

Mobile Computing

Sommersemester 2004
Hauptstudium Informatik
Hauptstudium Medieninformatik
Bachelor/Master Informatik

Michael Weber und Frank Kargl
Universität Ulm

Chapter 5:

Mobile operating systems

- Motivation
- Palm OS
- Symbian OS
- Windows CE
- Familiar Linux

Chapter 5:

Motivation
Palm OS
Symbian OS
Windows CE
Familiar Linux

Motivation

Tasks of an operating system

- Resource manager
 - Program execution
 - Memory, I/O
 - Interface for common tasks
 - File system
 - Communication
- ...comfortable implementation and safe programming

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Motivation

Mobile devices

- Scarce resources
 - No hard disk
 - Limited battery
- Limited hardware
 - Small memory (most likely transient memory)
 - Comparatively slow CPU
 - Limited CPU features, e.g. no MMU, no floating point unit

OS has to take these restrictions into account!

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Motivation

Additional OS tasks on mobile devices

- Emulate (or work around) missing hardware features
 - Multitasking, memory protection without MMU (current mobile CPUs provide a MMU)
 - File system replacement
- Provide libraries for common features
 - Ease implementation
 - Save memory (common library for all programs)
 - E.g. UI libraries

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Motivation

Mobile OS vs. embedded systems OS

- Mobile OS
 - Generic device
 - Multiple applications
 - Decent display/user interface
- Embedded system
 - Special purpose, highly specialized
 - Typically one application running
 - Usually no or very limited UI
 - Keep software (OS and application) as small as possible

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Common mobile OS

	Symbian	Win CE	Palm OS	Familiar Linux
Hardware platforms	ARM	ARM, XScale, x86 formerly: MIPS, PPC	Dragonball (MC680xx) ARM, XScale (≥5.0)	ARM
Prog.lang.	C++	C++, C#, VB	C, C++	C, C++, ...
Smartphns. EMEA	91,2%	7,8%	1%	-
PDA's	?	57,1%	32,3%	?
Typical battery runtime	~7h	~4-5h	~8h (ARM) ~12h (68k)	"similar to CE"

USA!?
(In Europe, Palm dominates)

Statistics: canalys.com, 4'04

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

ARM platform

- “old bull” for mobile devices
- ARM platform licensed to many companies (e.g. Intel XScale is based on ARM 5TE)
- “Thumb” instruction set: Short instruction words save memory (“*32-bit performance at 8/16-bit system cost*”)
- Several extensions (e.g. Jazelle – Java acceleration)
- Intelligent Energy Manager (IEM)
- All application cores have a MMU

Chapter 5:

Mobile operating systems

- Motivation
- Palm OS
- Symbian OS
- Windows CE
- Familiar Linux

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Palm OS

- Supported platform
 - Motorola Dragonball (up to Palm OS 4.0)
 - ARM (Palm OS 5.0+)
- Multitasking
 - Palm OS -4.0: Single usermode thread
→ cooperative multitasking
 - Palm OS 5.0+: “Real” multitasking
- Palm OS 5.0 backward compatibility

PACE (Palm Application Compatibility Environment)
provides environment for well-behaved 68k Palm OS programs

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Palm OS

History

- 1996: USRobotics Palm Pilot 1000, OS 1.0
- 1997: Palm Pilot Personal, OS 2.0
 - Background light; Prof.: TCP stack
- 1998: Palm III, OS 3.0
 - Flash ROM
- 1999: OS 3.1 – 3.3
 - Wireless connectivity, GPS/GSM interface
- 2000: OS 3.5
 - Color support!
- 2001: OS 4.0
 - New telephony library, Virtual File System
- OS 5.0
 - ARM support, hi-res displays, dynamic input areas
 - Graffiti 2
- 2003: OS 6.0
 - Enhanced multimedia support

Chapter 5:

Motivation

Palm OS

Symbian OS

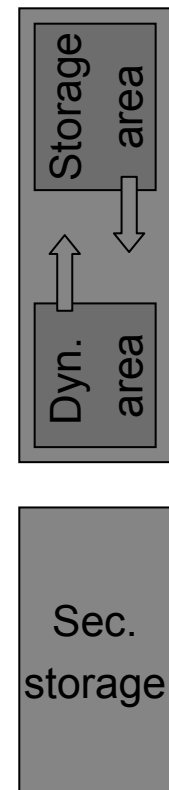
Windows CE

Familiar Linux

Palm OS

Memory management

- Main memory (volatile)
 - Dynamic area: Application stack and heap, OS stack and heap, global variables
 - Storage area: Program code, user data
 - Interface: Resource manager
- Secondary storage (persistent)
 - MMC, memory stick, ...
 - Data store only, program execution: Create temporary copy in RAM
 - Interface: Virtual File System (VFS, OS 4.0+)



Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Palm OS

- “Everything is database”
 - Databases = file system replacement
 - Programs executed “in place” (no copying)
- Common properties
 - Creator ID (4 chars): ID of creating application
 - Type (4 chars): Identifies purpose of resource
 - Name (32 chars): Globally unique “file” name
- Kinds of databases
 - Resource databases
 - Record databases

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Palm OS

- Resource databases
 - Resource = arbitrary data (up to 64k)
 - Database contains several resource entries
 - Programs are stored in resource databases
 - HotSync: Processed as one unit
- Record databases
 - Record = arbitrary data plus record info:
 - Number (0 – N-1) = array index
 - Unique ID (3 byte int) = ID for sync process
 - Flags (deleted, secret, locked, dirty=modified)
 - HotSync: Processed record by record

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Palm OS

Memory access

- Pointers
 - Points to (hopefully) allocated memory
 - Memory block is fixed
- Handles
 - Uniquely references allocated memory
 - Memory may be moved (if not locked)
 - Access → lock handle (returns pointer)
 - Database records stored in handles

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Palm OS

Program framework

- Start codes
 - Information on status of program
 - new start → initialize global variables
 - already running → reuse existing data
 - Special functions
 - Normal start
 - Save status (program ist forced to quit)
 - Find (does search query match with any saved data)
 - Go to (User selected search result from this program)

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Palm OS

Program framework

- Event loop
- System events
 - Pen strokes, Graffiti input, ...
 - Default handler exists (SysHandleEvent)
- Menu events
 - Default handler exists (MenuHandleEvent)
- User events
 - Events typically interesting for applications
 - If not processed: Fallback handler
FrmDispatchEvent

Chapter 5:

Motivation
Palm OS
Symbian OS
Windows CE
Familiar Linux

Palm OS – Hello World!

```
#include <System/SysAll.h>
#include <UI/UIAll.h>

DWord PilotMain( Word cmd, Ptr cmdPBP, Word launchFlags )
{
    EventType event;

    if (cmd == sysAppLaunchCmdNormalLaunch) {
        WinDrawChars( "Hello, world!", 13, 55, 60 );
        // Main event loop:
        do {
            // Doze until an event arrives.
            EvtGetEvent( &event, evtWaitForever );
            // System gets first chance to handle the event.
            SysHandleEvent( &event );

            // Normally, we would do other event processing here.

            // Return from PilotMain when an appStopEvent is received.
        } while (event.eType != appStopEvent);
    }
    return;
}
```

Chapter 5:

Mobile operating systems

- Motivation
- Palm OS
- Symbian OS
- Windows CE
- Familiar Linux

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Symbian OS

- Supported platforms
 - ARM (ARM 4+)
 - x86 (for the emulator)
- Preemptive multitasking
- 2GB addressable memory
- OS tailored for mobile phones
 - Small, ubiquitous devices
 - Occasionally connected to the Internet or other devices
 - Manufacturers differentiate their products, yet an independent, open software platform is desirable

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Symbian OS

- Process model
 - Process = Unit of resource reservation; own address space
 - Read-only memory globally shared
 - Thread = Unit of execution
 - Preemptive scheduling: The highest-priority thread that is ready to be executed will be scheduled by the kernel
 - Write memory protected by MMU (shared between threads)

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Symbian OS

Memory layout

- RAM (volatile)
 - Application stack and heap, OS stack and heap, global variables
 - (Virtually) no OS-imposed limits on stack/heap size
- Flash memory, removable MMC (persistent)
 - Data store
 - Program code

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Symbian OS

Implementation peculiarities

- Strict Model/View/Controller architecture
- Uses home-cooked exceptions (instead of C++ exception handling)
- Naming conventions: E.g.
 - Suffix “L” – function may leave
 - Suffix “LC” – function may leave, and add something to clean up on the stack
- “Descriptors” \approx strings with boundary checks

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Symbian OS

“Leaving” and left allocated memory

(rough idea of the concept)

- Cleanup-Stack clears allocated heap space
 - Allocation of memory → Reference in Cleanup-Stack
 - “leave”: Memory linked in Cleanup-Stack (associated with current stack level) is freed
 - Allocated memory: free()
 - Object instances: destructor is called
- No garbage collector

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Symbian OS

What if a constructor runs into an error?

- Memory allocations may fail any time
- Constructors (should) never fail
 - If they do, they won't be registered in Cleanup stack
- 2-phase object construction
 - Basic initialization in C++ constructor
 - Complex initialization in ConstructL()
 - Functions that may leave: Only in ConstructL()
 - ConstructL()s must be called manually

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Symbian OS

Active Objects

- Set of classes for event handling and -dispatching
 - Events dispatched in event loop
 - Asynchronous calls send results as messages
- ...although real multitasking available
 - Threads comparatively heavy
 - More control to programmer
 - (Supposedly) Less problems with deadlocks

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Symbian OS

- Common resources/services encapsulated in servers, single server per service
 - File system server
 - Socket server
 - Messaging server
 - ...
- Separate thread per server
- Communication with server: Asynchronous, message passing

Chapter 5:

Motivation
Palm OS
Symbian OS
Windows CE
Familiar Linux

Symbian OS – Hello World!

```
TUid CHelloWorldApplication::AppDllUid() const
{
    return KUidHelloWorld;
}

CApaDocument* CHelloWorldApplication::CreateDocumentL()
{
    return CHelloWorldDocument::NewL(*this);
}

EXPORT_C CApaApplication* NewApplication()
{
    return new CHelloWorldApplication;
}
```

Chapter 5:

Motivation
Palm OS
Symbian OS
Windows CE
Familiar Linux

Symbian OS – Hello World!

```
CEikAppUi* CHelloWorldDocument::CreateAppUiL()
{
    return new (ELeave) CHelloWorldAppUi;
}

/**
 * Symbian OS 2 phase constructor.
 * Constructs the CHelloWorldDocument using the constructor and
 * the ConstructL method.
 */
CHelloWorldDocument*
CHelloWorldDocument::NewL(CEikApplication& aApp)
{
    CHelloWorldDocument* self = new (ELeave)
    CHelloWorldDocument(aApp);
    CleanupStack::PushL(self);
    self->ConstructL();
    CleanupStack::Pop(self);
    return self;
}
```

Chapter 5:

Motivation
Palm OS
Symbian OS
Windows CE
Familiar Linux

Symbian OS – Hello World!

```
void CHelloWorldAppUi::HandleCommandL(TInt aCommand) {
    switch (aCommand) {
        case EEikCmdExit:
        case EAknSoftkeyExit:
            Exit();
            break;
        case EHelloWorldCommand1:
            HBufC* message = StringLoader::LoadLC(R_DIALOG_TEXT);
            // Pushes message onto the Cleanup Stack.
            CAknInformationNote* informationNote =
                new (ELeave) CAknInformationNote;
            informationNote->ExecuteLD(*message);
            CleanupStack::PopAndDestroy(message);
            // Removes message from the Cleanup Stack.
            break;
        default:
            // Abort application if we don't recognize the command
            LIT(applicationName, "HelloWorld");
            User::Panic(applicationName, KErrUnknown);
            break;
    }
}
```

Chapter 5:

Mobile operating systems

- Motivation
- Palm OS
- Symbian OS
- Windows CE
- Familiar Linux

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Windows CE

- Supported platforms
 - ARM, x86
 - Formerly MIPS, PPC, SuperH
(dropped in order to simplify development)
- Preemptive multitasking, 4GB addressable
- Naming
 - Windows CE 1.0 - 2.12
 - Microsoft Pocket PC
(based on Windows CE 3.0/Handheld PC 2000)
 - Microsoft Pocket PC 2002 (*Phone Edition* avail.)
 - Microsoft Windows Mobile 2003 for Pocket PC

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Windows CE

- Origins
 - 1992: WinPad (pad computer with handwriting recognition): OS “Microsoft at work”
 - 1990ies: Pulsar (PDA project) Effort to port WinNT
 - All efforts failed; new project: “Windows companion device”
 - Same object-oriented OS as WinPad
 - Too many bugs, not WinAPI compatible
- Rewrite with subset of Win32 API
- Pegasus = Windows CE 1.0 (1996)

Chapter 5:

Motivation
Palm OS
Symbian OS
Windows CE
Familiar Linux

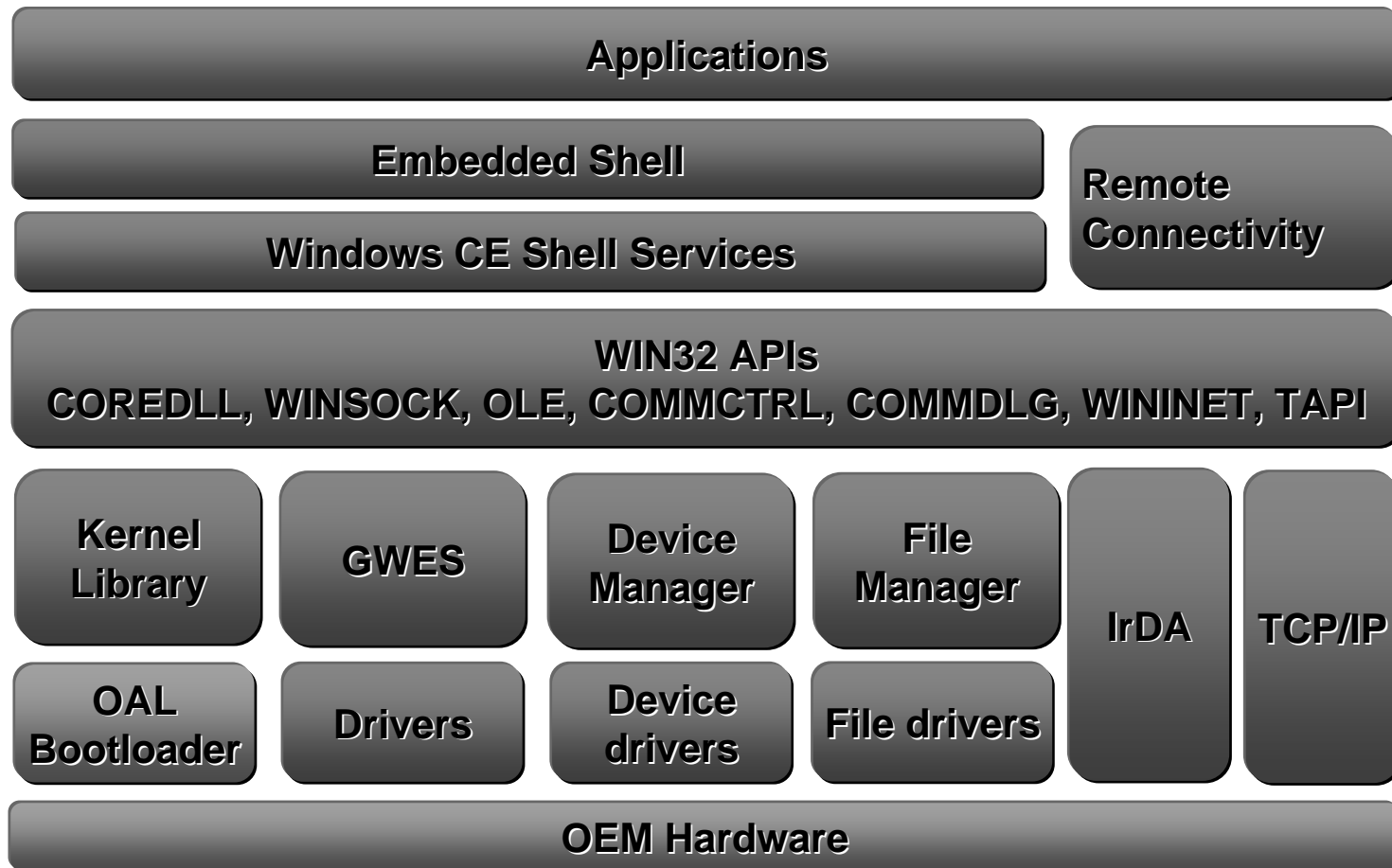
Windows CE

- Designed as embedded system framework
- Modular design
 - Unneeded modules may be removed
 - Core can be reduced to ~500kB
 - Graphical UI adds ~2MB, etc.
- Pocket PC: Microsoft-defined platform for PDAs
 - Certification rules more strict than for WinCE
 - Runs a variation of WinCE

Chapter 5:

- Motivation
- Palm OS
- Symbian OS
- Windows CE
- Familiar Linux

Windows CE



Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Windows CE

Memory architecture

- 2 GB virtual address space for programs
- 32 MB address space per process
- Large memory blocks: Memory mapped files
- Maximum of 32 processes
- 60 kB stack per process
- RAM split in Object Store / Program Memory

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Windows CE

Storage types

- Windows CE file system
 - User-installed applications
 - Data files
 - Compressed file system
- Windows CE database
 - Structured storage
 - API for insert/delete/update records
- System registry
 - Similar to desktop registry

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Windows CE

Dealing with scarce RAM/Program slots

- Close-Button does not terminate program: Program is simply suspended
- Lack of process slots: Least recently used program is terminated by system
- Low memory condition:
 - Boundary between object store and program memory is moved
 - Suspended programs are terminated
 - At least in theory...

Chapter 5:

Motivation
Palm OS
Symbian OS
Windows CE
Familiar Linux

Windows CE – Hello World!

```
CMainWindow MainWindow;

int WINAPI WinMain( HINSTANCE hInstance, HINSTANCE hPrevInstance,
                   LPTSTR lpCmdLine, int nCmdShow )
{
    MSG msg;
    HACCEL hAccelTable;

    // Perform application initialization
    if ( !MainWindow.InitInstance( hInstance, nCmdShow ) )
        return FALSE;
    hAccelTable = ::LoadAccelerators( hInstance, (LPCTSTR)IDC_LTWEIGHT );

    // Main message loop
    int status;
    while ( ( status = GetMessage( &msg, NULL, 0, 0 ) ) != 0 )
    {
        if ( status == -1 ) return -1;
        if ( !TranslateAccelerator( msg.hwnd, hAccelTable, &msg ) )
        {
            TranslateMessage( &msg );
            DispatchMessage( &msg );
        }
    }

    return msg.wParam;
}
```

Chapter 5:

Motivation
 Palm OS
 Symbian OS
 Windows CE
 Familiar Linux

Windows CE – Hello World!

```

BOOL CMainWindow::InitInstance( HINSTANCE hInstance, int nCmdShow )
{
    HWND  hWnd = NULL;
    TCHAR szTitle[MAX_LOADSTRING];           // The title bar text
    TCHAR szWindowClass[MAX_LOADSTRING];     // The window class name
    (...)
    // Initialize global strings
    LoadString( hInstance, IDC_LTWEIGHT, szWindowClass, MAX_LOADSTRING );
    LoadString( hInstance, IDS_APP_TITLE, szTitle, MAX_LOADSTRING );

    // If it is already running, then focus on the window
    hWnd = FindWindow(szWindowClass, szTitle);
    if ( hWnd )
    {
        SetForegroundWindow ((HWND) (((DWORD)hWnd) | 0x01));
        return 0;
    }
    MyRegisterClass( hInstance, szWindowClass );
    (...)
    hWnd = CreateWindow( szWindowClass, szTitle, WS_VISIBLE,
                        CW_USEDEFAULT, CW_USEDEFAULT, CW_USEDEFAULT,
                        CW_USEDEFAULT, NULL, NULL, hInstance, (void *)this );
    if ( !hWnd ) return FALSE;
    (...)
    ShowWindow( hWnd, nCmdShow );
    UpdateWindow( hWnd );

    return TRUE;
}

```

Chapter 5:

Motivation
 Palm OS
 Symbian OS
 Windows CE
 Familiar Linux

Windows CE – Hello World!

```

LRESULT CMainWindow::WndProc( HWND hWnd, UINT message,
    WPARAM wParam, LPARAM lParam, PBOOL pbProcessed )
{
    (...variable declarations...)
    // call the base class first
    LRESULT lResult = CBaseWindow::WndProc( hWnd, message,
        wParam, lParam, pbProcessed );
    BOOL bWasProcessed = *pbProcessed;
    *pbProcessed = TRUE;

    switch ( message )
    {
        case WM_COMMAND:
            wmId      = LOWORD(wParam);
            wmEvent = HIWORD(wParam);
            switch (wmId) // Parse the menu selections:
            {
                case IDOK:
                case IDM_EXIT:
                    SendMessage( hWnd, WM_ACTIVATE,
                        MAKEWPARAM(WA_INACTIVE, 0), (LPARAM)hWnd );
                    SendMessage ( hWnd, WM_CLOSE, 0, 0 );
                    break;
                default: // the message was not processed
                    *pbProcessed = bWasProcessed;
                    lResult = FALSE;
                    return lResult;
            }
            break;
    }
}

```

Chapter 5:

Motivation
 Palm OS
 Symbian OS
 Windows CE
 Familiar Linux

Windows CE – Hello World!

```

case WM_CREATE:
    hwndCB = CreateRpCommandBar( hWnd );
    break;
case WM_PAINT:
    RECT rt;
    hdc = BeginPaint( hWnd, &ps );
    GetClientRect( hWnd, &rt );
    LoadString( GetInstance(), IDS_HELLO, szHello, MAX_LOADSTRING );
    DrawText( hdc, szHello, _tcslen( szHello ), &rt,
              DT_SINGLELINE | DT_VCENTER | DT_CENTER );
    EndPaint( hWnd, &ps );
    break;
case WM_DESTROY:
    CommandBar_Destroy( hwndCB );
    PostQuitMessage( 0 );
    break;
case WM_SETTINGCHANGE:
    SHHandleWMSettingChange( hWnd, wParam, lParam, &s_sai );
    break;
default:
    // the message was not processed
    // indicate if the base class handled it
    *pbProcessed = bWasProcessed;
    lResult = FALSE;
    return lResult;
}
return lResult;
} /* CMainWindow::WndProc */

```

Chapter 5:

Mobile operating systems

- Motivation
- Palm OS
- Symbian OS
- Windows CE
- Familiar Linux

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Familiar Linux

- Linux Port to ARM platform
 - Kernel version 2.4
- Complete distribution
 - Binary and library compatible with Debian ARM
 - Repositories for online updates
- New bootloader
- Storage: JFFS2 (Journaling Flash File System)
- Temporary files in tmpfs system (dynamically adjusting ramdisk)
- Display: XFree's TinyX or Framebuffer

Chapter 5:

Motivation
 Palm OS
 Symbian OS
 Windows CE
 Familiar Linux

Familiar Linux

- First time installation: New bootloader
 - Flash root file system via
 - Serial cable
 - CF-Card (CF emulates hard disk logic)
 - Set up network
 - WLAN
 - Serial line + PPP
 - Get updates
- Done 😊



Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Familiar Linux

Small tool sets

- Libc alternatives
 - diet libc
 - newlib
- busybox
 - Common programs combined in one binary: ls, rm, mkdir, chown, cat, su, wget, ...
 - Usually less options than normal implementations
 - Implemented with size optimization in mind

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Familiar Linux

- Several project working on palmtop environments
- GPE Palmtop Environment
 - X Windows System (TinyX)
 - GTK+2.2 toolkit
- OPIE Open Palmtop Integrated Environment
 - Fork of Qtopia, QT Embedded toolkit
 - Display on framebuffer device (via QT)

Chapter 5:

- Motivation
- Palm OS
- Symbian OS
- Windows CE
- Familiar Linux

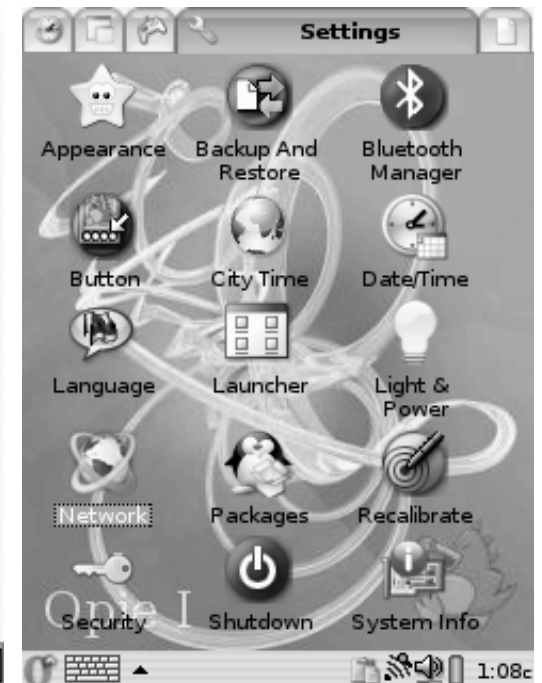
Familiar Linux



Pocket PC 2003



GPE



OPIE

Chapter 5:

Motivation

Palm OS

Symbian OS

Windows CE

Familiar Linux

Other Linux ports

- *μClinux*[™] – port to platforms without MMU
- Based on Kernel 2.0.x / 2.4.x
- Initially (1998) Linux port to 68k platform
- Support for many chipsets, e.g. Intel i960, Sigma Design DVD system, ...
- Drawbacks:
 - No autogrow stack, no brk(). Use mmem() instead
 - No memory protection
 - No fork(). vfork() starts new process, but blocks until its termination