DoVinci

Dortmund Virtualized Networked Campus Infrastructure

Olaf Spinczyk
Jochen Streicher
Horst Schirmeier
Michael Engel

Lehrstuhl XII
AG Eingebettete Systemsoftware
Fakultät für Informatik

http://ess.cs.uni-dortmund.de/DE/Teaching/PGs/dovinci/

Agenda

- **Einführung**
- Vision
- Hintergrundinformationen
- Durchführung
- Seminarthemen
- Nächste Schritte

- **Fragen**

DoVinci

- Dortmund Virtualized Networked Campus Infrastructure
  - Ein verteiltes Campus-Informationssystem basierend auf maßgeschneiderten virtuellen Maschinen
- Vision:
  - Abruf von Anwendungen für jeden auf dem Campus
    - Orientierung, aktuelle Informationen, Kommunikation
  - Bereitstellung der Anwendungen dezentral und unabhängig
    - Keine zentrale Registrierung, keine zentrale Infrastruktur notwendig
  - Ortsabhängige Dienste
    - z.B. Lageplan und Mitarbeiterliste eines Lehrstuhls
  - Zeitabhängige Dienste (z.B. aktueller Mensa-Speiseplan)
- Ausführungsplattform: Mobilgeräte
  - Netbooks oder Web-Tabletts
  - Nutzung vorhandener Virtualisierungsplattform
    - z.B. Xen oder Linux KVM auf Intel Atom oder VMware Mobile auf ARM

DoVinci


DoVinci VMs
Mensaplan
Bibliothek
Sonnendeck
Ersti-Info
Mensaplan 19.5.09
UB-Info OPAC

Agenda

- Einführung
- **Vision**
- Hintergrundinformationen
- Durchführung
- Seminarthemen
- Nächste Schritte

- **Fragen**

Lösungsansatz der Industrie

- Verwendung einer Hochsprachen-VM
  - z.B. Java Mobile Edition in Mobiltelefonen
- Probleme:
  - Kompatibilität zwischen verschiedenen Geräten
    - Unterschiedliche Versionen von JVM und Klassenbibliotheken
  - Ausführungsgeschwindigkeit
    - JIT-Compiler auf Mobilgeräten?
  - Nutzung spezieller Hardware (Kamera, Mikrofon, GPS, ...)
    - Spezifische Java Native Interface-Anbindung erforderlich
  - Keine vollständige Isolation einzelner Anwendungen
    - Ein JVM-Prozess im Host-Betriebssystem
  - Speicherverbrauch
    - Keine gerätespezifisch angepaßten Anwendungen
- Realisiert Anwendungen, die nur eine kleine, gemeinsame Teilmenge der Möglichkeiten nutzen

Unser Ansatz

- Virtualisierungs-Basisplattform auf Mobilgeräten
  - Einsatz von Systemvirtualisierung
  - Basierend auf existierenden Lösungen (z.B. Xen, Linux KVM)
- Anwendungen als „Appliances“
  - Virtuelle Maschine mit Anwendung, Bibliotheken, Betriebssystem und notwendigen Daten
  - Auslieferung über WLAN durch dedizierte Server
- Umgebung von Anwendungsanbieter bestimmbar
  - Betriebssystem, Programmier-sprache, Bibliotheken...
- Appliance-Maßschneiderung
  - Anpassung an Speichergröße
  - ...und andere Eigenschaften des Zielgerätes (Auflösung, Bandbreite, Eingabemethoden...)

Unser Ansatz

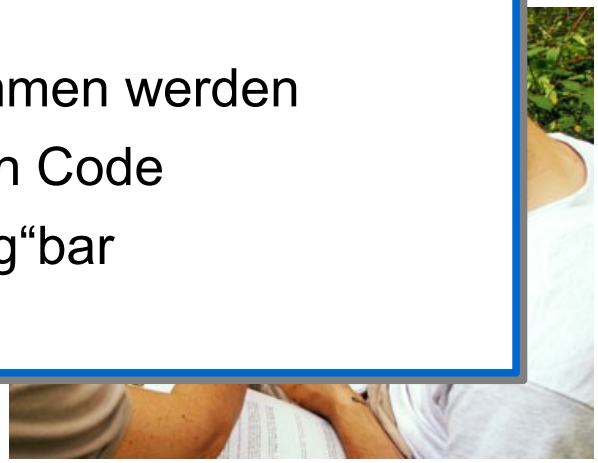
- Virtualisierungs-Basisplattform auf Mobilgeräten
  - Einsatz von Systemvirtualisierung
  - ...
- A...
  - ...
- M...
  - ...

Vorteile des DoVinci Ansatzes

- Interaktive, isolierte, vollständige Anwendungen ✓
- Maßschneiderung reduziert Ressourcenverbrauch ✓
- Offenes, verteiltes System ✓
- Unabhängigkeit von einer Systemplattform ✓

Darüber hinaus ...

- Software kann unverändert übernommen werden
- Besseres Zeitverhalten durch nativen Code
- Appliances beliebig lad- und „entsorg“bar
- ...



Die Highlights

- komplexe Aufgabe im Team lösen

- Virtualisierung auf Systemebene einsetzen
- Aufgaben in verschiedensten Bereichen der Informatik
  - Virtualisierung, Betriebssystem, Middleware, Anwendungen, Benutzerschnittstelle, Infrastruktur, ...
- Entwicklung eines komplexen verteilten Systems
- Mitarbeit an aktuellen Forschungsthemen
  - Virtualisierung in eingebetteten Systemen
  - Maßschneiderung virtueller Maschinen

Agenda

- Einführung
- Vision
- **Hintergrundinformationen**
- Durchführung
- Seminarthemen
- Nächste Schritte

- **Fragen**

Systemvirtualisierung

- Virtuelle Maschine, die ein **Hardwaresystem** nachbildet
  - Im Server- und Desktop-Bereich verbreitet: VMware, Xen etc.
  - Verwendet ein „komplettes“ Betriebssystem mit allen Libraries und Anwendungsprogrammen
- Keine neue Technologie
  - Von IBM schon in den 70er Jahren auf Großrechnern: VM/370
  - Seit 5-10 Jahren auf Desktops/Servern: VMware, VirtualBox
  - In eingebetteten Systemen: **aktuelles Forschungsgebiet**
- Open Source-Lösungen
  - Xen [3]: Universität Cambridge (UK), mittlerweile Citrix
    - Plattformunabhängig, eigenständige Software
  - KVM [2]: Linux-eigene Lösung
    - Linux Kernel-Modul
  - Qemu [6]: Portable System-Emulation
    - Emulator mit Hardware-Beschleunigung

Hochsprachen-Virtualisierung

- Virtuelle Maschine, die **Sprachumgebung** bereitstellt
  - ...aber keine vollständige Umgebung für ein Betriebssystem
    - Beispiel: Java VM
  - Meist abstraktere Maschinenbefehle vgl. mit „echten“ CPUs
    - Sog. „Bytecodes“
    - Oft als Stack-Maschine implementiert → einfachere Compiler
  - Ursprünglich: Performanceprobleme
    - Interpretation der Bytecodes
    - Mittlerweile: Beschleunigung durch Just-in-Time (JIT) Compiler
- Auf Laufzeitumgebung angewiesen
  - Windows, Linux, OS X + VM-Anwendung
  - „Native“ Java CPUs haben sich nicht durchgesetzt

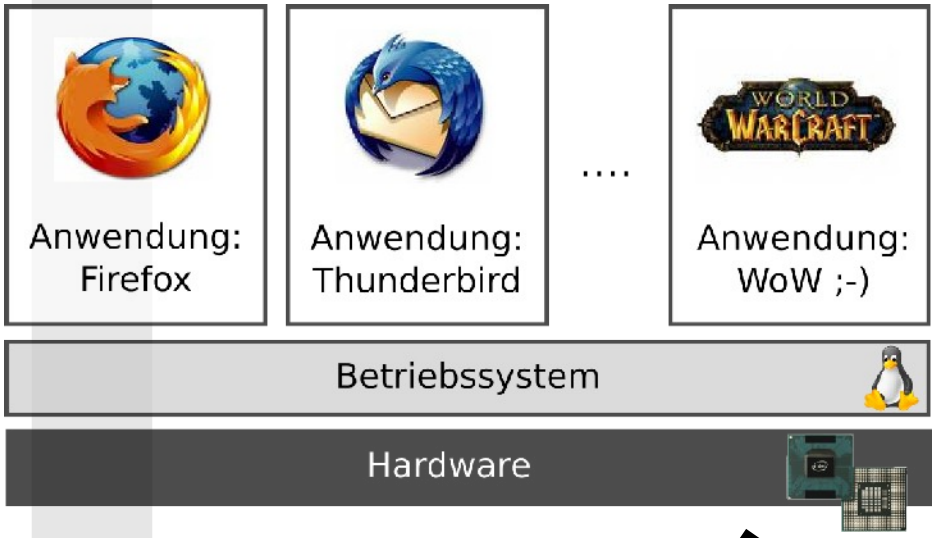
Pro und Contra

- Systemvirtualisierung
  - Vorteile:
    - **Unabhängigkeit** vom Betriebssystem: Linux, WindowsCE, ...
    - Vollständige **Isolation** zwischen einzelnen Systemen
    - Reduktion der **Abhängigkeiten** zwischen Software-Komponenten
  - Probleme:
    - Ressourcenknappheit in eingebetteten Systemen
      - Speicher, Rechenleistung, Energie, Bandbreite...
- Hochsprachen-Virtualisierung
  - Vorteile:
    - Unabhängigkeit von eigentlicher CPU-Architektur
    - Geringerer Speicherverbrauch
  - Probleme:
    - Keine vollständige Systemumgebung - Isolationsprobleme je nach VM
    - Performance und Speicheroverhead
    - Direkter Zugriff auf spezielle Hardware

Systemvirtualisierung: Details

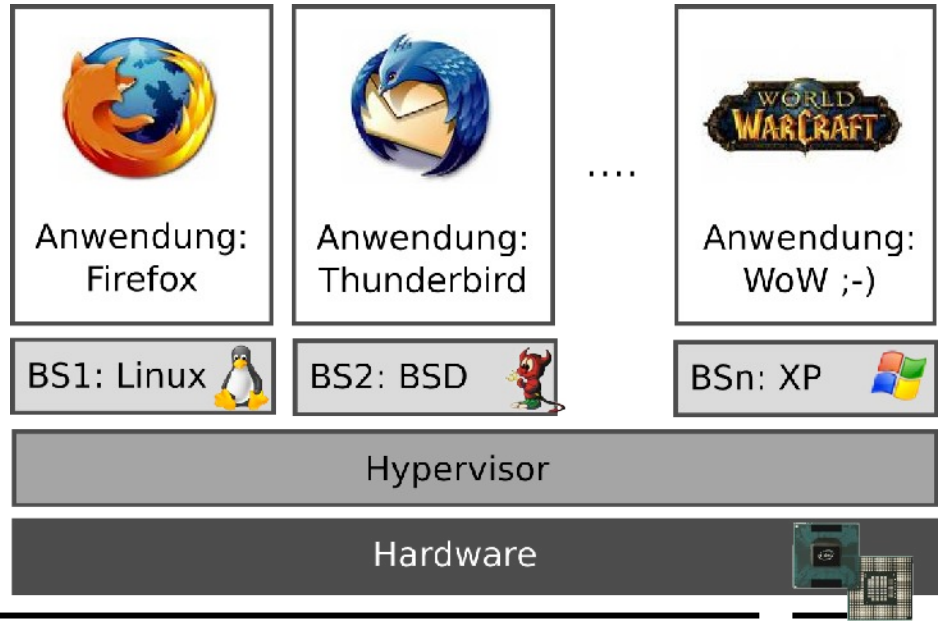
- Echte Betriebssysteme auf virtueller Hardware
  - Sog. „Gast“-Betriebssysteme
  - Problem: mehr als ein Betriebssystem **gleichzeitig**!
  - Jedes BS belegt Ressourcen normalerweise exklusiv
- Zugriff auf Ressourcen muss geregelt werden
  - Außerdem: Gast-Betriebssysteme starten/beenden
- Lösung: Hypervisor
  - „Meta“-System, kontrolliert die Ausführung der Gast-BSe
  - Muss privilegierte Operationen der Gast-BSe abfangen
  - Stellt Zugriff auf Ressourcen zur Verfügung
    - Rechenzeit, Haupt-/Massenspeicher, Netzwerk...
  - Soll möglichst wenig Ressourcen selbst beanspruchen
    - Speicher, Rechenleistung

Systemvirtualisierung



„Ist“

„Soll“



Virtualisierung in DoVinci

- System führt gerade zwei Appliances aus
  - Dritte Appliance wird aktuell vom Server abgerufen
- Basis: Mini-System zur Virtualisierungskontrolle
  - z.B. basierend auf Linux-Kernel mit KVM-Virtualisierung

Maßschneiderung

- Anpassung eines Systems auf bestimmte Anforderungen
  - Weglassen „überflüssiger“ Komponenten
    - Alle, die für die geg. Anforderungen nicht benötigt werden
  - Wichtig für eingebettete Systeme → Ressourcen sparen
- Konstruktive Maßschneiderung
  - Ausgangspunkt: **nichts** ☺
  - Aufbau des Systems „von Null auf“
  - Hinzufügen aller benötigter Komponenten
- „Destruktive“ Maßschneiderung
  - Ausgangspunkt: Vollständiges System
  - Entfernen aller **nicht** benötigten Komponenten

Betriebssystem-Maßschneiderung

- Was läßt sich maßschneidern?
  - Installieren/Deinstallieren einzelner **Software-Pakete**
  - **Einzelne Dateien** innerhalb von Paketen
  - **Funktionen** innerhalb von ausführbaren Programmen/Libraries
  - **Programmcode** innerhalb von Funktionen
- Wie stelle ich Abhängigkeiten fest?
  - Verschiedene Werkzeuge (unter Unix/Linux)
    - Paketebene: Paketverwaltungssystem (Debian apt, RedHat/SuSE RPM)
    - Dateiebene: `ldd` für Shared Libraries
    - Funktionsebene: Linker
- Aber...
  - Wie stelle ich allgemein fest, ob Dateien benötigt werden?
  - Wie kann ich auf Programmcode-Ebene maßschneidern?
  - Was ist mit dynamischen Anforderungen (Plugins nachladen)?

➔**Forschungsfragen!**

Betriebssystem-Maßschneiderung

- Paketabhängigkeiten in Debian

```
$ apt-cache depends bzip2
bzip2
 Depends: libbz2-1.0
 Depends: libc6
 Suggests: bzip2-doc
 Replaces: <libbz2>
```

- Shared-Library-Abhängigkeiten: **`ldd`**

```
$ ldd /bin/bzip2
        linux-gate.so.1 =>  (0xbfffe000)
        libbz2.so.1.0 => /lib/libbz2.so.1.0 (0xb7f64000)
        libc.so.6 => /lib/tls/i686/cmov/libc.so.6 (0xb7e30000)
        /lib/ld-linux.so.2 (0xb7f80000)
```

Abhängigkeiten

- Einfache Programme haben komplexe Abhängigkeiten...

```
$ ldd /usr/bin/xcalc
        linux-gate.so.1 =>  (0xffffe000)
        libXaw.so.7 => /usr/lib/libXaw.so.7 (0xf7ec9000)
        libXt.so.6 => /usr/lib/libXt.so.6 (0xf7e78000)
        libXmu.so.6 => /usr/lib/libXmu.so.6 (0xf7e63000)
        libX11.so.6 => /usr/lib/libX11.so.6 (0xf7d74000)
        libSM.so.6 => /usr/lib/libSM.so.6 (0xf7d6c000)
        libICE.so.6 => /usr/lib/libICE.so.6 (0xf7d55000)
        libm.so.6 => /lib/i686/cmov/libm.so.6 (0xf7d2f000)
        libc.so.6 => /lib/i686/cmov/libc.so.6 (0xf7bd3000)
        libdl.so.2 => /lib/i686/cmov/libdl.so.2 (0xf7bcf000)
        libXext.so.6 => /usr/lib/libXext.so.6 (0xf7bc1000)
        libXpm.so.4 => /usr/lib/libXpm.so.4 (0xf7bb1000)
        libxcb-xlib.so.0 => /usr/lib/libxcb-xlib.so.0 (0xf7baf000)
        libxcb.so.1 => /usr/lib/libxcb.so.1 (0xf7b96000)
        /lib/ld-linux.so.2 (0xf7f4a000)
        libXau.so.6 => /usr/lib/libXau.so.6 (0xf7b93000)
        libXdmcp.so.6 => /usr/lib/libXdmcp.so.6 (0xf7b8e000)
```

Maßschneiderung

- Was läßt sich erreichen?
  - Reduktion der Größe einer Appliance
- „Normale“ Linux-Distributionen
  - Mehrere Gigabytes (SuSE, Ubuntu, Debian, ...)
  - ...von denen viele ***nie*** benötigt werden
- Spezial-Linux-Distributionen
  - Einige (dutzend) Megabytes
  - z.B. „Damn Small Linux“: ca. 50 MB mit X11 und > 20 Applikationen
- Machbar
  - z.B. Linux-Kernel + X11-Server in 1 MB RAM [5]
- Herausforderung
  - **Automatische** Maßschneiderung von Systemen
  - Maßschneiderung von **proprietären** Systemen (z.B. Windows XP)

Agenda

- Einführung
- Vision
- Hintergrundinformationen
- **Durchführung**
- Seminarthemen
- Nächste Schritte

- **Fragen**

Ziele der Projektgruppe

- Entwurf und Entwicklung von Methoden zur Dienstlokalisierung in drahtlosen Netzwerken
- Entwurf von Anwendungsszenarien
  - für allgemeine sowie orts- bzw. zeitabhängige Dienste des Campus-Informationssystems
- Entwicklung der Virtualisierungsinfrastruktur
  - Einfache Verwaltungsschnittstelle
  - Zwei einfache Demonstrationsanwendungen

Winter semester

- Bereitstellung von Appliances
  - Entwicklung eines Servers
  - Erweiterung der Verwaltungsoberfläche der Clients (GUI)
  - Entwicklung komplexerer Beispielanwendungen
- Maßschneiderung von virtuellen Maschinen
- Test und Inbetriebnahme des Systems

Sommer semester

Zeitplanung detailliert

Woche	Inhalt	Arbeitsform
1-2	Einstieg in Literatur, Werkzeuge und Vorarbeiten	gemeinsames Seminar
3	Ideensammlung, Festlegung von Arbeitsaufträgen	Gruppen-/Plenarsitzungen
4-7	Entwurf und Entwicklung von Methoden zur Dienstlokalisierung in drahtlosen Netzwerken (Minimalziel 1)	Gruppenarbeit
8	Entwurf von Anwendungsszenarien für allgemeine sowie orts- bzw. zeitabhängige Dienste des Campus-Informationssystems	Gruppen-/Plenarsitzungen
9-13	Entwicklung der Virtualisierungsinfrastruktur mit einer einfachen Verwaltungsschnittstelle sowie zwei einfacher Demonstrationsanwendungen (Minimalziel 2)	Gruppenarbeit
14-15	Überprüfung der bisherigen Ergebnisse, Zwischenbericht	Gruppen-/Plenarsitzungen

Woche	Inhalt	Arbeitsform
1	Rekapitulation des Wintersemesters, ggf. Anpassung der Infrastruktur und Festlegung von Arbeitsaufträgen für das Sommersemester	Gruppen-/Plenarsitzungen
2-5	Provisioning von VMs: Entwicklung eines Servers, Erweiterung der Verwaltungsoberfläche auf Client-Seite (GUI) sowie Entwicklung komplexerer Beispielanwendungen (Minimalziel 3)	Gruppenarbeit
6	Bewertung/Korrektur des bisher entstandenen Systems, Aufteilung von Arbeitsaufträgen, Festlegung von Maßschneiderungszielen	Gruppen-/Plenarsitzungen
7-11	Maßschneiderung von virtuellen Maschinen (Minimalziel 4)	Gruppenarbeit
12-13	Test und Bewertung des Gesamtsystems	Gruppenarbeit
14-15	Abschlussbericht	Gruppen-/Plenarsitzungen

Dokumentation

- Ausarbeitung der Seminarthemen
- Kurzprotokolle in Gruppen- und Plenarsitzungen
- Entwurfsdokumente
- Zwischen- und Endbericht

... ganz schön viel Papier! ☹

Dokumentation

- Ausarbeitung der Seminarthemen
- Kurzprotokolle in Gruppen- und Plenarsitzungen
- Entwurfsdokumente
- Zwischen- und Endbericht

... ganz schön viel Papier! ☹

- Reduzierung auf das Nötigste
- kontinuierlich wachsende Berichte

Infrastruktur

- eigener Laborraum (OH-16 E06)
- Hard- und Software
  - Server mit Hardware-Virtualisierungs-Unterstützung (VT)
    - 8 Core intel Xeon mit 16 GB RAM
  - Arbeitsplätze: SunRay Thin Client
  - Notebooks und PCs mit VT
- Kooperationswerkzeuge
  - Mailingliste
  - Wiki
  - Versionsverwaltung (Subversion o.ä.)
  - Projektmanagement-Werkzeuge

Agenda

- Einführung
- Vision
- Hintergrundinformationen
- Durchführung
- **Seminarthemen**
- **Nächste Schritte**
- **Fragen**

Seminarthemen

- Virtualisierungs-**Grundlagen**
  - Virtualisierung auf Systemebene
  - Hardware-Unterstützung für Virtualisierung
- Maßschneiderung
  - Maßschneiderung auf verschiedenen Ebenen
  - Struktur und Umfang von Appliances
- Dienste in mobilen Netzen
  - Lokalisierung, Dienstefindung
- Grundlegende **Werkzeuge**
  - Virtualisierungssoftware: Xen, KVM, VMware
  - Konfigurierungswerkzeuge
- **Forschungsthemen**
  - Automatische Maßschneiderung, Anwendungsanalyse

Nächste Schritte

1. bis **Mittwoch, 3. Juni 12:00**: Online-Anmeldung!
   https://projektgruppen.cs.uni-dortmund.de/
2. optional: Mail bzgl. bevorzugter Mitstreiter an uns
3. eventuell Rückfragen von uns per Mail
4. Auswahlverfahren
5. ab 17.6.: Aushang der Listen
6. ab Juli: Erste Sitzung der Gruppe
   - Verteilung der Seminarthemen

Literatur

[1] Popek, Gerald J. and Goldberg, Robert P.:
„*Formal requirements for virtualizable third generation architectures*“
In: Commun. ACM Vol 17, Nr. 7, 1974, pp. 412-421

[2] A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori:
„*KVM: the Linux virtual machine monitor*“
In: OLS’07: The 2007 Ottawa Linux Symposium, pages 225–230, July 2007

[3] P. Barham, B. Dragovic, K. Fraser, St. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield:
„*Xen and the art of virtualization*“
In: SOSP ’03: Proceedings of the nineteenth ACM symposium on Operating systems principles, pages 164–177, ACM, 2003

[4] John Scott Robin and Cynthia E. Irvine:
„*Analysis of the Intel Pentium's Ability to Support a Secure Virtual Machine Monitor*“
In: Proc. 9th USENIX Security Symposium, 2000

[5] Dor Laor:
„*Virtualizing the Embedded World: Vista Over Linux in a Cell Phone?*“
http://www.linuxplanet.com/linuxplanet/reports/6490/4/

[6] Fabrice Bellard:
„*QEMU, a fast and portable dynamic translator*“
In ATEC'05: Proceedings of the USENIX Annual Technical Conference 2005

Fragen?

Kontakt

olaf.spinczyk@tu-dortmund.de	(0231) 755-6322
jochen.streicher@tu-dortmund.de	(0231) 755-6142
horst.schirmeier@tu-dortmund.de	(0231) 755-6142