

# Solutions IHM pour Linux embarqué

Pierre Ficheux ([pierre.ficheux@openwide.fr](mailto:pierre.ficheux@openwide.fr))

CTO Open Wide Ingénierie

Octobre 2011


- Présentation d'Open Wide
- IHM et embarqué
- Les approches possibles
- X11 et Framebuffer
- Les bibliothèques graphiques
  - DirectFB, SDL
- Les toolkits
  - Qt
  - EFL
  - GTK
- HTML et Flash
- Android

- SSII/SSL créée en septembre 2001 avec Thales et Schneider
- Indépendant depuis 2009
- Environ 100 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



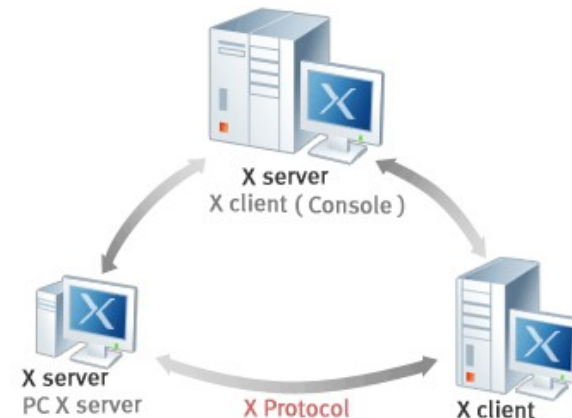
- IHM = affichage et saisies
- Ce fut longtemps un problème mineur car peu utilisé dans l'embarqué
  - Système autonome sans affichage (RTOS)
  - Configuration par réseau (SNMP, HTTP, ...)
- Evolution des systèmes, passage du RTOS au multimédia
  - Set-top box: OS21 → Linux
  - Smartphones
  - Industriel → début de l'utilisation d'Android
- L'IHM n'est (était) pas le sujet de prédilection des spécialistes du logiciel embarqué :-)

- Développement d'une application embarquée
  - Le cas « général » proche du desktop Linux
- Utilisation d'un navigateur web (HTMLx, Javascript)
  - Données en local → similaire au cas précédent
  - De plus en plus de possibilités avec HTML5 :)
- Interface en Adobe Flash
  - Propriétaire
  - Bien vu par les équipes marketing !
  - Consommateur de ressources
  - Fréquemment utilisé pour les set-top box

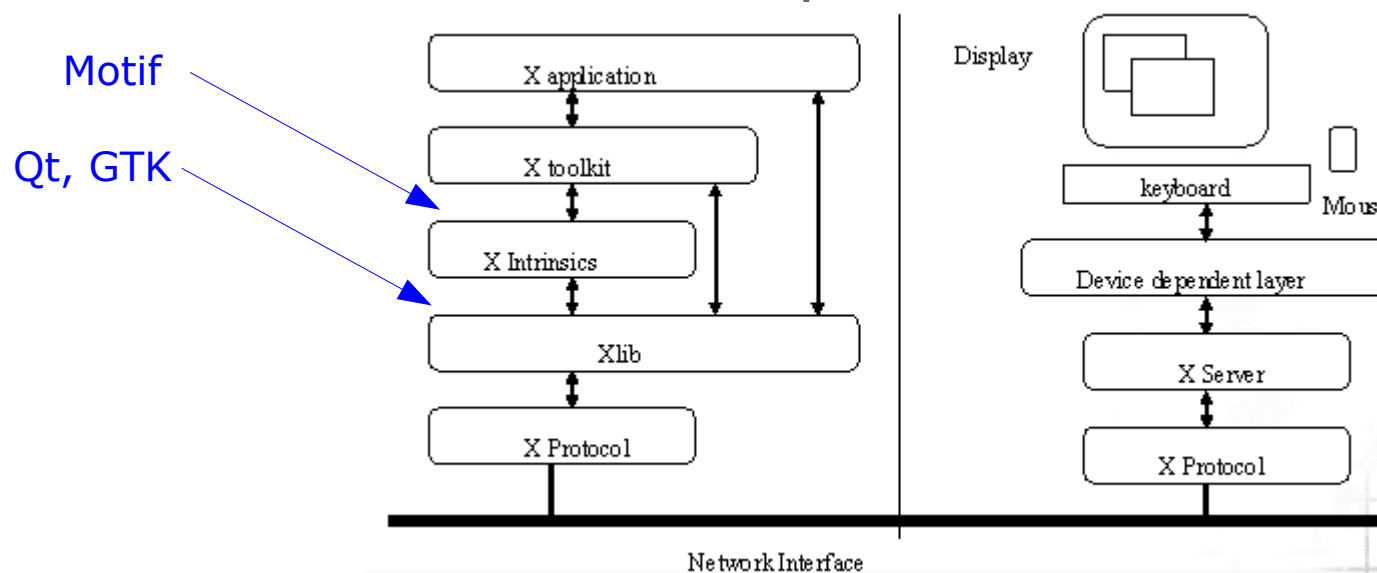
- Affichage fréquent sur des périphériques variés :
  - LCD monochrome « bas de gamme » 
  - Ecran souvent tactile (attention aux résolutions étranges)
  - Accélération graphique de plus en plus fréquente sur les cartes embarquées (Freescale i.MX, ...)
- Autres périphériques (USB ?)
  - Clavier spécial
  - Télécommande infra-rouge
  - Molette / manette
  - Périphériques spéciaux

- Linux est un UNIX
  - Mode texte par défaut
  - « X Window System » ou X11 à partir de 84
- Créé au MIT
- Système graphique réparti, modèle client/serveur → Xfree86 (x86), X.org
- Puissant mais lourd + API complexe
- Approche *répartie* rarement utilisée !

Utilisation de X11 →



- Initialement peu adapté à l'embarqué
- Retour grâce à plusieurs éléments :
  - Augmentation de la puissance des CPU embarqués
  - Utilisation de l'Atom/x86
  - Le pilote accéléré devient commun au desktop et à l'embarqué



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

```
$ cp /dev/fb0 copie_ecran.raw
```
- Avantages :
  - Léger (faible consommation RAM)
  - Démarrage rapide
- Inconvénients :
  - Pilote spécial → drivers/video
  - Peu « standard » par rapport à X11

- Exemples d'utilitaires/bibliothèques disponibles/compatibles
  - Bas niveau → fbset, fbinfo, fbdump, ...
  - SVGALIB → DOOM :-)
  - DirectFB (abstraction du FB)
  - SDL
  - Qt
  - X11 sur FB

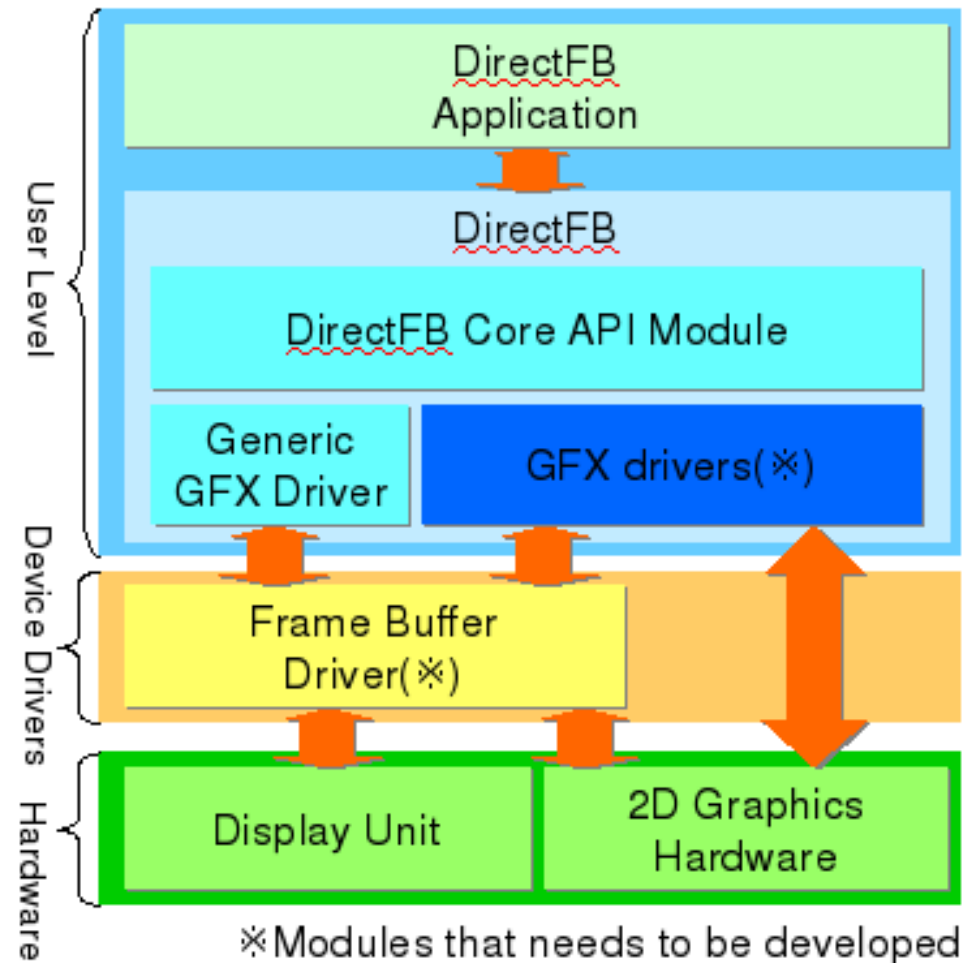
- Se placent « au dessus » de X11 ou du framebuffer
- Deux catégories
  - Les *bibliothèques d'abstraction* → portabilité mais pas d'objets graphiques évolués (SDL, DirectFB)
  - Les *toolkits* graphiques → fournissent des objets graphiques et peuvent se placer au dessus des bibliothèques d'abstraction

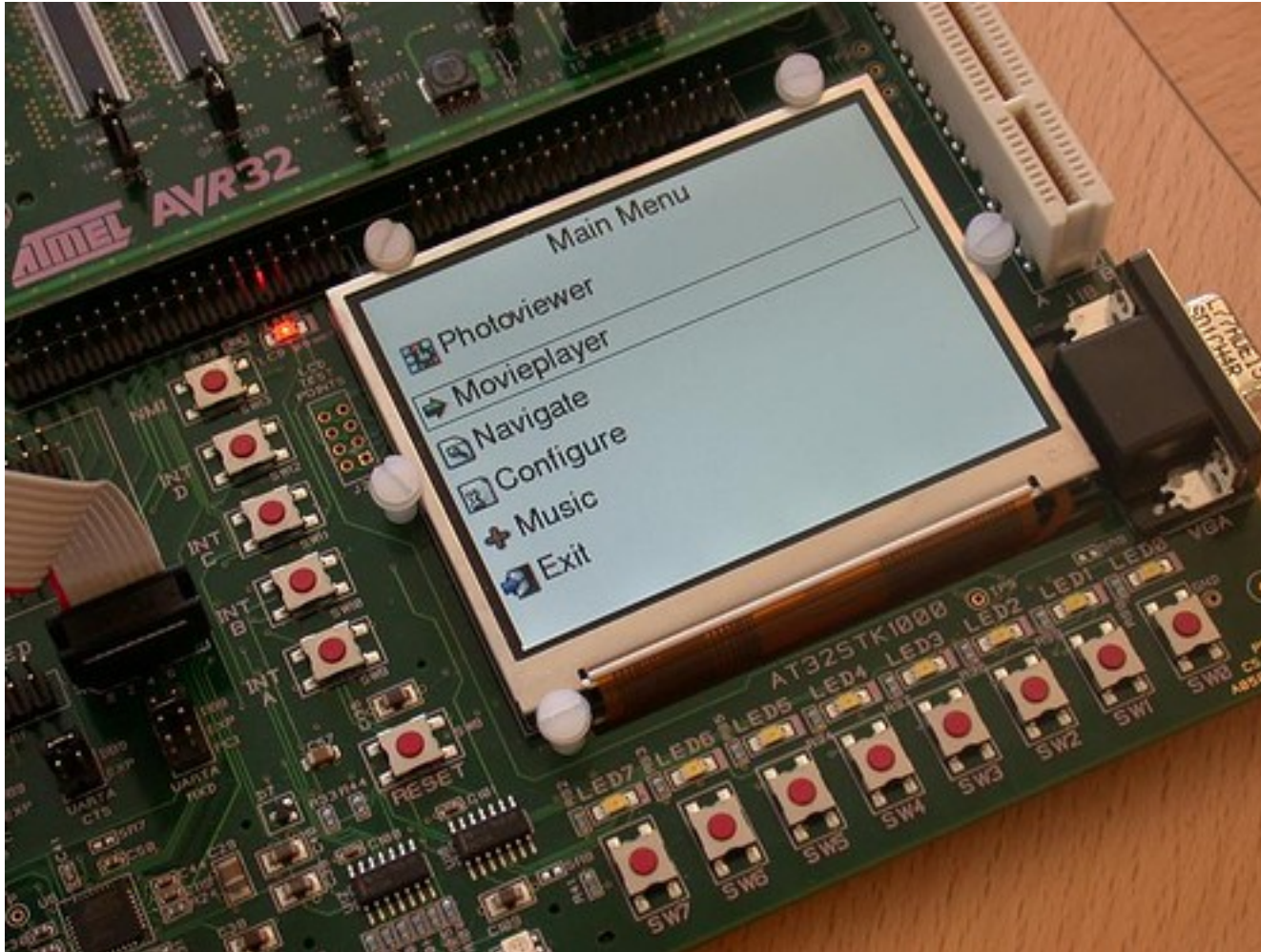
Qt → X11 (Xlib)

Qt → FB

Qt → DirectFB

- 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 FB accélérés





- 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) et légères
- Gestion basique de l'écran: fenêtres, transparence, polices de caractères, ...



- Fournissent un ensemble d'objets graphiques
  - Menus
  - Boutons
  - Boites de dialogues
  - WebView !
  - Mediaplayer
- Exemples:
  - Athena widgets, OSF-Motif (X11) → obsolète
  - Qt
  - EFL (Enlightenment, E17)
  - GTK+ (Gimp)
  - WxWidgets

- Toolkit C++ publiée par Trolltech en 1996 (X11)
- Outil multi-plateforme (Unix, Windows, MacOS, Symbian, ...)
- Connue grâce à KDE !
- Dernière version: 4.7.4
- Avantages :
  - X11, FB, DirectFB, QVFb, ...
  - Excellente documentation
  - Outil de conception d'interface (QtCreator)
- Inconvénients :
  - Lourd
  - Avenir / Nokia



- Toolkit C basé sur *Enlightenment* ou *E* (actuellement *E17*)
  - Avantages :
    - Peu gourmand en ressources, rapide
    - Multiplateforme
    - Esthétique, modulaire, configurable
  - Inconvénients
    - Peu connu
    - Moins de documentation que pour Qt
    - Pas de constructeur d'interface
- Le petit toolkit qui monte :-)





- Développé pour GIMP (Gimp ToolKit)
- Toolkit en C multiplateforme
- Fonctionne le plus souvent sur X11 → GNOME
- Assez peu utilisé dans l'embarqué
- Il existe des adaptations sur FB, mais peu maintenues

- Assure une certaine « indépendance » par rapport à la plateforme
- Maquettage aisé sur desktop → à double tranchant
- Attention, l'installation d'un navigateur (Webkit, Opera, ...) implique de bonnes capacités matérielles pour supporter les technologies récentes
- Le navigateur lui-même utilise un toolkit !
- Approche intéressante dans le mobile pour éviter les divers SDK, mais peut elle s'imposer dans l'embarqué ?

- Android n'est pas une IHM, c'est un OS
- API de programmation Java (SDK) mais pas de bytecode Java
- Système Linux – très - modifié (noyau, libc)
- NDK en C++ mais peu utilisé
- Raz de marée en téléphonie
- Apparition en informatique « industrielle » mais
  - Pas de TR
  - Compatibilité matérielle
- Tiré par l'existence de plateformes peu coûteuses (exemple: tablette SAMSUNG) → mais peu d'interfaces matérielles

