Hochverfügbare Java Web Anwendungen

Teil 1: Lastverteilung und Tomcat-Tuning

KNF Kongress 2005
20.11.2005

Michael Behrendt <mb@michael-behrendt.net>

Übersicht

- Motivation
- Typische Architektur von Web-Anwendungen
- Optimierungen
- Messen von Leistungskennzahlen
- Erweiterungen
- Quellenangaben

Michael Behrendt
mb@michael-behrendt.net

Motivation

Warum gibt es Probleme?

Michael Behrendt
mb@michael-behrendt.net

Motivation

- Anforderungen aus Kundensicht
  - Fachlich kompetente Anwendung
  - Geschwindigkeit der Anwendung
  - eCommerce kennt keinen Ladenschluss (365/24)
- Probleme
  - Geschwindigkeit wird erreicht durch Lastverteilung
  - Verteilte Systeme erhöhen die Ausfallwahrscheinlichkeit
  - Skalierbarkeit und steigende Benutzerzahlen

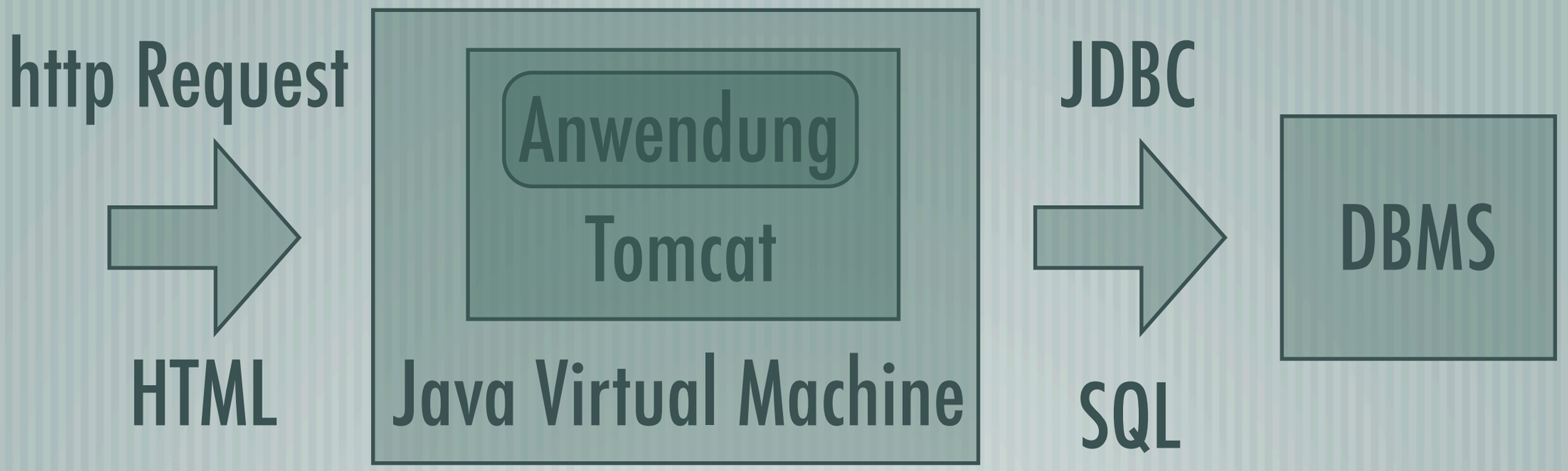
Architektur von Java

Web-Anwendungen

Wo ist was zu finden?

Michael Behrendt
mb@michael-behrendt.net

Java-Webanwendungen



Michael Behrendt
mb@michael-behrendt.net

Optimierungen

Was kann man besser machen?

Übersicht Optimierungen

- ISP
- Hardware
- Software
  - OS / JVM
  - Tanuki Service Wrapper
  - Tomcat Servlet Container

Michael Behrendt
mb@michael-behrendt.net

Optimierung > ISP

- Service Level Agreement 99,9 - 99,95%
- Peering
- 365 Tage/24h Zutritt und Personal vor Ort
- Hands-On Service, Schliess-System
- Stromversorgung
- Feuerlösch-Systeme

siehe auch [iX1]

Optimierung > Hardware

- Qualität der Hardware (Stichwort: Kondensatoren Problem)
- mehrere CPUs, RAID, SCSI-Platten, 15k UPM, Gigabit LAN
- Hardware Monitoring
  (Platten, Lüfter, Parity Speicher, Temperatur Not-Aus)
- RAID Managment im laufenden Betrieb
- Max. Arbeitsspeicher Ausbau
- Kühlung im Gehäuse

Optimierung > OS, JVM

- Betriebsystem
  - Max. adressierbarer Speicher (amd64, em64t Support)
  - Zertifiziert für Hardware (Treiber)
- Java Virtual Machine
  - Sun, IBM, BEA oder gcj JVM

Optimierung > Sun JVM

- Heap Size
  -Xmx8096M -Xms4048M
- PermGen Size
  -XX:PermSize=512M
  -XX:MaxPermSize=512M
- Garbage Collector
  -XX:+UseParallelGC

siehe auch [sun1]

Optimierung > jmx, mc4j

- Java Managment Extensions [sun2]
- “managing and monitoring” von MBeans über http
- JVM Parameter

  -Dcom.sun.management.jmxremote.port=8999
  -Dcom.sun.management.jmxremote.password.file=jmxremote.password
  -Dcom.sun.management.jmxremote.access.file=jmxremote.access
  -Dcom.sun.management.jmxremote.ssl=false
- Client mc4j [mc4j]

Optimierung > Tanuki

- Tanuki Service Wrapper [tan]
- “are you alive?” ping an JVM
- Token Filter “OutOfMemory”
- Ggfs. Neustart des JVM Prozesses
- Logfile Rotation
- Schützt nicht vor SIGSEGV innerhalb der JVM :-(

Michael Behrendt
mb@michael-behrendt.net

Optimierung > Tomcat

- web.xml von Tomcat
  - In-/Output Buffer erhöhen (4096/16384 Byte)
- server.xml
  - Anzahl der Worker Threads erhöhen

<Connector Parameter maxThreads="250" minSpareThreads="50" maxSpareThreads="75">

Optimierung > Tomcat 2

- server.xml
  - Warteschlange für TCP Verbindungen erhöhen
    <Connector Parameter acceptCount="200"
    connectionTimeout="20000">
  - DNS Lookup für Logfile Einträge ausschalten
    <Connector enableLookups="false">
  - Http Content Compression einschalten
    <Connector compression="on" compressionMinSize="2048"
    noCompressionUserAgents="gozilla, traviata"
    compressableMimeType="text/html,text/xml">

Optimierung > Tomcat 3

- Connection Pooling [dbcp]
  - JDBC Connections werden vom Container verwaltet
  - Dieser hat Pool von bereits geöffneten Verbindungen
  - Anwendung holt sich Connection per JNI lookup
  - Context Definition der Anwendung

<Context ...><Resource name="jdbc/myDB" auth="Container" type="javax.sql.DataSource"
maxActive="100" maxIdle="30" maxWait="10000" username="foo" password="bar"
driverClassName="com.mysql.jdbc.Driver" url="jdbc:mysql://localhost:3306/myDB"/></Context>

Messen von Leistungskennzahlen

Was kann das System wirklich?

Michael Behrendt
mb@michael-behrendt.net

Kennzahlen > JMeter

- Apache JMeter [jmeter]
- Flexible Simulation von Benutzer-Sessions
- Aufzeichnen von TestCases mittels http-Proxy
- Kann mit Session-IDs und Logins umgehen
- Auswertungen - Graphisch, Statistisch
- TestCases beliebig skalierbar

Michael Behrendt
mb@michael-behrendt.net

Erweiterungen

Was tun wenns brennt?

Michael Behrendt
mb@michael-behrendt.net

Erweiterungen

- Architektur der Anwendung
- Clustering auf mehreren Tomcats
- Clustering des DBMS (2. Teil)

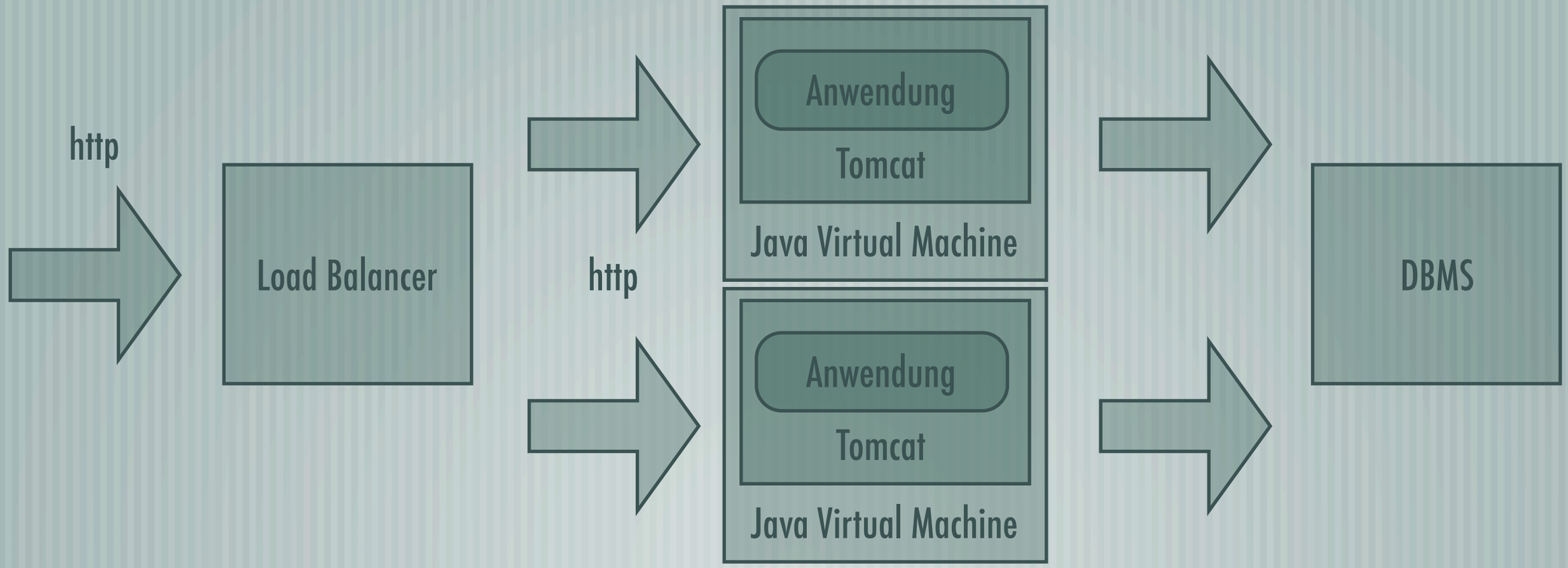
Erweiterung > Anwendung

- i.A. sehr Aufwendig diese im Nachhinein zu Ändern
- Wenn Probleme auftreten ist es i.d.R. für Änderungen zu spät
- Probleme müssen vorerst anders gelöst werden

Michael Behrendt
mb@michael-behrendt.net

Erweiterung > Web Cluster

- Idee: aus "n" werden "n+1" Tomcats
- LoadBalancer verteilt die Arbeit
- Weiterhin ein DBMS wegen Datenkonsistenz



Michael Behrendt
mb@michael-behrendt.net

Web Cluster > Sessions

- Session Replikation
  - Session kommt zum User/Request
  - Broadcast Verfahren oder Session in DBMS abgelegt
  - Session Elemente müssen Serializable implementieren
- Sticky Sessions
  - User/Request kommt zur Session
  - Wie soll LoadBalancer die Request-Folge zuordnen?

Web Cluster >Sessions & LB

- Wie erkennt ein LoadBalancer zu welchem Server ein Request gehört?
  - Cookie in http-Protokoll eingeschleust
  - Merkmal der SessionID durch Container
    <Engine jvmRoute="tomcat1">
  - NAT auf TCP Ebene
  - SSL ID

LB > Implementierungen

- Spezielle Hardware (siehe [iX2])
- Netzwerk Stack
  - ipvsadm unter Linux (siehe [vs], [teamix])
- Anwendungsebene
  - Apache httpd mit mod_rewrite
  - Tomcat Instanz im LoadBalancer Betrieb

Michael Behrendt
mb@michael-behrendt.net

LB > Apache mod_rewrite

- Apache httpd als Reverse-Proxy [bhowto]
- Zustandslos, weil Merkmal aus SessionID
- Gleichzeitiger httpd-Cache per mod_cache
- Statisches Web-Hosting zusätzlich möglich
- Server.Map Datei:

tomcat1 10.0.0.1:8080
tomcat2 10.0.0.2:8080
ALL 10.0.0.1:8080 | 10.0.0.2:8080

LB > Apache mod_rewrite

```
RewriteMap SERVERS rnd:server.map
<Location /webapp>
    RewriteEngine On
    RewriteCond "%{HTTP_COOKIE}"     "(^|;\s*)jsessionid=\w*\.(\w+)($|;)"
    RewriteRule "(.*)"   "http://${SERVERS:%2}%{REQUEST_URI}"  [P,L]
    RewriteRule "^.*;jsessionid=\w*\.(\w+)($|;)"
                "http://${SERVERS:$1}%{REQUEST_URI}"  [P,L]
    RewriteRule "(.*)"    "http://${SERVERS:ALL}%{REQUEST_URI}" [P,L]
</Location>
```

Zusammenfassung

- Optimierungen bestehen nicht nur aus Verteilten-Systemen
- Leistungsmessung als Bewertungskriterium
- Verteilte-Systeme
  - um Leistung weiter zu Skalieren
  - um Ausfall eines Knotens vorzubeugen

Quellenangaben

[iX1] iX, 08/2005, S. 123 ff, Datenfestungen

[sun1] http://java.sun.com/docs/hotspot/gc5.0/gc_tuning_5.html

[mc4j] http://mc4j.org

[tan] http://wrapper.tanukisoftware.org

[dbcp] http://jakarta.apache.org/commons/dbcp/

[jmeter] http://jakarta.apache.org/jmeter/

[vs] http://www.linuxvirtualserver.org/

[teamix] http://www.teamix.net

[bhowto] http://tomcat.apache.org/tomcat-5.0-doc/balancer-howto.html

[iX2] iX, 08/2005, S. 114 ff, Unter dem Joch

Hochverfügbare Java Web Anwendungen

Teil 2: Datenbank-Clustering mittels JDBC Middleware

KNF Kongress 2005
20.11.2005

Michael Behrendt <mb@michael-behrendt.net>

Übersicht

- Motivation
- Die Projekte c-jdbc / sequoria
- Szenarien
  - Normalbetrieb
  - Störungen
- Redundante Controller
- Quellenangabe

Michael Behrendt
mb@michael-behrendt.net

Motivation

Warum gibt es Probleme?

Motivation

- DBMS bleibt als kritische Stelle der Web-Anwendung bestehen
- Probleme
  - Skalierbarkeit der Anzahl von DBMS Abfragen
  - Ausfallsicherheit / Single Point of Failure

Replikation hilft nicht ...

- Postgresql [postgres]
  - Keine generische Lösung produktiv verfügbar
  - Spezielle Lösungen außerhalb der Transaktion
- MySQL mit MyISAM Backend [mysql]
  - Replikation nicht transparent zur Anwendung
  - Zusätzlich zum SQL-Dump muss Replikations-Zeitstempel notiert werden; Dump macht Lock Table
  - Replikation erfolgt asynchron

MySQL Cluster

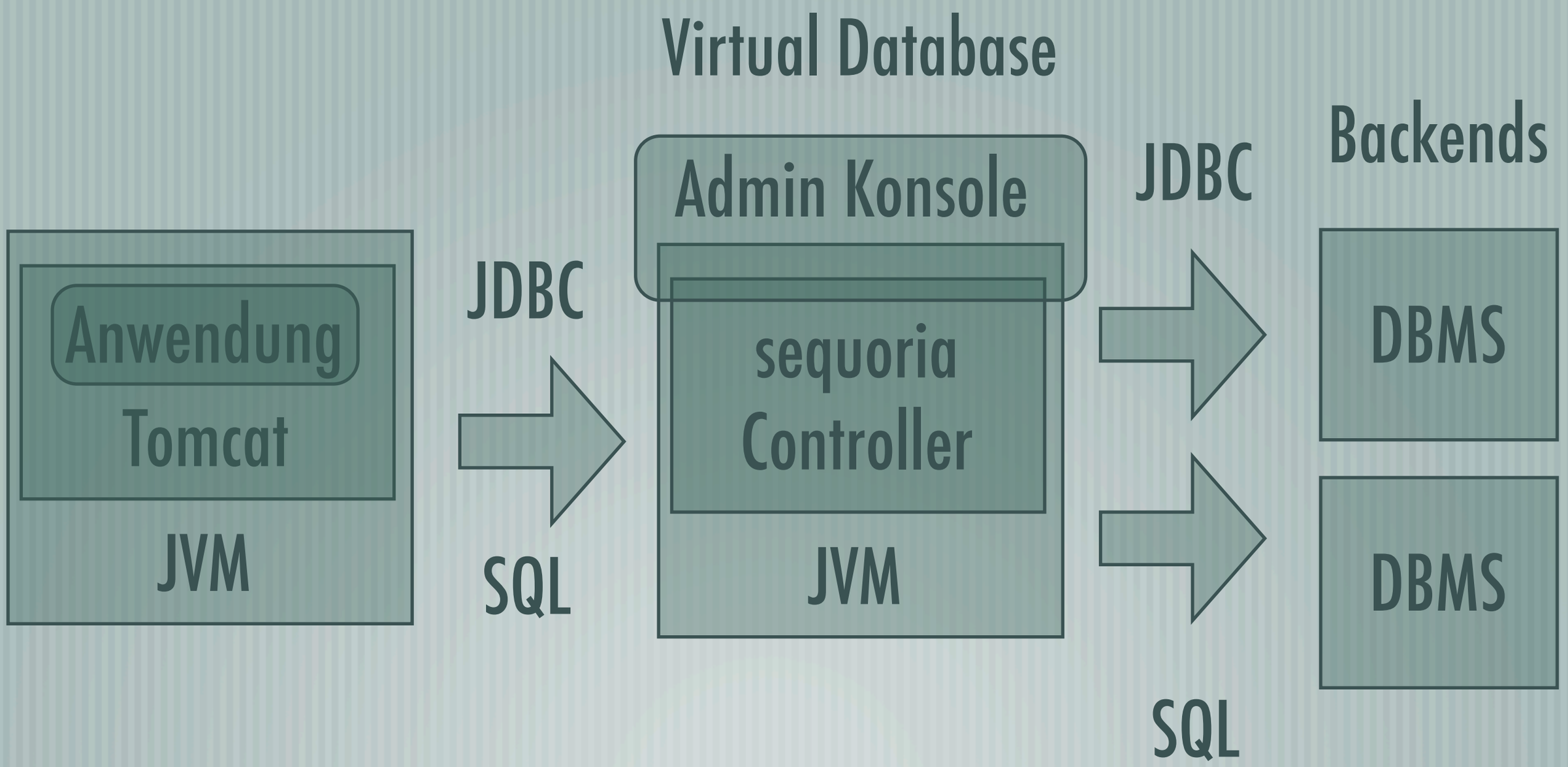
- Sämtliche Daten zur Laufzeit im Hauptspeicher
- Cluster erfordert NDB-Engine
- Performancevorteil von MyISAM nicht vorhanden
- Alle Datensätze feste Länge (VARCHAR = CHAR)
- Siehe [iX3]

Projekte

c-jdbc / sequoria

Der Ansatz ...

Projekt > Idee Middleware



Michael Behrendt
mb@michael-behrendt.net

Projekt>Vorteil Middleware

- Erfordert keine Änderungen an den Web-Applikationen oder Servlet Containern
- Single-Point-Of-Failure Lösung für alle DBMS Systeme
- Funktioniert mit jedem DBMS wegen JDBC Standard
- Controller hat Überblick über alle Transaktionen

Projekt > Unterschiede

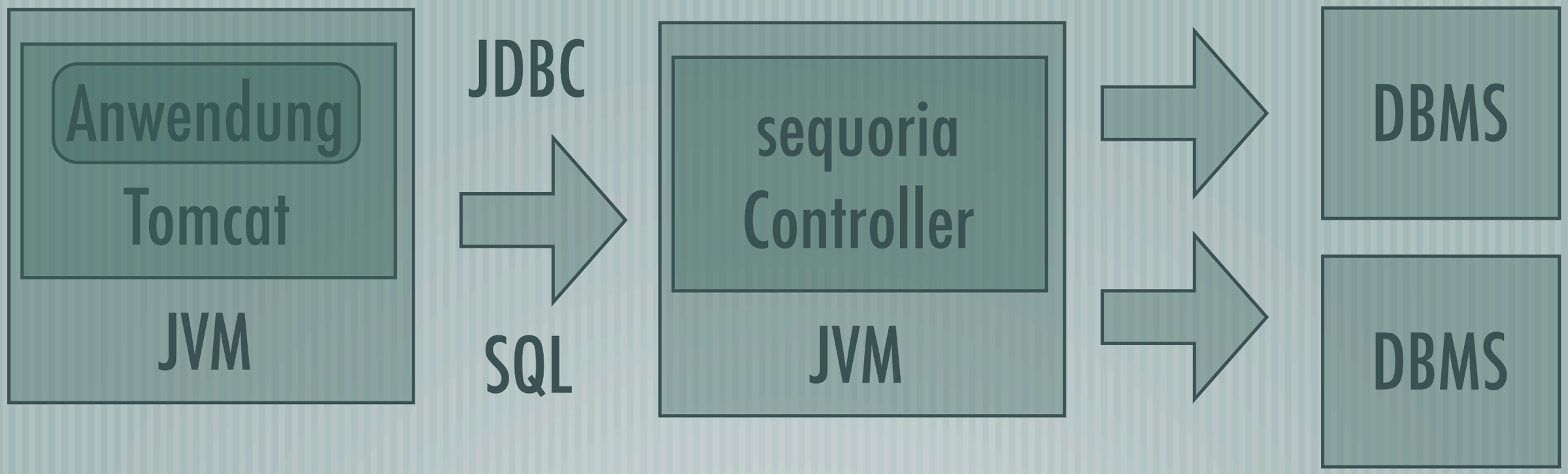
- c-jdbc [cjdbc]
  - LGPL
  - Entwicklung scheint einzuschlafen
- sequoria [sequoria] seit 10/2005
  - Apache v2 Licencse
  - Zukunft ?

Szenarien

Normalbetrieb

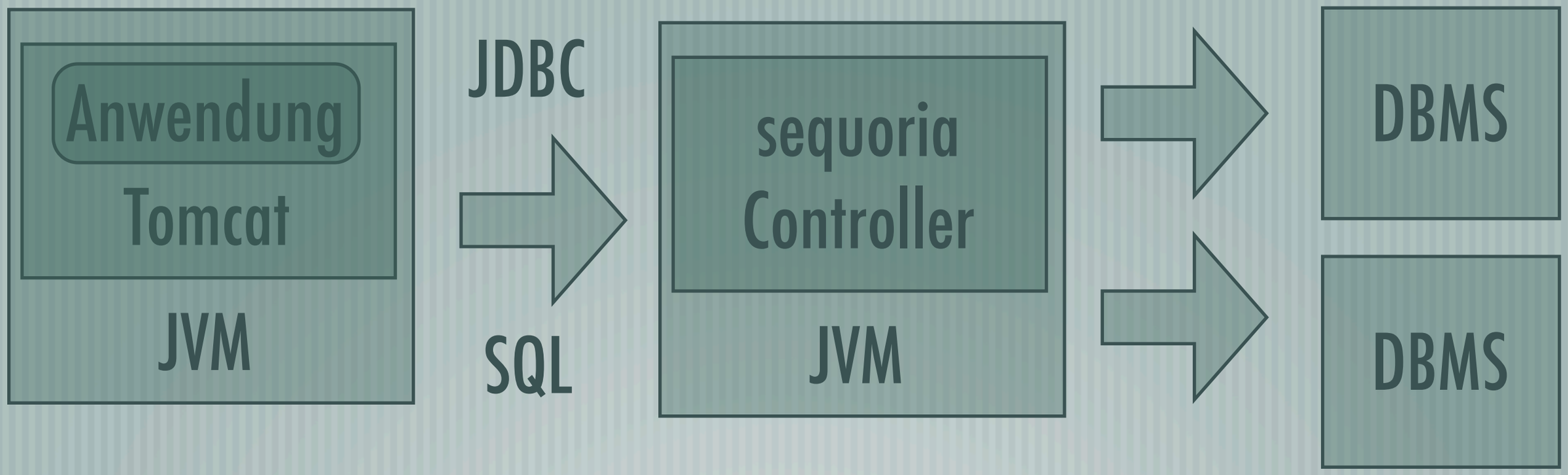
Wie funktioniert es?

Normalbetr. > SELECT ...



Vorteil:
Mehrere Requests können echt parallel bearbeitet werden
Controller kann Caching von Abfrageergebnissen durchführen

Normalbetr. > UPDATE...



Problem:
Alle Backends müssen synchronisiert geändert werden
Noch kein 2 Phasen Commit (Zukunft: Postgres 8.1?)

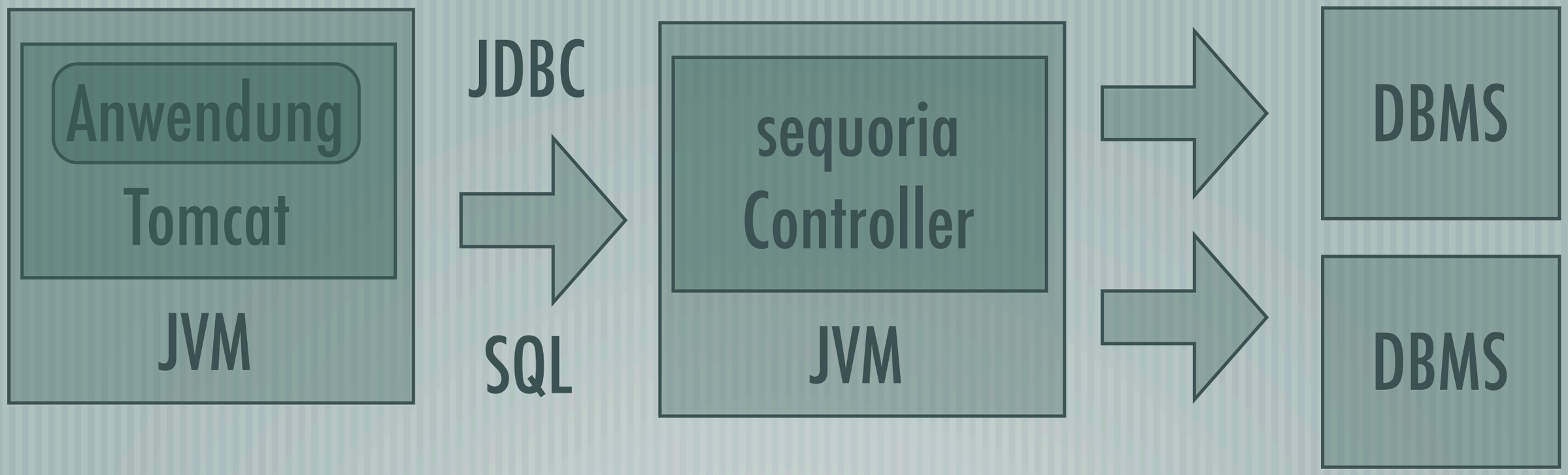
Szenarien

Störungen

Wie wird mit dem Ausnahmefall umgegangen?

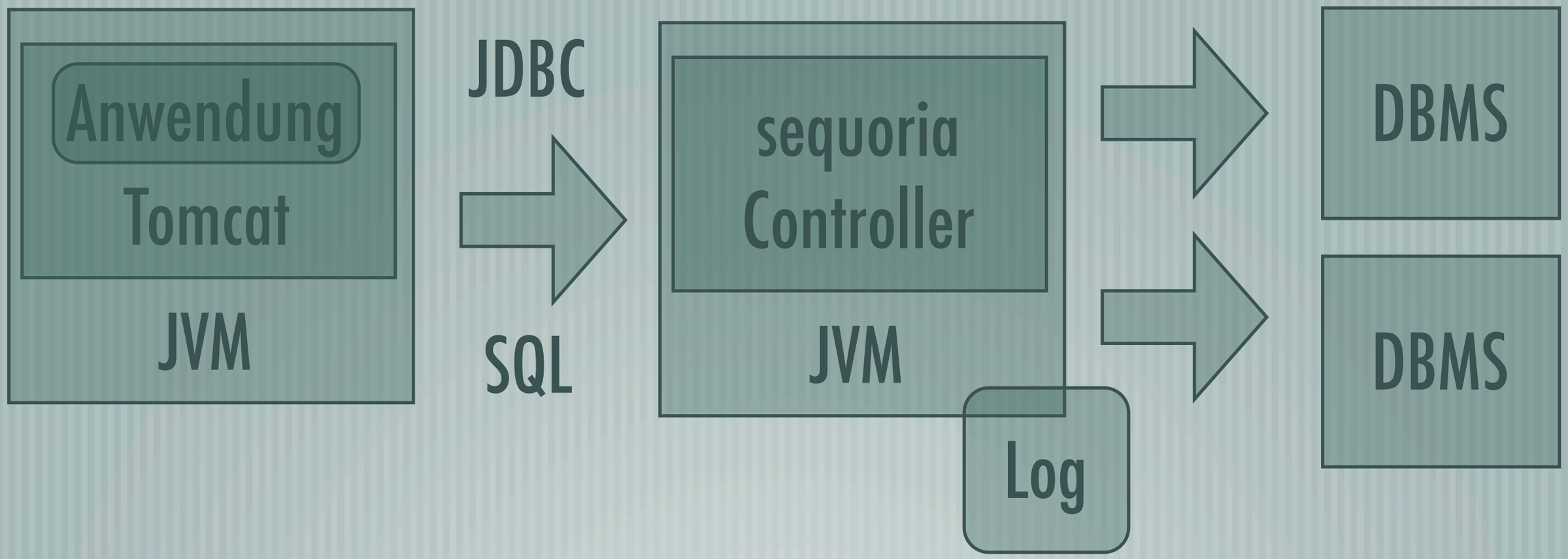
Michael Behrendt
mb@michael-behrendt.net

Ausfall eines Backends



System läuft einfach mit einem Backend weiter

Wiederanlauf Backend



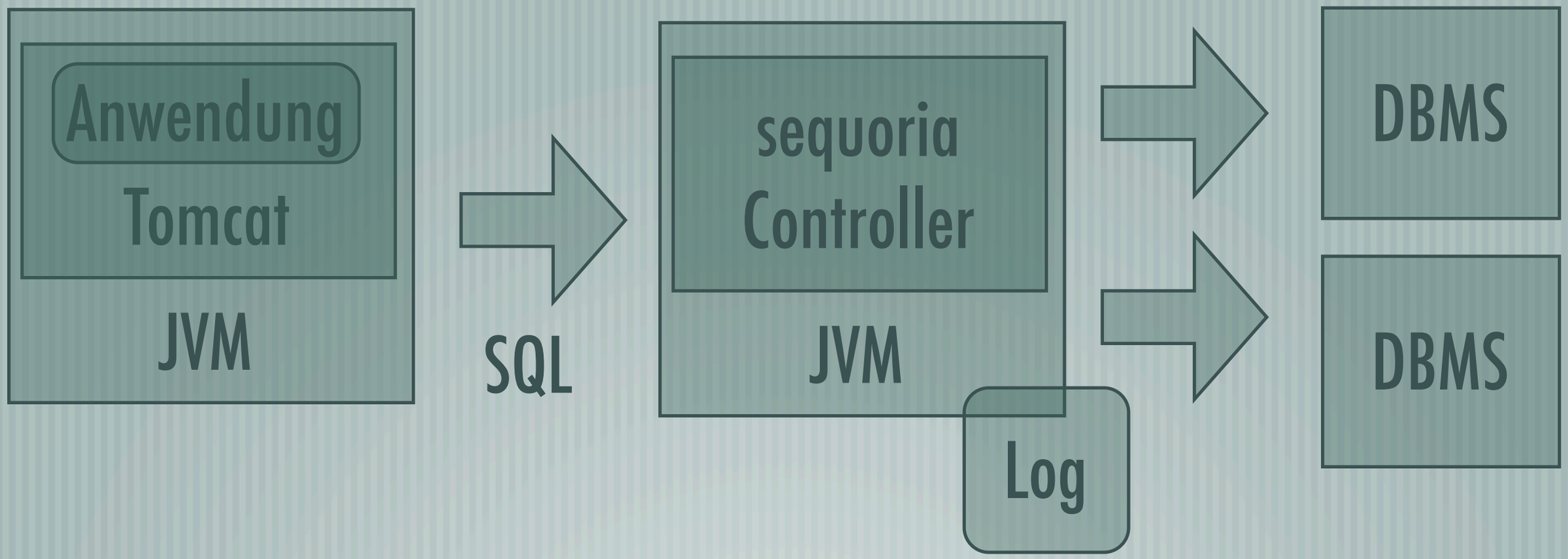
Controller kennt letzten Save-Point des Backends
Controller spielt alle Änderungen aus dem Log ein

Michael Behrendt
mb@michael-behrendt.net

Backup eines Backends

- Controller wird angewiesen ein Backup zu erstellen
- Controller
  - deaktiviert ein Backend
  - führt Backup aus
  - aktiviert ein Backend

Ausfall des Controllers



Problem: Controller ist Single-Poit-Of-Failure

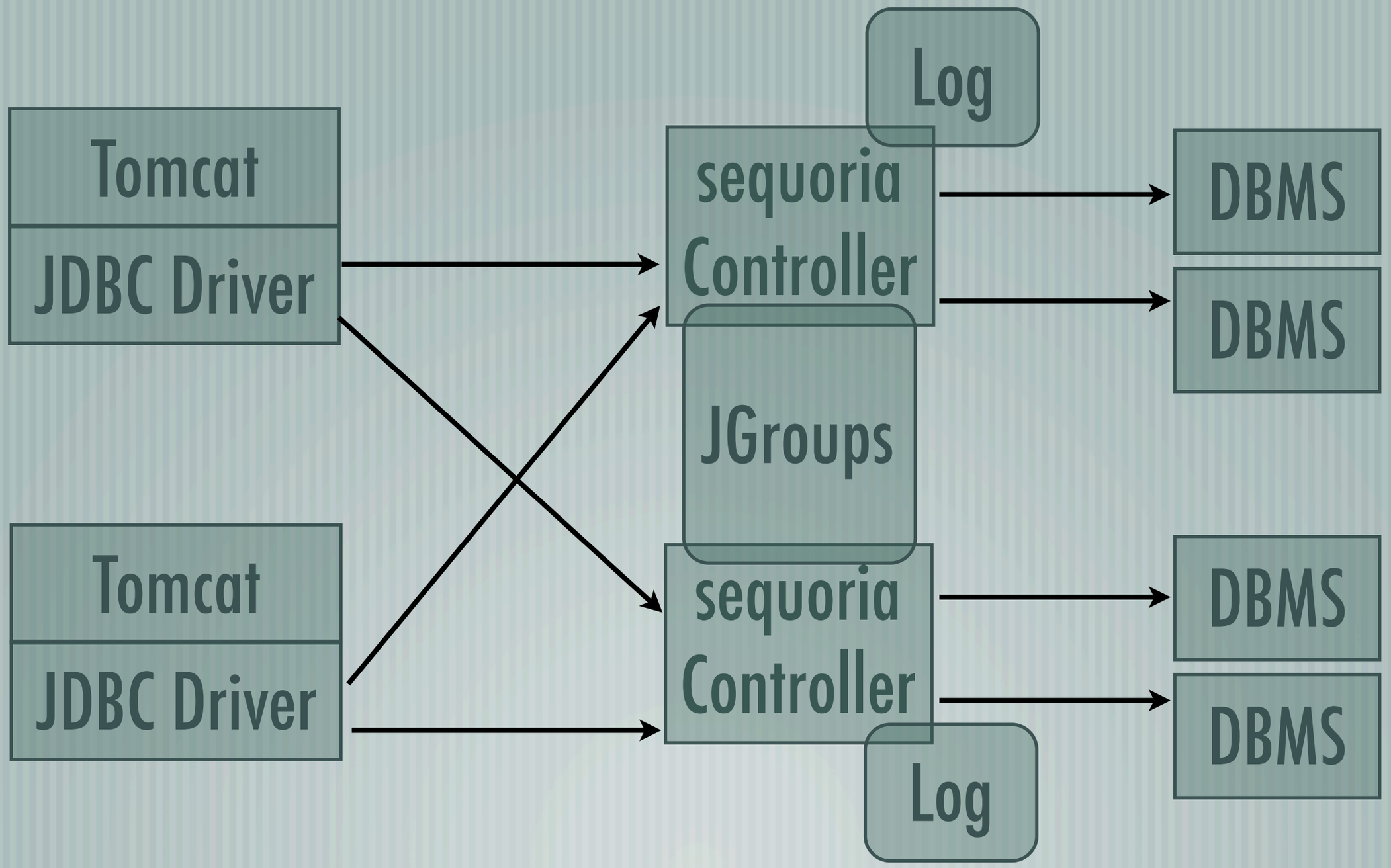
Michael Behrendt
mb@michael-behrendt.net

Redundante Controller

Wie wird der letzte Single-Point-Of-Failure vermieden?

Michael Behrendt
mb@michael-behrendt.net

Redundante Controller



Michael Behrendt
mb@michael-behrendt.net

Redundante Controller

- Im Tomcat wird sequoria JDBC Driver geladen
- Dieser kann Fallback auf anderen Controller ausführen
- Nativer JDBC Driver erst im sequoria Controller verwendet
- Controller synchronisieren sich über JGROUPS

JGROUPS

- "toolkit for reliable multicast communication" [jgroups]
- Erweitert IP Multicast um
  - Zuverlässigkeit
  - Gruppen (-verwaltung)
- Echtes IP Multicast Routing erfordert ggfs. zus. Aufwand
- IP Multicast an Ethernet Switch sehr einfach
  route add -net 224.0.0.0 -interface eth0

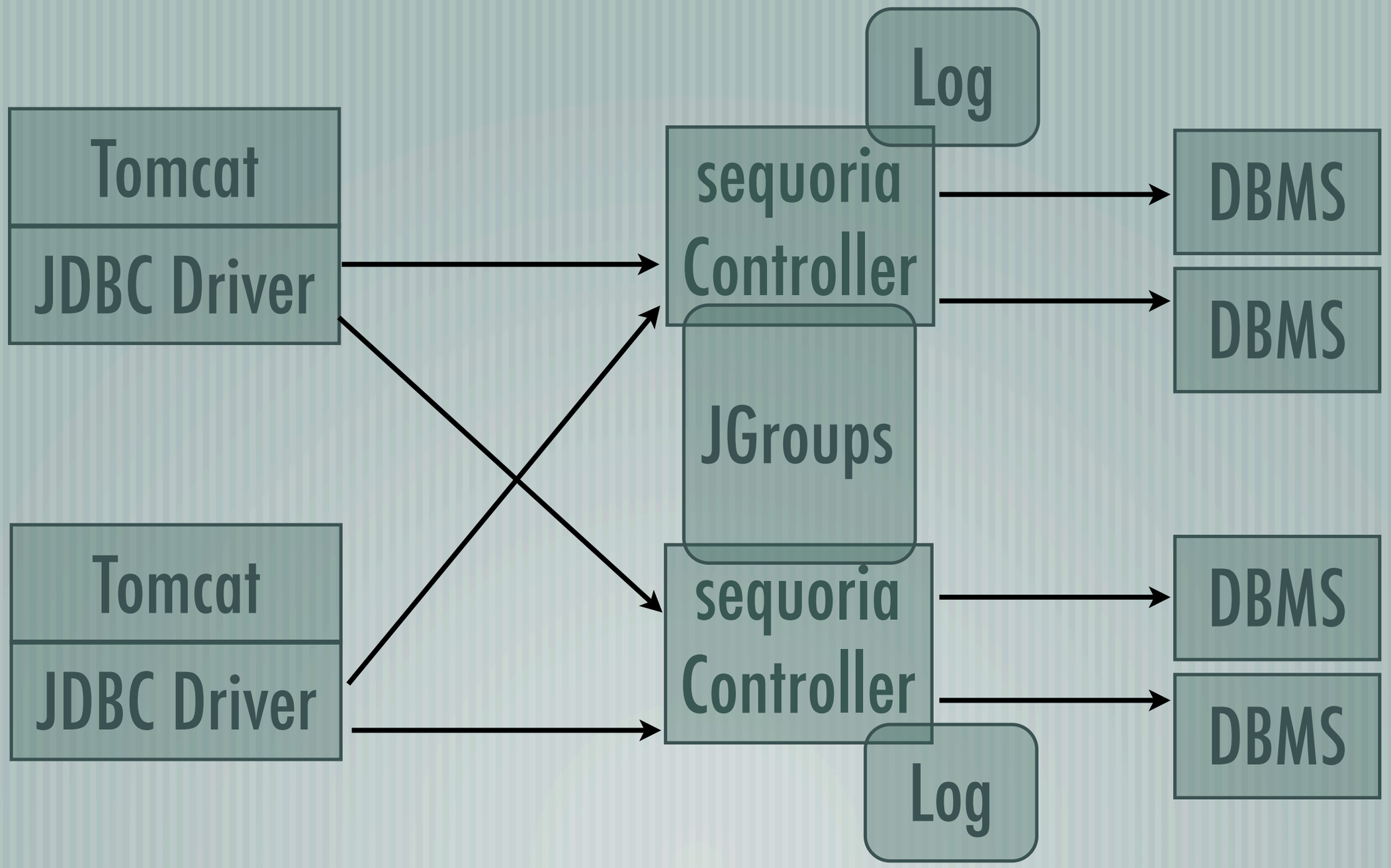
Szenarien

Störungen Teil 2

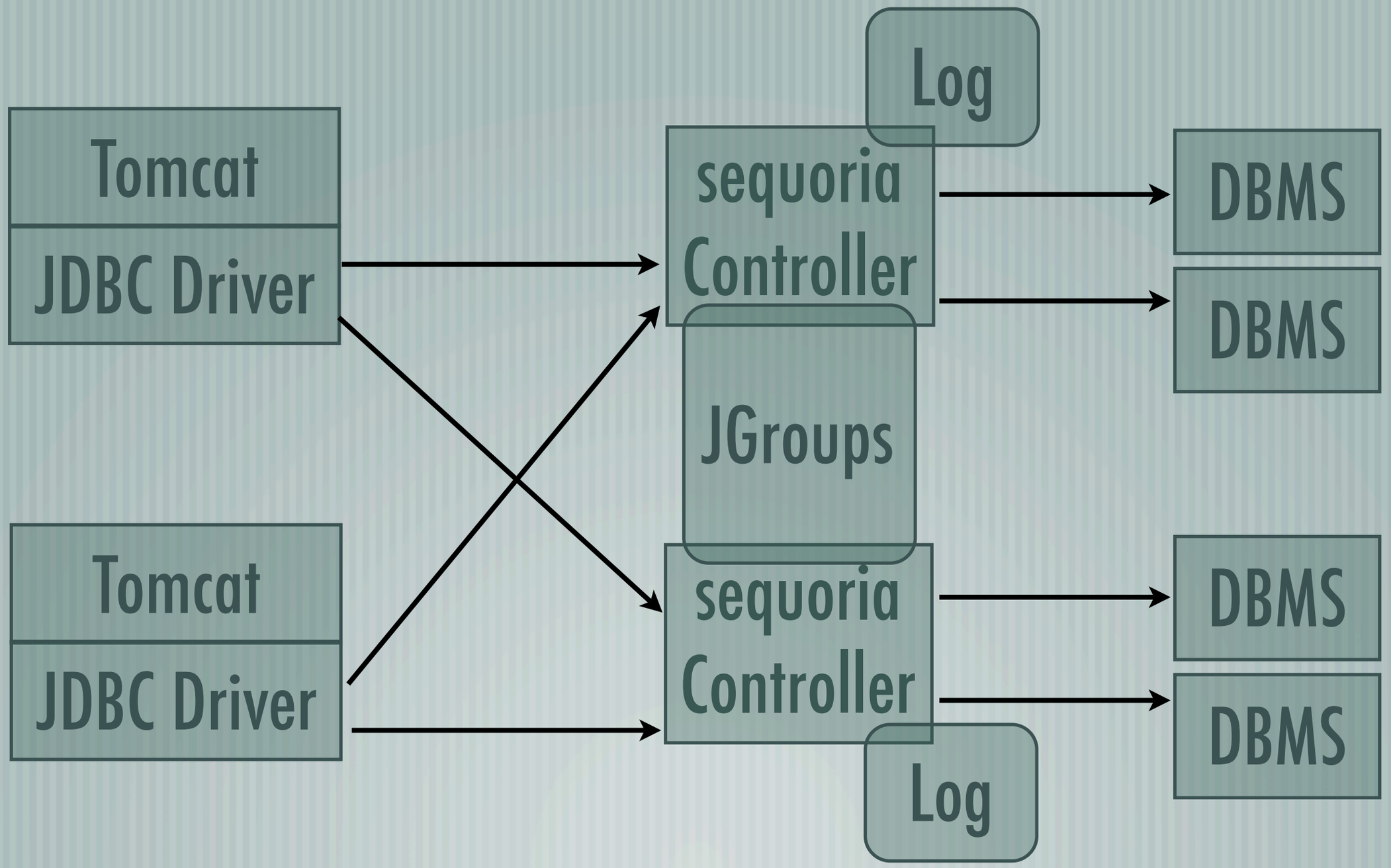
Wie wird mit dem Ausnahmefall umgegangen?

Michael Behrendt
mb@michael-behrendt.net

Ausfall eines Controllers



Wiederanlauf Controller



Michael Behrendt
mb@michael-behrendt.net

Zusammenfassung

- Es ist möglich Java-Web-Anwendungen ohne Single-Point-Of-Failure zu betreiben
- Dies nur unter Einsatz von Open Source Software
- Einfache Optimierungen mit wenig Aufwand möglich
- In der höchsten Ausbaustufe Administration relativ komplex

Michael Behrendt
mb@michael-behrendt.net

Quellenangaben

- [iX3] iX, 11/2005, S. 151 ff, Gemeinsam speichern
- [postgres] http://www.postgresql.org
- [mysql] http://www.mysql.com
- [cjdbc] http://c-jdbc.objectweb.org
- [sequoia] http://sequoia.continuent.org
- [jgroups] http://www.jgroups.org

Michael Behrendt
mb@michael-behrendt.net