Vorlesung
Software aus Komponenten

3. Komponentenmodelle

Prof. Dr. Hans-Gert Gräbe
Wintersemester 2012/13

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Auslagerungsdienst (externalization service)

- Linearisierung / Delinearisierung von Objekten
  - zueinander invers (erzeugt Objektkopie)
  - keine referenzielle Integrität
    - Wertkopie von Teilobjekten
    - Referenzen **nur** über ORB Referenzmechanismus
- zum Datenexport von Objekten in Dateien und Streams
- Schnittstelle *Streamable* des auszulagernden Objekts (AO)
- wird von Strom-Objekt gerufen, das selbst Schnittstelle *Stream* implementiert
  - über *externalize_to_stream* Methode des AO
  - erzeugt daraus ein lineares Objekt (LO), das Schnittstelle *StreamIO* implementiert
- es können ganze Graphen von Objekten ausgelagert werden.

Zeitdienst (time service)

- Synchronisierung der Uhren in verteilten Systemen
- Korrelation innerhalb sinnvoller Fehlerschranken, um zeitliche Kausalitäten über Systemgrenzen hinweg zu erhalten

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Eigenschaftendienst (properties service)

- dynamisches Binden von Eigenschaften an Objekte
- keine semantische Interpretation der Eigenschaften
- Schnittstelle *PropertySet* mit Methoden *add*, *modify*, *delete*
- diese können normal, read-only (löschbar, schreibgeschützt), fixed-normal (nicht löschbar) oder fixed-read-only sein

Anfragedienst (object query service)

- Dienst zum Auffinden von Objekten nach Attributen
- ähnlich Händlerdienst, sucht aber Objektinstanzen
- Unterstützt Object Query Language (OQL-93) der Object Database Management Group) sowie SQL mit Objekterweiterungen
- Definiert Schnittstelle eigener *Sammeldienst*-Objekte
  - Semantik geordneter Mengen (add, remove, enumerate)
  - spezielle Schnittstelle *Iterator* zur Auswertung solcher Objekte
- Anfrage-Objekt kapselt die Anfrage, welche in zwei Schritten beantwortet wird: Vorbereitung und Abarbeitung der Anfrage

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Anfragedienst (Fortsetzung)

- Vier Objekttypen:
  - Anfrage-Objekte (query object, QO) und Sammelanfragen (querable collections, QC)
  - Anfrage-Auswerter (query evaluator, QE) wertet QO oder QC aus und erzeugt Ergebnis-Sammelobjekt
  - Anfrage-Manager (query manager, QM) erzeugt QO oder QC und schickt sie an QE zur Beantwortung
- Das Objekt, das Anfrage generiert, benutzt *Iterator*-Schnittstelle zur Auswertung der Antwort

Sammeldienst (object collections service)

- Möglichkeit zum Bilden von Sammeltypen verschiedener Topologien, z.B. Mengen (bags, sets), Schlangen (queues), Listen (lists) oder Bäume (trees), entsprechend der Smalltalk-Klassifikation
- unklar, ob das nicht lieber auf Objektebene realisiert sein sollte
  - existieren effiziente Implementierungen dieser Datentypen auf Bibliotheksebene

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Standardkomponenten (CORBAfacilities)

Standardisierung von häufig benötigten Anwendungsbestandteilen

- Komponentenrahmen zur einfachen Integration von Anbieterlösungen

Abgrenzung von Bereichen horizontal oder vertikal

- horizontal: Fokus auf generellem Anwendungsmodell
  - Standards für Nutzerschnittstellen, System- und Aufgabenverwaltung
- vertikal: Focus auf bereichsspezifischen Einsatzfeldern
  - im Rahmen von OMG SIG‘s oder Domain Task Forces

Standards zur Integration häufig benötigter Dienste als „Plugins“ in bestehende Komponentenrahmen (component frameworks)

- vereinfacht und standardisiert das Vorgehen bei der Integration von Komponenten verschiedener Anwender

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Horizontale (generale) Standardkomponenten

Fokus auf generellem Anwendungsmodell. OMG hatte hier ursprünglich Standards für folgende Rahmen im Auge

- **Benutzerschnittstelle** (user interface)
- **Informationsverwaltung** (information management)
- **Systemverwaltung** (system management)
- **Aufgabenverwaltung** (task management)

Wird heute nur noch wenig vorangetrieben und stärker auf übergreifende Dienste konzentriert wie

- Internationalisierung, mobile Agenten, Zeit- und Druckdienst-Standards
- Fokus heute nicht auf Standard-Komponenten (ready to use), sondern auf Komponenten-Standards (Architektur)

Standardisierungsbemühungen der ursprünglichen Bereiche spielen heute praktisch so gut wie keine Rolle mehr.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Vertikale Standardkomponenten

Ursprünglich lag hier der Fokus auf Basisfunktionalitäten für unterschiedliche Marktsegmente. Ergebnisse bekommen zunehmend segmentüberschreitende Bedeutung

- Auch hier Komponenten-Standards statt Standard-Komponenten

Ausgehandelt werden dieses Standards in Aktivitäten verschiedener Domain Task Forces

- Business Enterprise Integration
- Command, Control, Communications
- Finanzbereich
- Bereich Gesundheitsvorsorge
- Lebenswissenschaften
- Produktionsstrukturen
- Telekommunikation usw.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Sun und Java

- Java: Geschichte und Konzepte
- Wichtige von Java unterstützte Grundkonzepte
- Die J2EE-Architektur
- Java Komponentenmodelle
- Java Servlets / Java Server Pages (JSP)
- Enterprise Java Beans
- Ein Beispiel

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Java: Geschichte und Konzepte

große Erfolgsstory

- 1995/96 erste Anfänge (Sun Microsystems)
- Um 2000 eines der am häufigsten gebrauchten Schlagworte

Zwei Ansätze, womit Java wirklich zur Killerapplikation wurde

1. Das Sicherheitskonzept

- Applet wird doppelt geprüft (Übersetzungszeit und Ladezeit)
- strenge Sicherheitsregeln (security policies), die mit keiner anderen Programmiersprache erreicht werden
- Sicherheit auch im compilierten Code eines JIT-Compilers
- Sicherheitsaussagen durch Erzeugerzertifikate möglich
  - „signed applets“ (unter Nutzerkontrolle)

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. Die Java virtual machine

- Plattformunabhängigkeit
- großer Vorteil für Applikationen, die übers Web verteilt werden
- Vorteil vor allem im Standard
  - Java class-File Format und Java JAR-Archiv-Format
- beides nicht neu, jedoch in der Kombination und zu diesem Zeitpunkt durchschlagend

Java 2 (seit 1998)

- Fokus auf Applets aufgegeben
- Applets heute nur noch marginal
- Plattform-Editionen – Funktionalitätsbündel für verschiedene Klassen von Nutzern
  - J2SE als Plattform für Einzelanwendungen
  - J2EE als Serverplattform (seit Ende 1999)
  - J2ME für mobile und eingebettete Anwendungen

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Java 2 (Fortsetzung)

- Formalisierung der Bezeichnungen Laufzeitumgebung (JRE), Entwicklungsumgebung (JDK) und Referenzimplementierung
- Referenzimplementierung der J2SE von Sun/Oracle auf der Basis der HotSpot-JVM
- J2EE als Standard mit Implementierungen von verschiedenen (unabhängigen) Anbietern
  - (nicht laufzeitoptimierte) Referenzimplementierung von Sun/Oracle als Beispiel-Implementierung im Quellcode verfügbar
- Prüfung der Unterstützung von Standards durch Kompatibilitätstest-Reihen
- Java BluePrints als Sammlung von Design-Richtlinien und Mustern, um spezielle Technologien zu unterstützen.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Wichtige von Java unterstützte Grundkonzepte

- **Methoden** (Verhalten, behavior) und **Attribute** (Status, state)
- **Schnittstellen:** Es können Interfaces und abstrakte Klassen definiert werden, die später (mittels ‚*implements*') implementiert werden sollen
  - Mehrfachvererbung von Schnittstellen (ohne Status und Verhalten)
- **Klassen:** Implementierungen von Schnittstellen. Es können von bestehenden Klassen spezielle Unterklassen (mittels ‚*extends*') abgeleitet werden.
  - Einfachvererbung von Implementierungen
  - vermeidet das Diamant-Problem
  - unveränderbare Implementierungen (final class, method, attribute)

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

- **Pakete** und **Pakethierarchien** als Modularisierungskonzept jenseits von Klassen
  - Namensgebung und Namensräume auf dieser Basis
  - keine Unterstützung von Mehrfachversionen
  - company-name.productname Präfix als Standard
  - Namensraum-Importe
- **Sichtbarkeitsklassen** von Attributen und Methoden (default, public, protected, private)
- **Ausnahmebehandlung** (exception handling): Es besteht die Möglichkeit, über Ausnahmen (mittels ‚try-catch'-Blöcken) vom Standard-Kontrollfluss abzuweichen
- **Threads und Synchronisation:** Nebenläufige Programmabläufe lassen sich mit Threads erzeugen und synchronisieren
- **Garbage Collection:** Nicht mehr referenzierte Objekte werden automatisch auf kontrollierbare Weise (‚*finalize*') zerstört
  - Anwender hat darauf aus Sicherheitserwägungen keinen Einfluss

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

- **Objektserialisierung:** Objekte, welche die Schnittstelle *Serializable* implementieren, können in einen Datenstrom geschrieben oder aus einem solchen aufgebaut werden (z.B. Speichern in eine Datei)
- **Ereignisse (events):** werden einige Folien später genauer besprochen

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

J2EE Architektur und Javas Komponentenmodelle für Middleware-Anwendungen

- Im Zentrum steht **Familie von Komponentenmodellen**
  - Client-Schicht: Anwenderkomponenten, JavaBeans, Applets
  - Webserver-Schicht: Servlets, JSP
  - Anwendungsserver-Schicht: EJB

- Integrations-Ebenen (Basisdienste):
  - Namens- und Verzeichnis-Infrastruktur (naming and directory interface, JNDI) sowie Nachrichten-Infrastruktur (Java messaging service, JMS) bilden die Klammern zwischen den verschiedenen Schichten
  - weitere I.-E.: Transaktionskoordinierung, Sicherheitsdienste

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

J2EE Architektur

Kontrollfluss: 

Datenfluss: