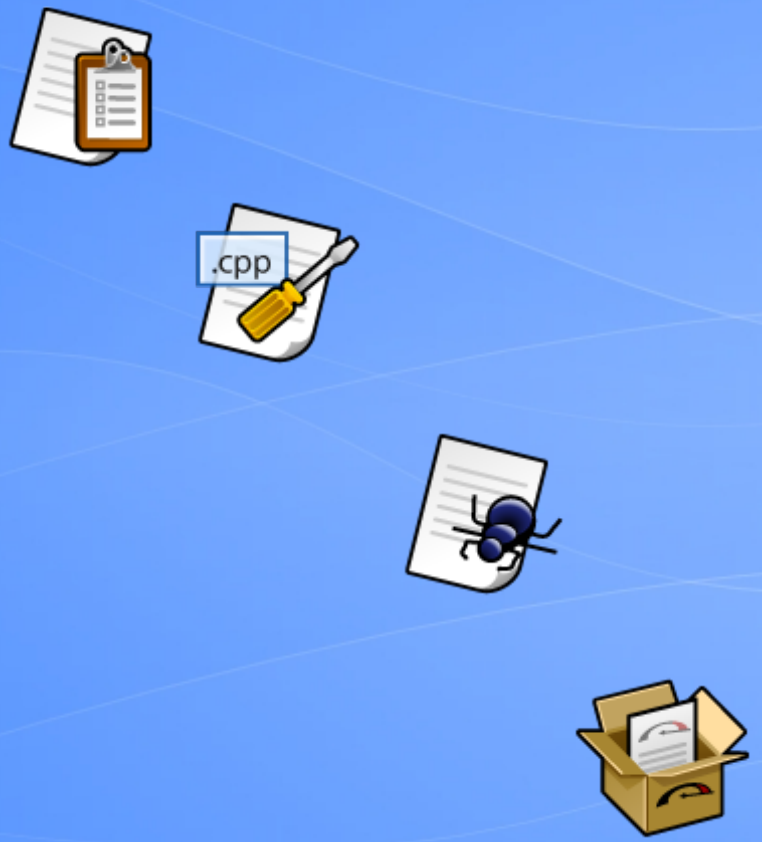
Desarrollo de Aplicaciones con Qt

Rodolfo Martínez
macr111080@yahoo.com.mx
LinuxCabal - Flisol2007

Ciclo de Vida del Desarrollo

- Requerimientos
- Análisis y Diseño
- Programación
- Pruebas
- Implementación
- Mantenimiento



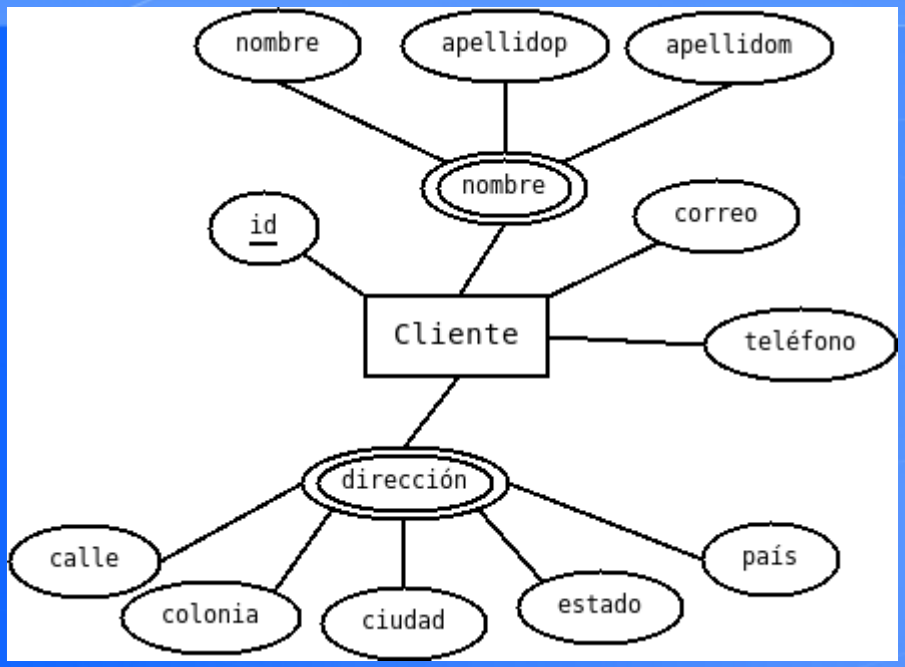
Requerimientos

- Crear una base de datos con la información de los clientes de “MiEmpresa”
- Mostrar la información de la base de datos en una interfaz gráfica.
- Agregar clientes nuevos desde la interfaz gráfica.
- La aplicación debe ser multiplataforma.

Análisis y Diseño

- Diccionario de datos, diagramas de entidad relación, UML, diagramas de Gantt, y todas esas cosas aburridas que nos enseñan en la escuela.
- Diseño de la interfaz gráfica.
- Qt como plataforma de desarrollo.
- MySQL como base de datos.
- Acceso modular (centralizado) a la base de datos a través de una librería dinámica.

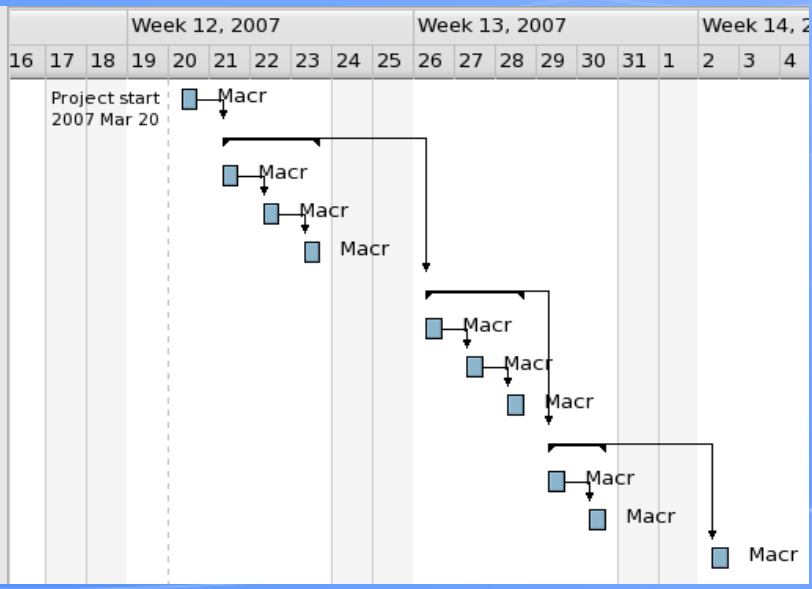
Diagramas



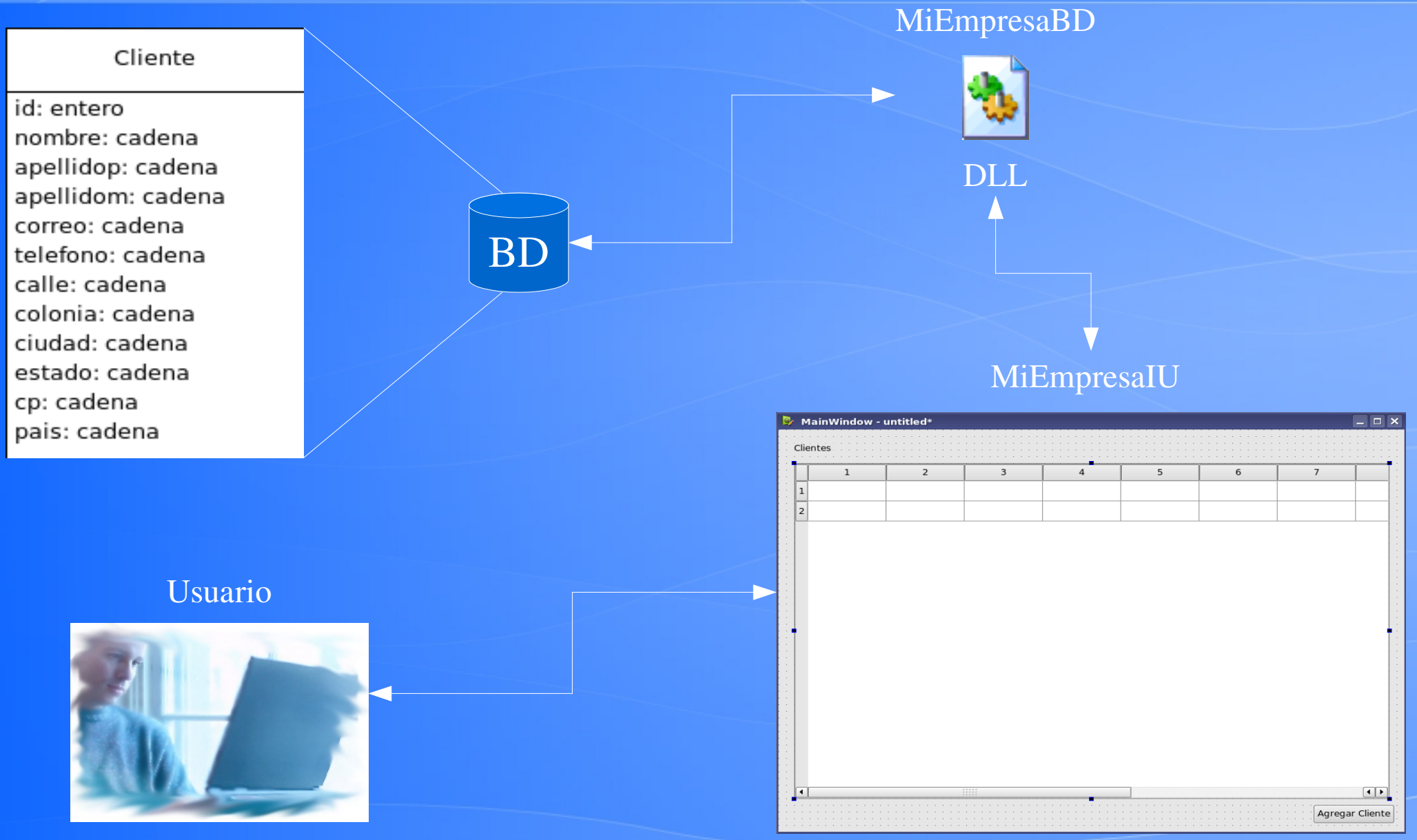
El diagrama de entidad relación los podemos crear con Dia o Kivio

La administración del proyecto la podemos llevar con Planner

WBS	Name	Work
1	Requerimientos	1d
2	**Análisis y diseño**	**3d**
2.1	Diccionario de datos	1d
2.2	Diagrama de entidad relación	1d
2.3	Diagrama de Gantt	1d
3	**Programación**	**3d**
3.1	Crear la base de datos	1d
3.2	Crear la interfaz para acceso a	1d
3.3	Crear la interfaz de usuario	1d
4	**Pruebas**	**2d**
4.1	Pruebas unitarias	1d
4.2	Casos de uso	1d
5	Implementación	1d



Estructura del programa



Qt y sus herramientas de desarrollo

- **Qt** es una plataforma de desarrollo basada en C++ la cual incluye una extensa librería de más de 400 clases (interfaz de usuario, base de datos, XML, conectividad, OpenGL, multithreading y más).
- **Qt Designer** es una interfaz para crear y diseñar interfaces gráficas.
- **Qt Linguist** es un conjunto de herramientas que facilita la traducción de la interfaz a varios idiomas.
- **Qt Assistant** es un visor para la documentación de Qt.
- **qmake** es un generador multiplataforma de archivos “Makefile”. Genera los árboles de dependencia y los “Makefiles” específicos para cada plataforma.

Los Archivos .pro

comun.pro

```
1  CONFIG += debug_and_release warn_on
2
3  CONFIG(debug, debug|release) {
4      TARGET_DIR = debug
5  } else {
6      TARGET_DIR = release
7  }
8
9  DESTDIR = $$TOP_DIR/$$TARGET_DIR/$$TEMPLATE
10
11 win32:MIEMPRESABD_LIB = -L$$TOP_DIR/$$TARGET_DIR/lib -lMiEmpresaBD1
12
13 unix:MIEMPRESABD_LIB = -L$$TOP_DIR/$$TARGET_DIR/lib -lMiEmpresaBD
```

MiEmpresaBD.pro

```
1  TEMPLATE = lib
2  VERSION = 1
3  CONFIG += dll
4  QT -= gui
5  QT += sql
6
7  DEFINES += VERSION=$$VERSION
8
9  TOP_DIR = ..
10 include($$TOP_DIR/comun.pro)
11
12 DEPENDPATH += . include
13
14 INCLUDEPATH += . include
15
16 HEADERS += include/Nombre.hpp \
17     include/Direccion.hpp \
18     include/Cliente.hpp \
19     include/MiEmpresaBD.hpp
20
21 SOURCES += Nombre.cpp \
22     Direccion.cpp \
23     Cliente.cpp \
24     MiEmpresaBD.cpp
```

Plantilla para librería

Quitamos el soporte para interfaz de usuario y agregamos el soporte para SQL

Plantilla para aplicación

Agregamos dependencia MiEmpresaBD

MiEmpresaIU.pro

```
1  TEMPLATE = app
2  TARGET = MiEmpresa
3  win32:CONFIG += console
4
5  TOP_DIR = ..
6  include($$TOP_DIR/comun.pro)
7
8  LIBS += $$MIEMPRESABD_LIB
9
10 DEPENDPATH += . $$TOP_DIR/MiEmpresaBD/include
11
12 INCLUDEPATH += . $$TOP_DIR/MiEmpresaBD/include
13
14 SOURCES += MiEmpresaIU.cpp main.cpp
15
16 HEADERS += MiEmpresaIU.hpp
```

qmake

Makefile

qmake

Makefile

Un Archivo .pro Global y el Orden de Compilación

MiEmpresa.pro

```
1  TEMPLATE = subdirs
2
3  SUBDIRS = \
4      MiEmpresaBD \
5      MiEmpresaIU
```

Utilizamos la platilla para subdirectorios y especificamos el orden de compilación en la variable SUBDIRS

Generamos los “Makefiles”:

```
qmake -r MiEmpresa.pro
```

y compilamos:

```
make
```

o nmake si utilizamos la versión para VS

Signals y Slots

```
void MiEmpresaIU::conectaSignalSlots(){
    connect(m_salir, SIGNAL(triggered()), qApp, SLOT(quit()));
    connect(m_agregarCliente, SIGNAL(triggered()), this, SLOT(menuAgregarCliente()));
```

```
public slots:
    /** Agrega un cliente a la base de datos
     * @param c Datos del cliente que ser&aacute; insertado en la base de datos*/
    void agregarCliente(Cliente const & c);

private slots:
    /** Muestra el dialogo para agregar un cliente */
    void menuAgregarCliente();
```

```
signals:
    /** Esta se&ntilde;al es emitida en la función obtenerDatos cuando el
     * usuario presiona el bot&oacute;n Aceptar del dialogo.
     * @param c Contiene los datos del cliente que ser&aacute; insertado en la
     * base de datos */
    void agregarCliente(Cliente const & c);

private slots:
    /** Obtiene los datos del dialogo despu&eacute;s de que el bot&oacute;n
     * Aceptar es presionado y crea un Cliente con estos datos, despu&eacute;s
     * emite la se&ntilde;al agregarCliente. */
    void obtenerDatos();
```

```
void MiEmpresaIU::menuAgregarCliente(){
    ClienteNuevo cn;// = new ClienteNuevo;

    connect(&cn, SIGNAL(agregarCliente(Cliente const &)),
            this, SLOT(agregarCliente(Cliente const &)));
```

```
connect(aceptar, SIGNAL(clicked()), this, SLOT(obtenerDatos()));
```

```
void ClienteNuevo::obtenerDatos(){
    Cliente c(0, Nombre(m_nombre.text(), m_apellidop.text(), m_apellidom.text

    emit agregarCliente(c);

    this->accept();
}
```

Documentación con doxygen

Doxygen es un sistema de documentación para C++, C, Java, y algunos otros lenguajes. Los comentarios van “encrustados” en el código fuente, de esta manera mantenemos un solo archivo para nuestro código fuente y su documentación. No confundir con la creación del manual de usuario o manual técnico.

```
 8  /** \brief Crea la interfaz de usuario.
 9   *
10   * Esta clase crea la interfaz de usuario y es el punto de entrada hacia el
11   * programa */
12  class MiEmpresaIU: public QMainWindow {
13  public:
14      /** Constructor por defecto */
15      MiEmpresaIU();
16      /** Destructor por defecto */
17      ~MiEmpresaIU();
18
19  private:
20      /// Controla la comunicación con la base de datos
21      MiEmpresaBD bd;
22      /** Crea la interfaz de usuario */
23      void crearIU();
24  };
```

```
[macr@macr]$ doxygen -g MiEmpresa.dox
[macr@macr]$ doxygen MiEmpresa.dox
```