projecte JPSSpan. <http://jpsspan.sourceforge.net/wiki/doku.php>