Présentation
et
Spécificités inspiratrices pour Linux & Co

RMLL 2008

Auteur : François Revol

Date : 04/07/2008

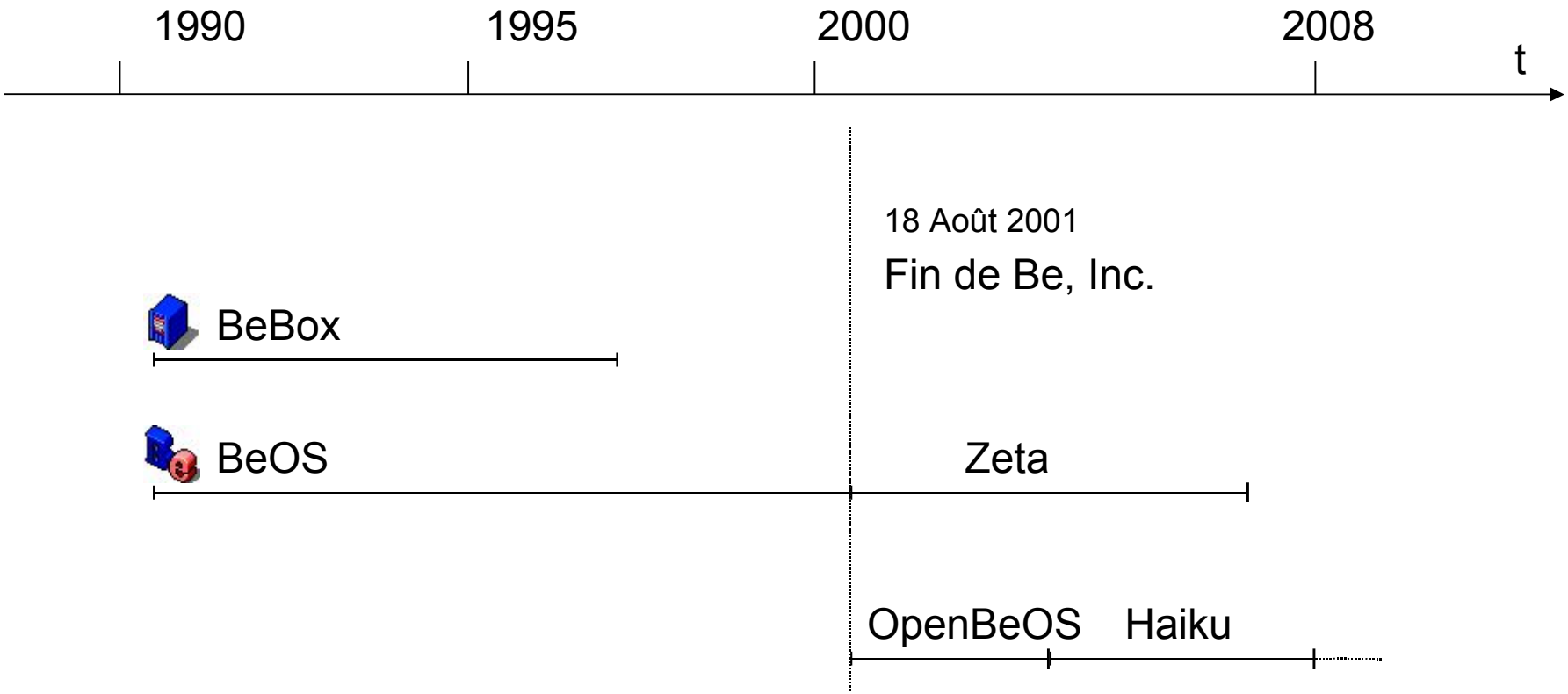
Haiku

« Armelle j'ai rencontrée
elle m'a plu
matin d'été »

Présentation

- Historique
- Caractéristiques Générales
- Principes
- Communauté
- Actions en cours

Historique


1990
1995
2000
2008
t
18 Août 2001
Fin de Be, Inc.
BeBox
BeOS
Zeta
OpenBeOS
Haiku

Caractéristiques Générales

- Perpétuer la philosophie de BeOS
  - R1 : compatibilité binaire BeOS (→ gcc2)
- Libre : Licence MIT
- Intégration d'autres projets Libres
  - GNU (glibc, bash, coreutils, ...)
  - FreeBSD (pilotes réseau, bientôt jemalloc)
  - FreeType
  - AntiGrain Geometry
  - FFmpeg (libavcodec)

Principes

- KISS: Keep It Smart & Simple (Simple et élégant)
  - Architecture
  - Interface graphique
  - Meilleurs réglages par défaut
- Ciblage poste de bureau multimédia
  - Faire une chose et la faire bien
- Modularité et Extensibilité

Communauté

- Équipes
  - Marketing/Communication
  - Kernel, Network, Printing...
- 20 développeurs principaux
- Contributeurs
- Projets associés
  - Portages (Gnash...)
  - Applications natives (BePDF, Pe, ...)

Actions en cours

- Portages
  - Webkit
  - NetSurf
  - CUPS
  - Java (supporté par OpenJDK)
  - XEmacs :-D
- Ouverture
  - FOSDEM, ... RMLL \o/
- GSoC
  - 5 projets
- Haiku Code Drive
  - 4 projets
- Bounties
  - SATA (fini)

Caractéristiques Techniques

- Noyau modulaire préemptif multithreadé
- Système de fichier avec méta-données typées et indexées
- API C++ cohérente (Kits)
- Greffons génériques (add-ons)
- Compatibilité POSIX
- Portabilité : x86, ppc, m68k (fun), arm?
- Concision (image 120Mo, noyau 140 klignes)

Spécificités

- Interface Graphique
- API
- Média
- Systèmes de Fichiers
- Noyau
- Pilotes

Interface Graphique

- MultiThreading → Réactivité
- UTF-8 (merci Plan9)
- Réplicants (> ActiveX, Widgets)
- Scripting (hey Linux set Hype of Dbus to false)
- Gestionnaire de Fichiers
  - Navigation « x-ray »
  - Attributs étendus
    - Type MIME
  - Queries

Interface Graphique

- Cohérence visuelle et fonctionnelle (~~X11~~)
- ~~EyeCandy~~ Sobriété & Réactivité
- Sliding tabs
- Bureau bleu « BeOS » #336698
  - Win9x : vert mais BSOD, Win2k... bleu ;-)

API

- Orientée Objet, C++
- Organisée en « Kits »
- Héritage multiple
  - BDirectory : public BNode, BEntryList
- QApplication == BApplication ?
- App Kit
  - BMessage, ...
- Interface Kit
  - BWindow
  - BView
    - BButton, ...
- Storage Kit
- Media Kit
  - BMediaFile, ...

API

- Translation Kit
  - Inspiré d'AmigaOS (datatypes.library)
  - Uniformise l'accès par des add-ons
    - Transparence sur libpng, libjpeg, ...
  - Bitmap, Vectoriel, Texte,
  - Sanity Translator : lire == scanner avec SANE
    - Gnome ?
  - GOCR Translator : lire texte == lire bitmap

Média

- Media Kit (API)
  - Media node (gstreamer ?)
- media_server et media_addon_server
- Mixer système soft (alsa, OSSv4 ?)
  - 1 réglage par flux (> Vista : par application)
- Support API bas niveau différentes
  - Old, multi, OSSv4
  - Transparent pour les applications natives

Systèmes de Fichiers (VFS)

- rootfs en RAM (/boot, /bin -> /boot/beos/bin)
- Node monitoring ([di]notify...)
  - Création, suppression, modif fichier, xattr, ...
  - Envoi de BMessage par le noyau
- Autres FS
  - Ext2, reiser (ro), NTFS, NFS2, ...
  - Userlandfs (même API que le noyau)
  - Cddafs
  - Googlefs

Systèmes de Fichier (OpenBFS)

- Tradition Unix (superblock, groupes d'allocations, i-nœuds)
- Optimisations multimédia (grands fichiers)
  - B+Tree, 64 bits (xfs... ext4!...)
  - block_run (extents, enfin dans ext4!)
- Et...

Systèmes de Fichier (OpenBFS)

- Méta-données (xattr)
  - Typées ((u)int, float, chaîne, icône, type mime... ==4CC)
  - Indexées (option)
  - Live Queries `'(BEOS:MIME=="text/x-mail")&&(MAIL:status=="New")'`
    - Spotlight ? Beagle / Tracker* ? *TM de Be, Inc en son temps
- SkyOS : base OpenBFS + indexeur

Noyau

- Pas µnoyau au sens Tannenbaum
- Mais très modulaire (même pci) → propreté
- SMP Préemptif (pas de giant lock!)
- Threads noyau (idle[cpu], réseau, usb, DPC...)
- Priorités 120 (0 → 99, > 100 = FIFO, RT mou)
- Kernel Debugger Land (stub GDB, hangman)
- Haiku: O(1) (GSoC2007, en cours)

Noyau

- Tickless

  - PIT mode 0 (ATIIXP bug...) (Haiku : APIC)
  - Même sans PM, perf >
  - → `add_timer()` → semaphore → `snooze(us)`

    `{ acquire_sem_etc(sleepSem, 0, B_TIMEOUT, us); }` → `[u]sleep()`
- Initrd ? Zbeos + tgz!
  - Eltorito → Stage2 + tgz → kernel + pilotes → /boot
    - CD BeOS (et Haiku) == LiveCD

Pilotes

- API **stables** (~~#ifdef~~ hell)
  - Table de fonctions (ld -lkernel.so ~~-lfoo~~) → design++;
- Haiku : nouveau framework en +
  - Noeuds et bus (à la BSD)
- ~~Configure &~~ Plug & Play

Pilotes

- Add-ons `/system/add-ons/kernel/{bus_manager/{pci,...},file_system,...}`
  - Modules `module_info={&std_ops, "foo/bar/v1", flags...}`
    - Bus `bus_module_info={module_info, &rescan, ...}`
      - PCI `pci_module_info={{{,"bus_manager/pci/v1",}, &rescan}, &write_io_8, ...}`

        ...
    - Fs...
  - Pilotes `/system/.../drivers/dev/foo/bar -> ../bin/bar`
    - insmod && modprobe -> /dev/null
    - Devfs
      - Node monitoring → device_watcher, Tracker (== udev - vi)

Pilotes : Exemple /dev/null

```
#include <Drivers.h>
#include <string.h>

#define DEVICE_NAME "null"
int32 api_version = B_CUR_DRIVER_API_VERSION;

static status_t null_open(const char *name, uint32 flags, void **cookie)
{
     *cookie = NULL;
     return B_OK;
}

static status_t null_close(void *cookie)
{ return B_OK; }

static status_t null_freecookie(void *cookie)
{ return B_OK; }

static status_t null_ioctl(void *cookie, uint32 op, void *buffer, size_t length)
{
     return EPERM;
}


static status_t null_read(void *cookie, off_t pos, void *buffer, size_t *_length)
{
     *_length = 0;
     return B_OK;
}

static status_t null_write(void *cookie, off_t pos, const void *buffer, size_t *_length)
{
     return B_OK;
}
```

Pilotes : Exemple /dev/null

```
status_t init_hardware()
{
	return B_OK;
}

const char **publish_devices(void)
{
	static const char *devices[] = {
		DEVICE_NAME,
		NULL
	};

	return devices;
}


device_hooks *find_device(const char *name)
{
	static device_hooks hooks = {
		&null_open,
		&null_close,
		&null_freecookie,
		&null_ioctl,
		&null_read,
		&null_write,
	};

	if (!strcmp(name, DEVICE_NAME))
		return &hooks;

	return NULL;
}
```

```
status_t init_driver(void)
{
	return B_OK;
}


void uninit_driver(void)
{
}
```

Conclusions

- BeOS pionnier du Multimédia grand-public
- C'est bon de voir que Linux nous donne raison
- BeOS bien sur inspiré par d'autres (SGI...)
- Haiku s'inspire de BeOS et le revendique
- Linux, OSX, ... héritent de BeOS
- Haiku aussi s'inspire des autres et expérimente
- L'hybridation est un phénomène naturel de la technosphère, Libre ou non.

Conclusion Générale

Si la biodiversité est essentielle à notre planète, la technodiversité est nécessaire la logisphère.

Besoin de vous

- Manque
  - Pilotes
  - Applications
- Déboggage
- Testeurs
  - Portage Falcon et ARM ;-D

Liens

- Cette Présentation
  - http://revolf.free.fr/RMLL/2008/Haiku
- Haiku
  - http://haiku-os.org
  - irc://#haiku@irc.freenode.org/
- François Revol
  - revol@free.fr

Questions ?

BeOS

et

Fournisseurs d'idées depuis 1990

Remerciements

- Olivier Coursière (soutien moral et technique)
- Thomas Petazzoni (nouveautés dans Linux 2.6)
- Be, Inc. pour BeOS et la BeBox
- La team Haiku
- L'album « Voices of Amiga » utilisé pour cette démo : http://www.jamendo.com/fr/album/2964/