MICO - eine Open-Source CORBA-Implementierung

Kay Römer
Institut für Informationssysteme
ETH Zürich

MICO - MICO IS CORBA

- Infrastruktur für verteilte Systeme
- Implementierung der CORBA-Spezifikation
- Hohe Flexibilität und Anpassbarkeit an spezielle Anwendungsumgebungen durch mikrokern-basiertes Design
- Ursprünglich als Anschauungsbeispiel für ein noch in Arbeit befindliches Verteilte-Systeme-Lehrbuch in Analogie zu A. Tanenbaums Betriebssysteme-Lehrbuch entwickelt
- Erste öffentliche Version im März 1997
- Open-Source-Projekt

Open-Source

- In den letzten Jahren Vielzahl erfolgreicher Open-Source-Projekte (Linux, Apache, ...)
- Teilweise in direkter Konkurrenz zu kommerziellen Produkten
- Synergien von vielen Benutzern und Entwicklern
- Erwartung: schnellere Entwicklung, höhere Qualität
  - Kürzere Entwicklungszyklen
  - Wünsche der Anwender fliessen schnell ein

Open-Source - Grundlagen

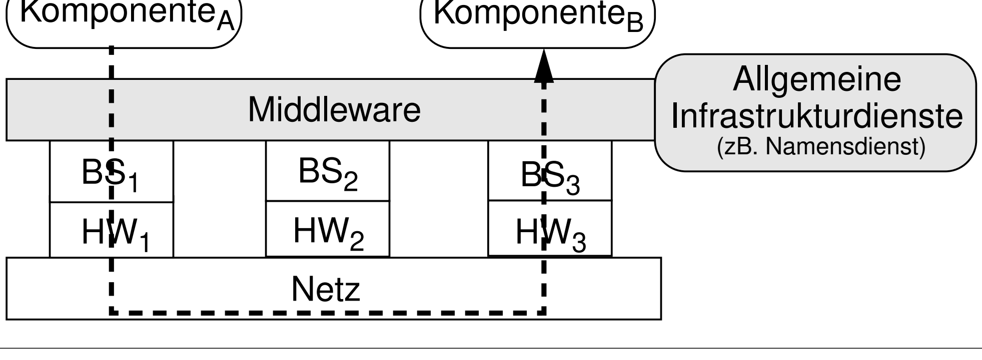
- Internet als Kommunikations-Infrastruktur
  - Mailinglisten
  - World Wide Web
  - Code Repositories
- Lizenz
  - Verfügbarkeit und freie Weitergabe des Source-Code
  - Keine Einschränkungen bzgl. der Anwender und des Anwendungszwecks
  - Jeder darf Varianten entwicklen
  - ...

Wozu MICO?

- Zunehmend entstehen grosse verteilte Systeme
- Vernetzte Systeme wie etwa das Internet bieten grosses Potential für Lösung komplexer Aufgaben
  - Beispiel: RSA Challange
- Ziel: Effiziente Entwicklung von Anwendungen für vernetzte Systeme
- Probleme dabei u.a.:
  - Heterogenität (verschiedene Hardware, Betriebssysteme, Programmiersprachen, Hersteller)
  - Komplexität durch Verteiltheit
  - Schutzbedarf

Problemlösung: Middleware

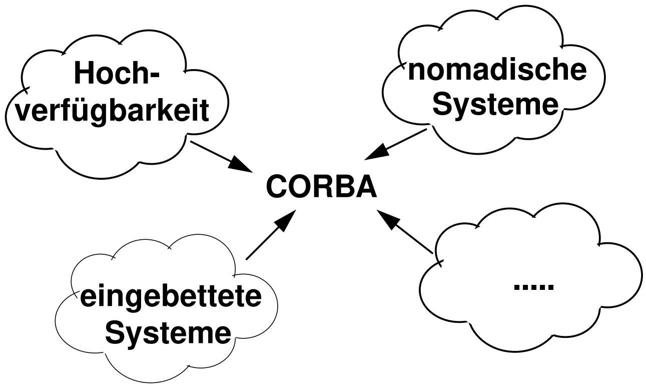
- Infrastruktur für verteilte Systeme
- Unterstützt Enwickler bei Behandlung der Probleme verteilter Systeme
- Erleichtert damit Entwicklung verteilter Anwendungen
- Logisch zwischen Betriebssystem und Anwendung



Middlewareplattform CORBA

- **C**ommon **O**bject **R**equest **B**roker **A**rchitecture
- Offene, frei verfügbare Spezifikation einer objektorientierten Middleware-Plattform
- Zentrale Komponente: Object Request Broker (ORB)
  - Verteilte Objekte als Berechnungseinheiten
  - Vermittlung von Methodenaufrufen auf Objekten
- Weitere Bestandteile:
  - System-Dienste
  - Horizontale und vertikale Anwender-Dienste

MICO - Motivation

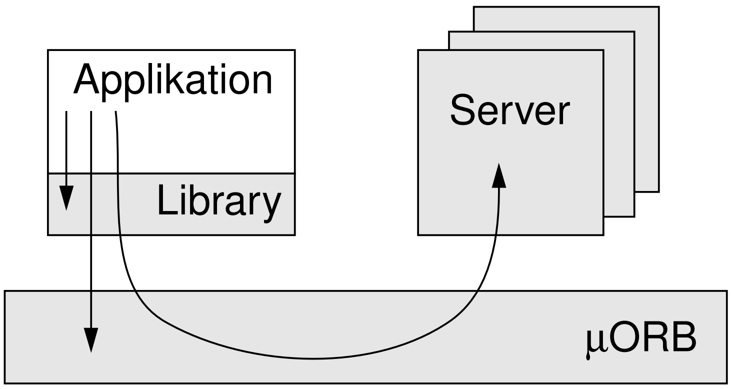


- Klassische monolithische CORBA-Implementierungen aufgrund spezieller Anforderungen für viele spezialisierte Anwendungsgebiete ungeeignet
- Ziel: generisches CORBA-System, das an verschiedenste Anwendungsbereiche angepasst werden kann

MICO - Erweiterbarkeit

- Nomadische Systeme
  - Häufige Disconnects: Kommunikation?
- Eingebettete Systeme
  - Begrenzte Ressourcen: Minimalkonfiguration
  - Homogenität: Potential für Optimierungen
  - Spezielle Hardware: Kommunikation?
- Hochverfügbarkeit
  - Redundanz: Gruppenkommunikation, Abstimmung

MICO - Mikrokern-Ansatz



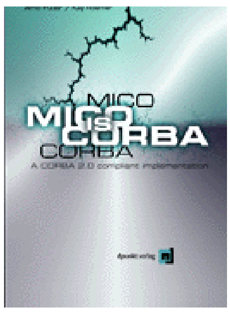
- Minimiere ORB-Funktionalität
- Verschiebe Funktionalität in
  - Server (separate Prozesse)
  - Laufzeit-Bibliotheken (im gleichen Prozess)

MICO - Funktionalitäten

- μORB-Funktionalität
  - Lokale Behandlung von Methodenaufrufen
  - Task-Scheduling
  - Bootstrapping
- Funktionalität in Servern/Libraries
  - Interprozesskommunikation
  - Prozessaktivierung
  - Scheduling-Verfahren
  - ...

MICO - Hintergrund

- 10/96: MICO als didaktisches Anschauungsbeispiel für verteilte Systeme in Analogie zu A. Tanenbaums Minix-Projekt und Betriebssysteme-Lehrbuch
- 4/97: Erste öffentliche Version: 25 kLoC
- 1/98: Version 2.0: 130 kLoC
- 1/98: Binärdistribution bei dpunkt-Verlag und MKP
- 5/99: Zertifizierung durch OpenGroup
- 10/99: Version 2.3: 300 kLoC
- 12/99: MICO/Eiffel gewinnt Wettbewerb "Eiffel Struggle ’99"



MICO als Open-Source-Projekt

- Lizensierung
  - Auf Basis der GNU General Public Licence (GPL) und Lesser General Public Licence (LPGL)
  - Ermöglicht Verwendung von MICO auch in kommerziellen Projekten
- Infrastruktur
  - Mailinglisten mit 1500 Teilnehmern
  - Beiträge von über 100 Entwicklern
  - Zentrales Repository mit Schreibzugriff für Kern-Entwickler

MICO - Projektkoordination

Diverse Regeln, die sich im Laufe des Projektes als praktisch erwiesen haben:

- Zentraler Koordinator für jede grössere Komponente, Arbeiten an dieser Komponente erfordern Absprache mit Koordinator
- Meta-Koordinator für projektglobale Entscheidungen
- Änderungen zunächst lokal testen, dann ins Repository einspielen
- Alle ein bis zwei Monate neue Version ("feature freeze", Testen, Freigeben, Ankündigen)

MICO - Aufgaben eines Koordinators

- Etwa eine Stunde pro Tag Mail/News lesen und beantworten
- Etwa eine Stunde pro Tag Patches einspielen und testen, Fehler beheben, kleinere Änderungen vornehmen
- Schubweise Arbeit an neuen Funktionen
- Alle ein bis zwei Monate etwa fünf Stunden Zeitaufwand für Freigabe einer neuen Version

MICO - Erfahrungen

E. S. Raymond: "OS fördert Zuverlässigkeit und Qualität von Software durch Ermöglichen von unabhängigem Peer-Review und schneller Evolution von Source-Code"

Unsere Erfahrungen:

- Schnelles, umfangreiches "negatives" Feedback: Fehler werden schnell gefunden und behoben
- Zögerliches positives Feedback: Einsatz (besonders in kommerziellen) Projekten nur zurückhaltend berichtet
- Maximaler Grad an paralleler Entwicklung durch Struktur des Projektes beschränkt
- Bildung von Ablegern für umfangreiche Neuentwicklungen mit späterer Reintegration

MICO - Projekte

- Ausbildung
  - Praktika Uni Frankfurt und TU Darmstadt
  - CORBA-Kurs ETH Zürich
- Forschung
  - MAQS (QoS-Unterstützung für CORBA)
  - CorbaWeb (Gateway WWW - CORBA)
  - CORBA-Interface zu SLS (Swiss Light Source)
- "Real World"-Projekte
  - KDE
  - Bodenkontrolle von Satelliten

Ausblick

- Unterstützung für Multi-Threading
- Unterstützung für CORBA 3
- Fertigstellung des Lehrbuches
- Einsatz in Forschung- und Ausbildungsprojekten im Bereich "Verteilte Systeme" an der ETH Zürich
- Details: `www.mico.org`