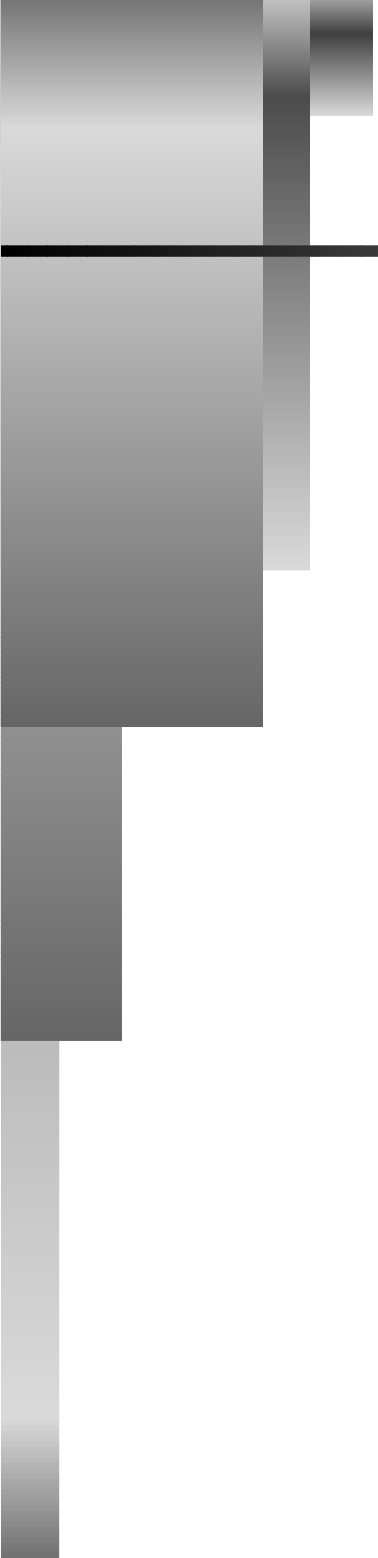




Building an Embedded Linux Prototype

Devin Carraway, Chuck Groom

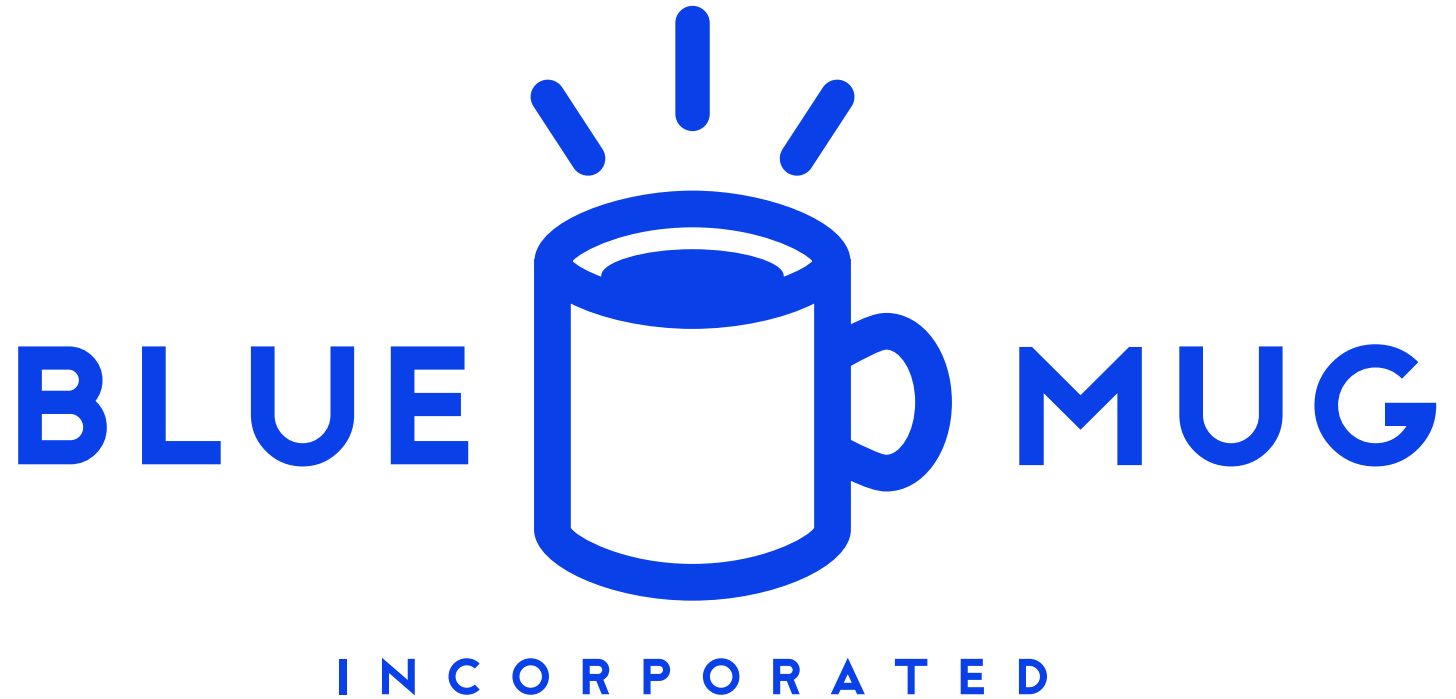
Blue Mug, Inc.

Contents copyright 2002 Blue Mug, Inc.

All rights reserved

- About Blue Mug, Inc.
- Project
- Hardware Selection
- Low-level
- User Interface
 - UI Design
 - Embeddable Linux GUIs
 - Modifying Gtk+

About Blue Mug, Inc.



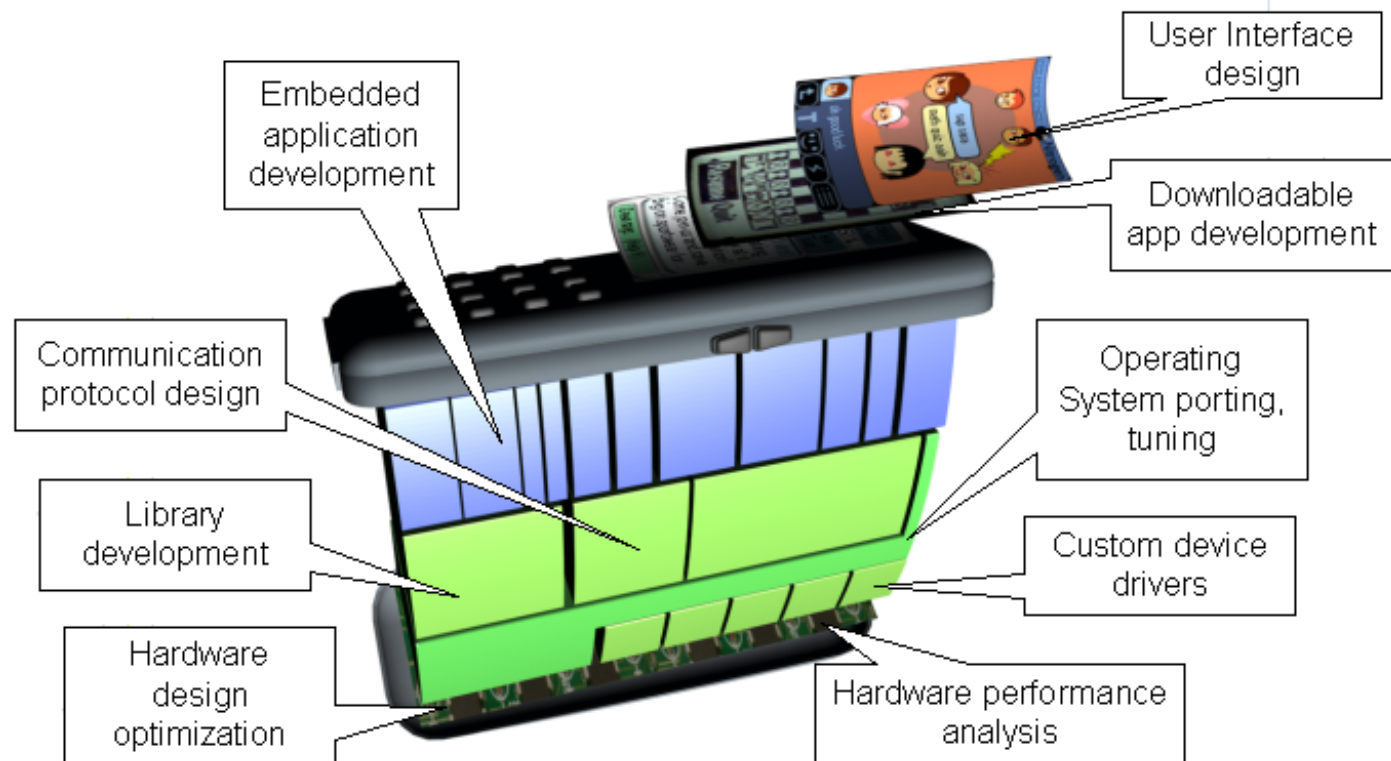
About Blue Mug, Inc.

Blue Mug creates software for mobile devices



About Blue Mug, Inc.

Blue Mug creates software for mobile devices



About Blue Mug, Inc.

Blue Mug creates software for mobile devices

- Located in Berkeley
- About 18 employees, 90% engineers
- Founded in 1999 (from Geoworks' Mobile OS Group)

About Blue Mug, Inc.

Blue Mug creates software for mobile devices

Linux is appealing for embedded solutions

- Business: Free (as in beer), actively developed technology
- Developer: sane platform
- Users: stable, fast

Blue Mug creates software for mobile devices

But we're not a Linux-only company

- GEOS-SC OS
- Palm OS
- Symbian OS
- μ ITRON RTOS
- J2ME, BREW
- Microprocessor projects

Our client asked us to create a prototype for a device:

- Low-cost (<\$100)
- Soft-key input
- Small gray-scale screen
- Palm-like battery life (22hrs)
- Can run simultaneous apps
- Multiple access points (modem, PCMCIA for Ethernet, Bluetooth, etc.)

Project Example Mockup



Which embeddable system-on-a-chip to use?
Considerations:

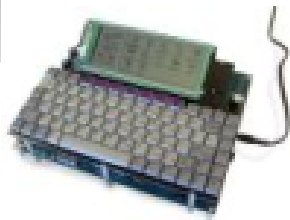
- Performance
- Price
- Power consumption

Which embeddable system-on-a-chip to use?

StrongARM, PPC use too much power, cost too much

MIPS, SH are struggling

Which embeddable system-on-a-chip to use?



ARM is cheap, low-power, reasonably fast.
We choose the Cirrus Logic EP7211 board

- 75Mhz ARM7
- 16Mb Flash, 16Mb RAM
- Low-power (170mw)
- Successor to PS7110 used in Psion Series 5, for which there is a Linux port.

System Overview

- Two 8Mb banks of Flash
 - Kernel in one bank
 - Root file system in other bank (mounted read-only)
- /tmp in RAM
- User files, add-on apps in RAM
- No swap!

- 8Mb Flash for all libraries, GUI, windowing system, and apps
 - JFFS2 and cramfs (compressed file systems) weren't ready at the time
 - Could compile in Thumb (16-bit) instruction set
 - Size-for-speed trade-off
 - Tricky; dynamic linking, c library...
 - For this project, we had to be extremely space-conscious

- 16Mb RAM
- What to when out of memory?
 - Difficult on desktop. Linux kills processes based on CPU usage, run time, and access to privileged I/O resources.
 - Easier on embedded systems
 - Known set of processes (eg. BeOS' "kill the browser" approach)
 - Tie into UI to display warning or errors
 - Require apps to be aware of low-memory situations
- Never ran out of RAM in testing

- Why not XIP (eXecute In Place)?
 - Opportunistic way to use less RAM
 - If page does not contain relocation address, run directly from Flash or disk
 - Pages cannot be compressed
 - Flash is slower than RAM, so apps may run slower
- XIP reduces RAM usage, but not Flash
- At time, Flash was expensive and hard to get and RAM was relatively cheap, we decided against XIP in favor of a compressed flash file system

Low-Level: Memory Mapping

EP7211 memory is non-contiguous

- Use kernel macros to map between actual and linear presentation of memory

Low-Level: Keyboard Driver

The EP7211 eval board has annoying keyboard

→ No hardware interrupt!

- Every 5 jiffies, generate interrupt
- Scan keyboard
- If key press, queue task to process keys
 - Can't process keys in timer process because it may take too long and cause the serial to drop

Low-Level: Which C Library?

C library is almost as big as kernel. Which C library to use?

- glibc: GNU C library, the standard
- sglibc: Patched glibc
- μ CLibc: Reduced-size, standard API
- Diet libc: Reduced-size, breaks API

Start with GLibC, move to sglibc.

User Interface: Design Principles

- The user interface (UI) can mean the success or failure of a consumer device
- You can't have a general-purpose mobile device GUI; it must fit device particulars

Good example: Palm UI fits small-screen, stylus-central organizer

Bad example: WinCE UI presents entire desktop interface on small screen

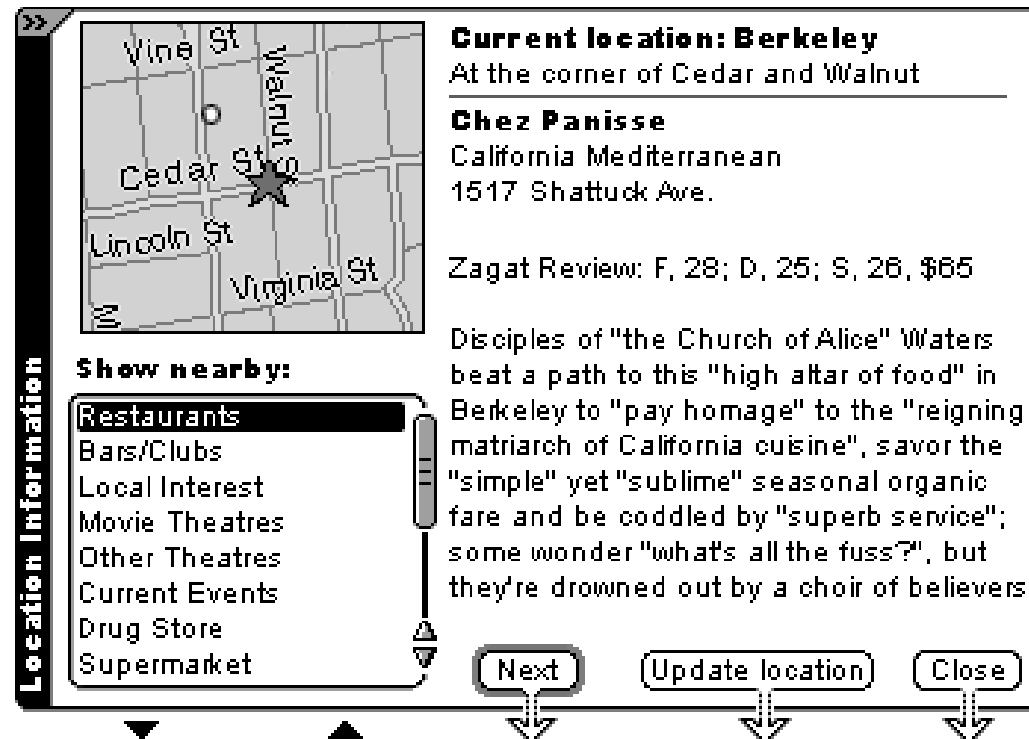
Always keep the user's goals in mind.

- Mobility means urgency
- Objective is to get job done
- Technology by itself isn't a feature
- Status notification only for things that matter
- A too-rich feature set makes the device feels unpredictable

User Interface: Givens

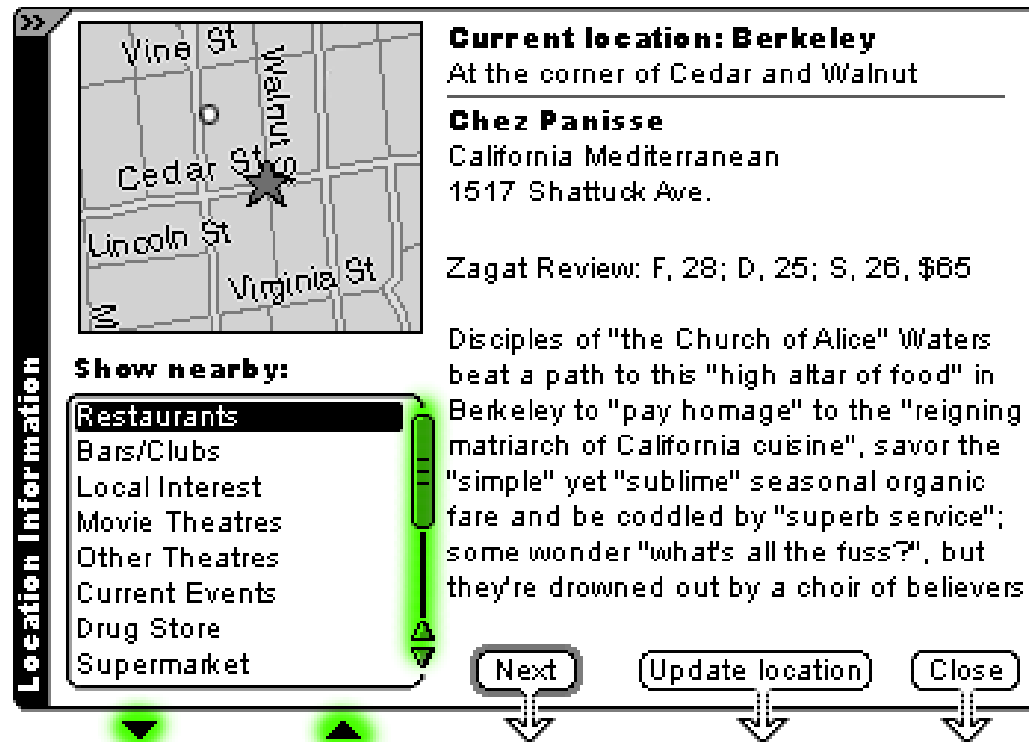
- Instant response to user interaction
- Always-on app model
- Primarily softkey control
- No touchscreen
- Cheap screen
 - Small
 - Grays cost power
 - Low-contrast
- “Walk up and use” interface
 - Limit user options
 - Borrow desktop elements as needed

User Interface: Our Design



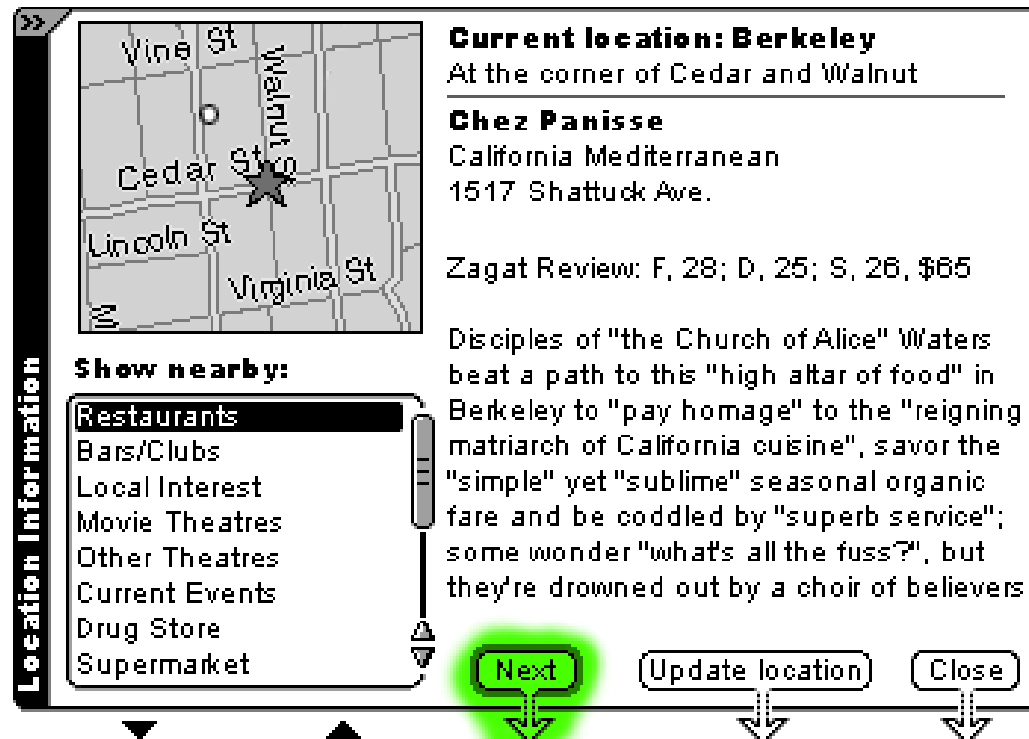
(This is a conceptual mockup)

User Interface: Our Design



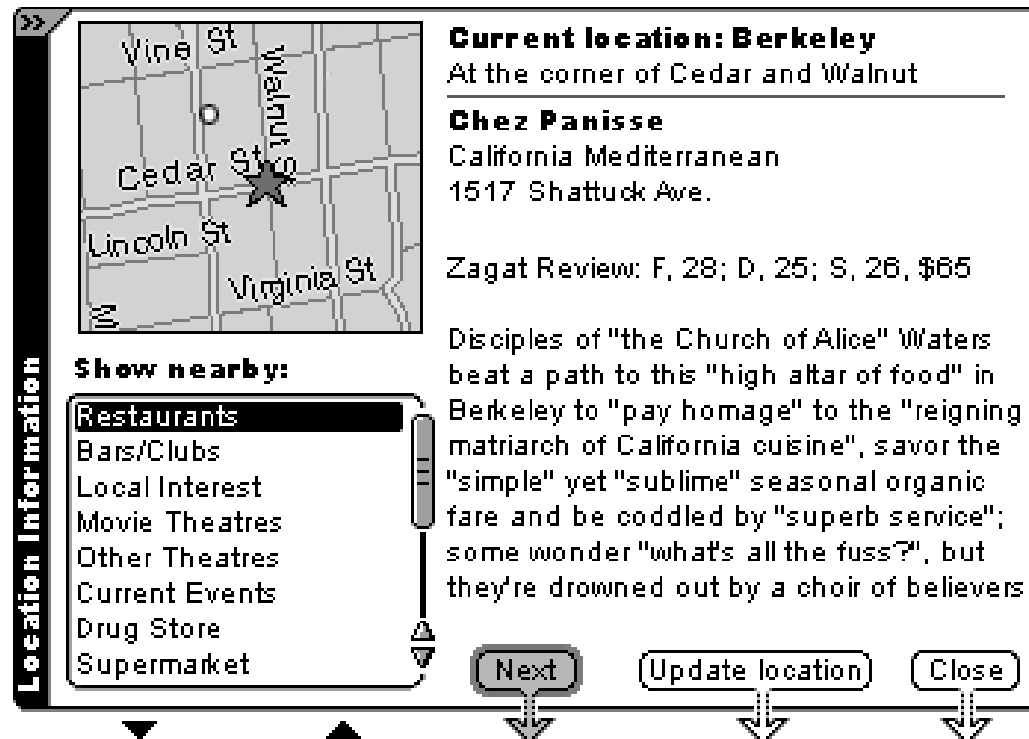
Use desktop GUI widgets with softkey control

User Interface: Our Design



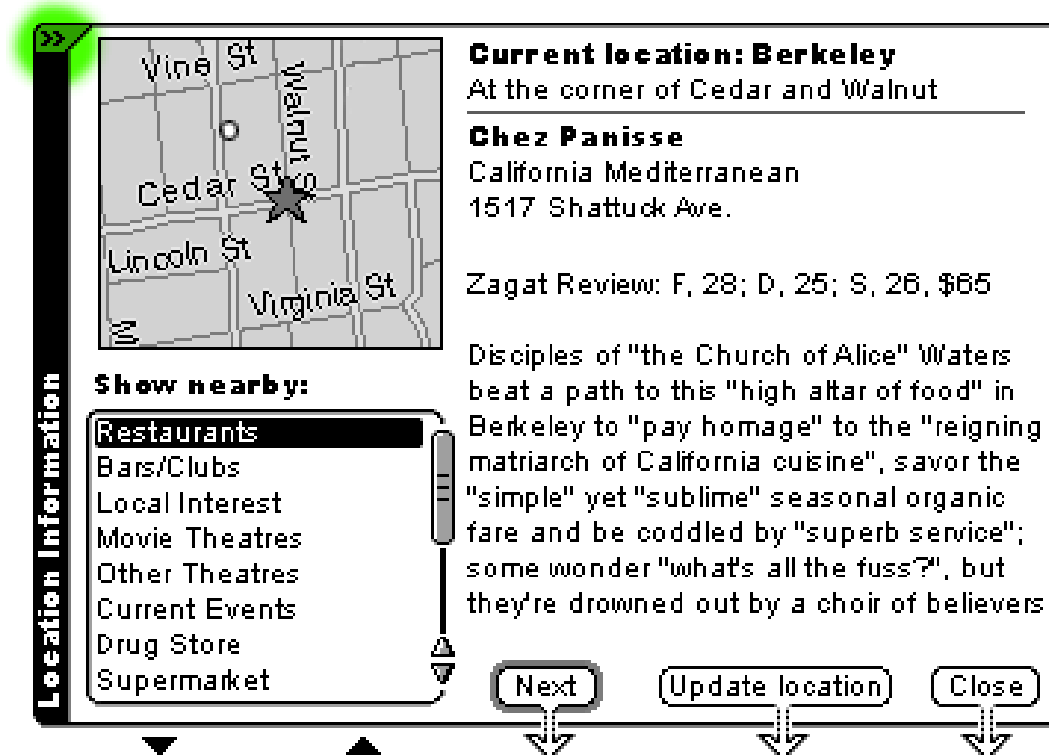
Use desktop GUI widgets with softkey control

User Interface: Our Design



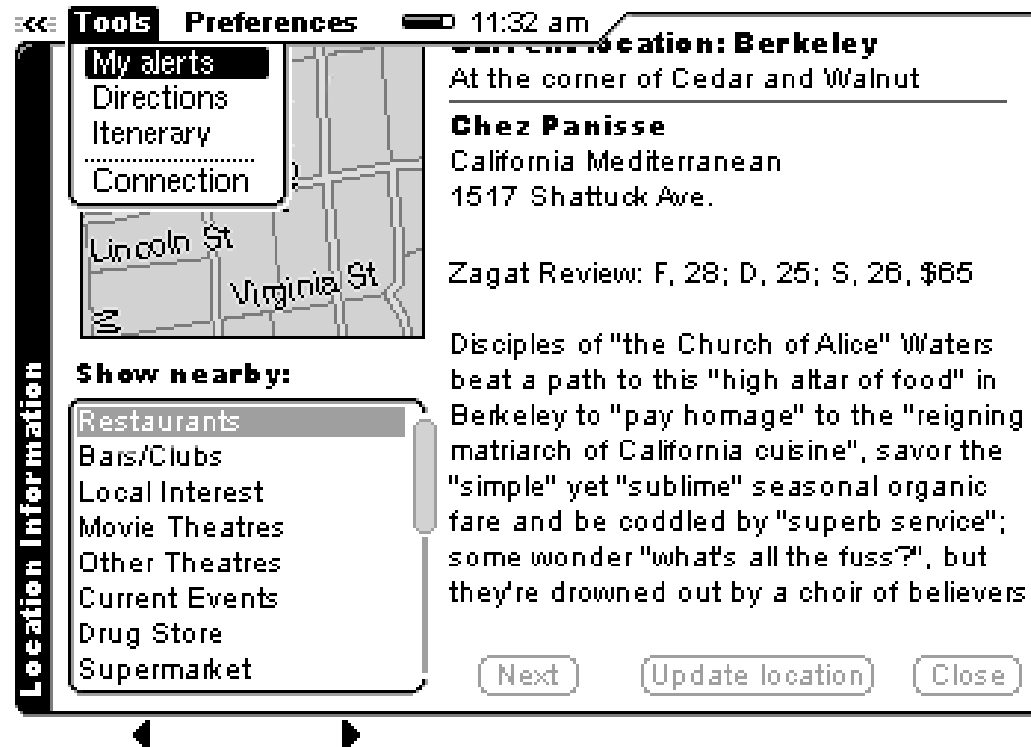
Use desktop GUI widgets with softkey control

User Interface: Our Design



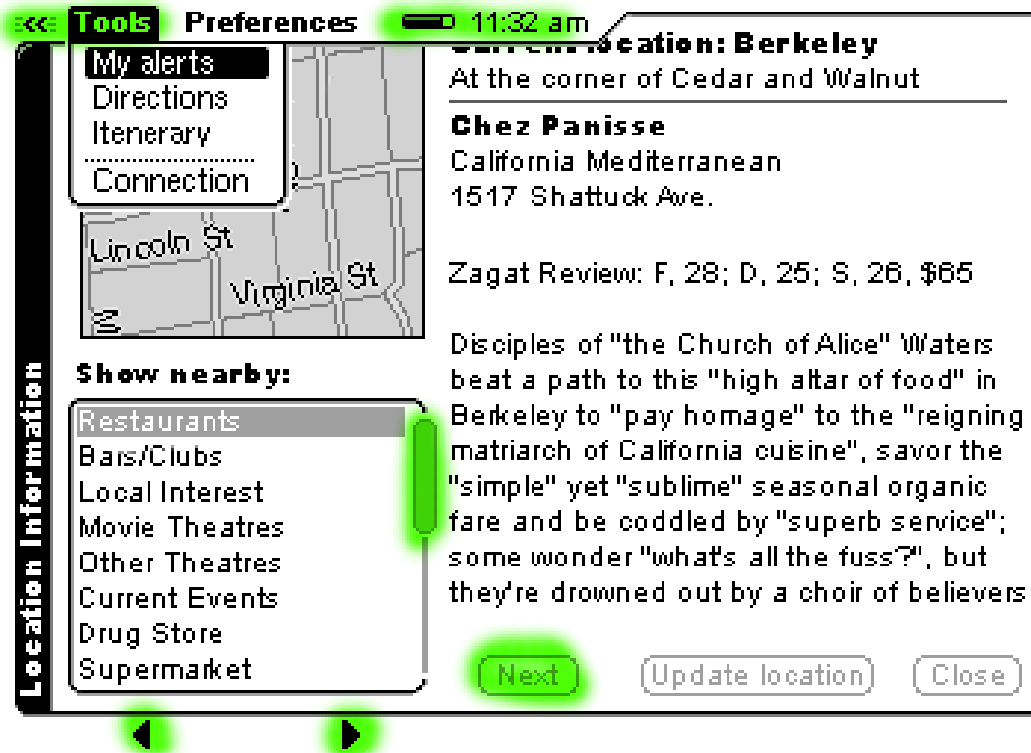
Place options in menu. Hide menu to save screen space, but indicate existence.

User Interface: Our Design



Menu bar includes time and battery.

User Interface: Our Design



Menu is modal and takes control of softkey bar. Other widgets are inactive.

User Interface: Other Elements

Other misc. design elements...

- The difference between softkeys and buttons
 - Softkeys can be stand-alone
 - Buttons affects pane
- Added “indeterminate” state to radio buttons, check boxes
- Dialogs
- When we launch an app, display “zoomy rectangle”

User Interface: Embeddable Linux GUIs

How do we implement this interface?

→ Tweak existing UI

How do we implement this interface?

→ Tweak existing UI

Criteria:

- Completeness
- Size
- Multiple apps can access framebuffer
- Language (C, C++)
- License

How do we implement this interface?

→ Tweak existing UI

- Gtk+
- Qt/e
- OpenGUI
- MiniGUI
- PicoGUI
- Microwindows
- ...

How do we implement this interface?

→ Tweak existing UI

Qt

- KDE Desktop
- Developed by TrollTech
- C++ framework
- Qt/E is reduced, runs on framebuffer
- QTopia app infrastructure
- Difficult to compile (circa Q1 2001)
- Dual-license

How do we implement this interface?

→ Tweak existing UI

Gtk+

- GNOME Desktop
- Open source project
- C
- Developed on X; also Gtk+/fb
- LGPL

User Interface: Embeddable Linux GUIs

How do we implement this interface?

→ Tweak existing UI

Decided on Gtk+ running on X

User Interface: Embeddable Linux GUIs

How do we implement this interface?

→ Tweak existing UI

Decided on Gtk+ running on X

X Windows! Eek!

- Client-server windowing system
- Network-transparent
- 20 years old
- Widely regarded as bloated and archaic

How do we implement this interface?

→ Tweak existing UI

Decided on Gtk+ running on X

We like X

- X is stable
- TinyX is, well, a tiny version of X
- Network-transparency is helpful
 - Quick UI analysis: run apps from desktop on device

User Interface: Embeddable Linux GUIs

How do we implement this interface?

→ Tweak existing UI

Decided on Gtk+ running on X

Modified AEWm window manager

- Vertical title bars
- Inter-app communication
- Application-level awareness of modal dialogs

User Interface: Modifying Gtk+

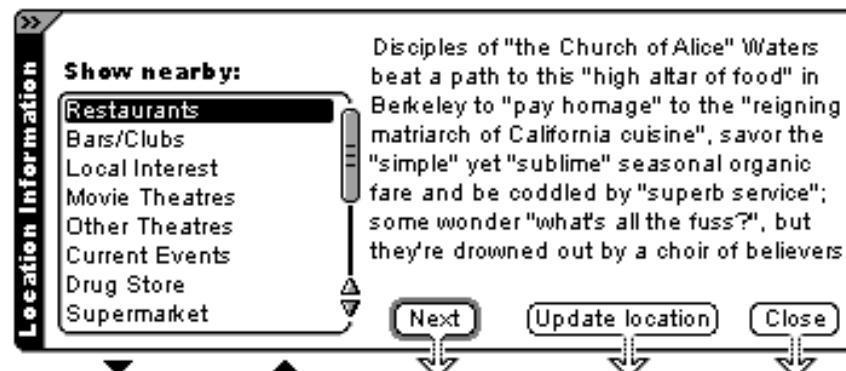
- Trim unnecessary widgets (eg. file dialog, color selection)
- Widget sizing
- Widget drawing
- GtkWindow
- Font management

Changes in-place, not sub-classed

2.9Mb footprint for Gtk+/X; this could be reduced to 2.4Mb.

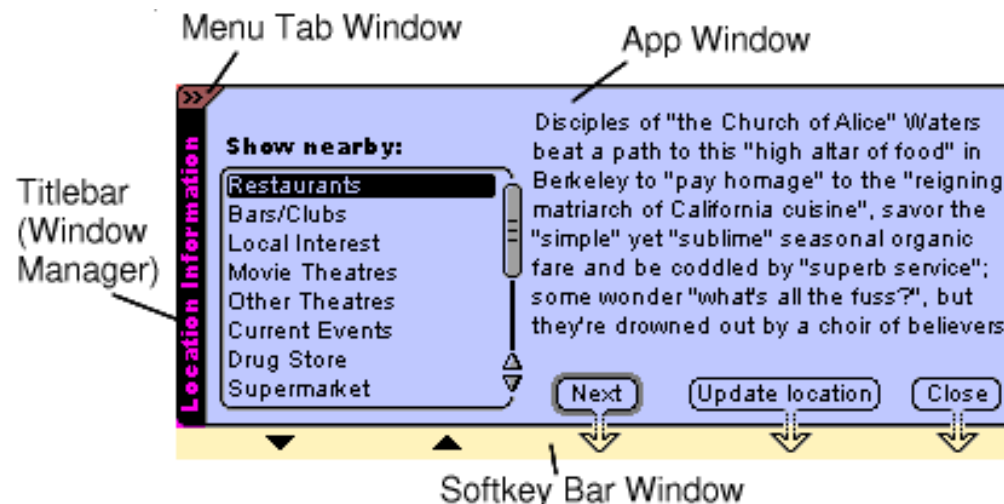
User Interface: *GtkWindow*

- Application window talks to window manager
- Application window has-a softkey bar
 - Not nested within widget
- API to register softkeys on application window
- Scrolling full-screen window



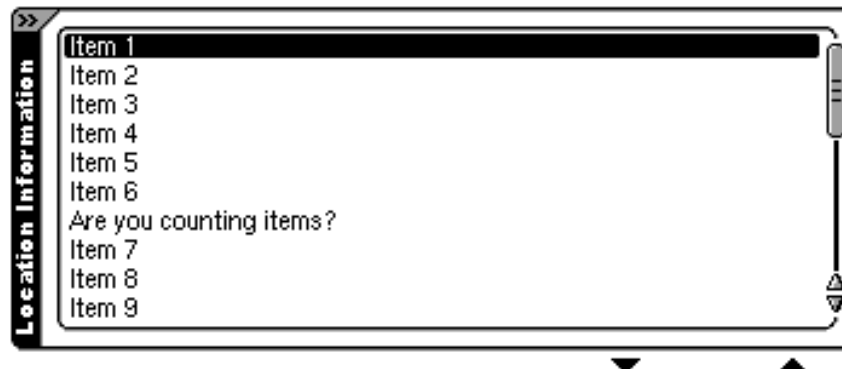
User Interface: *GtkWindow*

- Application window talks to window manager
- Application window has-a softkey bar
 - Not nested within widget
- API to register softkeys on application window
- Scrolling full-screen window



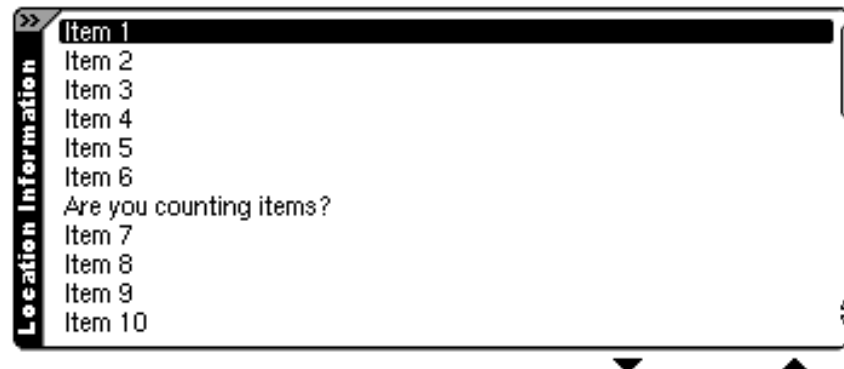
User Interface: GtkWindow

- Application window talks to window manager
- Application window has-a softkey bar
 - Not nested within widget
- API to register softkeys on application window
- Scrolling full-screen window



User Interface: GtkWindow

- Application window talks to window manager
- Application window has-a softkey bar
 - Not nested within widget
- API to register softkeys on application window
- Scrolling full-screen window



To change a font in stock Gtk+:

- Clone widget's `GtkStyle`
- Load a new X font, such as
`-adobe-helvetica-bold-r-normal-12-*-*-*-*p-*-*iso8859-1`

`GtkStyle` is fairly big, so this is expensive. And the developer has to know the specific font name.

To change a font in stock Gtk+:

- Clone widget's `GtkStyle`
- Load a new X font, such as
`-adobe-helvetica-bold-r-normal-12-*-*-*-*p-*-*iso8859-1`

We wrote API for requesting fonts by attribute relative to the base font.

```
gtk_widget_set_font_bold (widget, TRUE);  
gtk_widget_set_font_enlarge (widget, 1);
```

We added a `GdkFont * font` to `GtkWidget`. Use `widget->font` if possible, otherwise use `widget->style->font`

To change a font in stock Gtk+:

- Clone widget's `GtkStyle`
- Load a new X font, such as
`-adobe-helvetica-bold-r-normal-12-*-*-*-*p-*-*iso8859-1`

You can request font changes even before Gtk+ knows the base font.

User Interface: Performance

- Slow launch times
 - 2.4 seconds for most complicated app
 - Memory bandwidth bottleneck
 - For now, display eye candy when app is launched
 - In future, predictively launch applications

User Interface: Performance

- Slow launch times
- Loading pixmaps
 - XPM format is bulky
 - Gtk+ 1.2's XPM parser is terrible
 - Hack parser
 - Hand post-rendered pixmaps to X server

User Interface: Performance

- Slow launch times
- Loading pixmaps
- Floating point calculations
 - Floating point calculations are expensive on ARM
 - Gtk+ uses floating points for widget positioning
 - Integer math positioning gives a 3-12% speedup

Conclusion

- Client was happy with fully-functional prototype
- We're happy with our choice of Gtk+/X
- OSS made this project possible