

Utilisation de Qt pour les IHM embarquées Cas concret de Linux

Pierre Ficheux (pierre.ficheux@openwide.fr)

CTO Open Wide Ingénierie

Mars 2011

- Présentation d'Open Wide
- IHM et embarqué
- X11 et framebuffer
- DirectFB, SDL, GTK+, WxWidgets, EFL
- Qt
 - Histoire
 - Architecture
 - Compilation croisée
 - Environnement de développement QtCreator
 - Signaux/slots
 - I18N
 - IHM dynamique

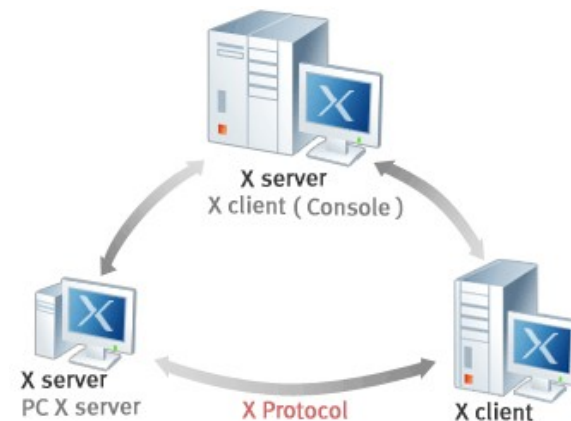
- SSII/SSL créée en septembre 2001 avec Thales et Schneider
- Indépendant depuis 2009
- Environ 90 salariés sur Paris et Lyon
- Industrialisation de composants open source
- Quatre activités :
 - OW : système d'information
 - OW outsourcing: hébergement
 - OW ingénierie: informatique industrielle
 - OW technologies: composants Java



- Ce fut longtemps un problème mineur car peu utilisé dans l'embarqué
 - Système autonome sans affichage
 - Configuration par réseau (SNMP, HTTP, ...)
- Evolution des systèmes, passage du RTOS au multimédia
 - Set-top box: OS21 → Linux
 - Smartphones
 - Industriel → début de l'utilisation d'Android

- Linux est un UNIX, l'IHM par défaut est X11, « X Window System »
- Créé au MIT en 1984
- Système graphique réparti, modèle client/serveur → XFree86, X.org
- Puissant mais très lourd
- Approche répartie rarement utilisée !
- Peu adapté à l'embarqué

Utilisation de X11 →



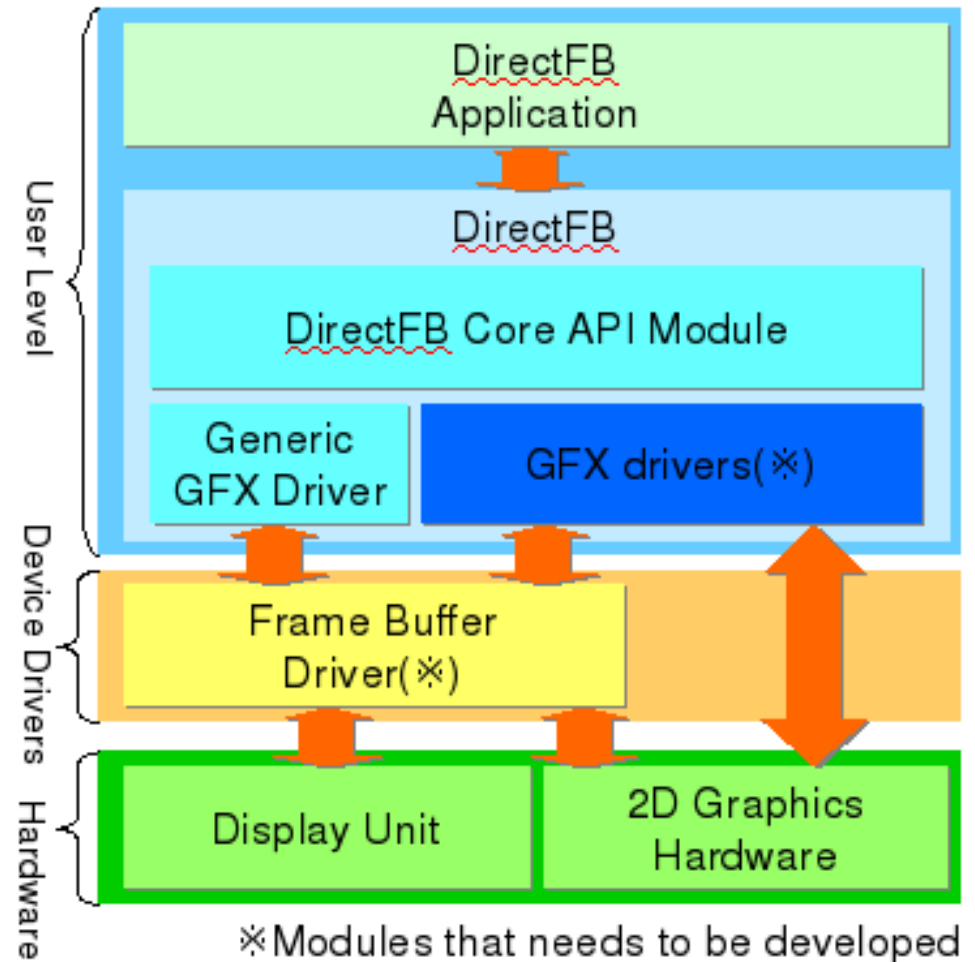
- Pilotage de la carte directement par le noyau (/dev/fb0 → plus de client/serveur)
- Mode VGA, SVGA, VESA ou accéléré
- Programmation très bas-niveau (pixel)

```
$ cp /dev/fb0 copie_ecran.raw
```
- Exemples d'utilitaires/bibliothèques disponibles:
 - fbset
 - fbi
 - fbdump
 - SVGALIB → DOOM :)
 - DirectFB, SDL

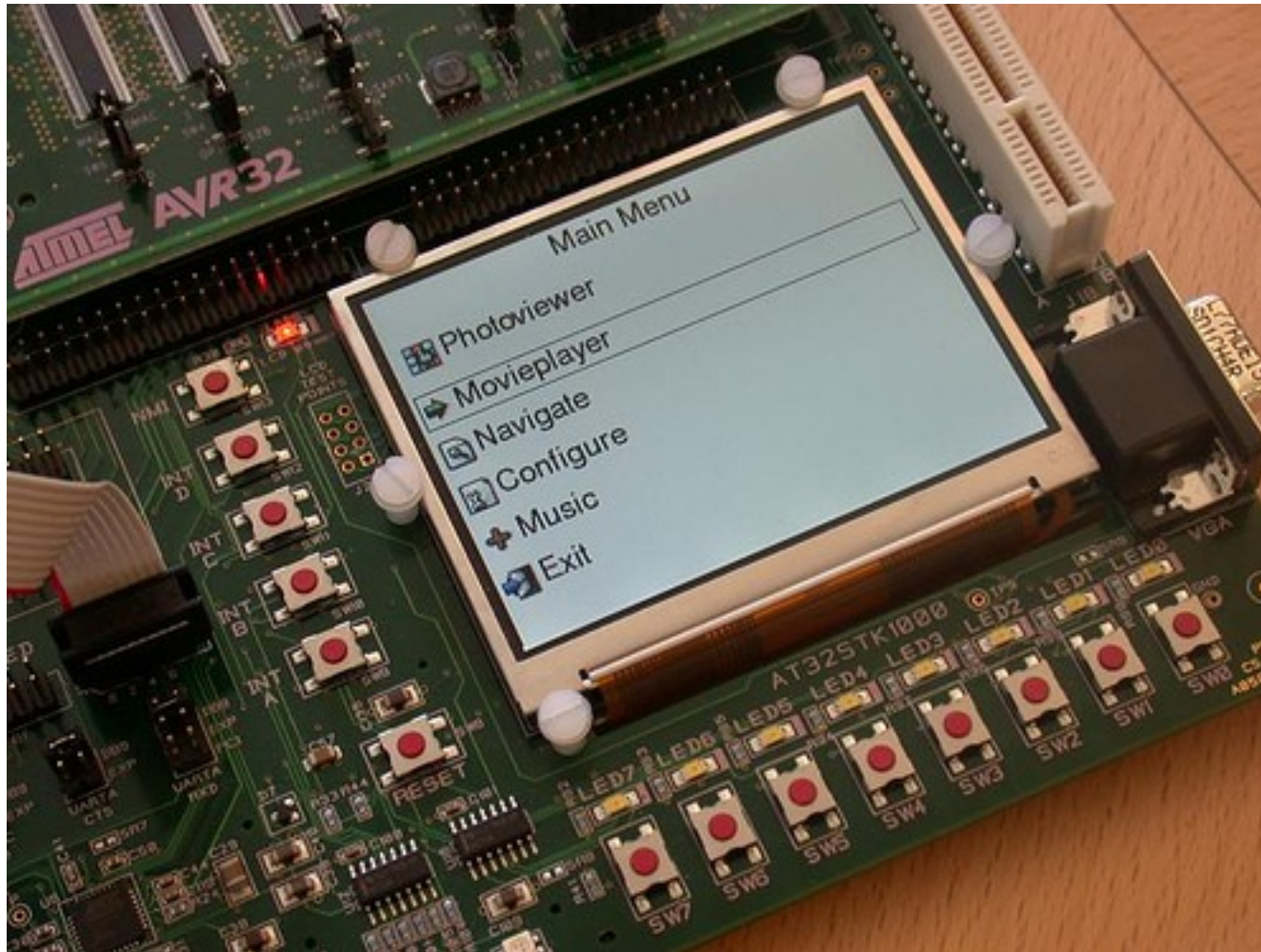
- Fournissent un ensemble d'objets graphiques
 - Menus
 - Boutons
 - Boites de dialogues
 - ...
- Exemples:
 - Athena widgets, OSF-Motif (X11)
 - GTK+ (Gimp)
 - WxWidgets
 - EFL (Enlightenment → Freebox HD)
 - Qt !!

- Fournit des primitives graphiques et audio
- Portables sur Linux, Windows, Mac OS X, QNX, WinCE, OS2...
- Pour Linux, utilisable sur framebuffer, DirectFB, X11
- Utilisée pour le portage d'applications graphiques (jeux)
- Gestion basique de l'écran: fenêtres, transparence, polices de caractères, ...
- Ce n'est pas un « toolkit »

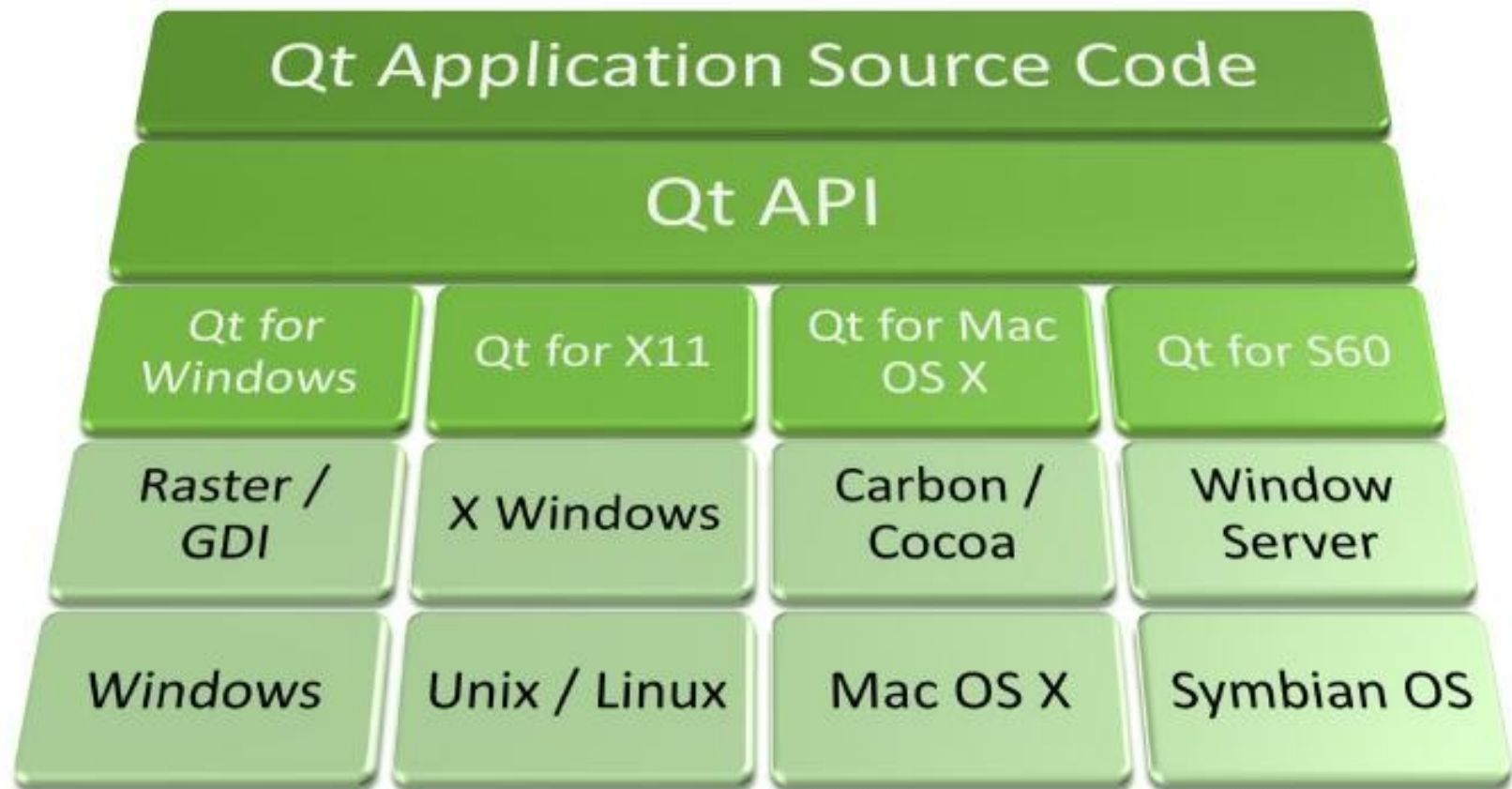
- Bibliothèque d'« abstraction » du framebuffer
- Fonctionne avec le framebuffer Linux mais également avec X11 (- -enable-x11)
- Prise en compte des entrées (souris, clavier, ...)
- Fournit des pilotes accélérés
- Ce n'est pas un « toolkit » !

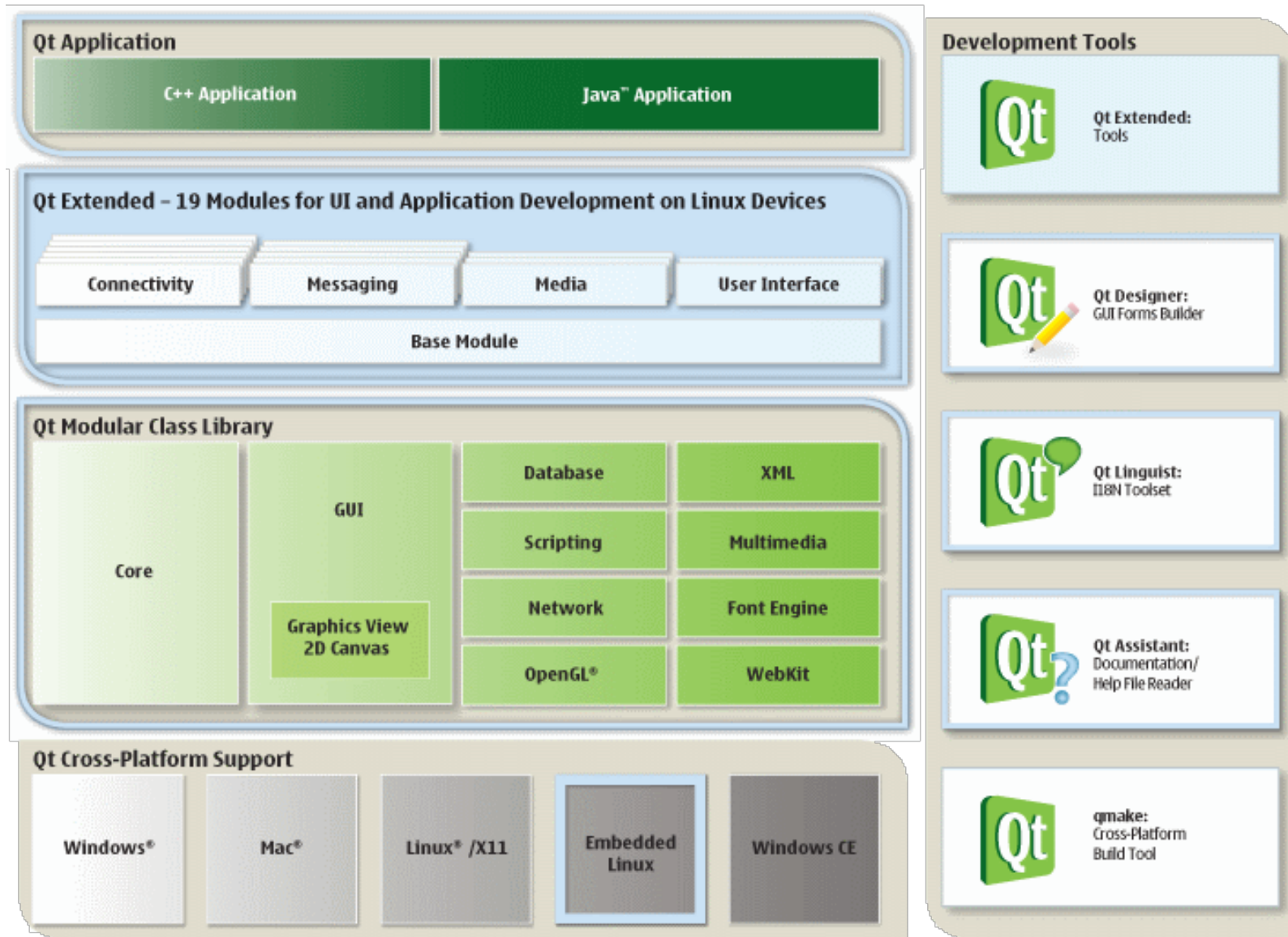




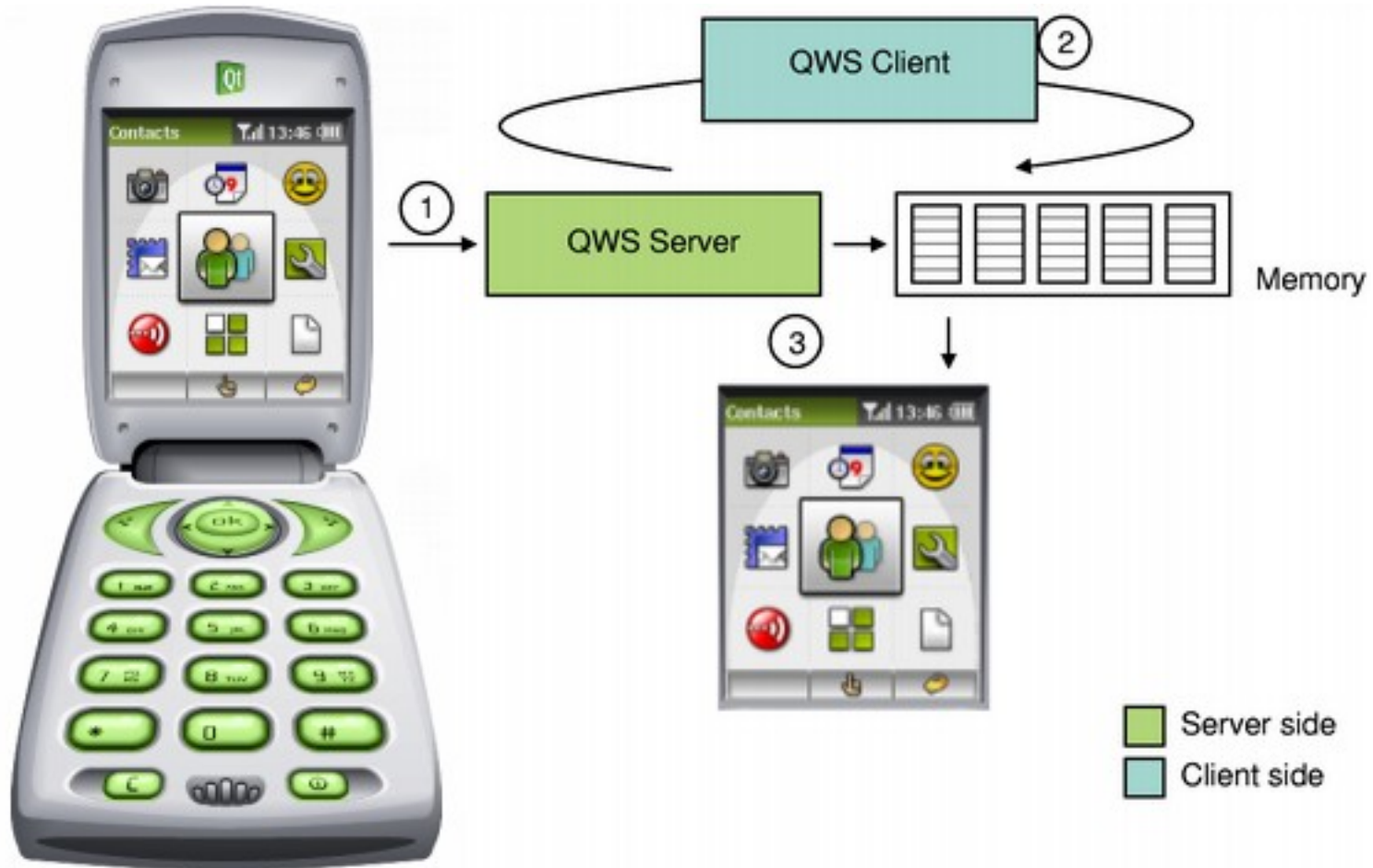


- Première version publiée par Trolltech en 1996
→ toolkit en C++
- Développé pour X11, 2 développeurs au départ, fondateurs de Trolltech
- Outil multi-plateforme (Unix, Windows, MacOS)
- Connu grâce à KDE !
- Qt2 en 2000 → Qtopia (PDA sous Linux « Zaurus » de SHARP)
- Jusqu'en 2008, double licence GPL/Propriétaire
- En 2008, achat par Nokia → LGPL
- Qt4: + Symbian, WinCE, Maemo, ...
- En 2011 rapprochement Nokia / M\$ → licence commerciale cédée à Digia PLX





- Client/serveur → QWS = Qt Window System
- Basé par défaut sur le framebuffer de Linux
- Peut utiliser un framebuffer « virtuel » (X11) avec QVFb
- Une application serveur (-qws), les autres clientes
- Le serveur gère en général l'affichage mais l'application peut accéder directement à l'écran pour des raisons de performances
- Ajout d'un pilote accéléré en dérivant QScreen et QRasterPaintEngine
- <http://doc.qt.nokia.com/4.7/qt-embedded-accel.html>



- Basée sur qmake (sur-couche à make similaire à GNU/Autotools)
- Utilisation par défaut du fichier `mkspecs/qws/linux-arm-g++` → définition de la chaîne de compilation croisée

```
$ ./configure -embedded arm -xplatform qws/linux-arm-g++
```

```
$ make; make install
```
- Configuration spéciale

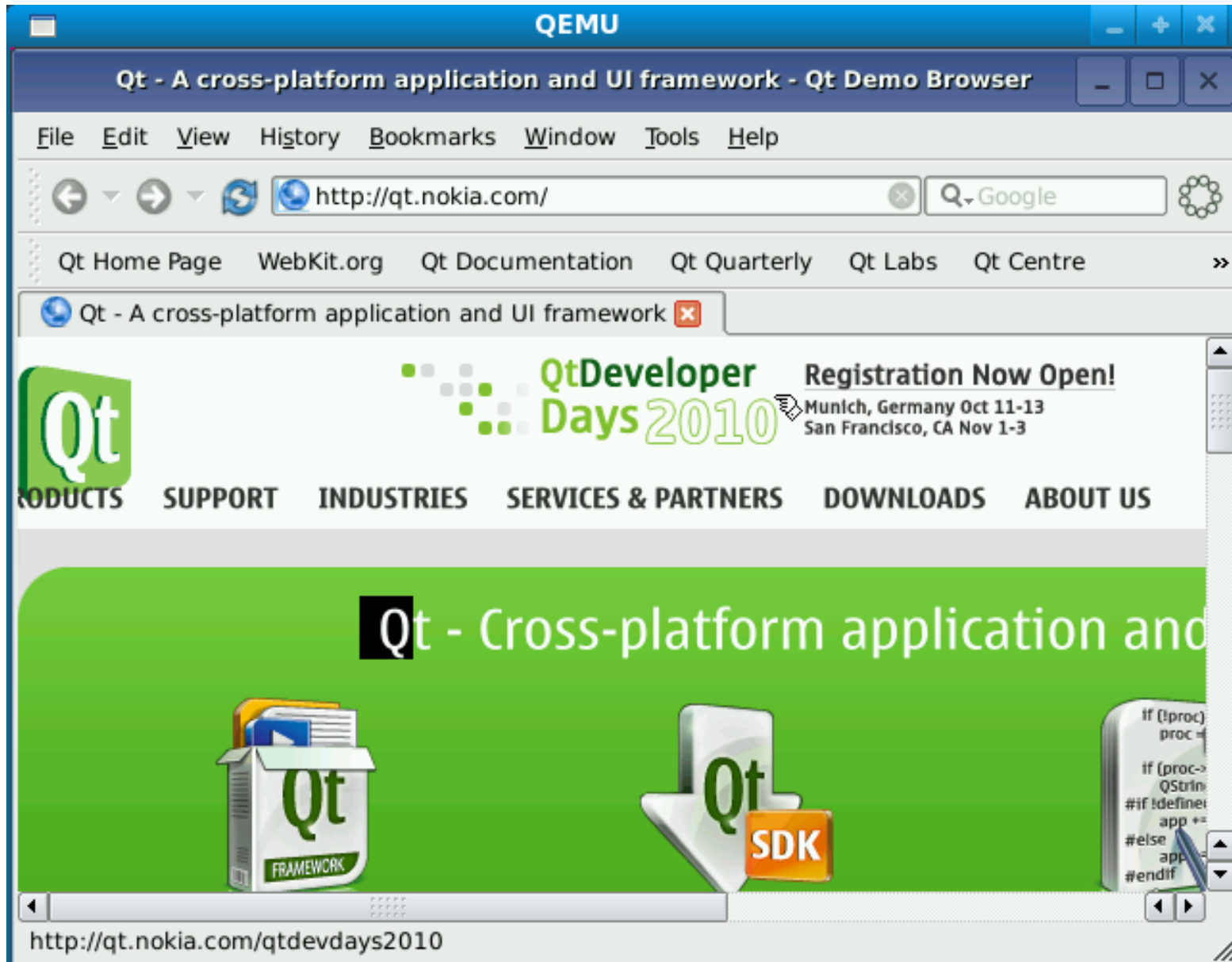
```
$ ./configure -embedded arm -xplatform qws/linux-myconfig-g++
```
- Intégré à Buildroot

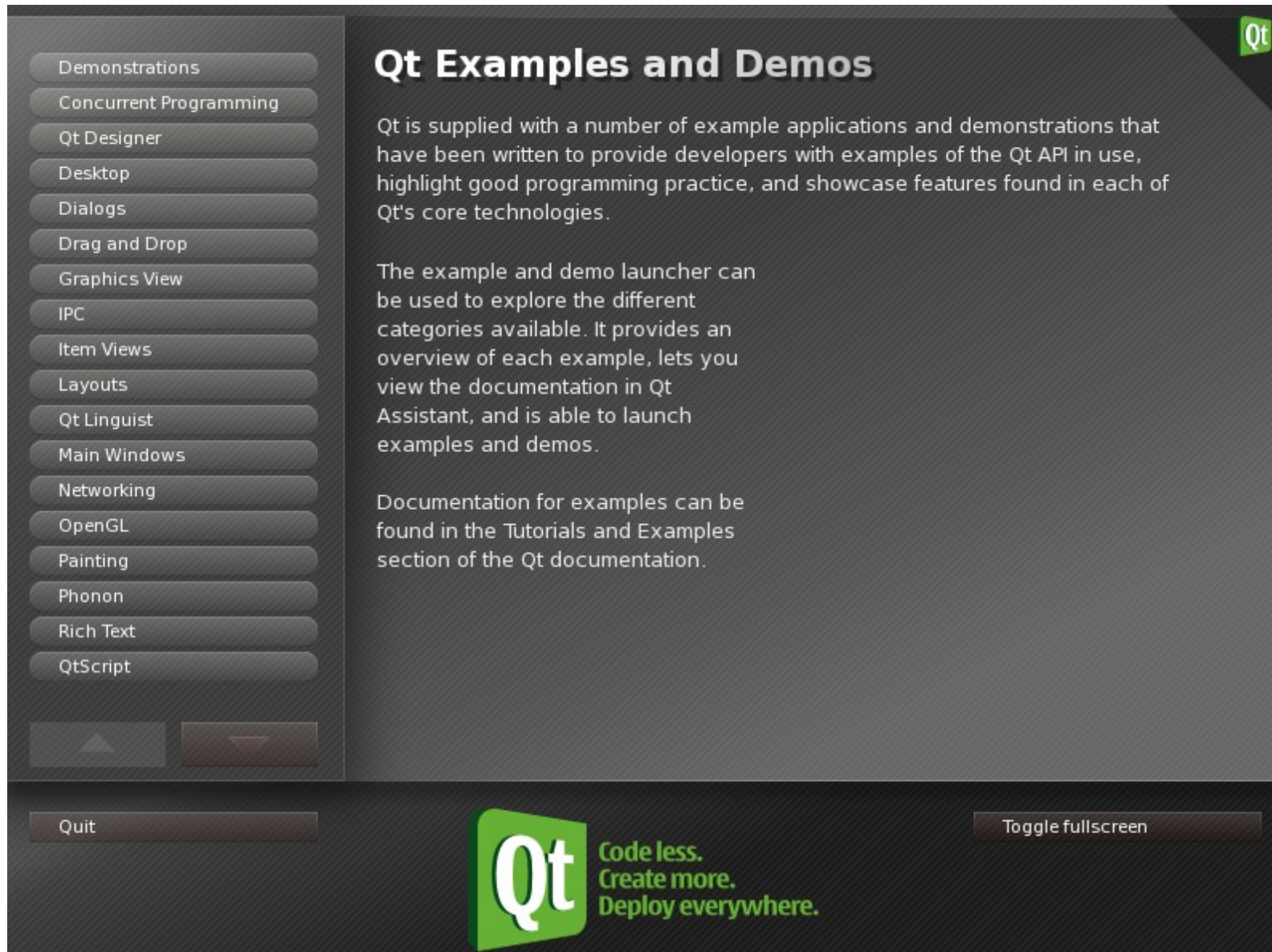
- Premier test possible avec QEMU ! (framebuffer non accéléré)

```
$ qemu-system-arm -M versatilepb -m 128  
-kernel zImage -initrd rootfs.gz  
-append "mem=128M"
```

- Nombreux programmes de test
- Ne pas oublier l'option -qws
- Utiliser la variable QT_QWS_FONTDIR pour les polices :

```
# export QT_QWS_FONTDIR=/usr/lib/fonts  
# /usr/examples/widgets/analogclock/analogclock -qws
```



A screenshot of the 'Qt Examples and Demos' application window. The window has a dark grey background with a subtle grid pattern. On the left side, there is a vertical list of category buttons: Demonstrations, Concurrent Programming, Qt Designer, Desktop, Dialogs, Drag and Drop, Graphics View, IPC, Item Views, Layouts, Qt Linguist, Main Windows, Networking, OpenGL, Painting, Phonon, Rich Text, and QtScript. Below these buttons are two arrow buttons for navigation. The main content area on the right is titled 'Qt Examples and Demos' and contains three paragraphs of text. At the bottom of the window, there is a 'Quit' button on the left, the Qt logo with the slogan 'Code less. Create more. Deploy everywhere.' in the center, and a 'Toggle fullscreen' button on the right. A small Qt logo is also visible in the top right corner of the main content area.

Qt Examples and Demos

Qt is supplied with a number of example applications and demonstrations that have been written to provide developers with examples of the Qt API in use, highlight good programming practice, and showcase features found in each of Qt's core technologies.

The example and demo launcher can be used to explore the different categories available. It provides an overview of each example, lets you view the documentation in Qt Assistant, and is able to launch examples and demos.

Documentation for examples can be found in the Tutorials and Examples section of the Qt documentation.

Quit



Code less.
Create more.
Deploy everywhere.

Toggle fullscreen

- LE point faible de Qt
- Consommation de l'exemple analogclock: 20 Mo
- Consommation de webkit: 50 Mo
- Consommation d'une démonstration DirectFB (df_andi) : 2 Mo
- Qt est destiné au développement d'applications « complexes » !

- Le même code source utilisable sur toutes les plateformes grâce à qmake :

```
$ qmake-qt4
```

```
$ make
```

```
$ file DemoNew
```

Linux/X11

DemoNew: ELF 32-bit LSB executable, **Intel 80386**, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.18, not stripped

```
$ make distclean
```

```
$ type qmake
```

```
qmake est haché (/home/pierre/buildroot-2010.05/output/staging/usr/bin/qmake)
```

```
$ qmake
```


```
$ make
```

```
$ file DemoNew
```

Buildroot (ARM)

DemoNew: ELF 32-bit LSB executable, **ARM**, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.33, not stripped

- Qt est PLUS qu'un toolkit graphique !
- Abstraction pour sockets, threads, Unicode, SQL, ...
- Les objets dérivent de QObject mais ne sont pas uniquement des objets graphiques, ex: QThread
- Les « signaux/slots » remplacent avantageusement les *callbacks*
- Prise en compte I18N
- Chargement dynamique d'IHM

- Outil WYSIWYG de création d'interface
- IDE de développement semblable à Eclipse
 - Edition du code
 - Compilation (croisée)
 - Mise au point (GDB)
- Les fichiers d'IHM Qt (.ui) sont au format XML
- Permet de définir des cibles X11, embedded, etc.
- Moins agréable que  :-)



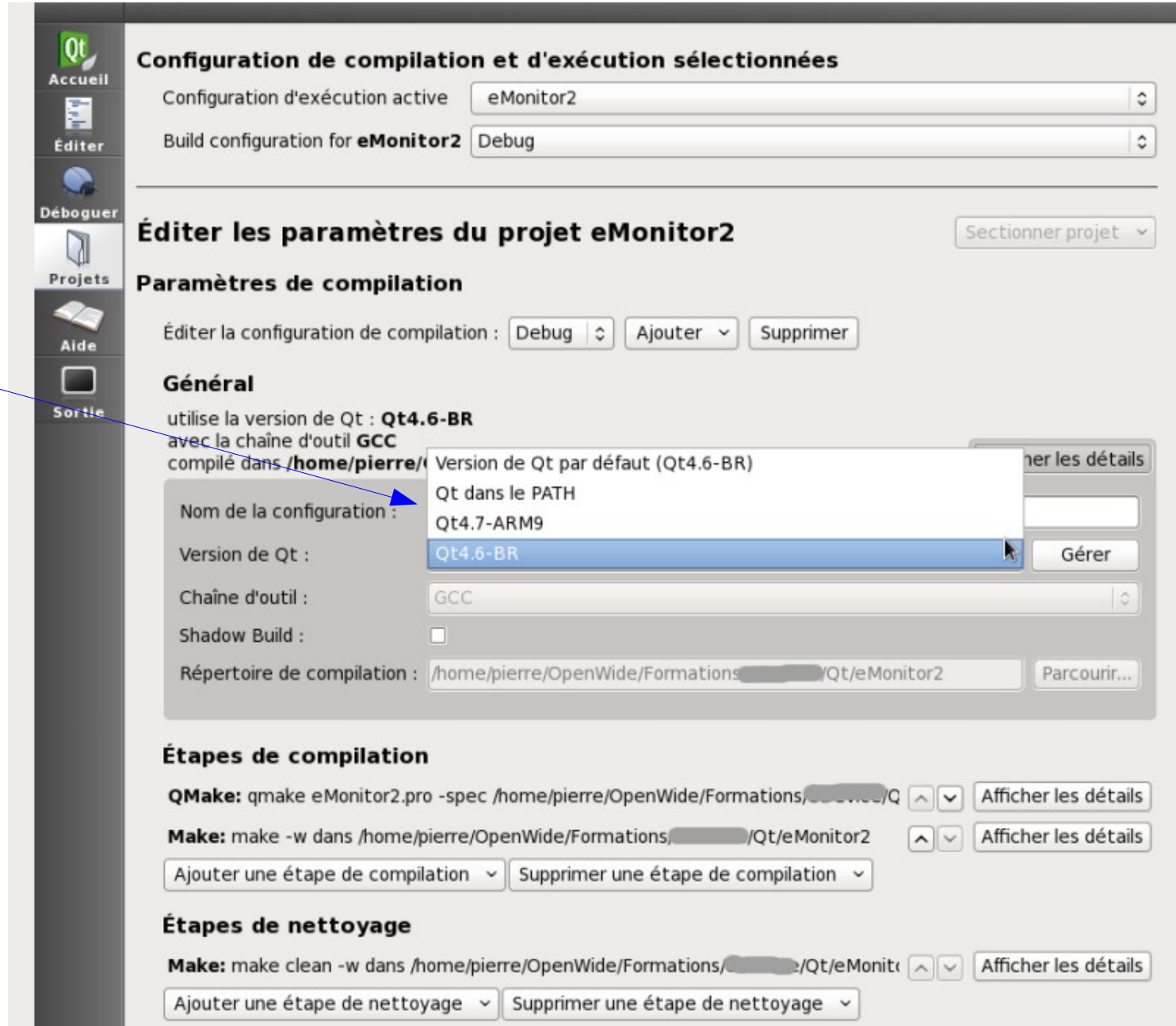
The screenshot shows the Qt Creator IDE interface. The main workspace displays a Qt widget design with a button containing the text "hello N900!". The left sidebar contains a widget palette with categories like Layouts, Spacers, Buttons, and Item Views. The right sidebar shows the Object Inspector and Property Inspector for the selected QPushButton widget.

Object Inspector:

Object	Class
MainWindow	QMainWindow
centralWidget	QWidget
pushButton	QPushButton

Property Inspector:

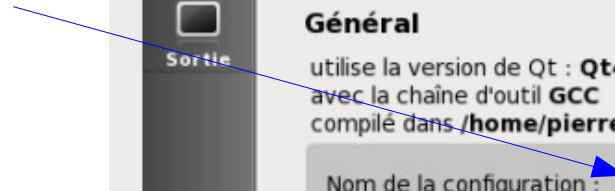
Property	Value
layoutDirection	LeftToRight
autoFillBackg...	<input type="checkbox"/>
styleSheet	
locale	Polish, Poland
inputMethod...	ImhNone
QAbstractButton	
text	hello N900!
icon	
iconSize	16 x 16
shortcut	
checkable	<input type="checkbox"/>
checked	<input type="checkbox"/>
autoRepeat	<input type="checkbox"/>
autoExclusive	<input type="checkbox"/>
autoRepeatD...	300
autoRepeatI...	100



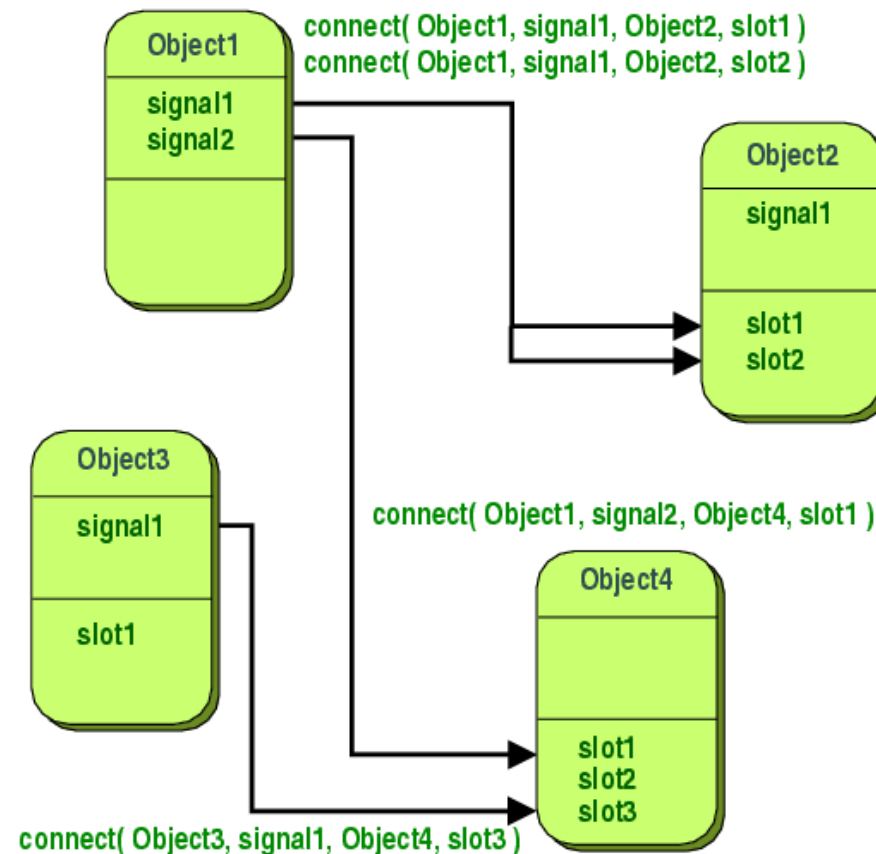
The image shows the QtCreator IDE configuration window for the project 'eMonitor2'. The window is titled 'Configuration de compilation et d'exécution sélectionnées'. It has a sidebar on the left with icons for 'Qt', 'Accueil', 'Éditer', 'Déboguer', 'Projets', 'Aide', and 'Sortie'. The main area is divided into several sections:

- Configuration de compilation et d'exécution sélectionnées:** Shows 'Configuration d'exécution active' set to 'eMonitor2' and 'Build configuration for eMonitor2' set to 'Debug'.
- Éditer les paramètres du projet eMonitor2:** Includes a 'Sectionner projet' dropdown.
- Paramètres de compilation:** Shows 'Éditer la configuration de compilation' set to 'Debug', with 'Ajouter' and 'Supprimer' buttons.
- Général:** Shows 'utilise la version de Qt : Qt4.6-BR avec la chaîne d'outil GCC compilé dans /home/pierre/'. A dropdown menu is open, showing options: 'Version de Qt par défaut (Qt4.6-BR)', 'Qt dans le PATH', 'Qt4.7-ARM9', and 'Qt4.6-BR' (which is selected). Other fields include 'Nom de la configuration', 'Version de Qt', 'Chaîne d'outil' (GCC), 'Shadow Build' (unchecked), and 'Répertoire de compilation' (/home/pierre/OpenWide/Formations/Qt/eMonitor2).
- Étapes de compilation:** Shows 'QMake' and 'Make' commands with 'Afficher les détails' buttons. There are also 'Ajouter une étape de compilation' and 'Supprimer une étape de compilation' buttons.
- Étapes de nettoyage:** Shows a 'Make' command for cleaning with an 'Afficher les détails' button, and 'Ajouter une étape de nettoyage' and 'Supprimer une étape de nettoyage' buttons.

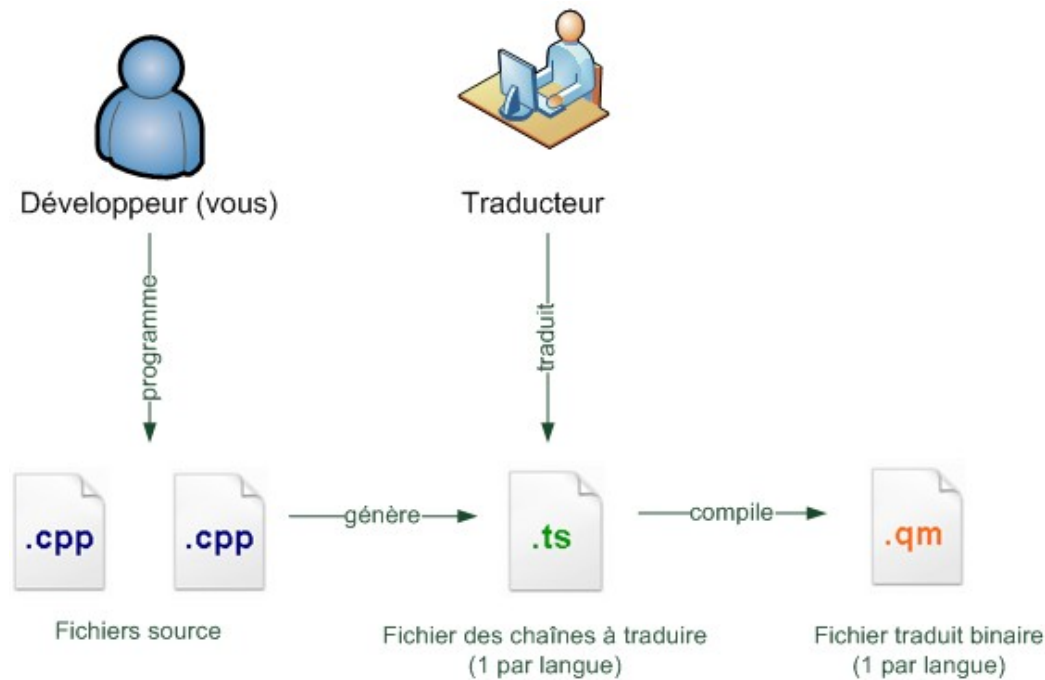
Sélection SDK

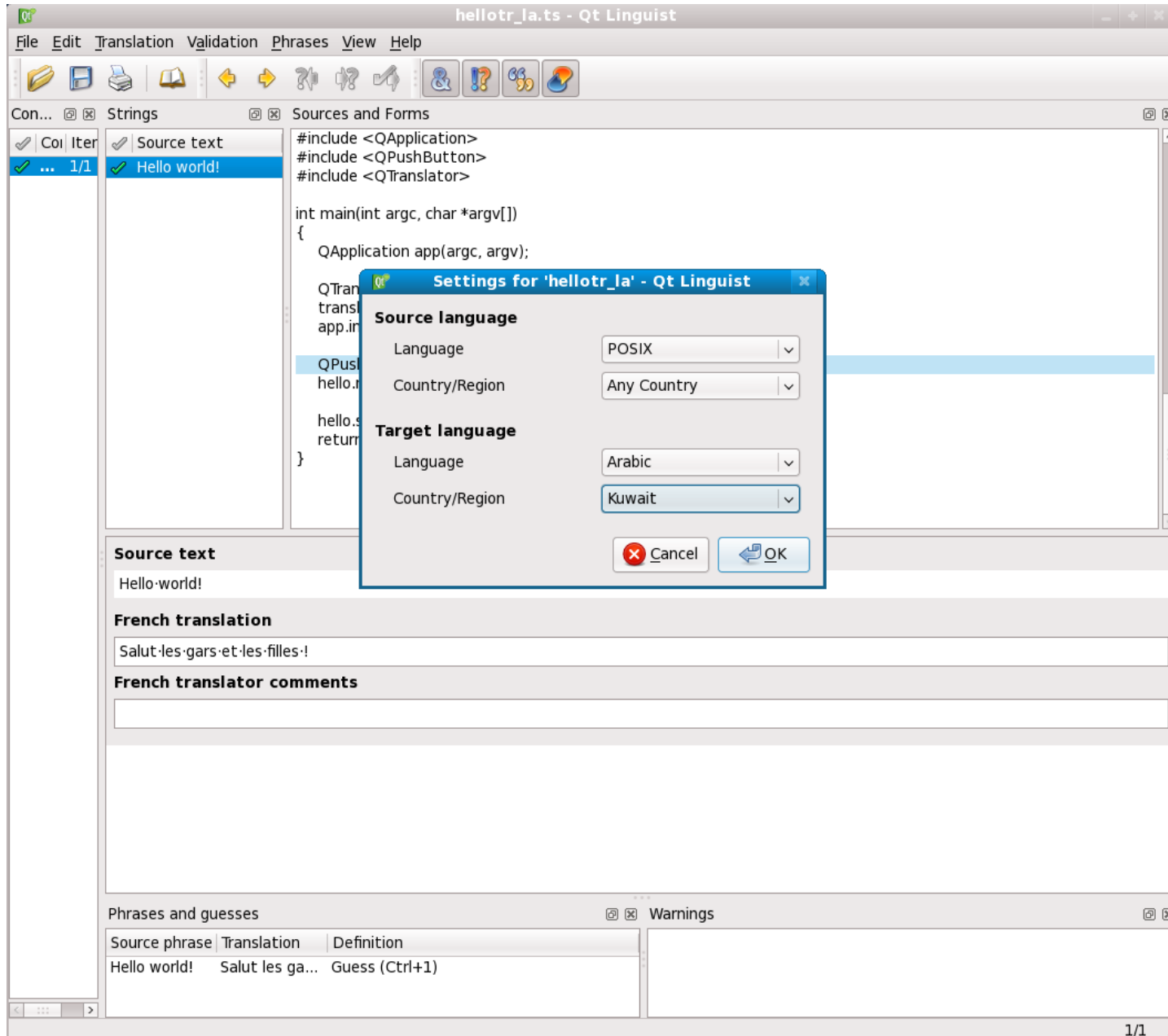


- Callback → un événement \Leftrightarrow une fonction
- Qt : Un signal (ex: clic bouton) est reçu par plusieurs « slots » après connexion



- Message extraits du code source (.cpp) → .ts (XML)
- Traduction avec Qt Linguist en compilé en .qm





hellotr_la.ts - Qt Linguist

File Edit Translation Validation Phrases View Help

Con... Strings Sources and Forms

Source text: Hello world!

French translation: Salut les gars et les filles!

French translator comments:

Source phrase	Translation	Definition
Hello world!	Salut les ga...	Guess (Ctrl+1)

Phrases and guesses Warnings

1/1

- Chargement d'un .ui au démarrage → pas de re-compilation des .cpp
- Utilisation de la classe QUiLoader

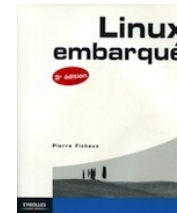
Fichier créé avec QtCreator

```
QUiLoader loader;  
QFile file("form.ui");  
file.open(QFile::ReadOnly);  
QWidget *widget = loader.load(&file);
```

- GTK+ : encore mal adapté au framebuffer
- WxWidgets : idem
- EFL : très bon outil (plus léger que Qt) mais peu de support, pas d'outil de construction. Bon support framebuffer et cible croisée.

REMARQUE :

La puissance des cartes embarquées augmentant, il est parfois possible d'utiliser X11



- Linux embarqué version 3 : <http://www.eyrolles.com/Informatique/Livre/linux-embarque-9782212124521>
- <http://www.slideshare.net/qtbynokia/qt-on-real-time-operating-systems>
- <http://www.docstoc.com/docs/56800911/directfb-introppt---Report-of-W>
- <http://qt.developpez.com/doc/4.6/requirements-embedded-linux/>
- <http://doc.qt.nokia.com/4.7/qt-embedded-accel.html>
- <http://doc.qt.nokia.com/4.7-snapshot/qt-embedded-crosscompiling.html>
- <http://doc.qt.nokia.com/4.7-snapshot/qt-embedded-linux.html>

