



innoQ

Bundle-Bee

Eine OSGI-basierte Grid-Plattform

Phillip.Ghadir@innoq.com

Philipp.Haussleiter@innoq.com

Agenda

- Arten von Grid-Plattformen
- Positionierung von Bundle-Bee
- Das Programmier-Modell
- Java sei Dank - Grundlegende Mechanismen
- Bundle-Bee-Roadmap

Arten von Grid- Plattformen



Arten von Grid-Plattformen

Computing

- Beispiele
 - Google Index Cluster
 - Super-Computer

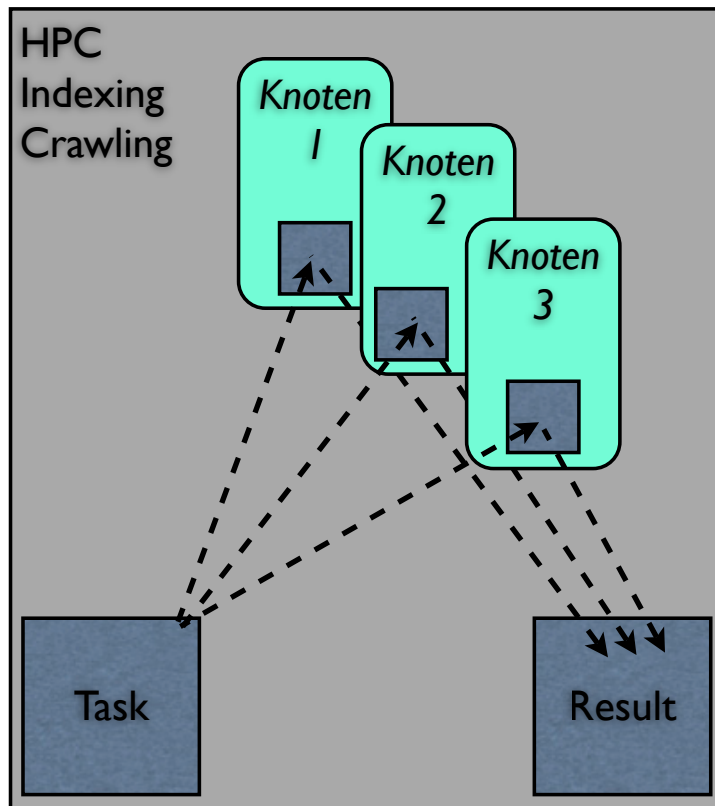
Data

- Beispiel
 - Amazon S3

On-Demand

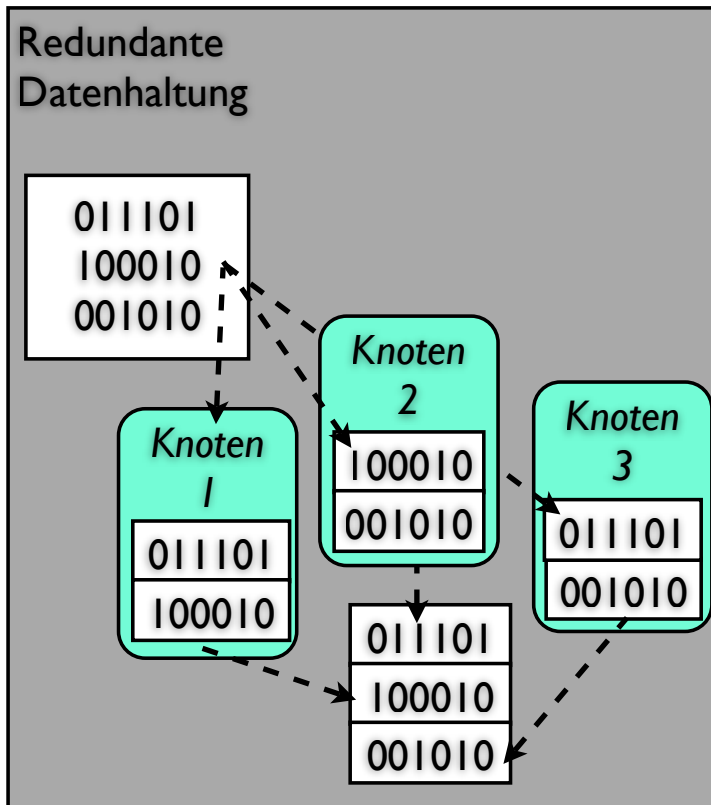
- Beispiele
 - SunGrid
 - Amazon EC2

Computing Grids



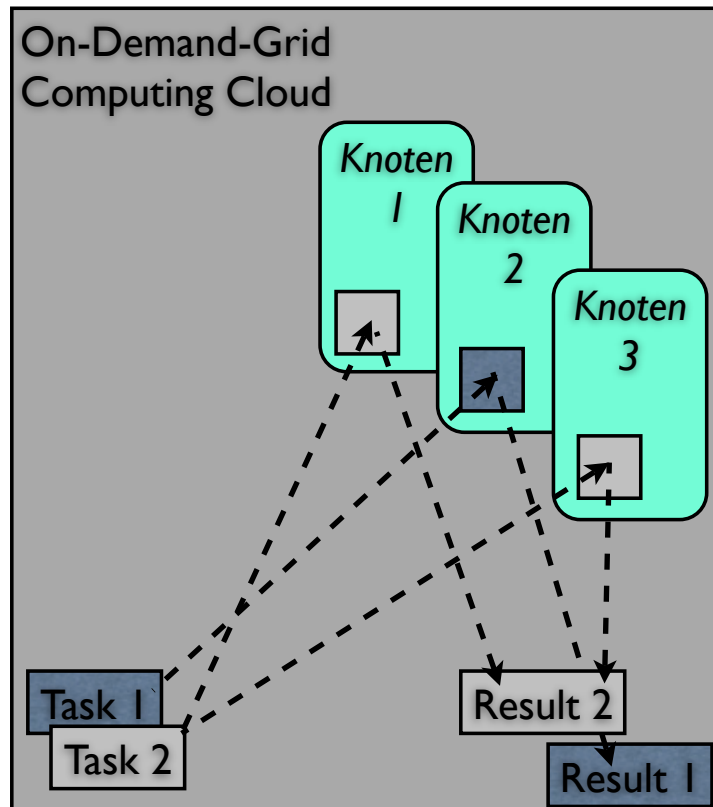
- parallele Verarbeitung von identischen Aufgaben
- gleichmäßige Nutzung von CPU/RAM
- gleichmäßige Ausnutzung aller Knoten
- Aufgabe muss „zerteilbar“ sein
- Knoten meistens hinsichtlich Durchsatz optimiert.
- Skalierend über Prozessoren

Data Grids



- Speicherung von redundanten Datenpaketen innerhalb des Grids
- Automatische Synchronisation der Knoten untereinander
- transparenter Zugriff auf die gespeicherten Daten
- Skalierend über Festspeicher
- brauchbar für große Daten (e.g. viele Datenpakete)
- wenig performant (Netzwerklatenz)

On-Demand-Grids



- Ausführung von unterschiedlichen Tasks auf verschiedenen Knoten
- Skalierend über Prozesse
- Ausfallsicherheit durch redundante Systeme
- „jeder Knoten kann alles“
- Anfragen/Ergebnisse müssen verteilt und zugeordnet werden.
- hohe, gleichmäßige Auslastung des Grids

Positionierung von Bundle-Bee



Positionierung von Bundle-Bee

- Nutzung von Verarbeitungskapazitäten in “gewöhnlichen” OSGI-Runtimes
- Verwendung des üblichen OSGI-Programmiermodells
- Nutzung von Verteilungs- und Management-Infrastruktur
- Schön-Wetter-Flug => Zero Configuration

Das Programmier- Modell



Allgemeine OSGI- Regeln

- Modularisierung über Bundles
- Bundles = JARs mit deklarierten Abhängigkeiten & Angeboten
- Keine Komposition von Paketen sondern stets per Assoziation

Spezifische Regeln für verteilte Systeme

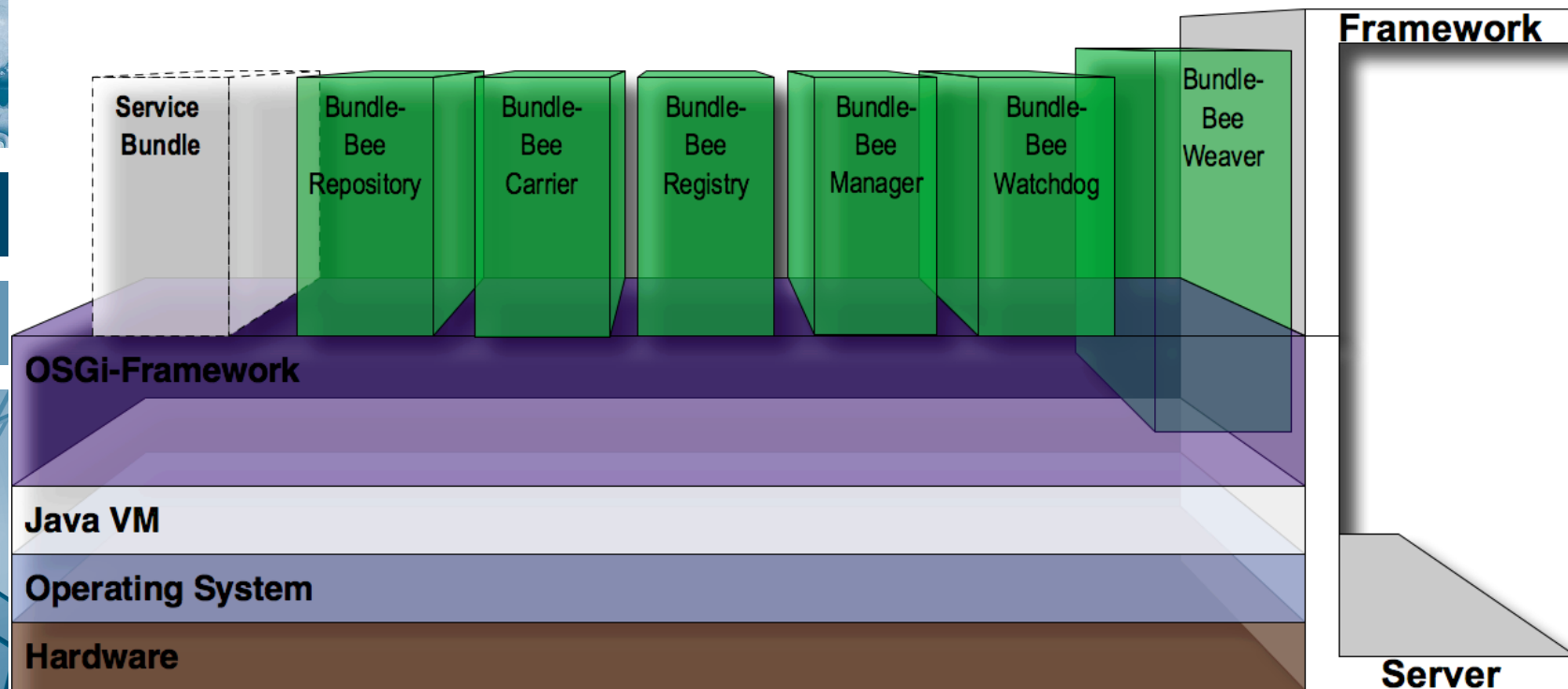
- Keine Zugriffe auf das lokale Dateisystem
- Keine Sockets
- Kein eigenes Ressourcen-Management
- Keine eigenen Threads
- Kein eigenes Class-Loading

Grundlegende Mechanismen



Bundle-Bee

Architektur-Übersicht



OSGI-Framework

com.example.testbundle

Classloading-Hook

processClass

testbundle

```
beep(){  
  [ ]  
}  
start(){  
  [ ]  
}
```

start

Bundle-Bee
Class-Loader

Packages

Javassist

Aspekte

classloading-Hook:

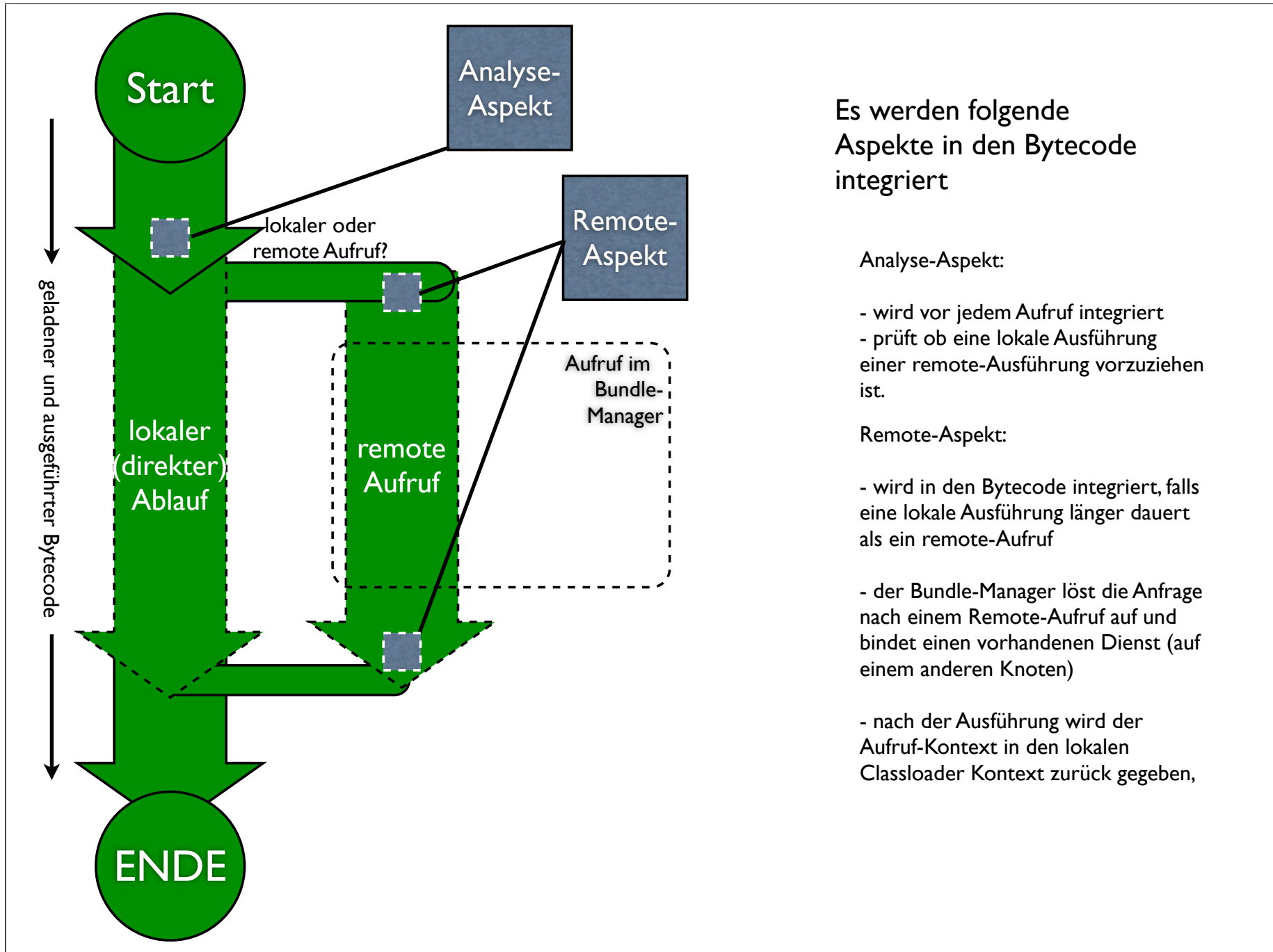
wird vom OSGI-Framework ausgeführt
bevor eine Klasse geladen wird.

process-Class:

javassist-Modifikator, um den Bytecode einer
Klasse zu verändern.

eigene Packages (welche zusätzliche Klassen
beinhalten), können mit Hilfe eines eigenen,
modifizierten Class-Loaders in die Runtime
geladen werden.

die verschiedenen Aspekte, welche in den
vorhandenen Bytecode eingewebt werden
sollen, werden ebenfalls mit der Hilfe von
javassist-Methoden hinzugefügt.



Es werden folgende Aspekte in den Bytecode integriert

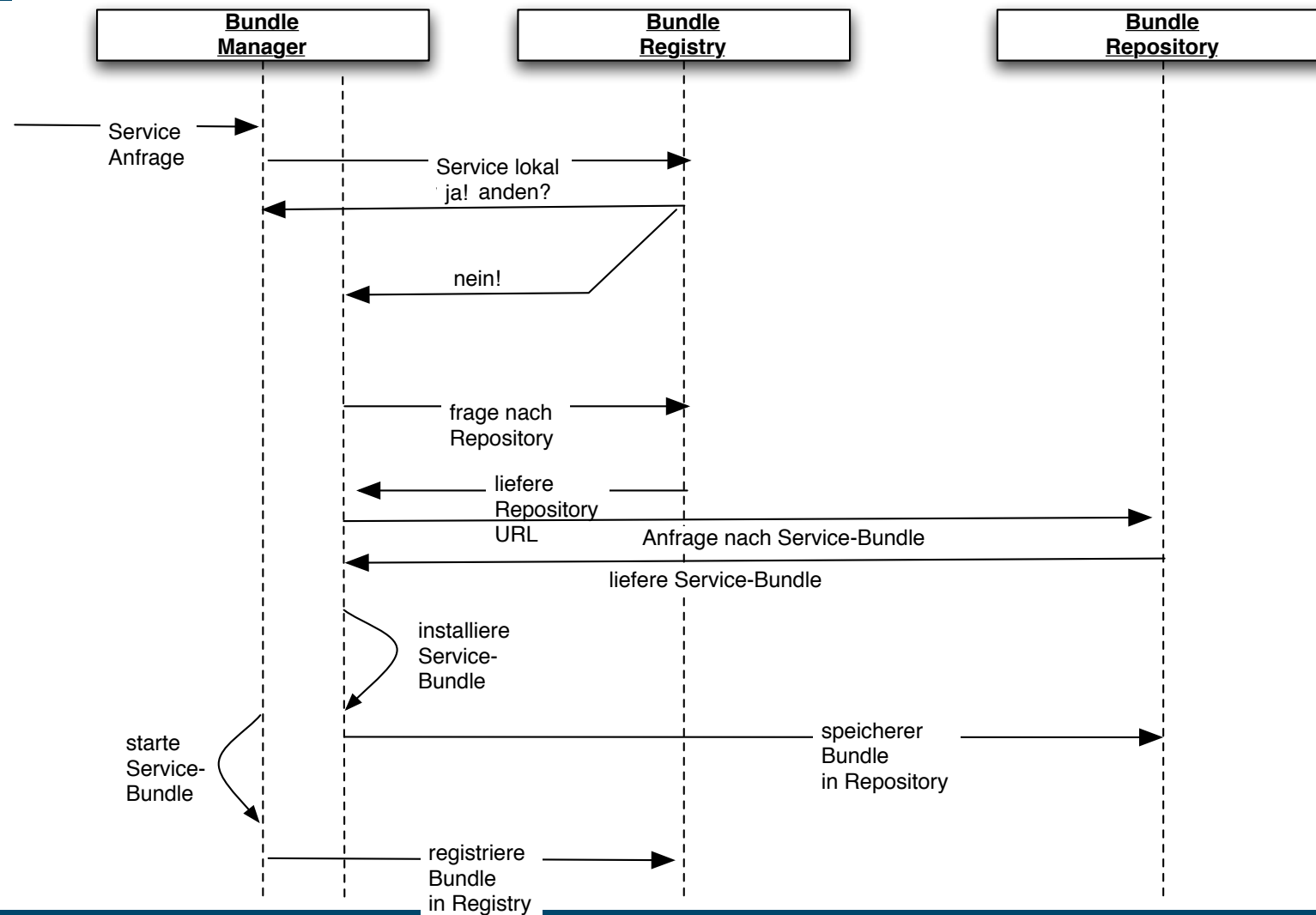
Analyse-Aspekt:

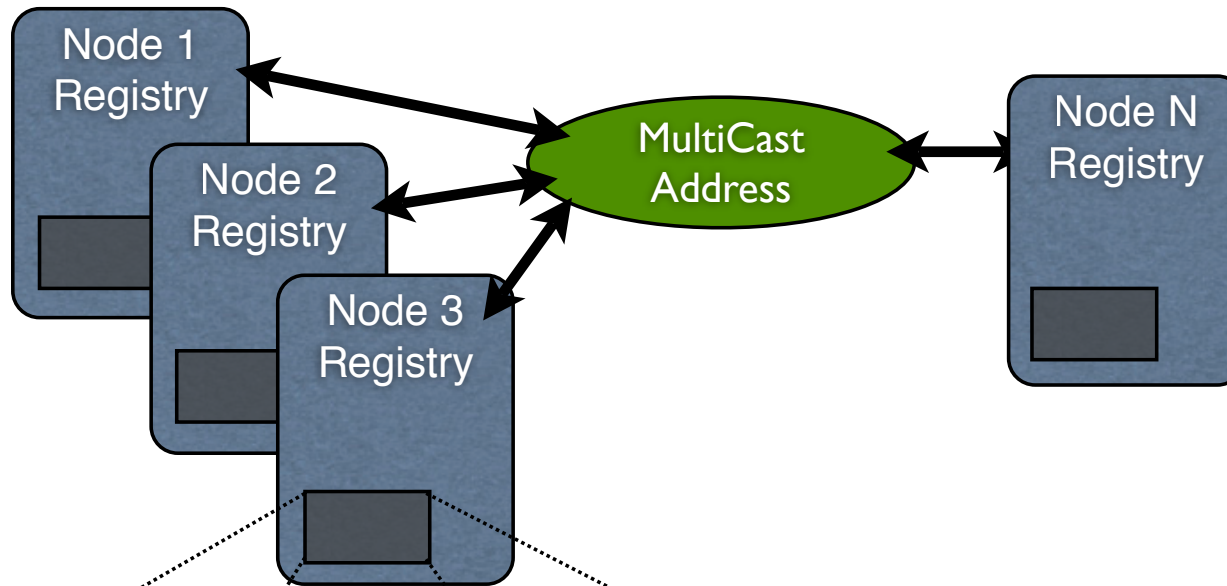
- wird vor jedem Aufruf integriert
- prüft ob eine lokale Ausführung einer remote-Ausführung vorzuziehen ist.

Remote-Aspekt:

- wird in den Bytecode integriert, falls eine lokale Ausführung länger dauert als ein remote-Aufruf
- der Bundle-Manager löst die Anfrage nach einem Remote-Aufruf auf und bindet einen vorhandenen Dienst (auf einem anderen Knoten)
- nach der Ausführung wird der Aufruf-Kontext in den lokalen Classloader Kontext zurück gegeben,

Lokal oder Remote?





distributed Hashmap

Node 1	<192.168.100.1>	Repository
Node 2	<192.168.100.2>	Manager
Node 3	<192.168.100.3>	ServiceA
Node 3	<192.168.100.3>	Repository
Node 2	<192.168.100.2>	ServiceB

- jeder Knoten enthält eine eigene lokale Registry

- jede Registry hält eine Map vor, in der Informationen aus dem Grid gespeichert sind.

- Bei einer (lokalen) Änderung werden alle anderen Grid-Nodes über Multicast-Nachrichten über die Änderung informiert.

Aktueller Stand

- Bundle_Bee_0_5_0 (Meilenstein 1) released
- Machbarkeitsnachweis:
 - Einweben der Anbindung an den Grid
 - Verwaltung lokaler und entfernter Bundles
 - “Zero”-Configuration
- Equinox-abhängig

Roadmap

- M1
 - Grundlagen für Verteilung
 - Setup “ohne Konfiguration”
- M2
 - Berechtigungskonzept
- M3
 - Monitoring
 - Auto-Rekonfiguration

Zusammenfassung

- Bundle-Bee ist ein Grid auf Basis von OSGI
- Bundle-Bee ist als Machbarkeitsstudie für Spielzwecke einsetzbar
- Bundle-Bee M2 ist für ernstere Vorhaben geeignet
- hehres Ziel:
Write OSGI-compliant - Run in the Grid



innoQ

ENDE!

Vielen Dank für Ihre Aufmerksamkeit!

innoQ

innoQ Deutschland GmbH	innoQ Schweiz GmbH
Halskestraße 17	Gewerbestrasse 11
D-40880 Ratingen	CH-6330 Cham
Phone +49 21 02 77 162-100	Phone +41 41 743 01 11
info@innoq.com · www.innoq.com	