MANEJO INTEGRADO DE INFORMACIÓN HIDROGEOLÓGICA

(MIIH)

Alumno: Ruben Fernando Pacheco Valerio

Tutor PFM: Gregorio Robles Martínez

Desarrollo de Aplicaciones de Software Libre, Prueba de Evaluación Continua 4 (PEC4)

Máster Software Libre, Universidad Oberta de Catalunya (UOC)

Enero 2008

RESUMEN

En este documento se resume las principales actividades realizadas en el marco de la asignatura Desarrollo de Aplicaciones en Software Libre, del Máster de Software Libre de la Universidad Oberta de Catalunya.

El trabajo se centro en dos actividades principales: la primera, la realización de tareas relacionadas a la creación de un proyecto de software libre para una aplicación que permite el almacenamiento, gestión e interpretación de información hidrogeológica; en tanto que la segunda -íntimamente relacionada con la anterior[1]-, la colaboración en tareas de desarrollo del proyecto QGis (http://www.qgis.org).

El software se denominó MIIH (Manejo Integrado de Información Hidrogeológica), fue realizado principalmente en el lenguaje C++ utilizando la plataforma de desarrollo Qt (http://trolltech.com/products/qt) para la interfase gráfica del usuario y acceso a base de datos, y es extensible por módulos (actualmente en C++ y en el futuro también en Python). Los datos hidrogeológicos y del usuario son almacenados en una base de datos en SQLite.

Se trata de un proyecto de porte medio, de largo aliento[2], y se utilizó como plataforma para la generación de la comunidad asociada a la aplicación a SourceForge (http://sourceforge.net).

La plataforma para el desarrollo y soporte de la aplicación se encuentra alojada en http://sourceforge.net/projects/miih/, el sitio web en la dirección http://miih.sourceforge.net/, en tanto que la documentación asociada al desarrollo (API) en http://miih.sourceforge.net/api_doc/.

Las tareas de colaboración con el proyecto QGis se centraron en el reporte de bugs, solución de los problemas encontrados (patchs) y en el pasaje de código fuente de la aplicación que utiliza todavía Qt3 (utilizando Q3Support) a Qt4.

1 La estructura de la aplicación fue utilizado como modelo para el desarrollo del software.
2 En el momento de la realización de este informe la aplicación se encuentra todavía en estado alpha.

Índice de contenido

Índice de ilustraciones

CAPÍTULO 1: INTRODUCCIÓN

El trabajo del curso "Desarrollo de software libre" se centro en dos actividades principales: la primera, la realización de tareas relacionadas a la creación de un proyecto de software libre, para la generación de una aplicación que permite el almacenamiento, gestión e interpretación de información hidrogeológica. Este documento expone los pasos seguidos para desarrollar ese proyecto.

La segunda tarea -íntimamente relacionada con la primera[3]-, estuvo centrada en la colaboración en el desarrollo con el proyecto QGis (http://www.qgis.org).

Como parte de la presentación y tratando de facilitar la comprensión, se expone un resumen del QGis y las actividades realizadas para dicho proyecto en un capítulo por separado.

1.1 JUSTIFICACIÓN

A pesar de la importancia que presenta el volumen de agua dulce almacenada como agua subterránea y la imperiosa necesidad de su gestión adecuada, no existen herramientas que permitan realizar en forma integrada las tareas básicas de almacenamiento, análisis y gestión requeridas. Algunas herramientas gratis, realizadas por distintas empresas ó instituciones en el mundo, pueden ser utilizadas para solucionar parcialmente el problema, en tanto que otras mas o menos completas fueron desarrolladas hace muchos años y no han evolucionado en la medida de lo esperado.

Se pueden adquirir a costes elevados productos que tratan de solucionar -en base a software propietario- los distintos aspectos involucrados. De cualquier manera, y en todos los casos, el pasaje de datos (en forma mas o menos periódica) de un sistema al otro es una tarea tediosa e ineficiente que solo puede ser aplicada de forma satisfactoria en pequeños proyectos, donde se manejen volúmenes de datos reducidos.

Se hace necesario -y de cierta forma no aplazable- la creación de una base de datos capaz de almacenar toda la información relacionada y un conjunto de herramientas que faciliten -a distintos niveles- las tareas de almacenamiento, gestión e interpretación de la información.

La necesidad de que todas las herramientas generadas puedan ser replicadas, compartidas, modificadas, extendidas y redistribuidas (a empresas, organismos, instituciones, etc.) conduce indefectiblemente hacia la generación de un proyecto basado en tecnologías libres.

Por otro lado, en relación a la segunda tarea del curso, la aplicación QGis es un proyecto de software libre que ha evolucionado muy favorablemente en cuanto al número de funcionalidades en los últimos 2 años, pero todavía presenta errores en el código fuente en algunos de sus elementos centrales. Entendiendo que es una excelente opción para trabajos con sistemas de información geográfica se trato de colaborar de la mejor manera posible en el desarrollo del mismo.

1.2 PUNTO DE PARTIDA

Se entiende que las premisas básicas del proyecto de fin de curso son el aprendizaje, investigación y aplicación de los conocimientos adquiridos en las distintas materias cursadas a lo largo del Máster.

En virtud de ello, se trató de seleccionar un proyecto de interés y un conjunto de herramientas que fueran altamente utilizados en el mundo de software libre y que resultaran además un

3 El proyecto en cuestión fue utilizado como modelo para el desarrollo de la aplicación.

desafió para el estudiante[4].

Con ese rumbo y con los requerimientos impuestos por la aplicación, se seleccionó como lenguaje de programación C++ y como bibliotecas para el desarrollo las STL de C++ y las Qt de Trolltech.

1.3 OBJETIVOS

1.3.1 GENERALES

Generar herramientas que faciliten la gestión de los recursos naturales, recursos hídricos y en particular la temática relacionada con el agua subterránea. Promocionar la modalidad de software libre y desarrollo comunitario en el ambiente académico especializado relacionado con el tema.

Introducir al alumno en forma práctica en el desarrollo de aplicaciones en software libre.

1.3.2 PARTICULARES

Creación de una base de datos capaz de almacenar y gestionar los datos en forma adecuada (relacional y transaccional).

Creación de una aplicación que facilite la gestión, análisis, interpretación e intercambio de datos hidrogeológicos.

Crear un conjunto de bibliotecas que puedan ser utilizadas por otras aplicaciones.

1.4 ÁREAS DE INTERÉS

Como principales áreas de interés para el uso del producto se presentan los organismos o instituciones dedicadas a la investigación y gestión de recursos naturales y en particular los recursos hídricos.

1.5 METODOLOGÍA

La metodología utilizada para el trabajo fue: 1 - análisis de requerimientos; 2 – selección de herramientas; 3 – implementación del sitio comunitario de desarrollo; 4 – investigación de proyectos con tecnologías similares en software libre; 5 – generación de la arquitectura de la aplicación; 6 – desarrollo; 7 – promoción.

1.6 PLANIFICACIÓN

Las grandes tareas identificadas al comienzo del proyecto fueron:

1. Creación del sitio de desarrollo en SourceForge (listas, sitio web, etc.);
2. Promoción de aplicación para generar la comunidad del proyecto;
3. Propuesta de lineamientos para el desarrollo;
4. Discusión y propuesta de estructura de aplicación;
5. Desarrollo de aplicación;

4 Es decir se trató de seleccionar un lenguaje de programación y bibliotecas donde el alumno no tuviera experiencia.

6. Desarrollo de bibliotecas;
7. Desarrollo de módulos.

El cronograma inicial propuesto se muestra a continuación:

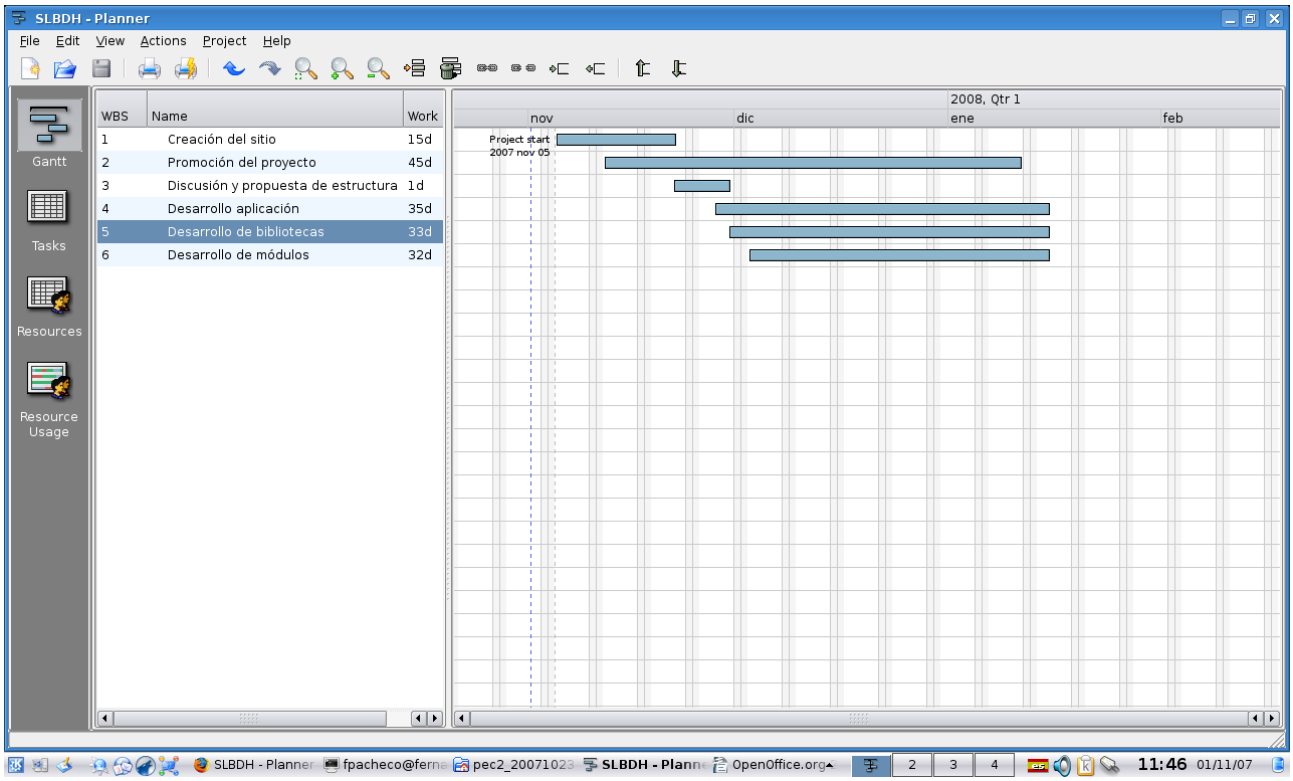


Imagen 1: Cronograma de trabajo propuesto.

CAPÍTULO 2: COLABORACIÓN PROYECTO QGIS

En el marco de la integración del alumno a un proyecto de software libre se seleccionó el proyecto QGis.

La selección estuvo altamente condicionada a la preferencia del alumno por los sistemas de información geográfica (SIG) y en particular al conocimiento y uso del SIG GRASS.

Se presenta a continuación una descripción somera del proyecto y de la arquitectura de la aplicación para luego paras a describir las tareas realizadas en el marco del proyecto. Las conclusiones del trabajo realizado se exponen en el capítulo de conclusiones.

2.1 DESCRIPCIÓN DEL PROYECTO

QGgis (http://www.qgis.org/) es un software de sistema de información geográfica desarrollado en C++ con las bibliotecas Qt.

El desarrollo comenzó en el año 2002 y la versión actual es la 0.9.1, lanzada a principios del mes de enero.

Es capaz de leer y editar archivos Shape File (estándar creado por la empresa ESRI)[5], presenta buenas funcionalidades de acceso a bases de datos espaciales en PostrgreSQL (http://www.postgresql.org)/PostGIS[6] (http://postgis.refractions.net/) y utiliza -al igual que la mayoría de los proyectos de software libre relacionados con información geográfica- las bibliotecas PROJ4 (http://proj.maptools.org/), GEOS (http://geos.refractions.net/), GDAL y OGR (http://www.gdal.org/).

La aplicación es extensible por módulos -plugins-, contando entre ellos con uno de acceso a la mayoría de las funcionalidades disponibles en GRASS[7]. Este módulo actualiza la interfase del usuario con GRASS (mejorada en los últimos años pero todavía en Tcl/Tk) conservando su potencia, presentándolo de forma mas moderna y amigable.

El proyecto se encuentra bien documentado, con información para desarrolladores nuevos disponible en una Wiki.

Utiliza como sistema de control de versiones a Subversion, en tanto que la planificación del proyecto, la Wiki de desarrolladores y el seguimientos de fallos se realizan por medio de Trac (http://trac.edgewall.org/).

El proyecto insume el esfuerzo de 26 desarrolladores por año y tiene algo mas de 104k líneas de código.

2.2 ARQUITECTURA DE LA APLICACIÓN

Una descripción de alto nivel de la aplicación puede obtenerse a partir de la siguiente imagen.

5 Dicho formato resulta muy popular para el intercambio de información entre los distintos programas SIG.

6 Módulo construido para PostgreSQL siguiendo los lineamientos del Open Geoespatial Consortium que permite el almacenamiento y gestión de datos espaciales directamente sobre el motor de bas de datos (bases de datos espaciales, geodatabases, etc.).

7 Excelente y potente software de información geográfico desarrollado por el Ministerio de Defensa de Estados Unidos, liberado en la versión 3 -actualmente en la versión 6.3-, que se continúa desarrollando en la actualidad bajo la modalidad de software libre (el sitio principal de desarrollo se encuentra http://grass.itc.it/).

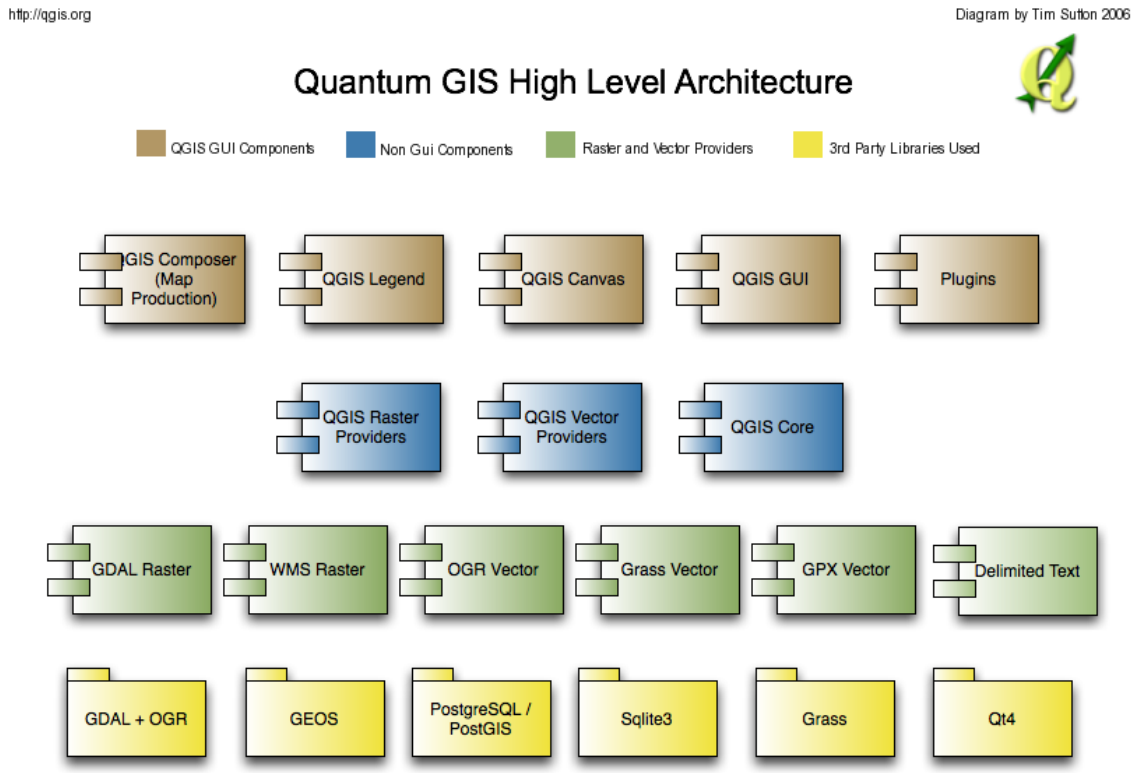


Imagen 2: Visión esquemática de alto nivel de la arquitectura de QGis según Tim Sutton (extraído de http://blog.qgis.org/?q=node/37).

2.3 PRINCIPALES TAREAS REALIZADAS

El trabajo comenzó con el estudio de la infraestructura y organización del proyecto así como las fechas de lanzamiento de nuevas versiones. Resulta relevante la documentación preparada para nuevos desarrolladores donde se indican los principales puntos a tener en cuenta para los nuevos desarrollos (http://wiki.qgis.org/qgiswiki/New_Developers_Welcome_Pack). Se comenzaron leer en forma diaria las listas de correo y se comenzó a participar en el canal QGis del IRC.

Al comienzo de curso el alumno no contaba todavía con el suficiente conocimiento en la estructura del código fuente de la aplicación ni en el manejo adecuado de las bibliotecas Qt. Sin embargo se contaba con varios años de trabajo en GRASS y como usuario de QGis.

Por tal motivo se continuó solicitando una cuenta para el reporte de errores. En base a ello se comenzó la búsqueda de espacios donde se pudiera colaborar con el código.

Se comenzó con el desarrollo de un módulo que permitiera la conversión de coordenadas directamente desde el portapapeles o archivos csv. Sin embargo, en base a las conversaciones mantenidas vía correo electrónico y en IRC con Tim Sutton y a las necesidades existentes, se opto por colaborar en el pasaje de código fuente con dependencia a la versión 3 de las bibliotecas Qt a la versión 4[8].

Parte del trabajo se encuentra documentado en el sistema de seguimiento de errores del proyecto. Todas las modificaciones fueron enviadas en formato patch (svn diff) al sistema de seguimientos de fallos y/ó al correo electrónico de Tim Sutton.

Se ha asumido el compromiso de participar en la migración de todo el código a la versión 4 antes

8 La versión estable al momento de escribir este informe es la 4.3.3.

del lanzamiento de la versión 1.0 de la aplicación.

Dicha tarea involucra la revisión y modificación de una buena parte del código fuente de la aplicación y sus módulos[9]. Todavía no se han establecidos los criterios generales -indispensables para una modificación de este tipo- para realizar dicha migración así como tampoco se han distribuido tareas.

9 Se menciona a modo de ejemplo que el módulo de GRASS posee, en al mayoría de sus archivos dependencias de este tipo.

CAPÍTULO 3: APLICACIÓN

3.1 ANTECEDENTES

No se tiene conocimiento de la existencia de antecedentes de trabajo en proyectos de software libre de una aplicación similar en el tema.

Se destaca en la década de los años 90 el programa Ground Water for Windows [GWW001] desarrollado bajo el auspicio de Naciones Unidas. Sin embargo dicho programa -interesante en la época que se lanzó- no se actualizo y todavía puede ser adquirido por un costo que ronda los U$S 250, sin mayores modificaciones en los últimos 10 años.

Por otro lado existen aplicaciones comerciales mas o menos reconocidas y difundidas orientada a cada uno de los aspectos relacionados a la gestión del agua subterránea.

Se pueden nombrar a modo de ejemplo AquaChem (http://www.waterloohydrogeologic.com/software/aquachem/aquachem_ov.htm) orientado a los aspectos relacionados con la calidad del agua (interpretación, gráficos y modelación) y varios orientados a la interpretación de ensayos de bombeo como AQTSOLV (http://www.aqtesolv.com/) y AquiferTest.

Se deduce que el mercado ofrece soluciones parciales a la interpretación de datos, que deben ser adquiridos -en forma separada- por precios generalmente superiores a los U$S 500.

El usuario se encuentra formateando y copiando información de un programa al otro[10]. Esta situación resulta engorrosa e ineficiente.

3.2 ALCANCE

La realización y el lanzamiento de una aplicación que permita almacenar, gestionar e interpretar toda la información hidrogeológica en forma adecuada es una tarea que se extiende mas allá de los objetivos del curso y del máster[11].

Se trata de un proyecto de porte medio, de largo aliento, donde se deben abarcar aspectos relacionados con el diseño y creación de la base de datos; diseño de la arquitectura de la aplicación; diseño de las interfases de gestión de datos; diseño de consultas especificas útiles para el usuario; diseño de utilidades para visualización de perfiles constructivos, litológicos e hidrogeológicos de las perforaciones; interpretación de ensayos de bombeos; validación e interpretación de datos hidroquímicos; visualización de datos de geofísica superficial y de pozo; estandarización de formatos de intercambio de datos con otras aplicaciones y generación de utilitarios; gestión de aspectos relacionados como son el agua superficial, el agua de lluvia y muestras de roca.

Cada uno de estos aspectos deberán ser abordados (o tenidos en cuenta) a lo largo de las distintas etapas del proyecto.

Sin embargo por razones lógicas y de plazos, se abordo en esta instancia la creación de la estructura de la base de datos relacionada con el módulo a construir, la definición de la estructura de la aplicación y creación de un prototipo con las principales funcionalidades estructurales del diseño.

10 Generalmente no se utiliza ODBC para la obtención de datos y cada uno de los programas posee un formato de ingreso predefinido.

11 Al menos en términos de plazos.

3.3 REQUERIMIENTOS

Los requerimientos mínimos impuestos para aplicación fueron: 1 - el sistema será capaz de ejecutarse en plataformas GNU/Linux, Windows (lo más amplio posible 98/NT/2000/Me/2003/98/XP/Vista) y Mac OS X[12]; 2 - La aplicación será extensible por módulos (plugins) que podrán ser desarrollados en un espectro -lo más amplio posible- de lenguajes populares. En principio, se podrán desarrollar módulos en el lenguaje C++. No se descarta a mediano plazo, la posibilidad de generar extensiones a la aplicación en lenguajes mas amigables como por ejemplo Python; 3 – La aplicación deberá contar con una base de datos adecuada para el almacenamiento de la información; 4 – Los requerimientos para la instalación y ejecución deberán ser mínimos, por ejemplo no se desea que el usuario deba instalar y configurar previamente una base de datos en un motor de base de datos[13].

La aplicación será capaz de gestionar toda la información almacenada en la base de datos hidrogeológica, así como la información relacionada a los usuarios del sistema, los parámetros de configuración y utilidades de distintos tipos (exportación, importación, cambios de coordenadas).

Resulta muy importante contar con módulos de interpretación de datos. Se destacan entre ellos (listado a modo de ejemplo, no exhaustivo):

- cálculos de balances hidroquímicos y su clasificación, diagramas hidroquímicos de Piper, Stiff, Schoeller, Balances, regresión, series temporales, etc.;
- diagramas litológicos y constructivos de las perforaciones almacenadas;
- interpretación de ensayos de bombeos.

3.4 HERRAMIENTAS

Se describen a continuación el conjunto de herramientas evaluadas y utilizadas.

3.4.1 SELECCIÓN DE LA LICENCIA[14]

Se seleccionó la licencia GPL pues en términos generales maximiza la libertad de los usuarios finales. Se desea además que los desarrollos posteriores y/o derivados continúen siendo libre por lo que la cláusula copyleft resulta adecuada.

En cuanto a las bibliotecas, se optó que estén bajo la misma licencia. Sin embargo, no se descarta en el futuro que alguna de las bibliotecas nuevas (principalmente las específicas a la hidrogeología) que utilice el programa sean liberadas bajo otro tipo de licencia, como por ejemplo la LGPL. Permitiendo de esa manera su enlace desde aplicaciones comerciales.

3.4.2 SISTEMA OPERATIVO DONDE SE REALIZA EL DESARROLLO

Todo el desarrollo fue realizado en sistema operativo GNU/Linux con distribuciones Debian y Ubuntu. Además de ser el sistema operativo que utiliza el alumno, se entiende que presenta mejores utilidades para las tareas de administración (en modo consola).

3.4.3 MOTOR DE BASE DE DATOS

Los principales motores de bases de datos en software libre se encuentran disponibles para los

12 El alumno posee un profundo desconocimiento en cuanto al sistema operativo MAC. Se agrega aquí como un requerimiento pues se tiene conocimeinto que las bibliotecas Qt funcionan correctamente sobre esa plataforma, pero se aclara que no se realizaron pruebas de ningún tipo con ese sistema operativo.

13 Se entiende que el público objetivo son usuarios de Windows que no poseen conocimientos avanzados de computación y base de datos.

14 Se presenta a la licencia aquí como una herramienta para la popularización del software y la mejora del conocimiento y gestión del recurso a través de la libertad del uso de la aplicación.

sistemas operativos Windows, Linux y Mac (PostgreSQL, MySQL, Firebid, SQLite, etc). Sin embargo, teniendo en cuenta las premisas de la aplicación, es decir, software de escritorio, de fácil instalación y configuración y sin requerimientos previos, se opto por SQLite (http://www.sqlite.org/).

El principal inconveniente que surge de esta selección es la incapacidad de SQLite de asegurar las claves foráneas. Esta restricción fue solventada con la creación de triggers.

No se evaluaron las posibilidades que presentan versiones "embedded" de Firebird y MySQL. Sin embargo debido a que la sintaxis SQL resulta casi idéntico[15] para las base de datos analizadas y que Qt presenta clases de abstracción adecuadas, podrá modificarse el motor de base de datos sin comprometer las funcionalidades de la aplicación.

3.4.4 Creación de la estructura de la base de datos

Se generó la estructura de la base de datos -en SQLite y en PostgreSQL- relacionada con el desarrollo inicial de la aplicación. Además de las respectivas consolas de administración -sqlite y psql- se utilizó SQLite Manager y pgadmin3.

Sqlite Manager es un módulo para Firefox que resulto particularmente útil en las tareas de administración de la base de datos debido a que la consola de sqlite presenta poca funcionalidades (si se la compara con psql).

En la figura siguiente se muestra una captura de la pantalla de SQLite Manager donde se muestra la estructura de una tabla de la base de datos.

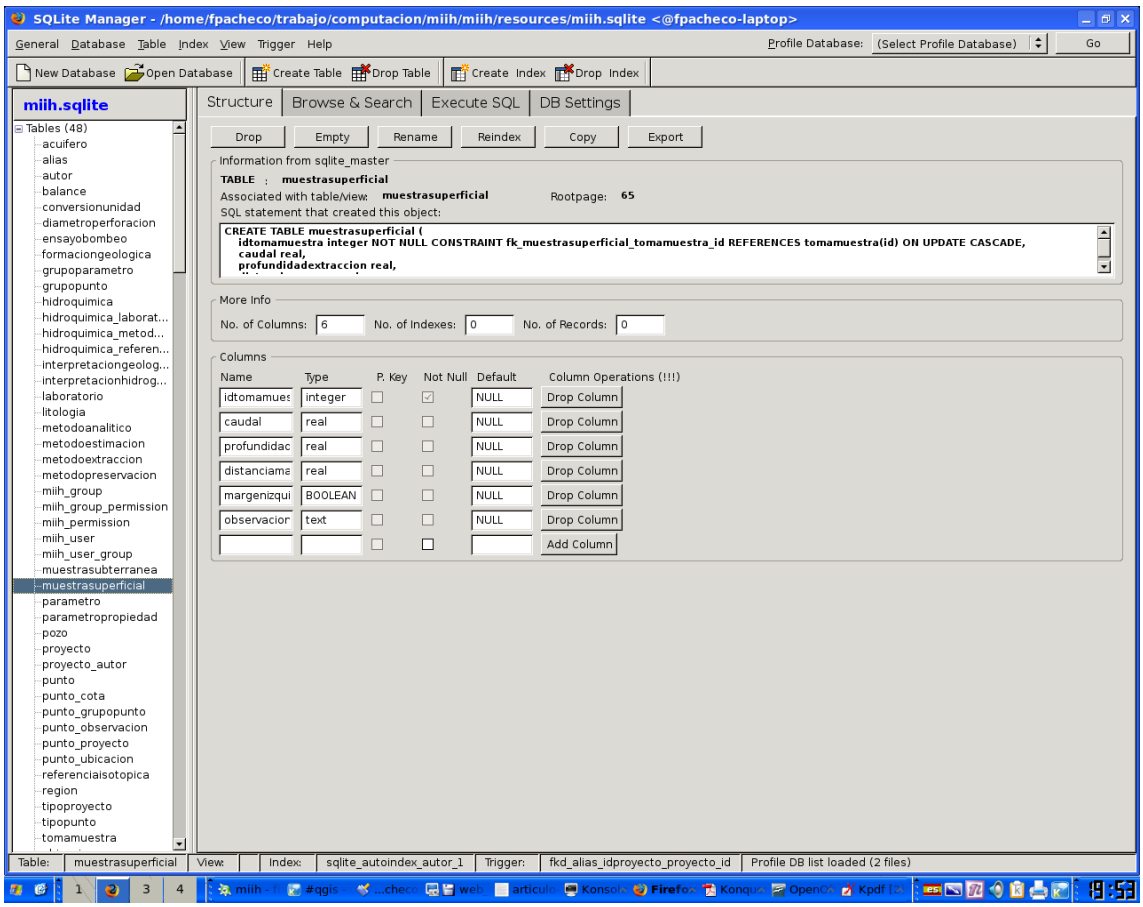


Imagen 3: Tareas de administración de la base de datos realizadas con SQLite Manager.

15 En realidad debería ser idéntico pero algunos motores tienen tipos de datos especiales que se denominan distinto.

3.4.5 Lenguaje de programación

En base a los objetivos del trabajo, a los requerimientos de la aplicación y tratando de utilizar un lenguaje de programación donde el alumno no tuviera experiencia, se seleccionó C++ (http://www.open-std.org/JTC1/sc22/WG21/).

3.4.6 Bibliotecas

Los requerimientos establecidos para la aplicación imponen la utilización de bibliotecas multi plataformas o generadores de lenguajes intermedios con máquinas virtuales capaces de ejecutarse en los sistemas operativos objetivo.

Como bibliotecas multiplataforma se evaluaron wxWindows, GTK+ y Qt (las dos últimas ampliamente utilizadas). En tanto que como máquinas virtuales se evaluaron Java y Mono (implementación del EMCA 335).

En base a que se trata de un proyecto de investigación y al desconocimiento que tenía el alumno en las bibliotecas Qt se optó por su utilización. El aprendizaje de las mismas fue realizado en base a la lectura del código fuente, la documentación disponible en la distribución y bibliografía seleccionada ([BQT001] y [BQT002]).

Las bibliotecas Qt son desarrolladas por al empresa Trolltech y se distribuyen bajo licencias duales. Debido a que se desea realizar una aplicación en software libre se seleccionó la versión Open Source. Dicha plataforma se encuentra excelentemente documentada y posee herramientas modernas para la mayoría de las tareas del programador.

A modo de ejemplo, sin ser exhaustivos, pueden mencionarse algunos programas populares (comerciales y libres) que las utilizan como KDE, GoogleEarth, Opera, Skype y QGis.

Se utilizo además como elemento de apoyo las bibliotecas STL (Standard Template Library) de C++ adecuadas para el almacenamiento y gestión de datos estructurados de gran tamaño.

3.4.7 Herramientas de desarrollo

Se evaluaron los distintos editores de código disponibles para el lenguaje C++ y las bibliotecas utilizadas. Se destacan vim, gvim, kate, Kdevelop y Edyuk. No se evaluaron Eclipse ni NetBeans.

Se optó por kdevelop debido a su madurez (disponible desde 1998) y funcionalidades. A pesar de ello kdevelop todavía no resulta todo lo flexible que se quisiera como IDE de desarrollo. Se establecieron los formatos del código fuente para kdevelop y vim.

Como editor de interfases gráficas para el usuario se seleccionó designer que es la herramienta distribuida por Trolltech junto con las bibliotecas Qt. Presenta funcionalidades adecuadas para los fines perseguidos.

Como herramienta preconfiguración se seleccionó cmake (http://www.cmake.org/). La selección estuvo altamente condicionada a los conocimientos adquiridos con el proyecto QGis y a lo interesante que resulta su utilización.

3.4.8 Sistema de control de versiones

SourceForge tiene disponible en forma integrada y genérica la posibilidad de utilización de CVS (Concurrent Versions System) y Subversion. El segundo aparece como el sustituto natural del primero. Sin embargo algunos proyectos de gran envergadura en software libre no se encuentran satisfechos con sus funcionalidades y están migrando a otros sistemas.

Subversion resultó adecuado para los objetivos del proyecto.

Kdevelop presenta herramientas adecuadas para la administración de Subversion, sin embargo,

todas las tareas de administración y actualización se realizan desde la línea de comando.

El código fuente se encuentra disponible en el repositorio -solo como lectura para usuarios no registrados- para ser descargado con el comando *svn checkout* *https://miih.svn.sourceforge.net/svnroot/miih/trunk* *miih*.

3.4.9 Documentación del código fuente

La documentación del código fuente es una etapa fundamental en el desarrollo de aplicaciones. Si el código no se documenta durante la creación, resultan escasas las probabilidades que se documente con posterioridad.

En este caso, el código fuente "heredado" del proyecto QGis que ya se encontraba documentado con Doxygen (http://www.stack.nl/~dimitri/doxygen/index.html). En virtud de ello, y en base a que esa herramienta se considera adecuada para los fines perseguidos[16], se selecciono Doxygen como herramienta de documentación de código C++ (y a futuro de Python).

Dicha aplicación puede generar salidas a HTML, RTF, PostScript, PDF con enlaces, HTML comprimido y ayudas de sistemas Unix.

La última versión disponible de la documentación generada en formato HTML se encuentra en el sitio web (http://miih.sourceforge.net/api_doc/). Se muestra a continuación una captura de la pantalla de Firefox en el sitio.

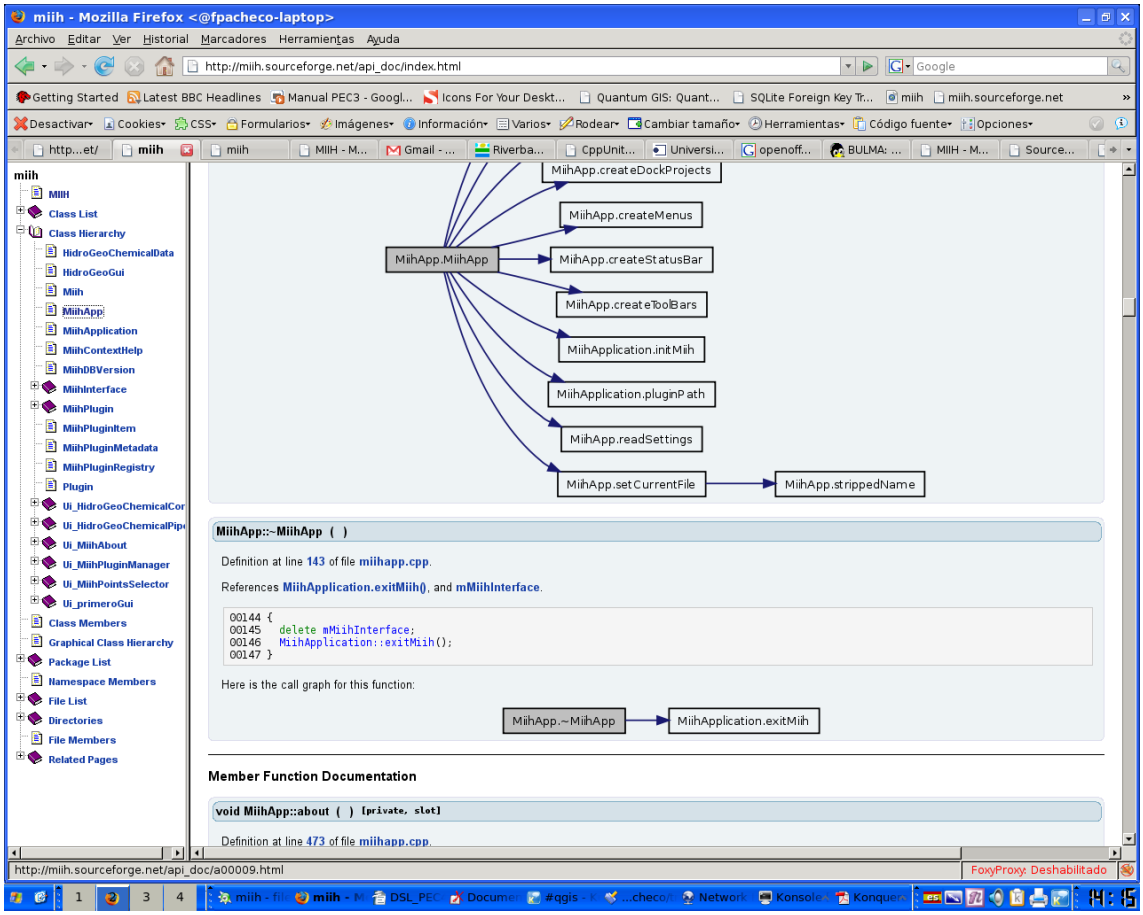


Imagen 4: Documentación de código fuente (http://miih.sourceforge.net/api_doc/).

16 Permite además documentar lista de tareas y relacionados descriptas en el código fuente.

3.5 PLATAFORMA DE DESARROLLO COMUNITARIO

Se optó por utilizar como plataforma para el desarrollo comunitario al sitio SourceForge debido a que posee un conjunto de herramientas mas o menos completa, adecuado para las expectativas iniciales de trabajo, y presenta un importante número de proyectos de software libre.

En primera instancia fue necesario familiarizarse con interfase de trabajo de SourceForge. Luego de ello, se selecciono como sistema de control de versiones a Subversion, se habilitaron el sistema de seguimientos de fallos, tareas, las listas de correo, foros de discusión y el sitio web.

Se comenzó con la elaboración de un sitio web estático que presenta las funcionalidades mínimas. En la medida que se vaya creciendo en tamaño y usuarios, se podrá migrar el sitio a un sistema de manejo de contenido, que permita llevar de mejor manera los temas, noticias y servicios.

En la figura siguiente se muestra una captura del sitio web.

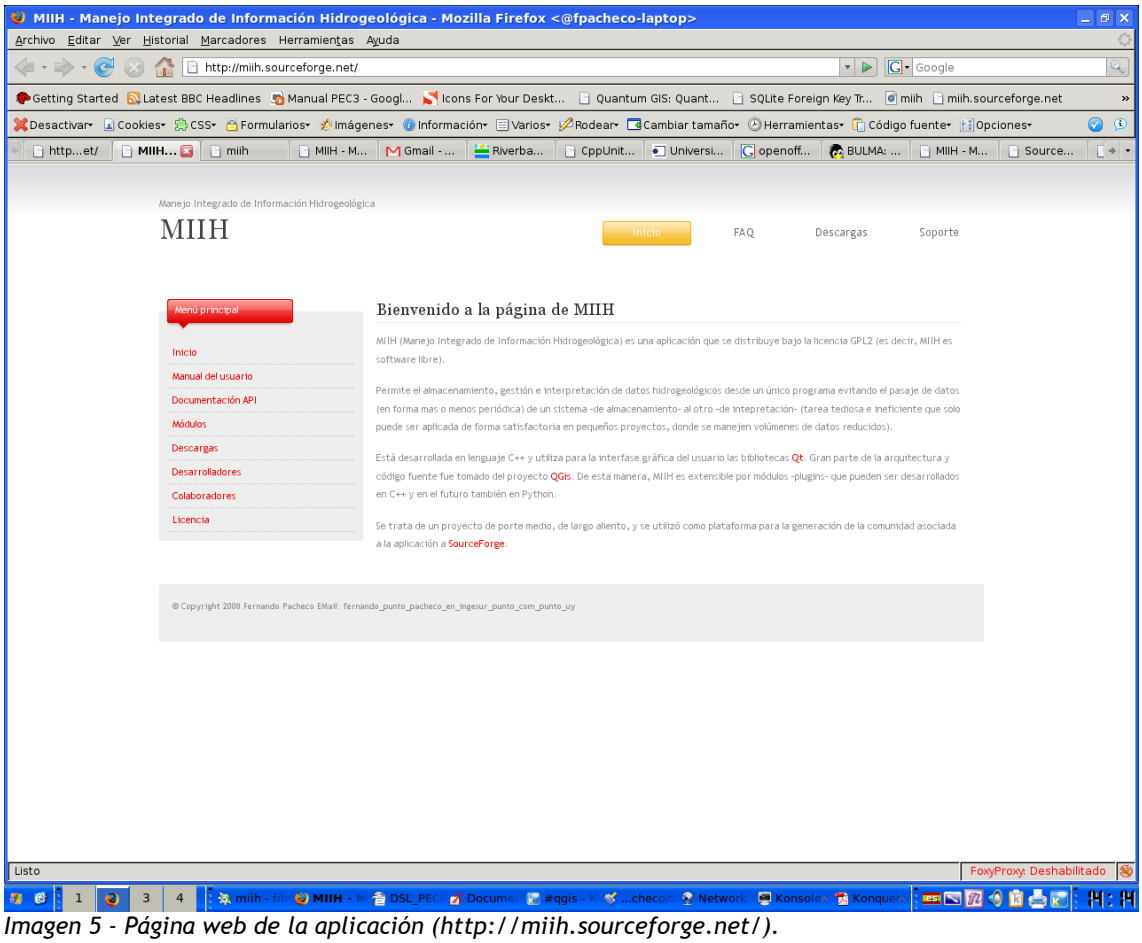


Imagen 5 - Página web de la aplicación (http://miih.sourceforge.net/).

3.6 ESTRUCTURA DE LA BASE DE DATOS

La base de datos fue diseñada en PostgreSQL pensando en las restricciones que impone SQLite.

Los archivos SQL con la información para la creación de la BD en PostgreSQL y SQLite y los esquemas y documentación en formato .dot (grapviz), png y html se encuentran disponibles en el repositorio en el directorio sql. Los mismos fueron generados a partir de la BD de PostgreSQL con la aplicación postgres-autodoc (http://www.rbt.ca/autodoc/index.html).

En la figura siguiente se muestra un esquema general de la base datos.

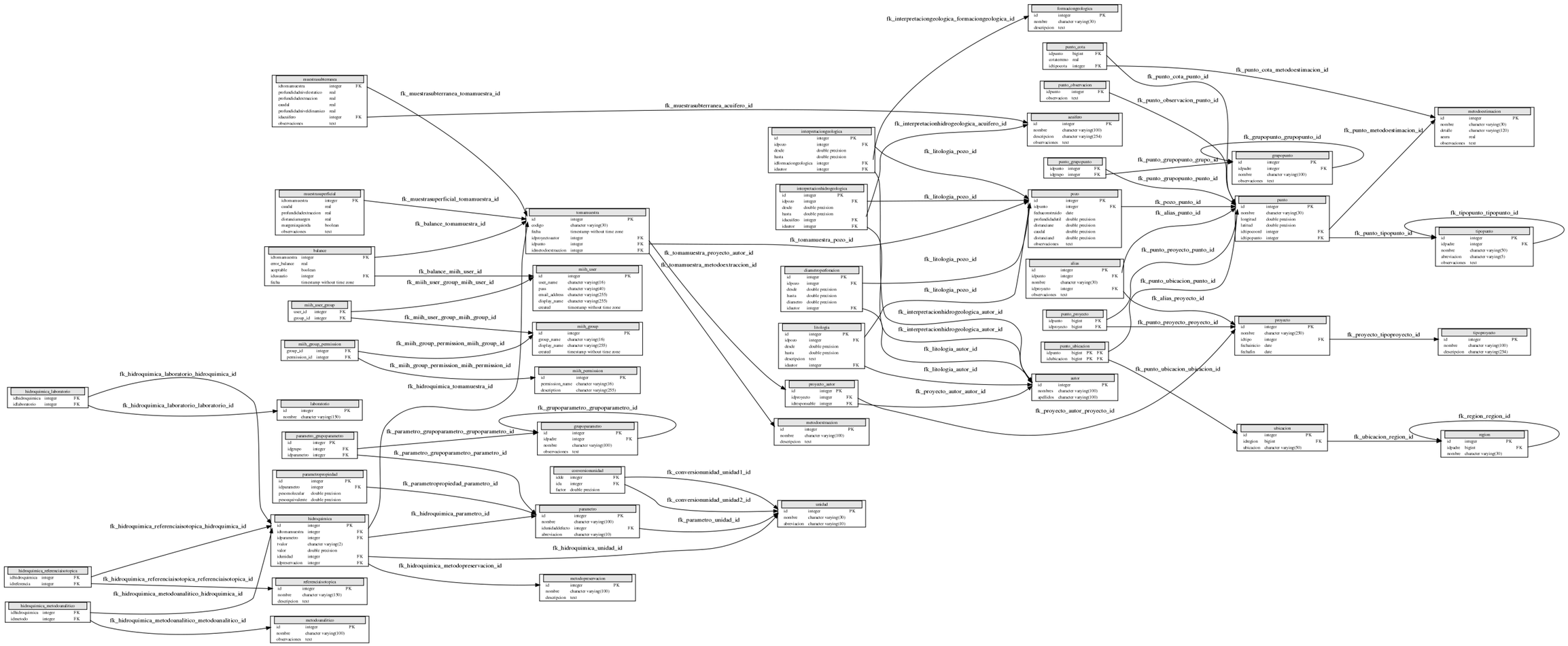


Imagen 6: Esquema de la base de datos.

Se presenta en la tabla siguiente un resumen de las principales tablas:

Nombre	*Detalle*
punto	Tabla donde asigna un identificador único para cada punto de la BD y se almacena datos de coordenadas geográficas, el método de estimación de esas coordenadas (por ejemplo estimadas a mano, a partir de la cartografía 1:1000, con GPS de mano, con GPS geodésico) y el tipo de punto (superficial, subterráneo, perforación, manantial, lluvia, roca, etc.).
pozo	Tabla que tiene relación de clave foránea con la tabla punto donde se almacenan los datos del identificador único del pozo, la fecha de construcción del pozo y los datos relativos a esa instancia.
alias	Otros nombres asignados a ese punto por otras instituciones, proyectos, publicaciones, etc..
region	Regiones y subregiones del area de estudio ó país.
ubicacion	Dirección (zona, calle y numero) de la ubicación del punto.
proyecto	Datos básicos de los proyectos donde se realizan actividades de investigación y se relevan puntos.
autor	Datos básicos de personas que realizan actividades en lso distintos proyectos.
acuifero	Nombre y descripción de los acuíferos detectados en la región.
tomamuestra	Datos básicos del acto de extracción de una muestra sobre cualquier tipo de punto. Es decir, en cada punto se puede sacar algún tipo de muestra (en una fecha específica), con datos adicionales dependiendo del tipo de punto, que se preserva de alguna manera, se envían a algún laboratorio, donde se analizan por algún método analítico, obteniéndose algún resultado.
muestrasubterranea	Datos adicionales asociados al acto de extracción de una muestra para el caso de una perforación.
hidroquimica	Datos básicos de los resultados de los parámetros muestreados en el punto, realizados por algún laboratorio, con algún método analítico, asociado a la toma de muestra respectiva.
parametros	Nombres de los parámetros solicitados al laboratorio. Generalmente componentes químicos, aniones, cationes, isotopos, compuestos complejos, plagicidas, etc..
grupoparametro	Agrupaciones mas o menos lógicas de los distintos parámetros. Por ejemplo para el caso de una muestra de agua, se pueden dividir en grupos tales como mayoritarios, minoritarios, trazas, gases nobles, isótopos naturales, etc..
grupopunto	Agrupaciones mas o menos logicas realizadas por el usuarios para acceder y trabajar rápidamente con un grupo de puntos. Los grupos tienen una relación de pertenencia con los proyectos pero esta tabla permite tomar puntos arbritariamente.
muestrasuperficial	Datos adicionales asociados al acto de extracción de una muestra para el caso de una muestra de un rio, arroyo, lago u otro item

	asociado al agua superficial.
litologia	Cada perforación (pozo) fue realizada en algún momento por una empresa, institución o particular. Como buenas prácticas de perforación se entiende la necesidad de describir los distintos estratos perforados (al menos cada metro) cuya información se almacena en esta tabla.
formaciongeologica	Formaciones geológicas presentes en la región o país.
interpretaciongeologica	La descripción realizada en el momento de la realización de la perforación puede ser asociada ó interpretada a formaciones geológicas (una formación puede ocupar mas de un estrato). En esta tabla se almacena el dato del usuario que lo realiza y la información de la asociación realizada.
interpretacionhidrogeologica	De forma similar se pueden realizar interpretaciones de las descripciones y geología asociándolos a un acuífero determinado (un acuífero puede ocupar mas de un estrato y mas de una formación).

3.7 Estructura de la aplicación

En la figura siguiente se expone una visión de alto nivel de la estructura de la aplicación creada.

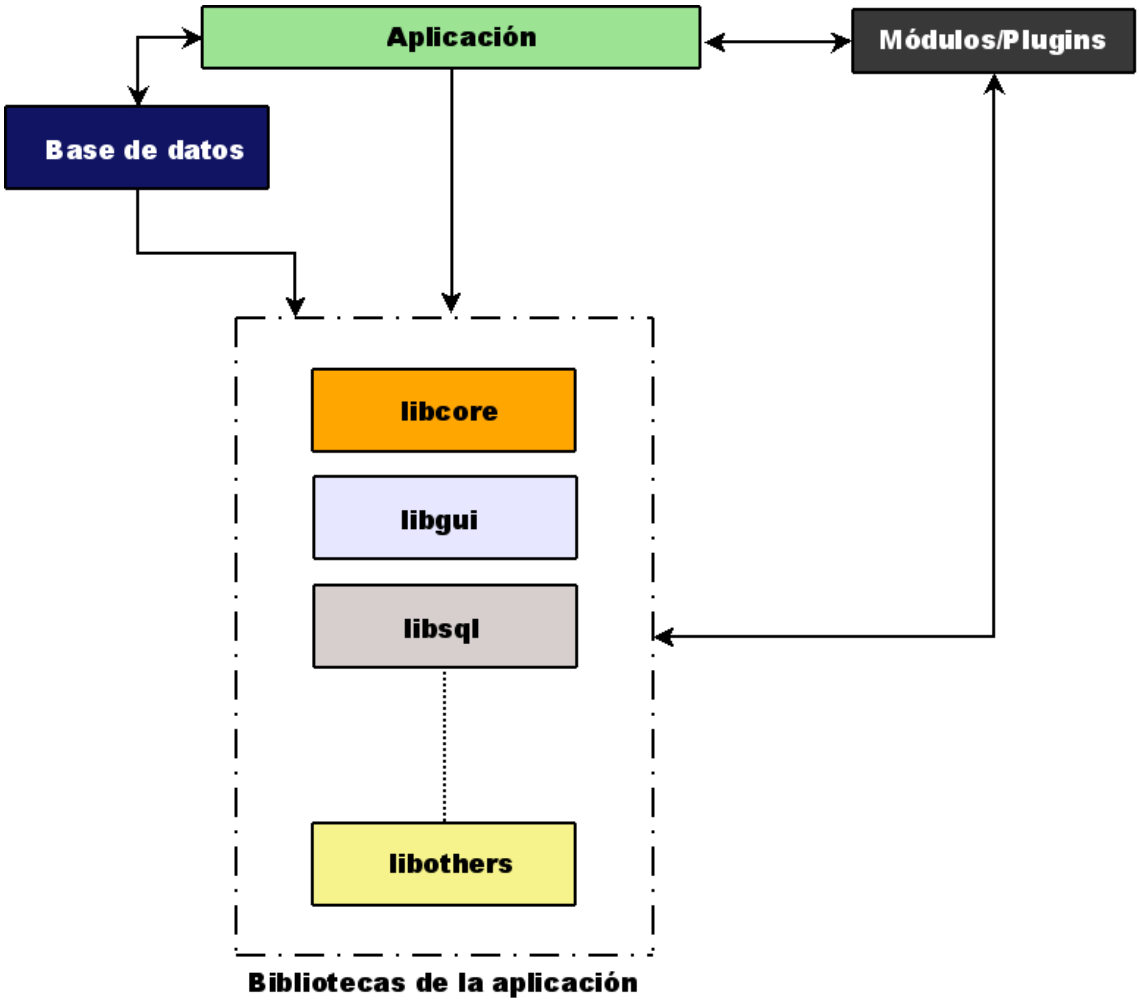


Imagen 7: Esquema de la estructura general de la aplicación.

La estructura de la base de datos y los módulos se exponen en ítem aparte por lo que se exponen a continuación las funcionalidades actuales y/o deseadas de cada una de las bibliotecas presentadas.

En la biblioteca "*libcore*" se almacenan[17], como lo indica su nombre, los aspectos principales de las aplicación. Se destaca la presencia de miembros estáticos que quedan disponibles para ser consultadas desde las otras componentes, a partir de funciones y procedimientos generados para tal fin.

En la biblioteca "*libgui*" se almacenan un conjunto de controles (widgets) específicos de la aplicación que pueden ser utilizados desde la aplicación central y/o desde los módulos. En ella también se definen las interfases expuestas por la aplicación, disponible para el uso en los módulos.

La biblioteca "*libsql*"[18] almacena los procedimientos y reglas para la gestión de los datos de la base de datos. Se prevé la utilización de las clases QsqlDatabase con los drivers presente en Qt para el acceso a SQLite.

17 Y quedan disponibles para aquellos que la enlazan.
18 Todavía en fase muy preliminar.

El modelo de datos a utilizar será QslQueryModel[19] en la mayoría de los casos, en tanto que cuando sea conveniente se podrán utilizar QsqlTable y QsqlRelationalModel.

Estos modelos interactúan en forma convencional con los controles en los que normalmente se exponen los datos al usuario.

Por último, se pretende representar con "*libothers*" a otras bibliotecas de la aplicación que pueden ser construidas en el futuro (seguramente serán necesarias) donde se almacenarán funciones específicas, por ejemplo de la temática hidrogeológica[20].

3.8 MÓDULOS

La estructura modular de la aplicación garantiza su extensibilidad.

Este elemento permitirá que a lo largo de la vida útil del producto, se adicionen nuevas funcionalidades sin mayores esfuerzos y en forma independiente de la aplicación central[21].

Las funcionalidades necesarias para la comunicación de los módulos y la aplicación se encuentran definidas en los archivos miihinterface.h y miihinterface.cpp[22]. En ellos se exponen las funciones y métodos que permiten acceder a la interfase gráfica principal.

Se tomo como referencia el generador básico diseñado por el proyecto QGis, un archivo sencillo en python que utiliza plantillas, con el cual se puede generar un esqueleto mínimo para la creación de un nuevo módulo.

Se modificaron las plantillas para que incluyeran las nuevas bibliotecas de la aplicación (libsql), así como las cabeceras adecuadas.

Con esa modificación se genero un módulo de ejemplo denominado primero.

Además de ello se comenzó con el trabajo de un módulo algo mas complejo, para la validación e interpretación de de datos hidroquímicos.

3.9 TEST

No se han realizados test para el código fuente. La selección de una plataforma de testeo resulta prioritaria. Se han evaluado someramente CppUnit y las herramientas nativas de Qt.

3.10 INTERNACIONALIZACIÓN E IDIOMA

La aplicación fue construida en idioma ingles.

La plataforma Qt viene con un conjunto de utilidades que permiten -a partir de ciertas reglas básicas presentes en el código fuente- la extracción del texto a traducir. La regla principal de trabajo es la utilización de la función tr() sobre todo el texto que se desee traducir.

Utilizando el programa lupdate, se puede extraer posteriormente el texto requerido. Los parámetros que se deben pasar a dicho programa pueden ser el archivo de proyecto qmake (desde donde se extrae el listado de archivos a procesar) o utilizar directamente el listado generado en forma independiente. El archivo extraído tiene formato XML.

En base a este último y utilizando otra aplicación disponible en la plataforma denominada Qt Linguist se realiza la traducción. La aplicación permite además visualizar los formularios y

19 Extendido (subclassing) para proporcionar funcionalidades de escritura y visualización.
20 Pueden ser bibliotecas construidas desde el inicio o intermediarios sencillos a bibliotecas complejas existentes.
21 Inclusive desde otros proyectos independientes.
22 En virtud que la aplicación no se encuentra totalmente definida, se estima que dichas interfases pueden presentar modificaciones a mediano plazo.

controles traducidos. Luego de ello se puede generar el archivo binario que será utilizado por el programa para modificar el idioma de la aplicación al deseado por el usuario.

Con este procedimiento y con el llamado a las funciones adecuadas en el código fuente que toman y establecen la localización del sistema se logra el objetivo perseguido.

Todo el software fue formateado de la manera requerida para que permita su visualización en distintos idiomas.

Se generó un script (muy sencillo) que permite extraer el texto a traducir y se realizaron las pruebas de traducción.

Vale la pena aclarar que este sistema resulta similar al de otras utilidades disponibles en Linux, y en particular, las bibliotecas GTK+ que utilizan directamente xgettext.

En la figura siguiente se muestra una imagen de la aplicación Qt Linguist en las pruebas realizadas.

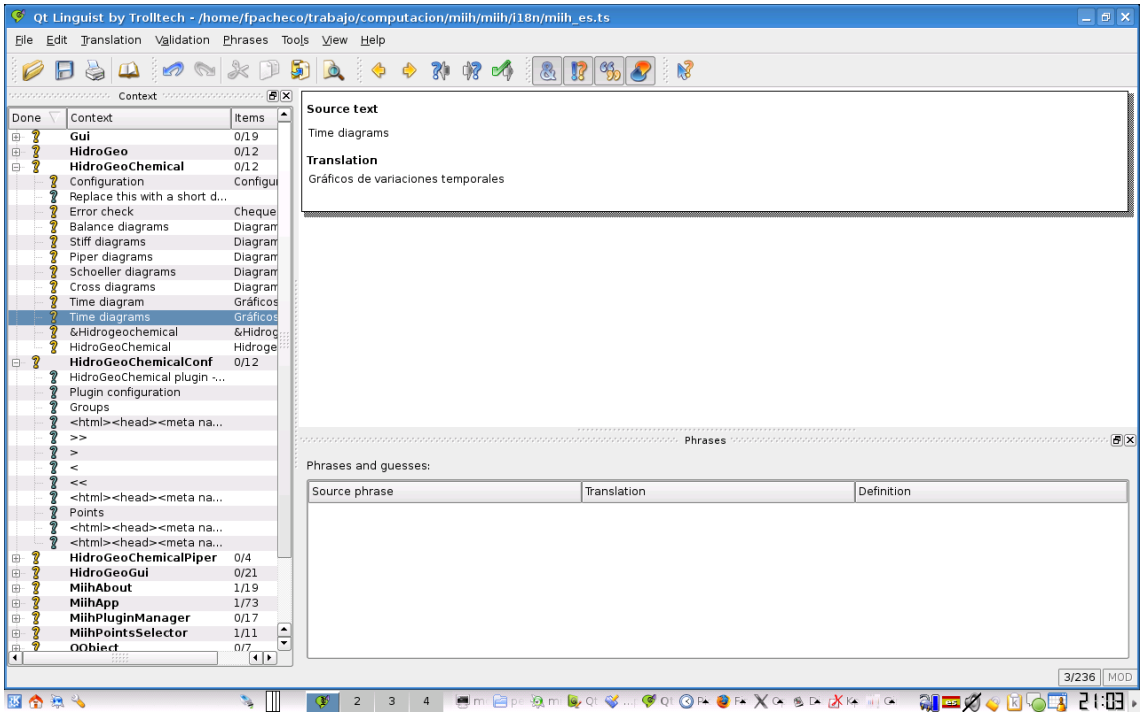


Imagen 8: Ejemplo de proceso de traducción con QLinguist.

3.11 Estado de arte

La aplicación se encuentra todavía en estado de franco desarrollo. Solamente se han abordado los aspectos relacionados con su esqueleto, es decir: la aplicación principal, la separación en bibliotecas, los módulos y el esbozo de la interfase con el usuario.

Como ya fue mencionado todavía quedan por incorporar test al código anterior, incorporar -respecto a los test- una modalidad similar a la de “extreme programming” para el nuevo código[23], así como construir la biblioteca de acceso a datos de la base de datos.

Respecto a la base de datos, se destaca que todavía no esta completa pues fueron abordados solamente los temas necesarios para su coherencia, la construcción inicial de la aplicación y el

23 Es decir pensar el test, plasmarlo y luego escribir el código de la aplicación.

modulo HidroGeoChemical.

En la figura siguiente se muestra el prototipo funcionando en Ubuntu GNU/Linux. Se puede ver además la ventana de configuración del modulo HidroGeoChemical que será el primer módulo operativo útil en el momento del lanzamiento.

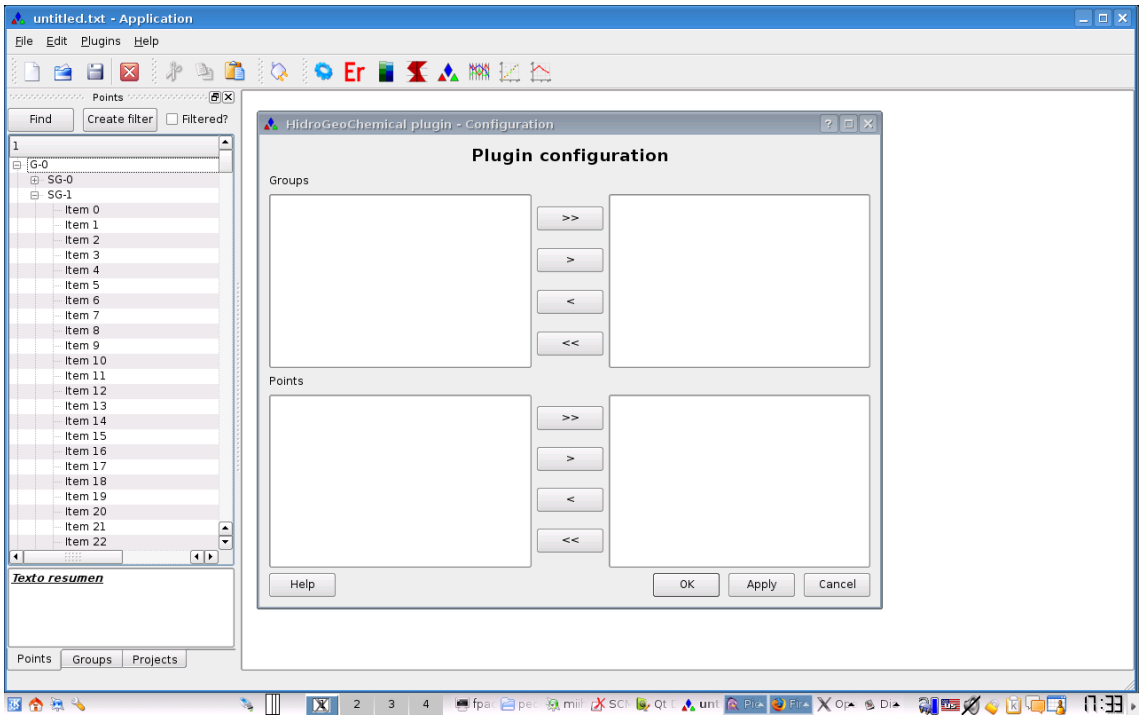
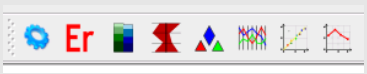


Imagen 9: Prototipo de MIIH ejecutando la etapa de configuración del módulo HidroGeoChemical en Ubuntu GNU/Linux.

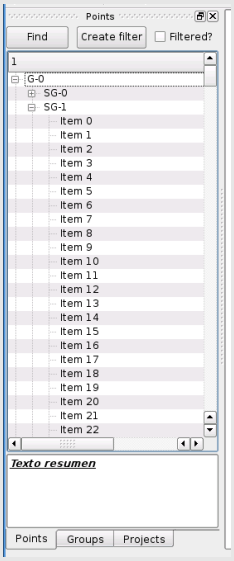
A continuación se describen brevemente los aspectos relevantes:

Barras de herramientas de la aplicación (es decir creadas en el ámbito de la aplicación).

Barra de herramienta disponible en la aplicación para módulos sencillos, es decir, la barra se hace disponible desde la aplicación y los módulos pueden agregar botones cuando son cargados. Resulta adecuada para módulos que no requieren mas de un botón para su funcionamiento.



Barra de herramientas generada por módulos que requieren mas de un botón para su funcionamiento (en este caso particular se muestra la barra del módulo HidroGeoChemical).



Panel (dock) de selección, filtrado, información de puntos, grupos y proyectos en la base de datos. Cada control tendrá un menú contextualizado que le permitirá realizar las acciones necesarias.

3.12 AYUDA DE LA APLICACIÓN

A pesar de la "herencia" que presenta el código del sistema de ayuda del proyecto QGis, se han realizado evaluaciones posteriores donde se concluye que la utilización del sistema de ayuda provisto por Qt (Assistant) resulta mas conveniente.

Las razones que pesan a la hora de la conclusión son los de: 1 – el sistema de ayuda de Qt no necesita que el usuario provea datos del programa que se debe utilizar para abrir la ayuda; 2 – presenta la posibilidad de incluir, realizar búsquedas y distribuir -como parte de la aplicación- libros, publicaciones y otros elementos técnicos relacionados con la temática del agua subterránea.

En base a ello, previo al lanzamiento de la versión inicial se pondrá en funcionamiento este sistema.

Se entiende, que la ayuda del sistema no solo esta compuesta por los archivos de ayuda y que en algunos casos la ayuda contextual y el adecuado diseño de la interfase son los mejores elementos para guiar al usuario.

3.13 DISTRIBUCIÓN DE LA APLICACIÓN

3.13.1 INSTALACIÓN A PARTIR DEL CÓDIGO FUENTE DISPONIBLE EN EL REPOSITORIO

A continuación se detallan los pasos a seguir para la obtención y compilación de la aplicación a partir del código fuente presente en el repositorio.

No se hacen diferencias explicitas en cuanto a sistemas operativos ya que las bibliotecas y utilitarios necesarios para la descarga y compilación se encuentran disponibles para Windows y Linux[24]. Todo el procedimiento se realiza en GNU/LInux en modo de consola, en tanto que se muestran capturas de pantalla de Windows cuando se considera necesario.

Los paso necesarios previos a la compilación son:

- descargar e instalar las bibliotecas Qt (se deberá decir que si a la instalación de MinGW) ;
- descargar e instalar cmake;

24 Se advierte nuevamente que no se cuenta con MAC OS X por lo que no se indica el procedimiento para la compilación de la aplicación en ese SO.

- descargar e instalar un cliente svn (por ejemplo TortoiseSVN en Windows y svn en linux).

Descargar la última versión de la aplicación desde el repositorio con el comando svn checkout https://miih.svn.sourceforge.net/svnroot/miih/trunk miih.

Se muestra a continuación una captura de pantalla de la obtención del código fuente desde el repositorio en Windows XP utilizando TortoiseSVN.

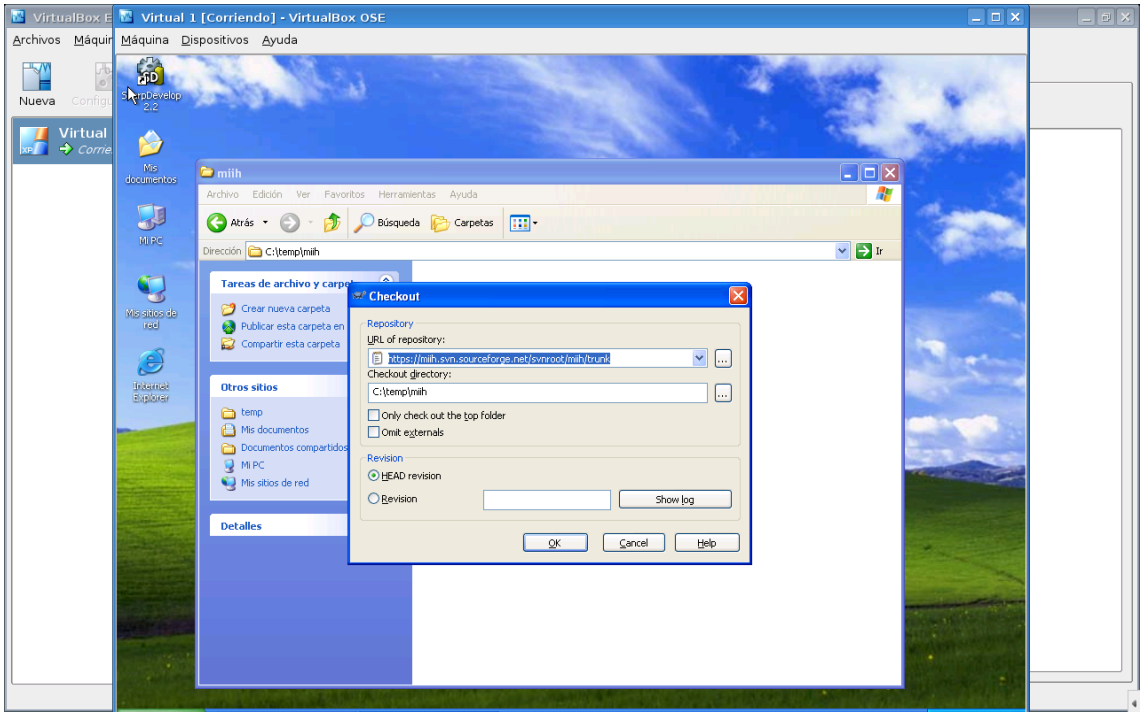


Imagen 10: Obtención del código fuente en Windows XP con TortoiseSVN.

Ejecutar *cmake .* sobre el directorio de la aplicación para la búsqueda, configuración e inclusión de archivos necesarios. En la imagen siguiente se muestra el proceso en Windows XP.

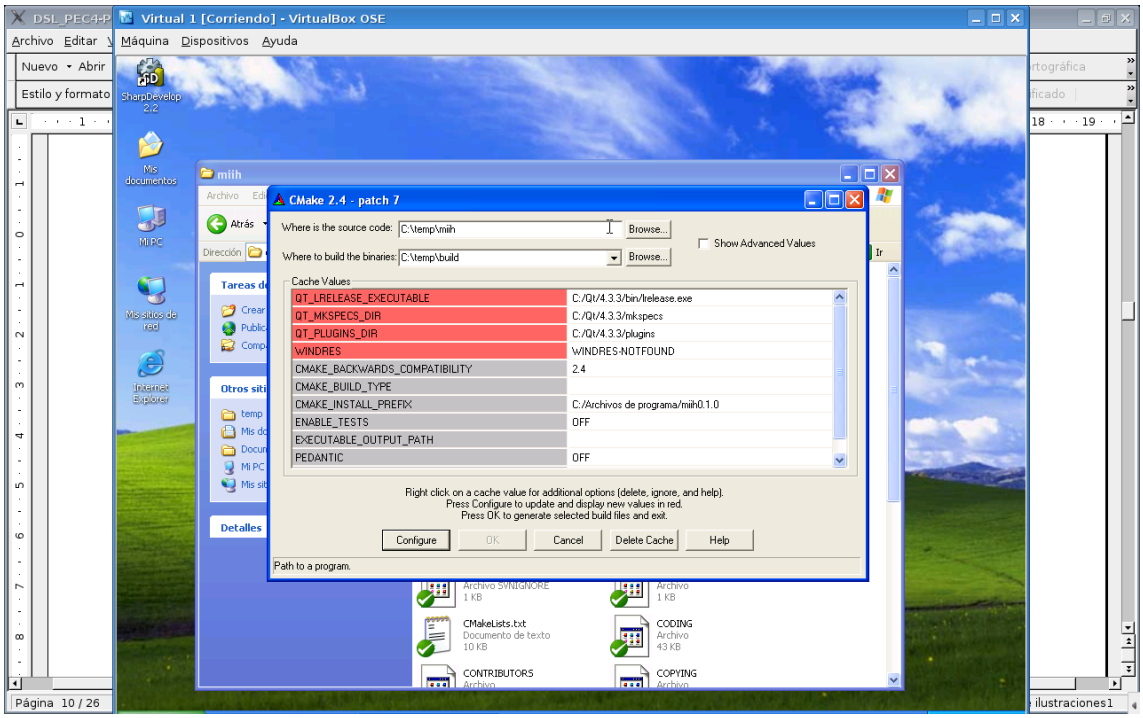


Imagen 11: Ejecución de cmake en Windows XP.

Ejecutar el comando *make* (mingw32-make en Windows) para que se realice la compilación.

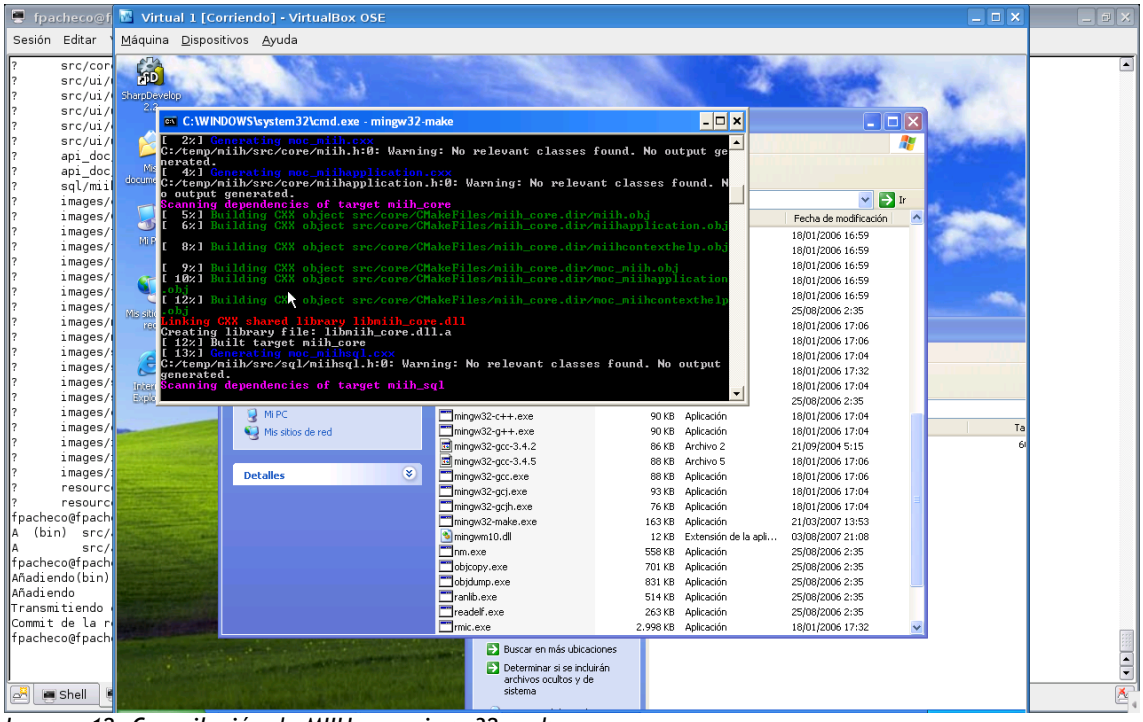


Imagen 12: Compilación de MIIH con mingw32-make.

Luego que culmine la compilación se deberá ejecutar *sudo make install* en linux ó mingw32-make install en Windows para instalar el producto.

Se presentan, generalmente, 3 errores en el proceso preconfiguración y compilación en Windows:

1. cmake no encuentra qmake. Este inconveniente se soluciona fácilmente indicando el camino hasta el ejecutable qmake, presente en el directorio bin de la distribución de Qt;
2. cmake no encuentra windres. Este inconveniente se soluciona fácilmente indicando el camino hasta el ejecutable, instalado en el directorio bin de la distribución MinGW;
3. el usuario con poca experiencia en el uso de MinGW no sabe que el comando make se denomina mingw32-make y que en algunos casos el directorio bin de la distribución de MinGW no se encuentra en el PATH.

Luego que se ha realizado la compilación e instalación, se presenta generalmente un último inconveniente: las bibliotecas Qt requeridas y mingw no se encuentran. Resulta adecuado agregarlas al PATH, copiarlas al directorio del proyecto (poco recomendable) o generar un script que las agregue antes de al ejecución de la aplicación.

En la figura siguiente se muestra una captura de pantalla de la aplicación ejecutándose en una máquina virtual con Windows XP.

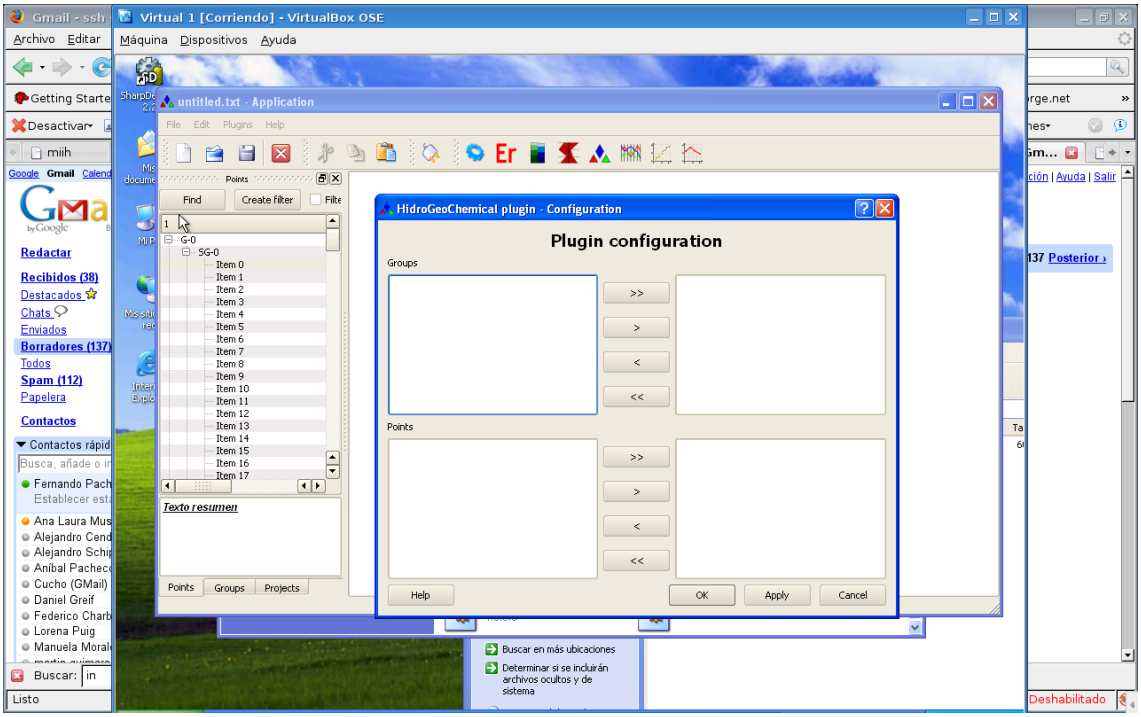


Imagen 13: Prototipo de MIIH ejecutándose en Windows XP.

3.13.2 Instalación a partir de la distribución binaria

No se cuenta con una distribución de binarios (instalables) para ninguno de los sistemas operativos objetivo.

Se comenzará con la creación de una distribución, para instalación, para los sistemas operativos Windows, cuando se cuente con una versión con las funcionalidades mínimas requeridas. Todavía no se ha evaluado las aplicaciones disponibles para dicha operación.

A corto plazo no se prevé la generación de paquetes con binarios para distribuciones GNU/Linux (deb y/ó rpm).

3.14 Principales tareas por realizar

La aplicación se encuentra todavía en estado de franco desarrollo y no se espera que sea lanzada antes del 31 de marzo del 2008.

Como tareas principales relevantes inmediatas se destacan:

- modificar el método de programación, adoptando lo antes posible un sistema de testeo;
- culminar la interfase de usuario;
- modificar el sistema de ayuda de la aplicación;
- avanzar en la implementación de la biblioteca libsql, eso implica realizar las consultas necesarias para el usuarios y personalizar los modelos disponibles en Qt para que sean capaces de realizar modificaciones. En base a ello mejorar la biblioteca libgui;
- con un prototipo en funcionamiento (previo al lanzamiento), continuar con la publicidad apuntando a conseguir colaboradores para test, documentación (manual del usuario), traducción a idiomas seleccionados y mejora del sitio web previo al lanzamiento;
- resolver los inconvenientes menores presentes en la ejecución en Windows (bibliotecas necesarias de la plataforma Qt y mingw no presentes en el PATH) y seleccionar una aplicación para realizar la distribución en ese sistema operativo.

3.15 Mantenimiento

El mantenimiento de aplicación, el seguimiento de errores y los nuevos desarrollos serán realizados y coordinados a través de la página del proyecto.

Los usuarios generales podrán reportar errores y enviar archivos patch para repararlos y/o mejoras, en tanto que existirán usuarios con privilegios de escritura sobre el repositorio que serán los encargados del desarrollo general y mantenimiento. Estas tareas pueden llevarse a acabo en forma adecuada con las herramientas disponibles en el sitio.

3.16 Promoción de la aplicación

Las tareas relacionadas a la promoción, discusión y participación del proyecto identificadas al comienzo del proyecto fueron:

1. sobre la página web del sitio de desarrollo por medio de listas de correos (de noticias, de usuarios, de desarrolladores, de documentación, etc.), noticias y demás;
2. a nivel personal tratando de involucrar a referentes académicos relacionados con el tema;
3. a nivel académico a través de publicaciones científicas con el apoyo universitario;
4. a nivel académico a través de la promoción y trabajos con de alumnos de grado y postgrado de la aplicación;
5. a nivel de sitios de promoción de noticias de aplicaciones (por ejemplo http://www.freshmeat.net/);
6. a otros niveles por medio de los usuarios;
7. a nivel de usuarios de otras aplicaciones relacionadas promocionando las posibilidades de intercambio semiautomático de datos;

8. a través de la búsqueda de financiación internacional (para desarrollos específicos) relacionado con la temática.

Debido a la carga en horas de desarrollo y a la imposibilidad de mostrar hasta la fecha un prototipo con las funcionalidades básicas se opto por profundizar en los puntos 4 y 6. En el momento que se cuente una versión compilada para Windows con funcionalidades básicas se profundizarán los puntos 5 y 8.

CAPÍTULO 4: CONCLUSIONES

Se presenta a continuación las principales conclusiones obtenidas a lo largo del curso y del trabajo realizado.

Se han separado en conclusiones generales y particulares. Las primeras están orientadas al proyecto y herramientas en general, en tanto que las segundas abarcan aspectos relacionados a las tareas especificas realizadas y a la aplicación en si misma.

4.1 GENERALES

- La plataforma de desarrollo Qt se presenta como una excelente herramienta para la creación de aplicaciones en los entornos de software libre. Se encuentra muy bien documentada y posee funcionalidades adecuadas -a distintos niveles- para la mayoría de las tareas que realiza el programador. La curva de aprendizaje es buena y se logra en poco tiempo una comprensión general de las mismas. La disponibilidad del código fuente es un factor determinante para la velocidad de aprendizaje.
- El soporte a contenedores de datos estructurados (vector, set, map, etc.) presente en Qt resulta pobre. Debido a ello se utilizó en al mayoría de los casos la STL de C++.
- La posibilidad de contar con proyectos de SL que puedan ser utilizados como punto de partida en algunos de los aspectos a desarrollar se presenta como una ventaja importante.
- SourceForge se presenta como un sitio adecuado para dar soporte a proyectos pequeños y medios de desarrollo comunitario. No se encontraron dificultades en la administración.
- Las estimaciones de tiempo previstas en la planificación no fueron adecuadas. En general las tareas llevaron mas tiempo que lo estimado.

4.2 PARTICULARES

- A pesar del escaso tiempo con el que se contaba se logró montar el esqueleto de una aplicación compleja. Las dos razones que contribuyeron al éxito son: 1 – las bibliotecas Qt facilitan el desarrollo; 2 – se contó con una aplicación ya realizada que permitió abordar el tema de otra manera.
- No se encontraron inconvenientes para la preconfiguración, compilación y ejecución de la aplicación en Windows y Linux.
- Se debe cambiar la metodología de desarrollo. Se deben utilizar test para cada uno de los elementos funcionales de código, ya que la utilización de mensajes o un debugger resulta sumamente engorrosa y se debe realizar nuevamente cada vez que se cambia el código fuente.
- QGis se presenta como una excelente herramienta en el ámbito de los sistemas de información geográfica. La potencia que le brinda la expansibilidad por módulos es su principal fortaleza. Entre ellos se destaca el modulo de acceso a GRASS, con corto tiempo de vida y con mucho por mejorar, que revitaliza la interfaz de ese SIG con el usuarios. Su principal debilidad es la gran cantidad de errores que se presentan en cada una de las versiones. Sería conveniente que para el lanzamiento de la versión 1.0 se eliminaran aquellos que tienen mayor impacto sobre el usuario.

- La aplicación ha crecido notablemente en cuanto a sus funcionalidades, pero presenta inconvenientes de fallos que en algunos casos resultan contraproducentes y desestimulan su uso por parte del usuario. En tal sentido se entiende que esa situación debe ser corregida lo antes posible.
- Gran parte del código fuente del proyecto QGis todavía presenta dependencias con la versión 3 de las bibliotecas Qt, es decir utiliza Q3Support. El pasaje de dicho código a la versión 4 requiere un esfuerzo importante para el cual se deberán establecer criterios básicos de sustitución adecuada para cada uno de los controles y modelos involucrados.

BIBLIOGRAFÍA

GWW001: BRATICEVIC, D.; KARANJAC, J., GROUND WATER SOFTWARE FOR WINDOWS (GWW).
BQT001: JOHAN, THELIN, FOUNDATION OF QT DEVELOPMENT.
BQT002: BLANCHETTE, JASMIN; SUMMERFIELD, MARK, C++ GUI PROGRAMMING WITH QT 4.

INTERNET

TROLLTECH. QT OPEN SOURCE EDITION. VERSION 4.3.3. HTTP://TROLLTECH.COM/DOWNLOADS/OPENSOURCE.
DOXYGEN, DIMITRI VAN HEESCH. DOXYGEN PROJECT. HTTP://WWW.STACK.NL/~DIMITRI/DOXYGEN/.
SQLITE3. SQLITE PROJECT. HTTP://WWW.SQLITE.ORG/.
POSTGRESQL. POSTGRESQL PROJECT. HTTP://WWW.POSTGRESQL.ORG/
QGis. QGis Project. http://www.qgis.org/