

# C-JDBC: a High Performance Database Clustering Middleware

**Nicolas Modrzyk**

Nicolas.Modrzyk@inrialpes.fr



# Outline - Motivations

**Motivations**

**Use-Cases**

**C-JDBC concepts**

**Performance**

**Monitoring**

**Community**

**Conclusion**



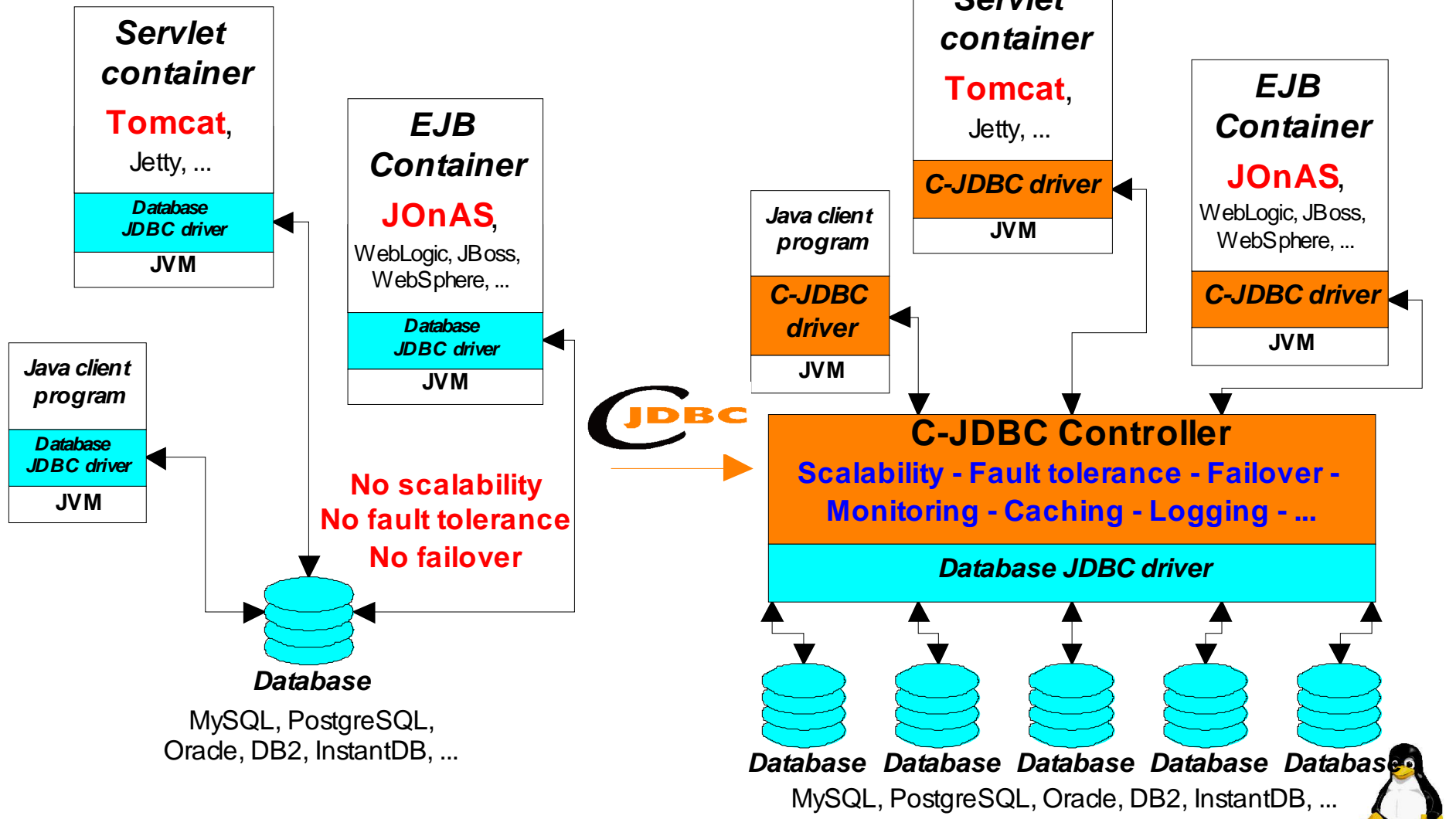
## J2EE performance scalability bounded by database performance

### Database tier must be

- scalable
- fault tolerant (high availability + failover)
- without modifying the client application
- using open source databases
- on commodity hardware



# What is



# Redundant Array of Inexpensive Databases

## RAIDb controller

- gives the view of a single database to the client
- balance the load on the database backends

## RAIDb levels

- RAIDb-0: full partitioning
- RAIDb-1: full mirroring
- RAIDb-2: partial replication
- composition possible



# Middleware implementing RAIDb

## Two components

- generic JDBC 2.0 driver (C-JDBC driver)
- C-JDBC Controller

## C-JDBC Controller provides

- performance scalability
- high availability
- failover
- caching, logging, monitoring, ...

## Supports heterogeneous databases



# Outline - Use-Cases

**Motivations**

**Use-Cases**

**C-JDBC concepts**

**Performance**

**Monitoring**

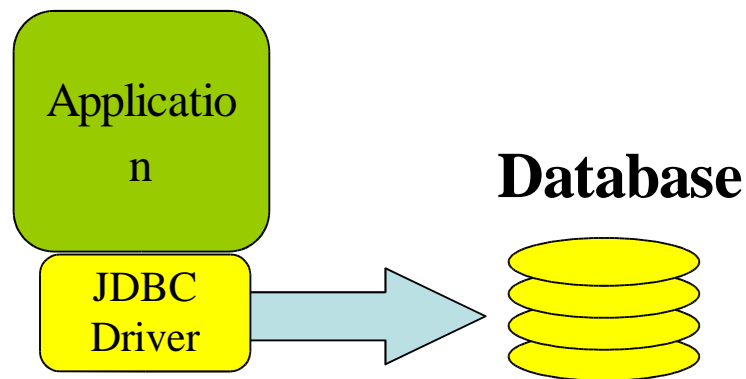
**Community**

**Conclusion**



# What C-JDBC offers

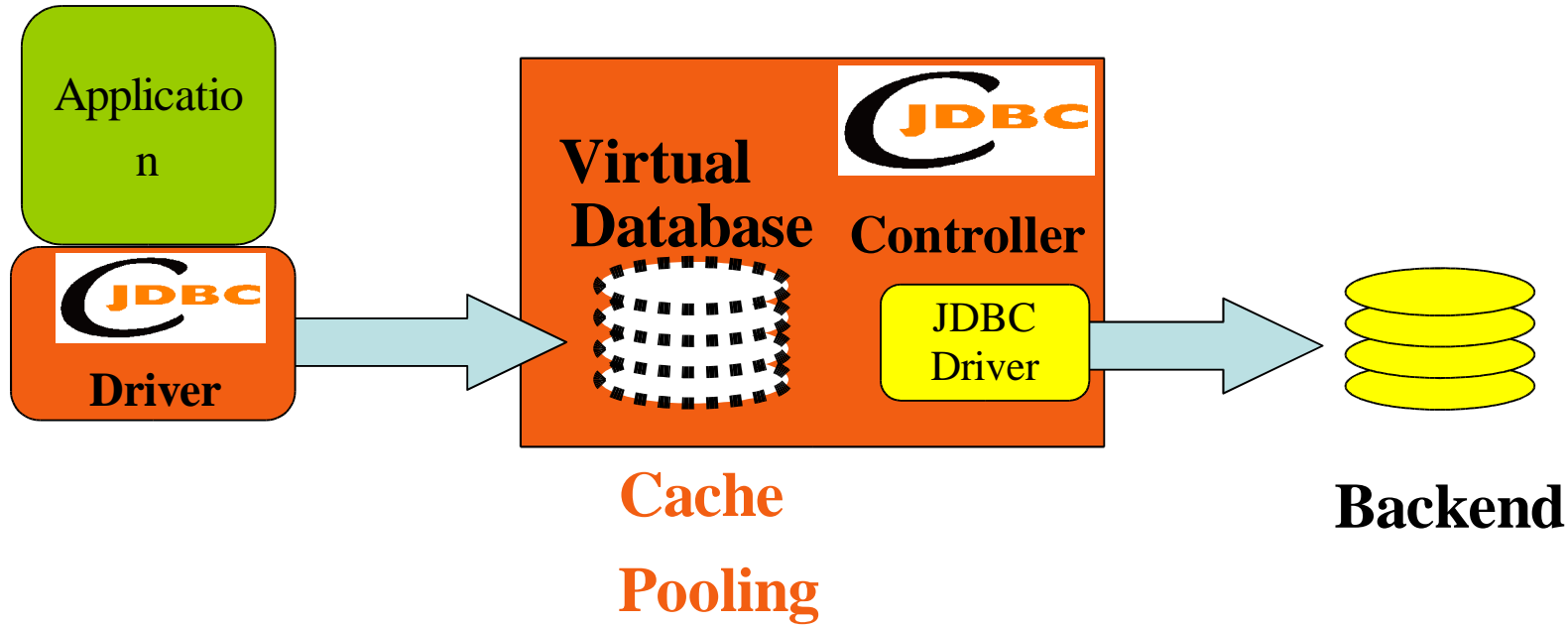
Usually, we do this:





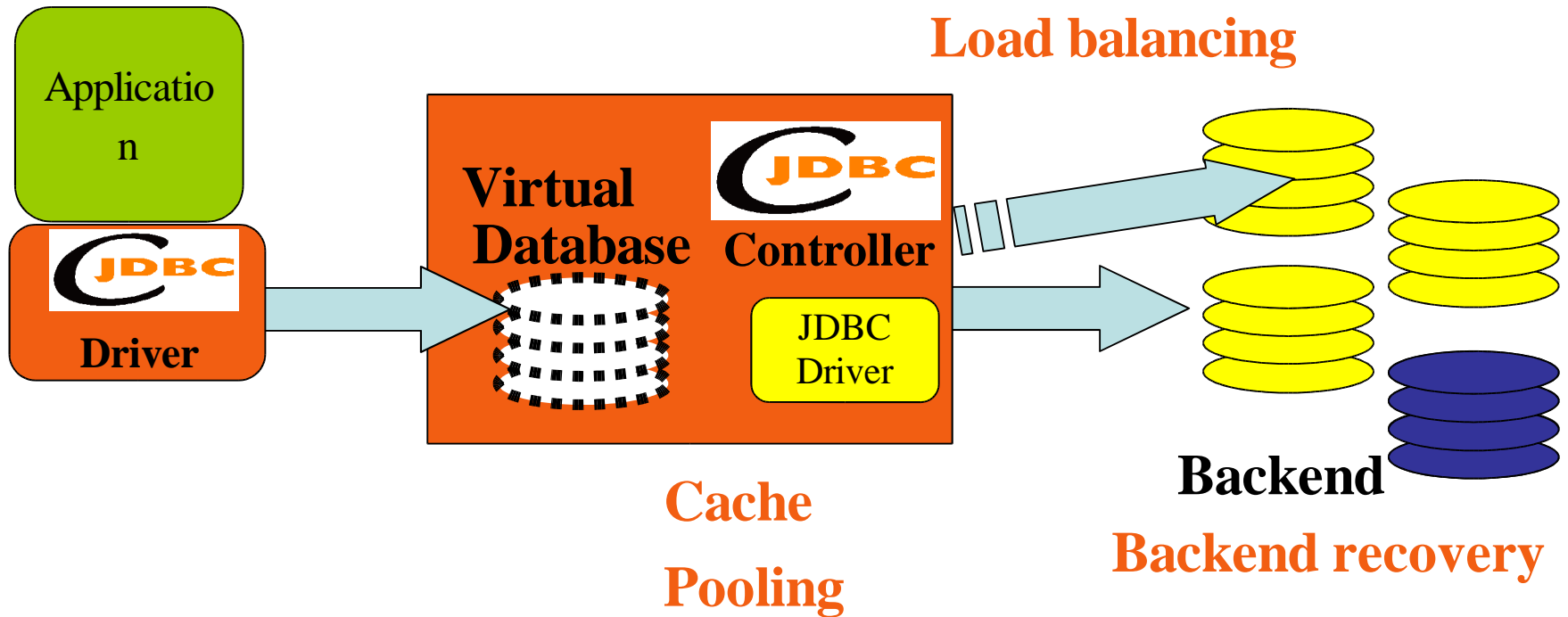
# What C-JDBC offers

## Now we have this:



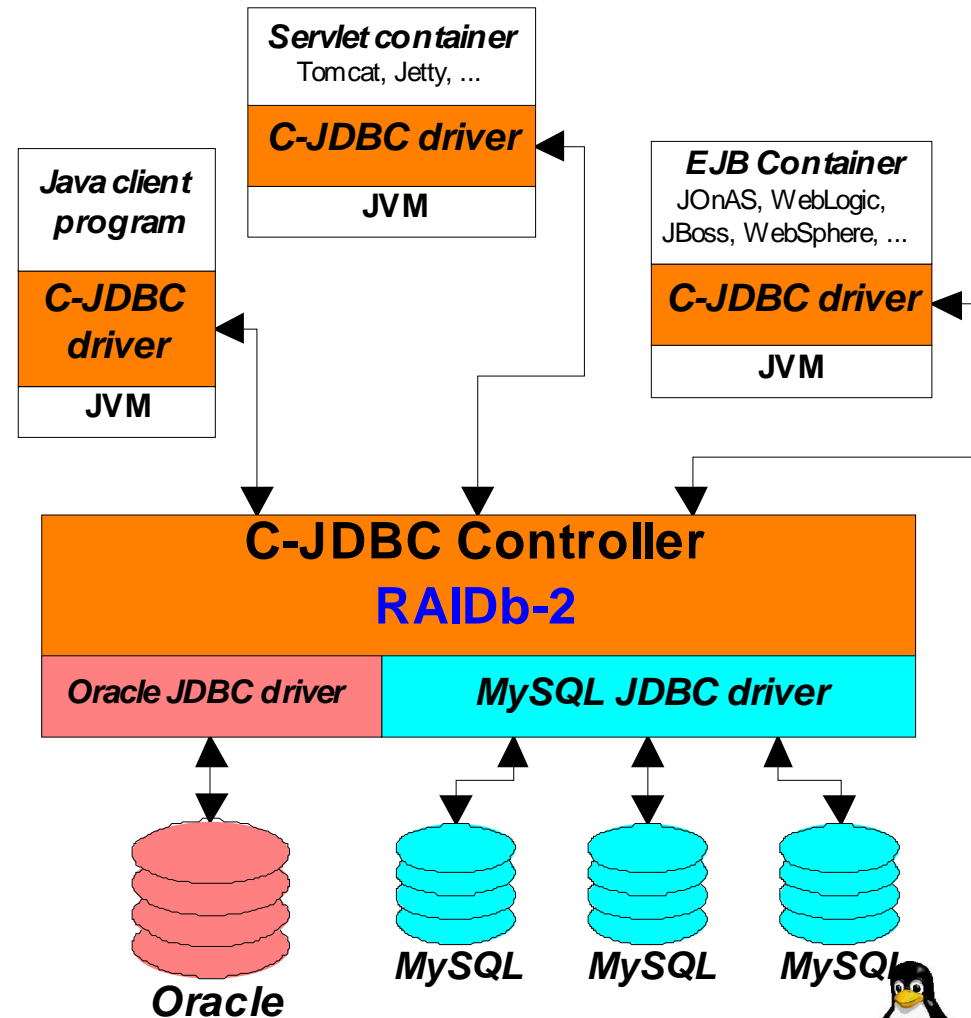
# What C-JDBC offers

And, finally we have all this:



# Heterogeneity support

application already  
written for a specific  
[commercial] database  
user defined rules  
for on-the-fly query  
rewriting to execute  
on heterogeneous  
backends



# Outline - C-JDBC concepts

**Motivations**

**Use-Cases**

**C-JDBC concepts**

**Performance**

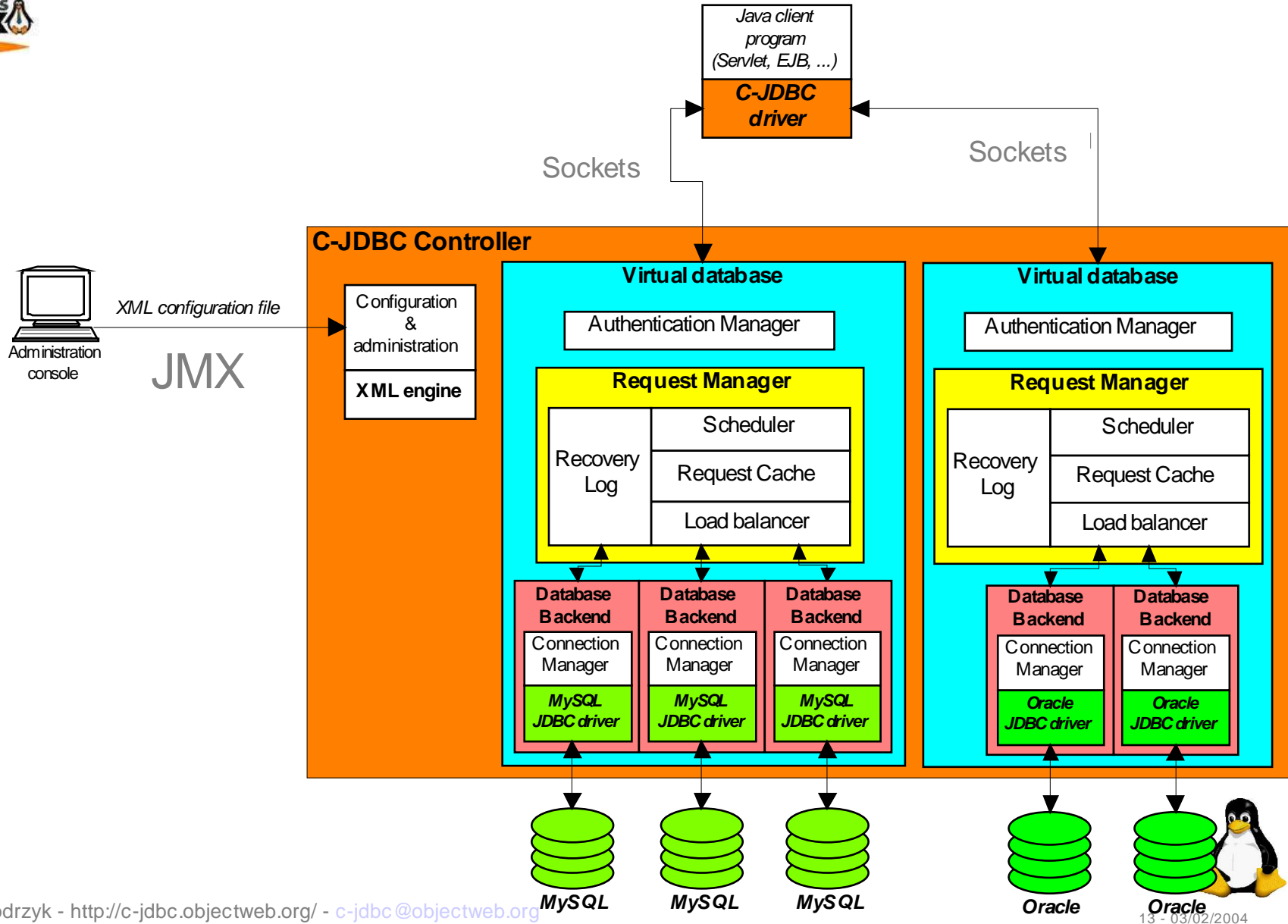
**Monitoring**

**Community**

**Conclusion**



# Controller



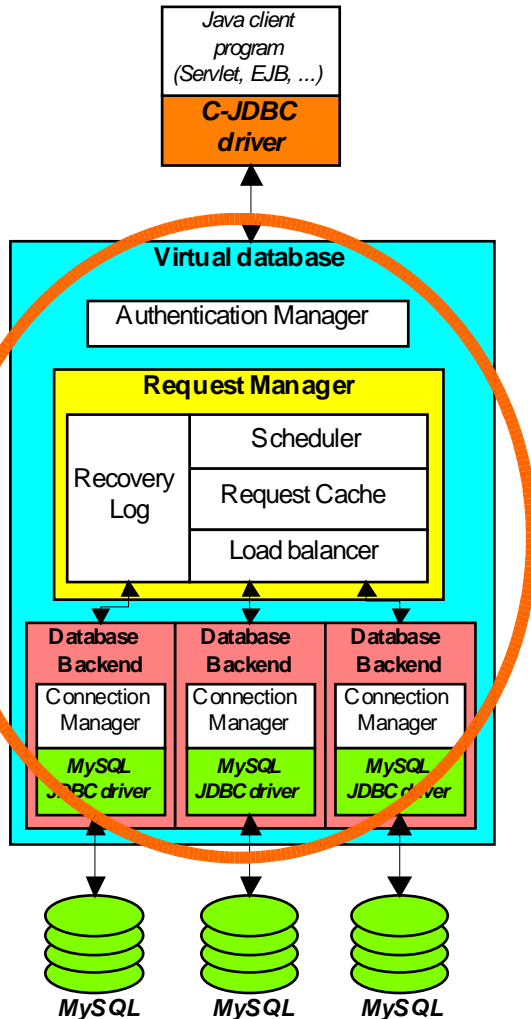
# Virtual Database

gives the view of a single database

establishes the mapping between the database name used by the application and the backend specific settings

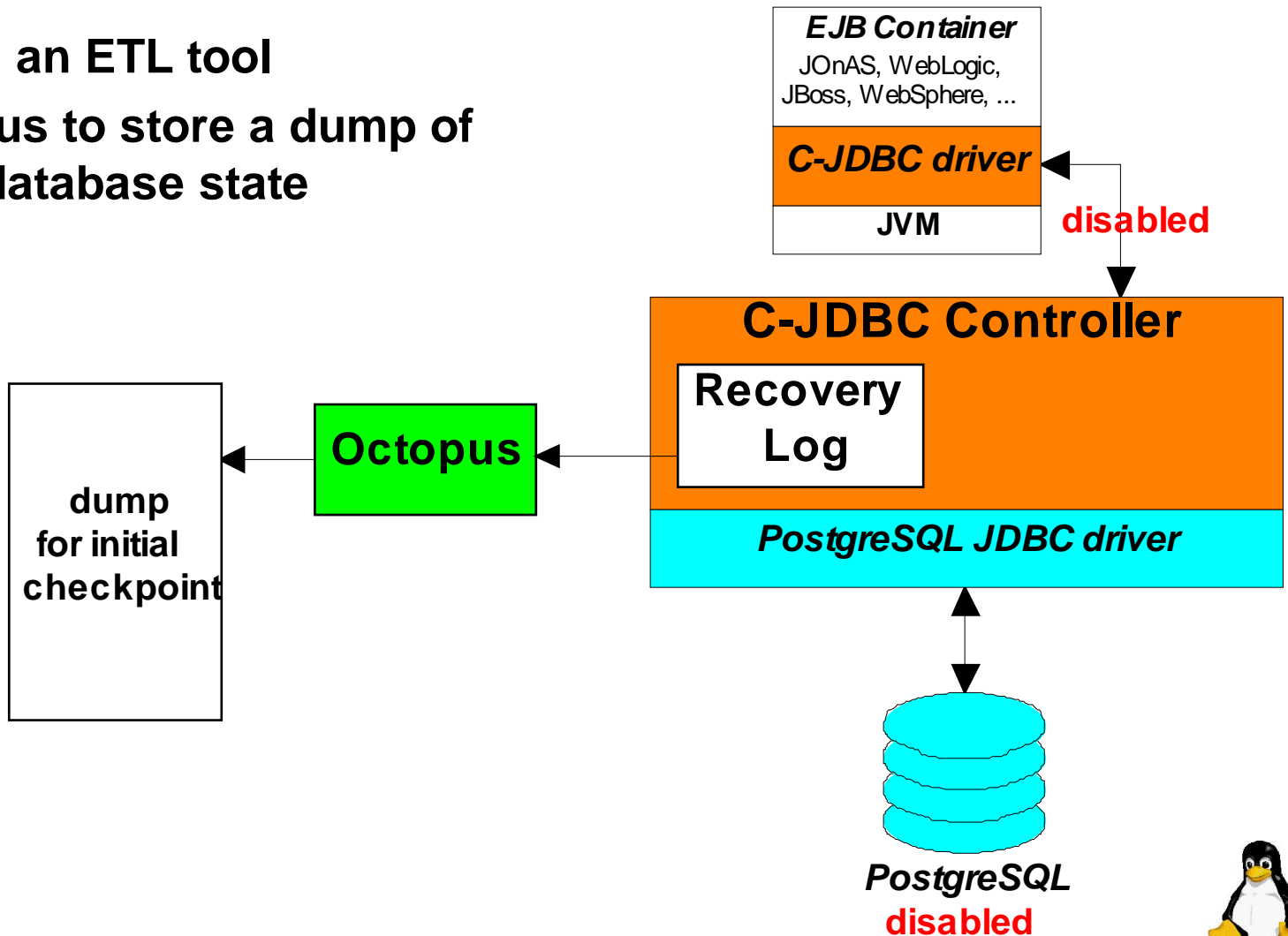
backends can be added and removed dynamically

configured using an XML configuration file



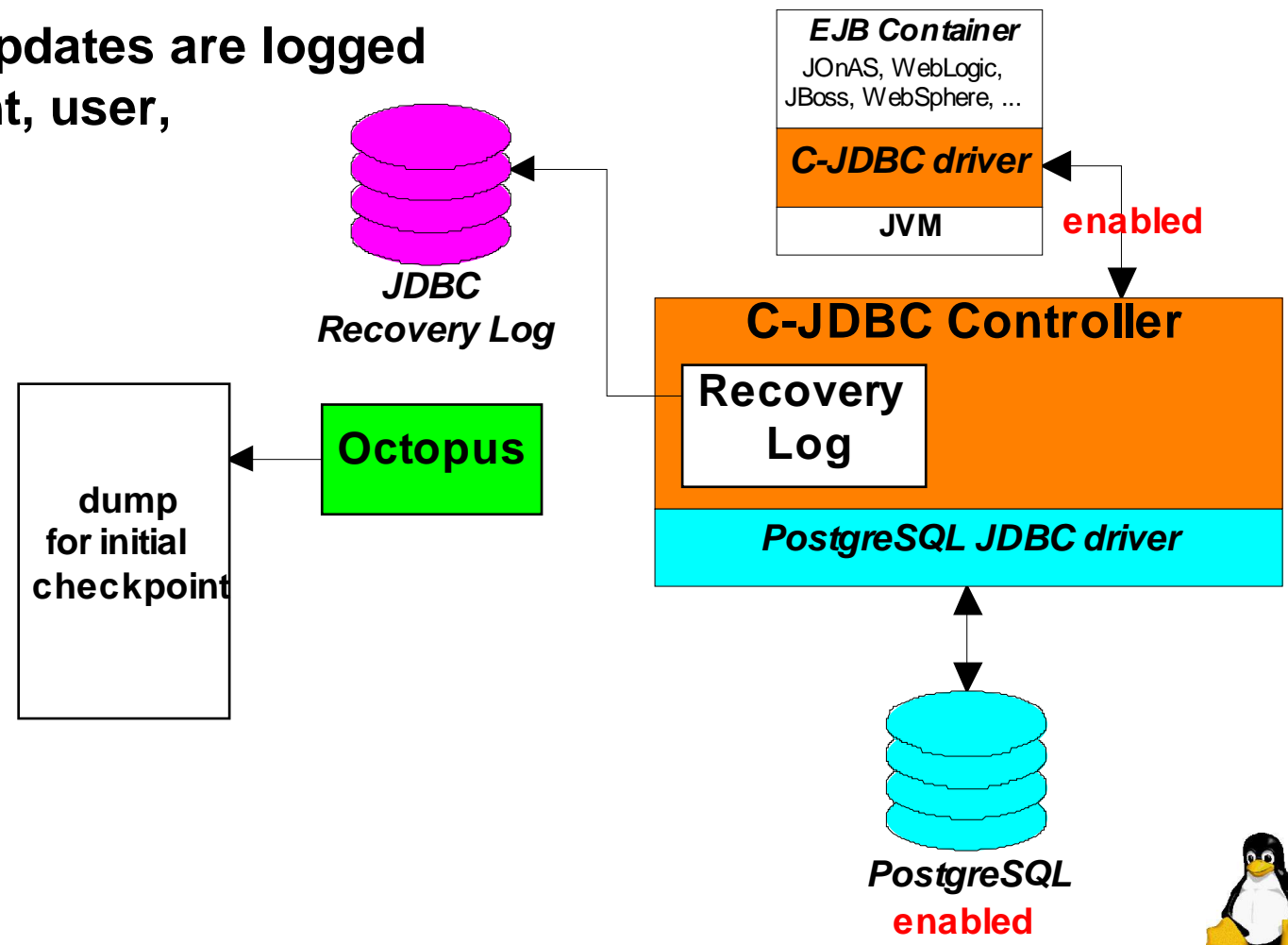
# Building the initial state

- ➔ Octopus is an ETL tool
- ➔ Use Octopus to store a dump of the initial database state



# Journaling

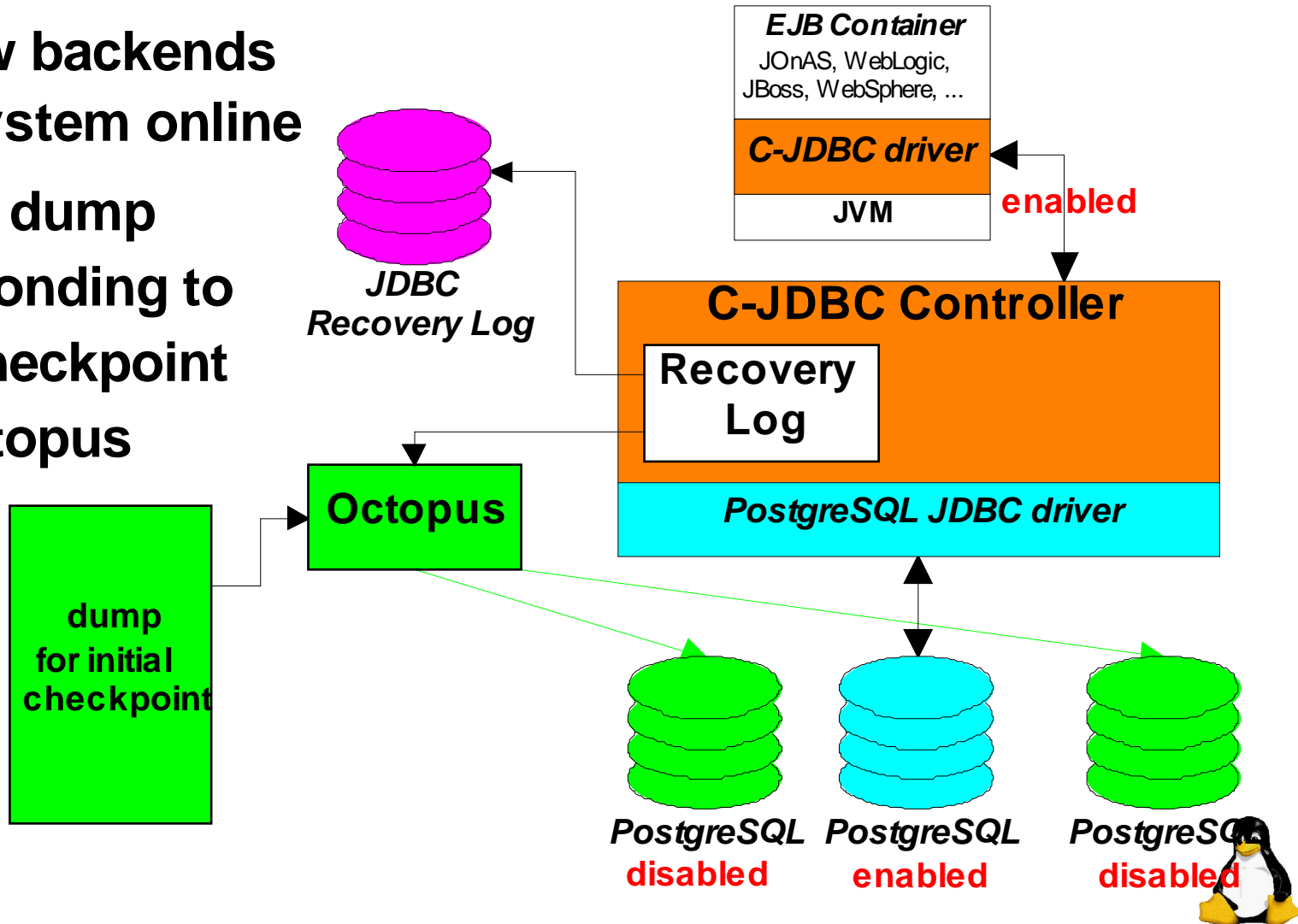
- ➔ Backend is enabled
- ➔ All database updates are logged (SQL statement, user, transaction, ..)





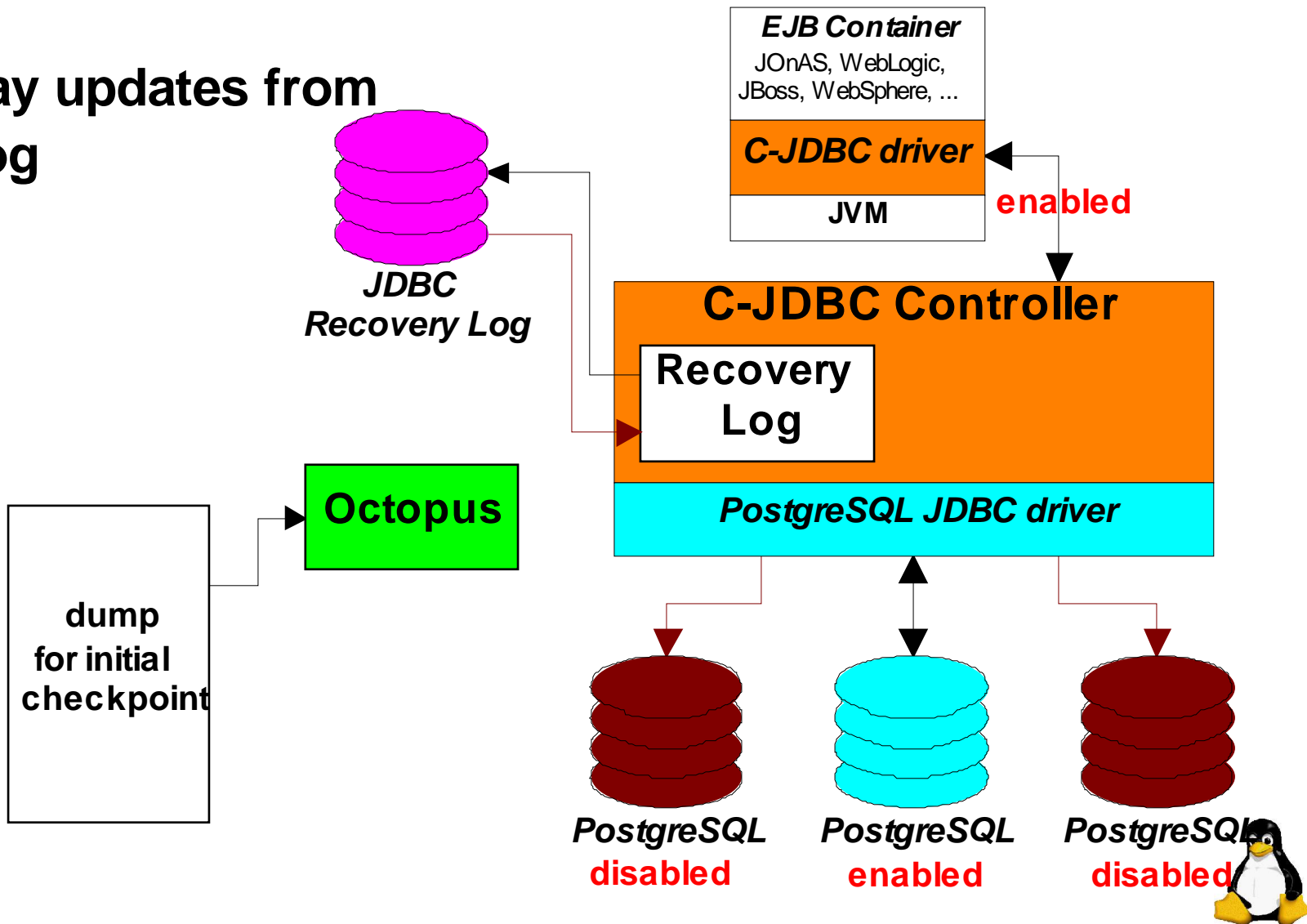
# Adding backend on the fly

- ➔ Add new backends while system online
- ➔ Restore dump corresponding to initial checkpoint with Octopus



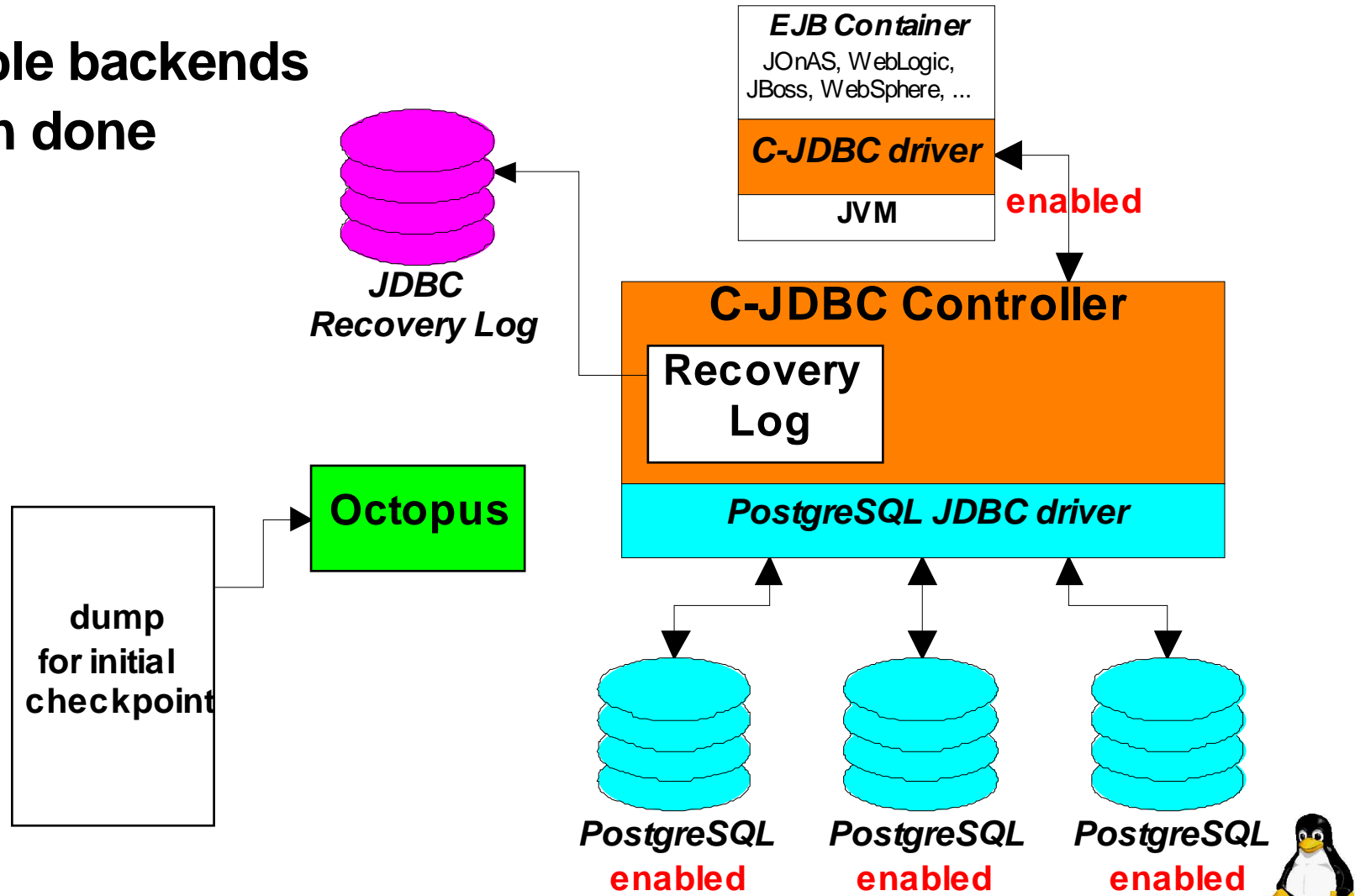
# Synchronizing backends

➔ **Replay updates from the log**



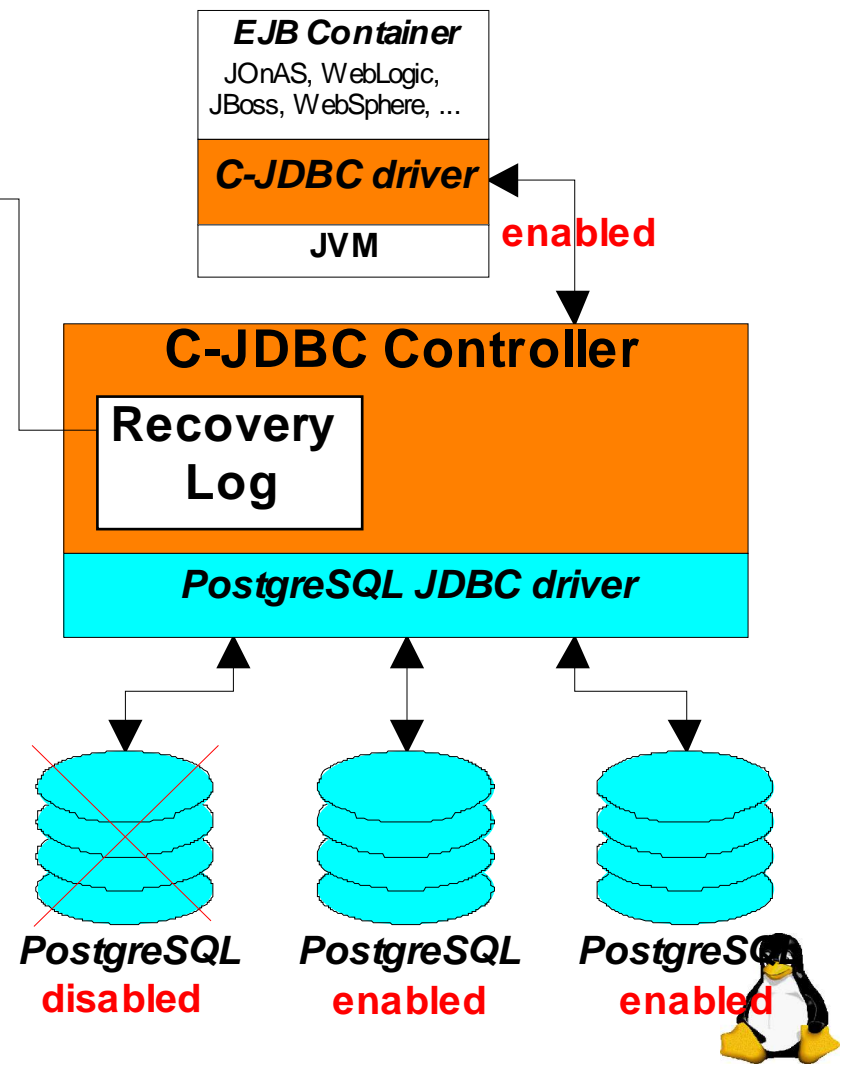
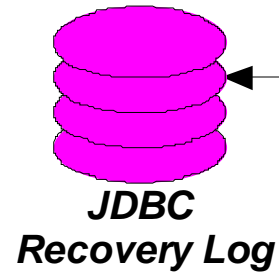
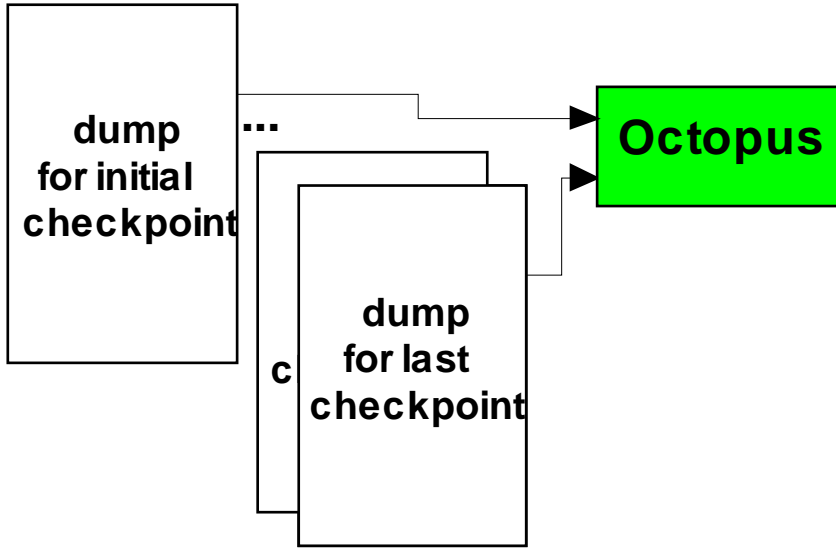
# Expanded Cluster

➔ **Enable backends when done**



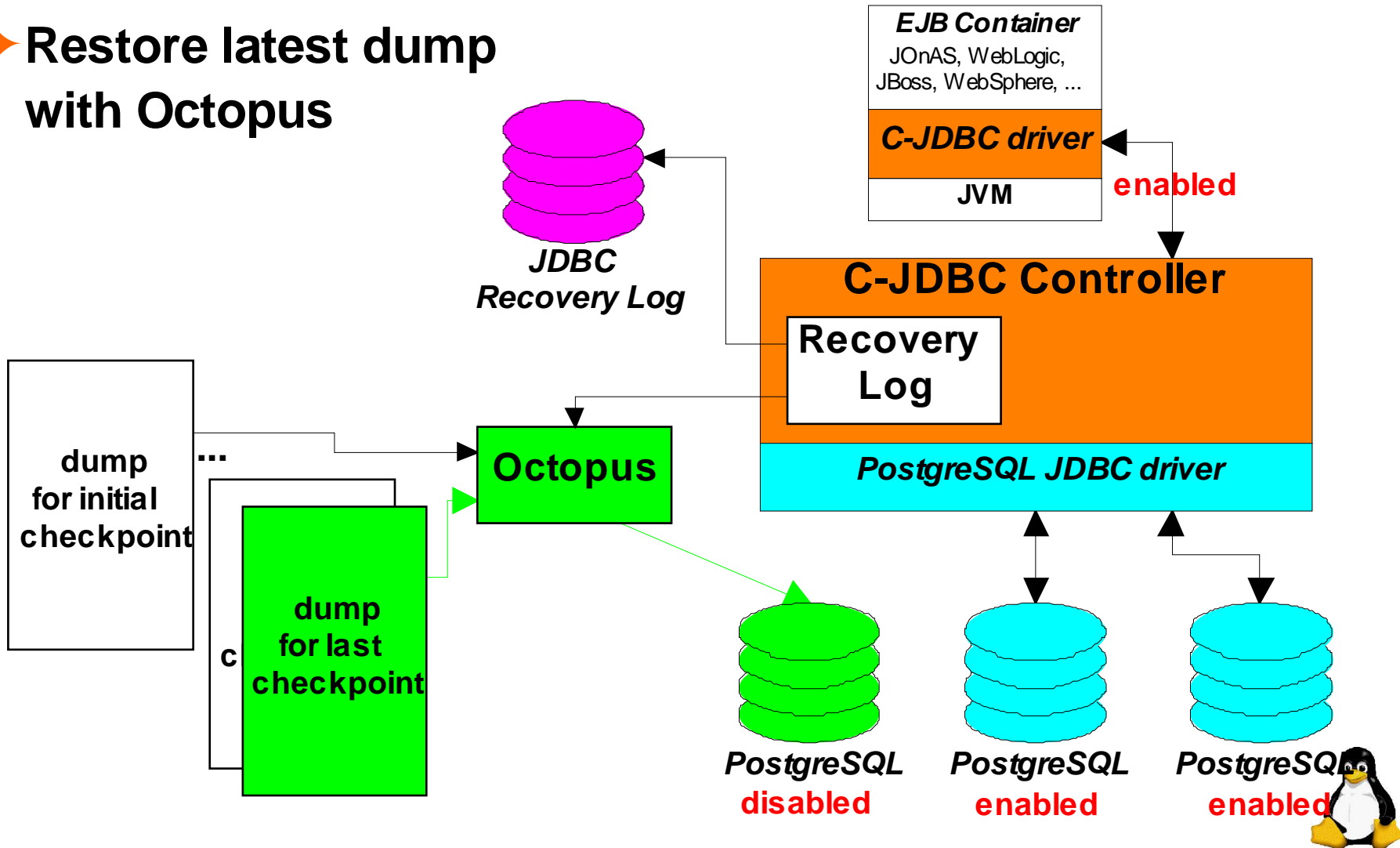
# Handling a backend failure

- ➔ A node fails!
- ➔ Automatically disabled but should be fixed or changed by administrator



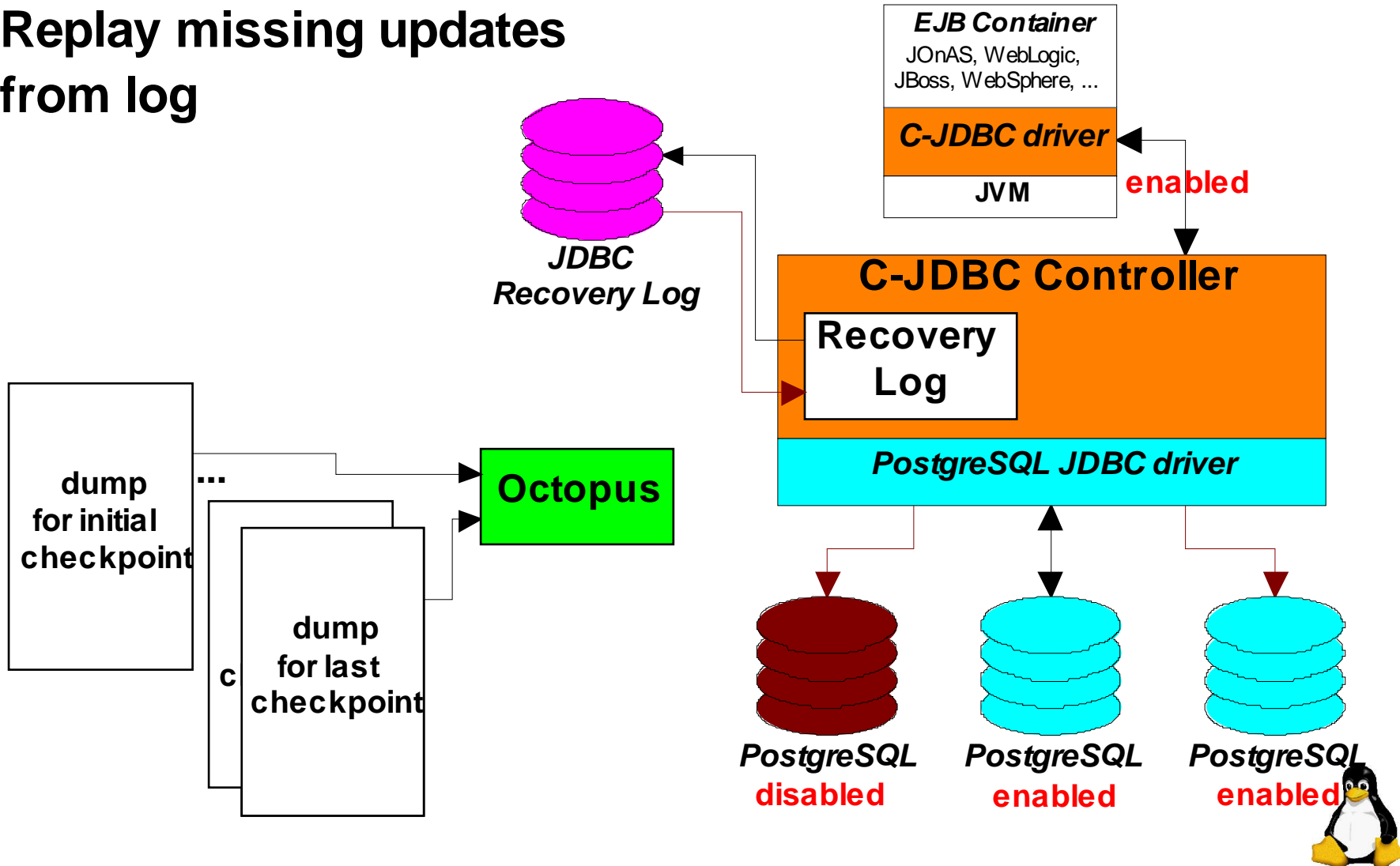
# Restoring a backend

➔ Restore latest dump with Octopus



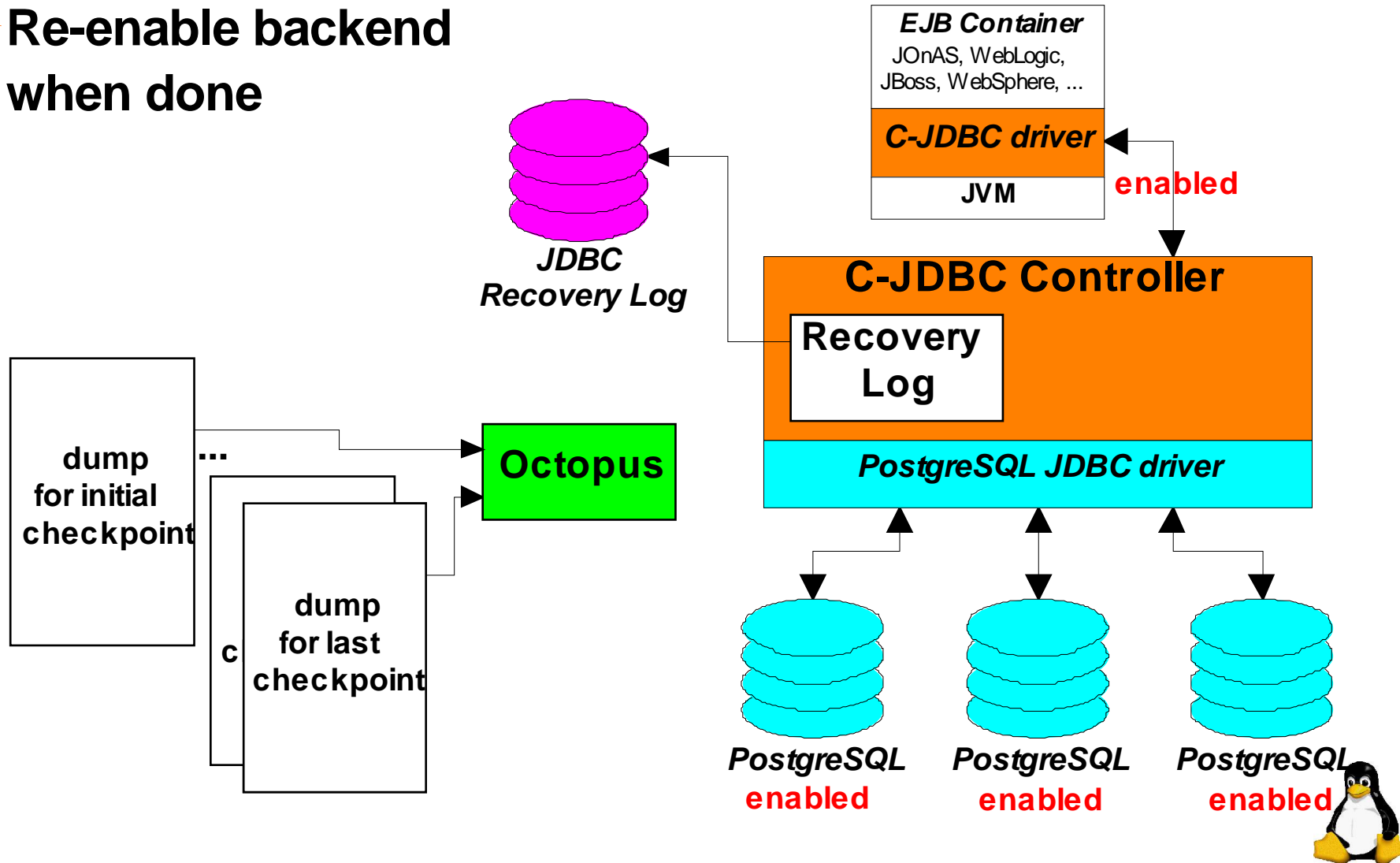
# Re-synchronization

## ➔ Replay missing updates from log



# Healed Cluster

➔ **Re-enable backend when done**



# Outline - Performance

**Motivations**

**Use-Cases**

**C-JDBC concepts**

**Performance**

**Monitoring**

**Community**

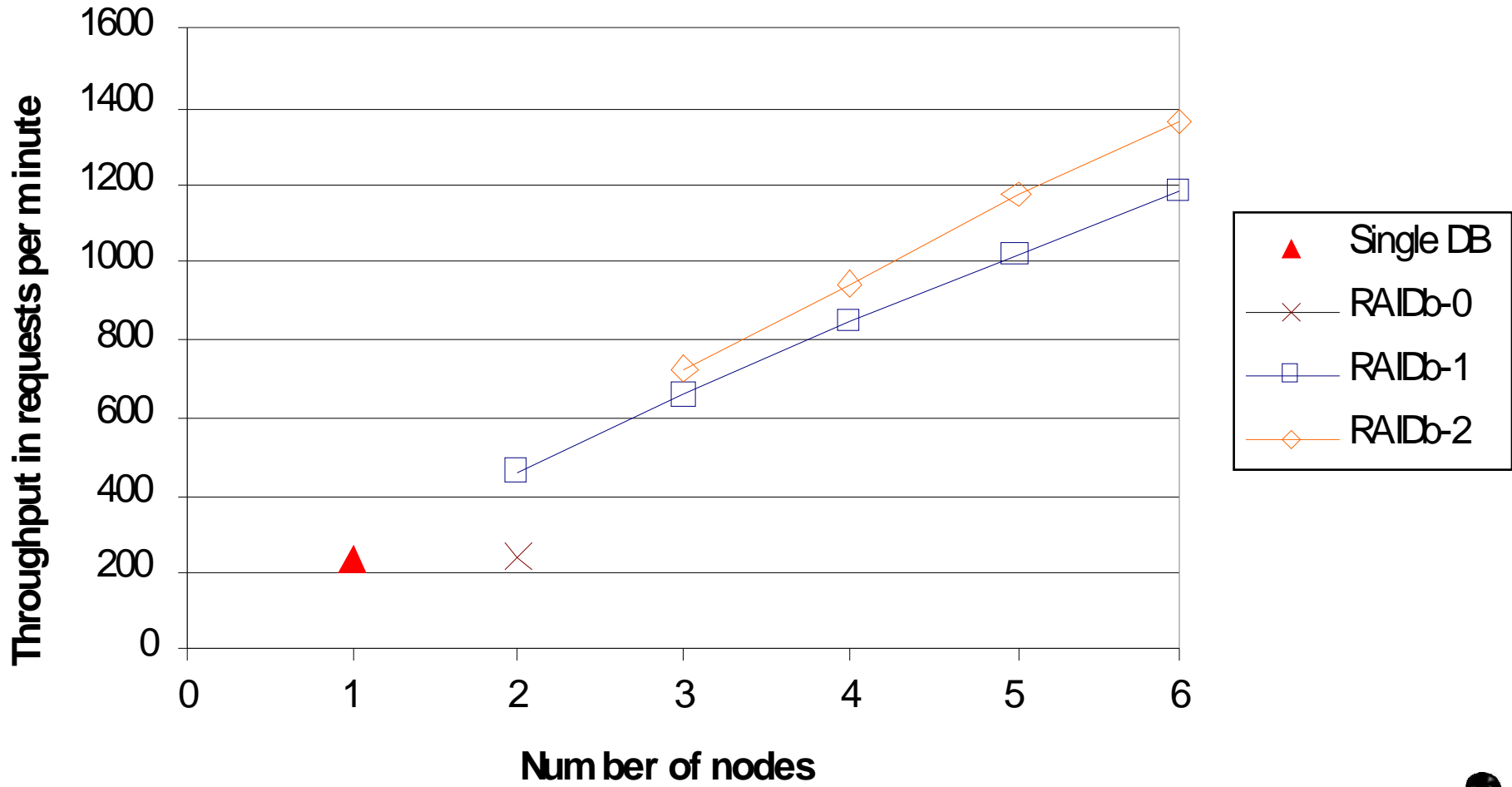
**Conclusion**





# TPC-W Performance

(Amazon.com)

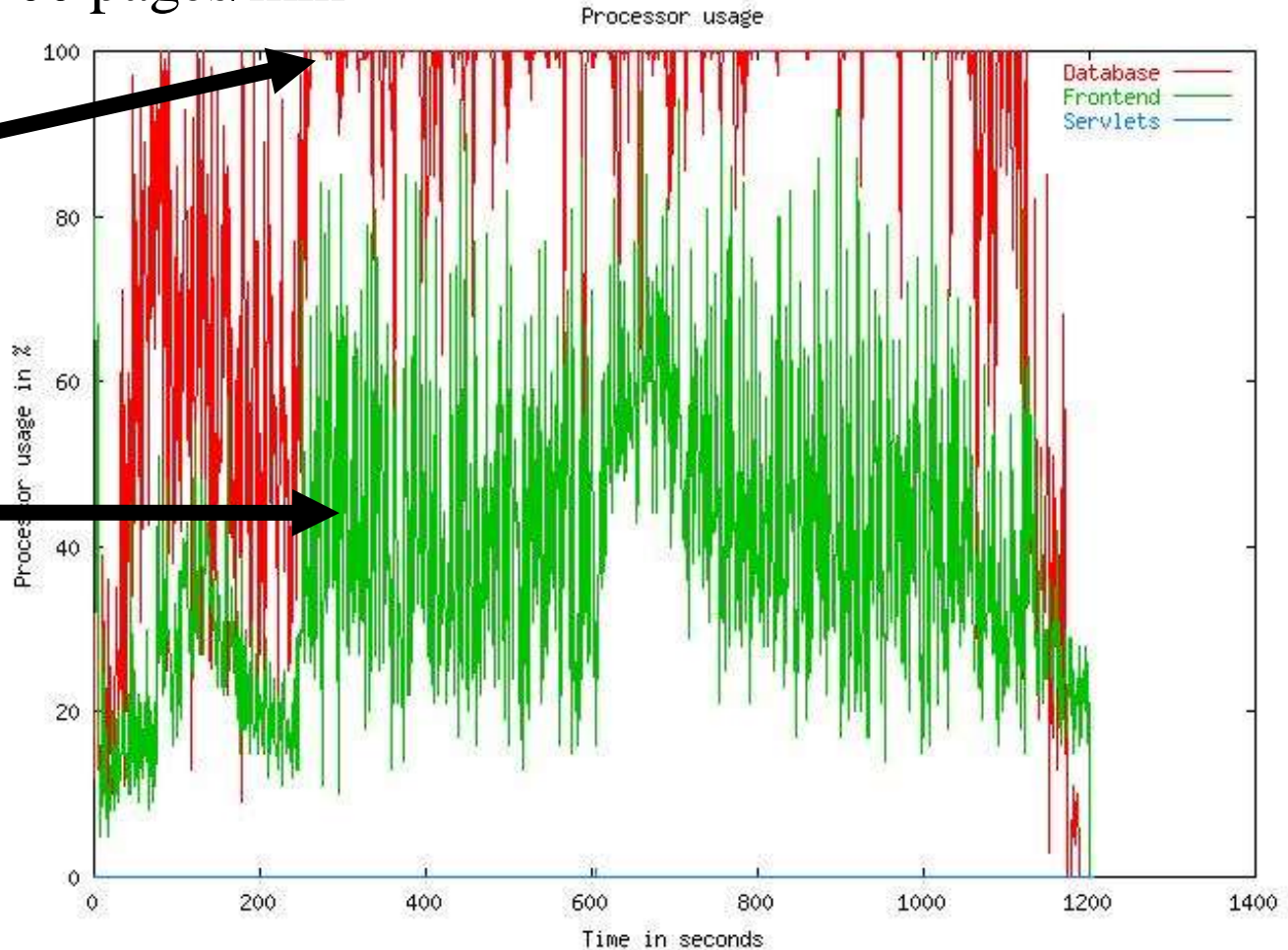


# RUBiS- Tomcat without C-JDBC caching

**Throughput: 3900 pages/min**

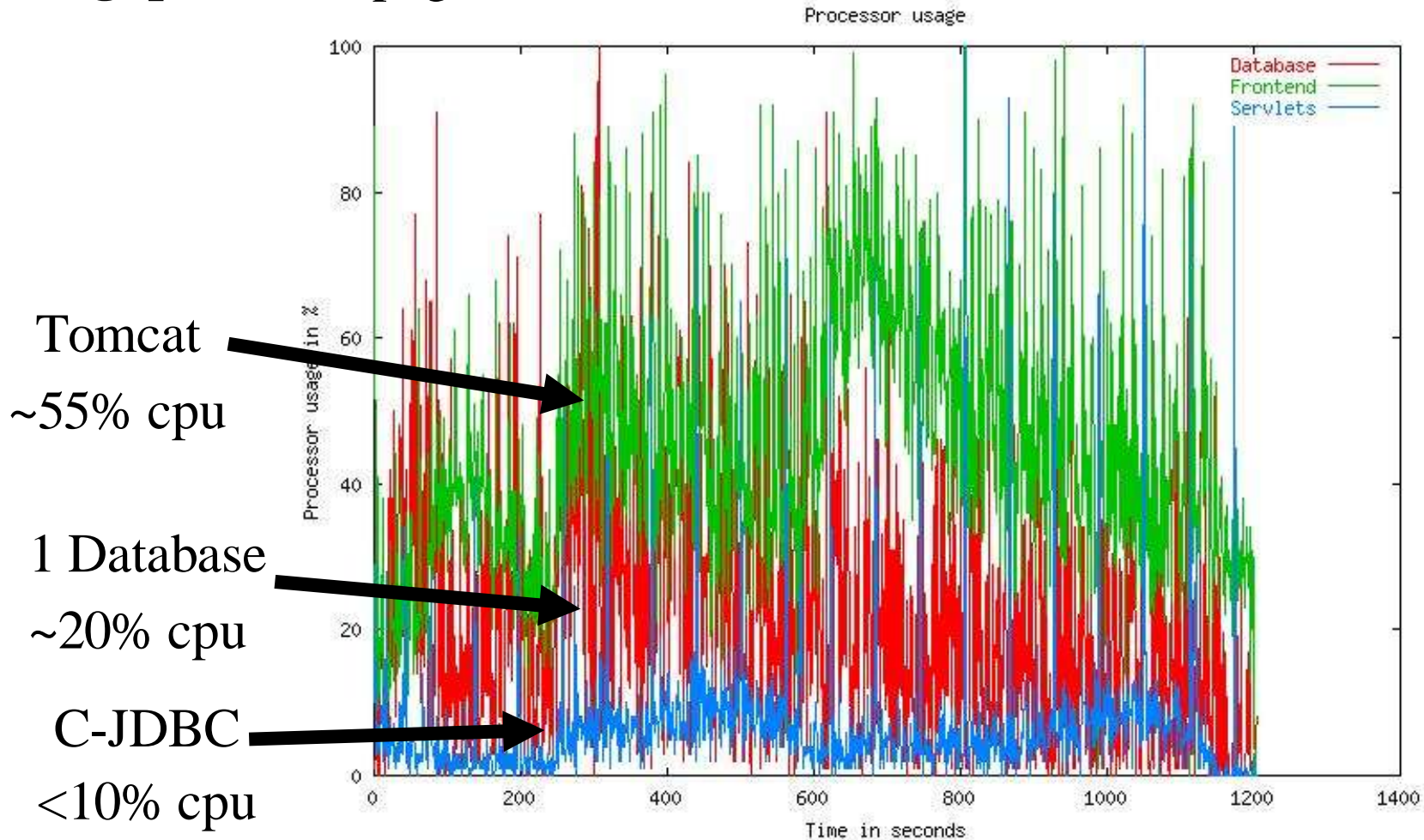
1 Database  
100% cpu

Tomcat  
~50% cpu



# RUBiS- Tomcat with C-JDBC caching

**Throughput: 4200 pages/min**



# Outline - Monitoring

**Motivations**

**Use-Cases**

**C-JDBC concepts**

**Performance**

**Monitoring**

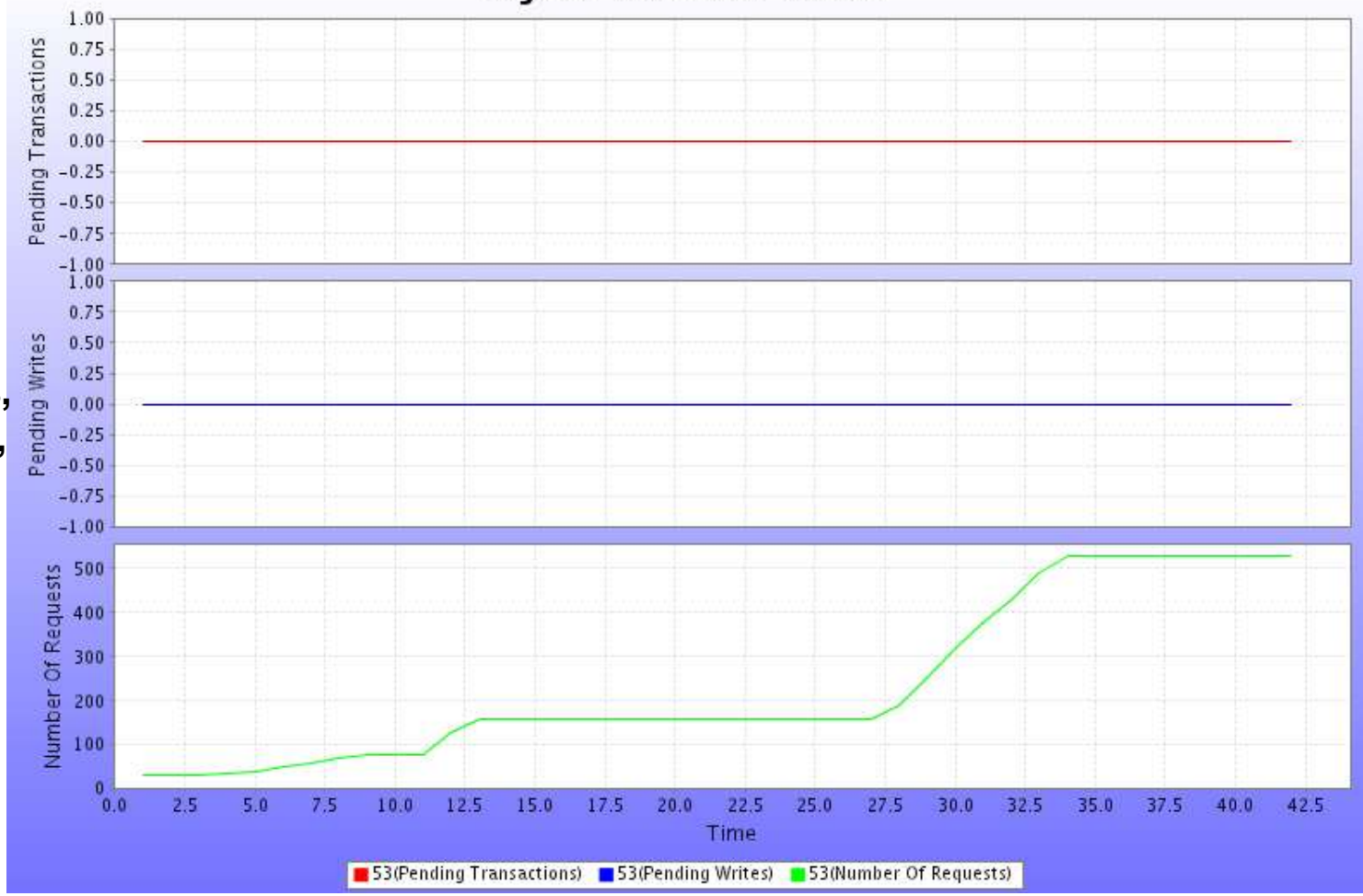
**Community**

**Conclusion**



# Monitoring/Trace

## C-JDBC Scheduler Viewer



Trace, save,  
get statistic  
content of  
different  
modules

Controller,  
database, users,  
backend, cache,  
load, memory  
usage ...



# SQL Console: Squirrel

Execute a set of atomic sql requests

Verify content of clustered database

Verify cluster schemas

Squirrel SQL Client Version 1.1final1 <2>

Connect to: myCJDBCdb (port 25322)

Drivers: C-JDBC, HyperSonic SQL

Aliases: myCJDBCdb (port 25322), myHSQLdb (port 9001), myHSQLdb (port 9002)

myCJDBCdb (port 25322) as user

ID	FIRSTN...	LASTNAME	STREET	CITY
0	Laura	Steel	429 Seventh Av.	Dallas
1	Susanne	King	366 - 20th Ave.	Oltten
2	Anne	Miller	20 Upland Pl.	Lyon
3	Michael	Clancy	542 Upland Pl.	San Francisco
4	Sylva	Ringer	365 College Av.	Dallas

Query 1 elapsed time (seconds) - Total: 0.165, SQL query: 0.146, Building output: 0.019

1,23

15.3 MB/198.5 - 4:14:16 PM CET

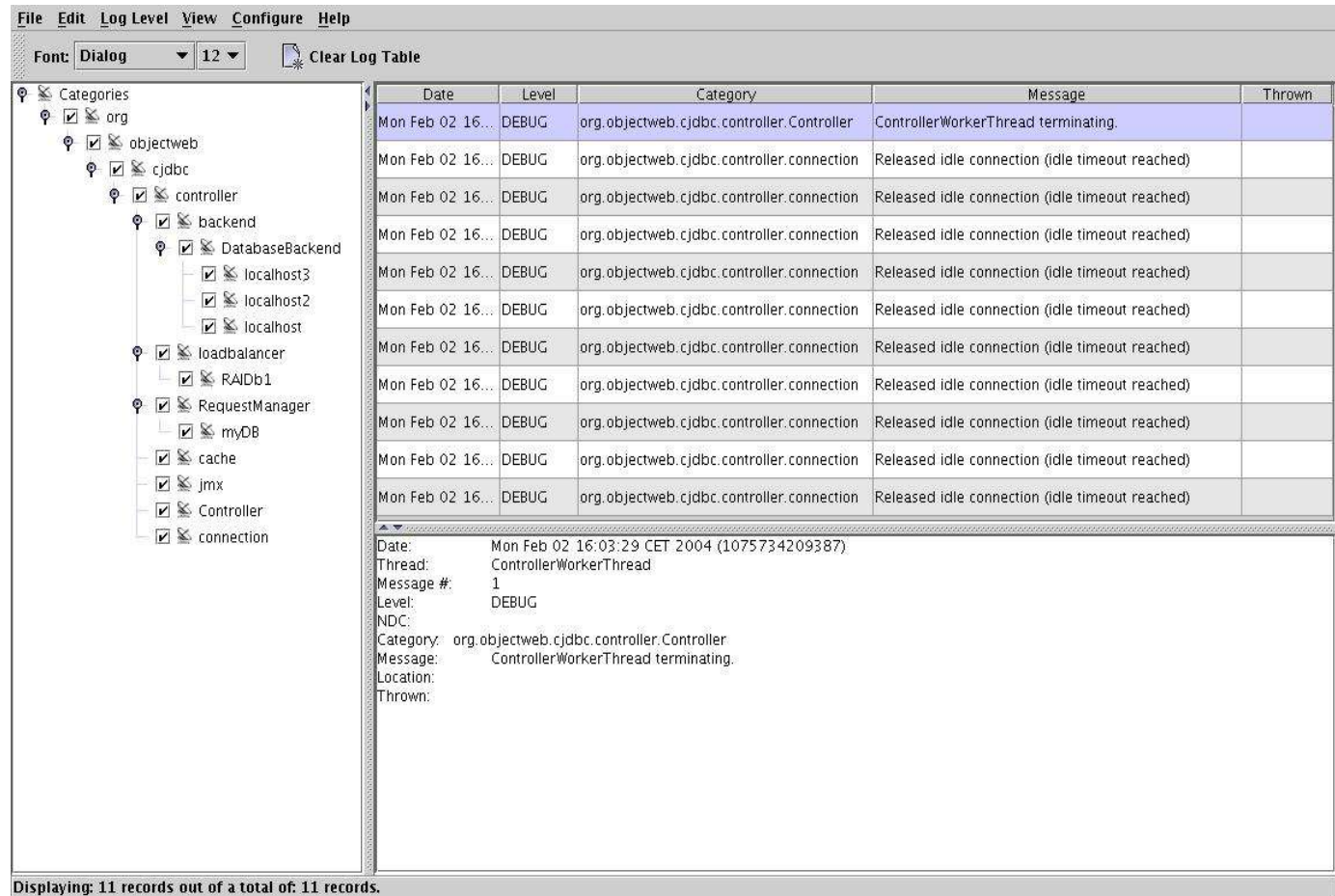




# View graphic remote logs

– **Watch execution:**

- per backend
- per controller
- per virtual database



The screenshot shows the ObjectWeb log viewer interface. The top menu includes File, Edit, Log Level, View, Configure, and Help. Below the menu, there are controls for font (Dialog, 12) and a 'Clear Log Table' button. The left pane shows a tree view of categories with checkboxes for selection. The right pane displays a table of log records.

Date	Level	Category	Message	Thrown
Mon Feb 02 16...	DEBUG	org.objectweb.cjdbc.controller.Controller	ControllerWorkerThread terminating.	
Mon Feb 02 16...	DEBUG	org.objectweb.cjdbc.controller.connection	Released idle connection (idle timeout reached)	
Mon Feb 02 16...	DEBUG	org.objectweb.cjdbc.controller.connection	Released idle connection (idle timeout reached)	
Mon Feb 02 16...	DEBUG	org.objectweb.cjdbc.controller.connection	Released idle connection (idle timeout reached)	
Mon Feb 02 16...	DEBUG	org.objectweb.cjdbc.controller.connection	Released idle connection (idle timeout reached)	
Mon Feb 02 16...	DEBUG	org.objectweb.cjdbc.controller.connection	Released idle connection (idle timeout reached)	
Mon Feb 02 16...	DEBUG	org.objectweb.cjdbc.controller.connection	Released idle connection (idle timeout reached)	
Mon Feb 02 16...	DEBUG	org.objectweb.cjdbc.controller.connection	Released idle connection (idle timeout reached)	
Mon Feb 02 16...	DEBUG	org.objectweb.cjdbc.controller.connection	Released idle connection (idle timeout reached)	
Mon Feb 02 16...	DEBUG	org.objectweb.cjdbc.controller.connection	Released idle connection (idle timeout reached)	
Mon Feb 02 16...	DEBUG	org.objectweb.cjdbc.controller.connection	Released idle connection (idle timeout reached)	

Below the table, there is a detailed view of a selected log entry:

```

Date: Mon Feb 02 16:03:29 CET 2004 (1075734209387)
Thread: ControllerWorkerThread
Message #: 1
Level: DEBUG
NDC:
Category: org.objectweb.cjdbc.controller.Controller
Message: ControllerWorkerThread terminating.
Location:
Thrown:
    
```

At the bottom of the window, it says: "Displaying: 11 records out of a total of: 11 records."



# Outline - Community

**Motivations**

**Use-Cases**

**C-JDBC concepts**

**Performance**

**Monitoring**

**Community**

**Conclusion**





## Downloads

- total : 11260 downloads since may 2003
- 2004 : > 1300 downloads
- Top 5 of the most downloaded ObjectWeb project

## Mailing lists

- [c-jdbc@objectweb.org](mailto:c-jdbc@objectweb.org): 124 subscribers

## Team

- 11 committers
- 1 full-time INRIA engineer



## **Mathieu Peltier (ObjectWeb)**

- build scripts, automatic installer, JUnit test

## **Julie Marguerite (ObjectWeb)**

- JDBCRecoveryLog, automatic schema detection

## **Christiana Amza (Rice University),**

## **Anupam Chanda (Rice University), Sara Bouchenak (EPF Lausanne)**

- SQL query caching

## **Guillaume Bort (INRIA Lorraine)**

- JBoss support

## **Marek Prochazka (INRIA Rhone-Alpes)**

- Datasource implementation

## **Greg Ward (dplanet.ch)**

- Sybase support, design, debug

## **Marc Wick (monte-bre.ch)**

- HSQL support, design debug and ideas

## **Duncan Smith (mightybot.com)**

- IP binding, security concerns, console, JMX, distributed management

## **Vadim Kassin (Kazakhstan Stock Exchange)**

- Autogenerated keys support



# Outline - Conclusion

**Motivations**

**Use-Cases**

**C-JDBC concepts**

**Performance**

**Monitoring**

**Community**

**Conclusion**



## C-JDBC 1.0 rc1 release

- Generic JDBC 2.0 driver
- Schedulers and load balancers for RAIDb 0, 1 and 2
- Fine grain query caching and sql monitoring
- JDBC recovery log
- Logger/request player
- Java installer
- User documentation
- Octopus integration



# On-going work and efforts

Listen to the needs of users, quick answers on the mailing list

Horizontal scalability

Fully featured administration console

Graphical configuration and deployment of  
centralized/distributed backends and controllers (offline/online)

Dynamic reconfiguration

Automated Load testing, report page updated by users

RPM packaging (Jpackage version 1.0b15 done)

C-ODBC (asked by a lot of people)



# Take this message at home

## Database Clustering Middleware (100% java)

### Based on JDBC Standard

### No code modification (application or database)

### Open source (LGPL)



# Questions & Answers

---

Thanks to all users and  
contributors ...

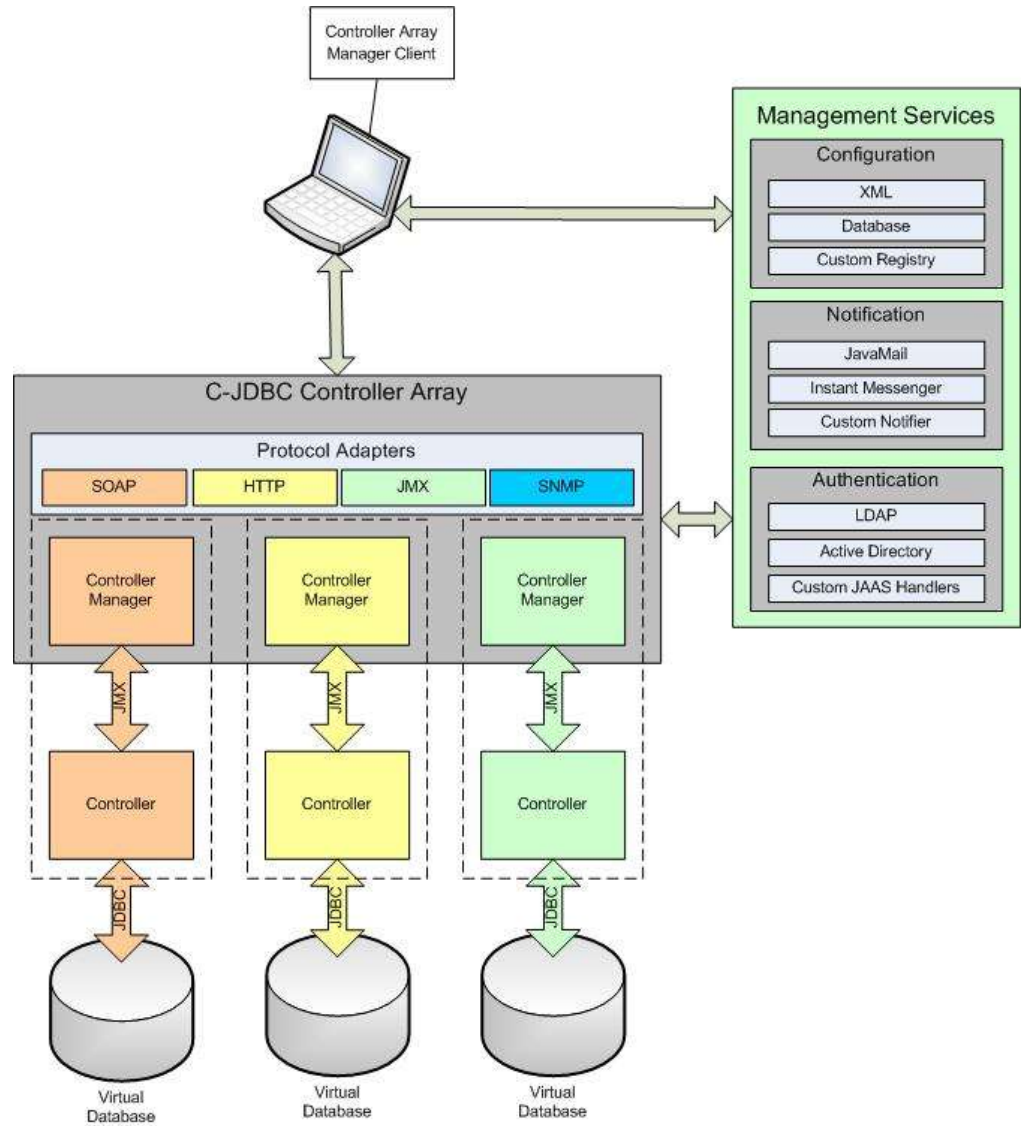
<http://c-jdbc.objectweb.org>



# Prototype

## C-JDBC Management Framework

## Shared design





# Request cache

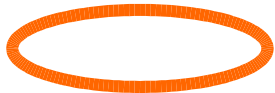
**caches results from SQL requests**

**improved SQL statement analysis to limit cache invalidations**

- table based invalidations
- column based invalidations
- single-row SELECT optimization

**request parsing possible in the C-JDBC driver**

- offload the controller
- parsing caching in the driver



# Load balancer

## RAIDb-0

- query directed to the backend having the needed tables

## RAIDb-1

- read executed by current thread
- write executed in parallel by a dedicated thread per backend
- result returned if one, majority or all commit
- if one node fails but others succeed, failing node is disabled

## RAIDb-2

- same as RAIDb-1 except that writes are sent only to nodes owning the written table



# Connection Manager

## Connection pooling for a backend

- Simple : no pooling
- RandomWait : blocking pool
- FailFast : non-blocking pool
- VariablePool : dynamic pool

## Connection pools defined on a per login basis

- resource management per login
- dedicated connections for admin



# Scheduler

**Manages concurrency control**

**Specific implementations for Single DB, RAIDb 0, 1 and 2**

**Query-level**

**Optimistic and pessimistic transaction level**

- uses the database schema that is automatically fetched from backends



# Recovery Log

**Checkpoints are associated with database dumps**

**Record all updates and transaction markers since a checkpoint**

**Used to resynchronize a database from a checkpoint**

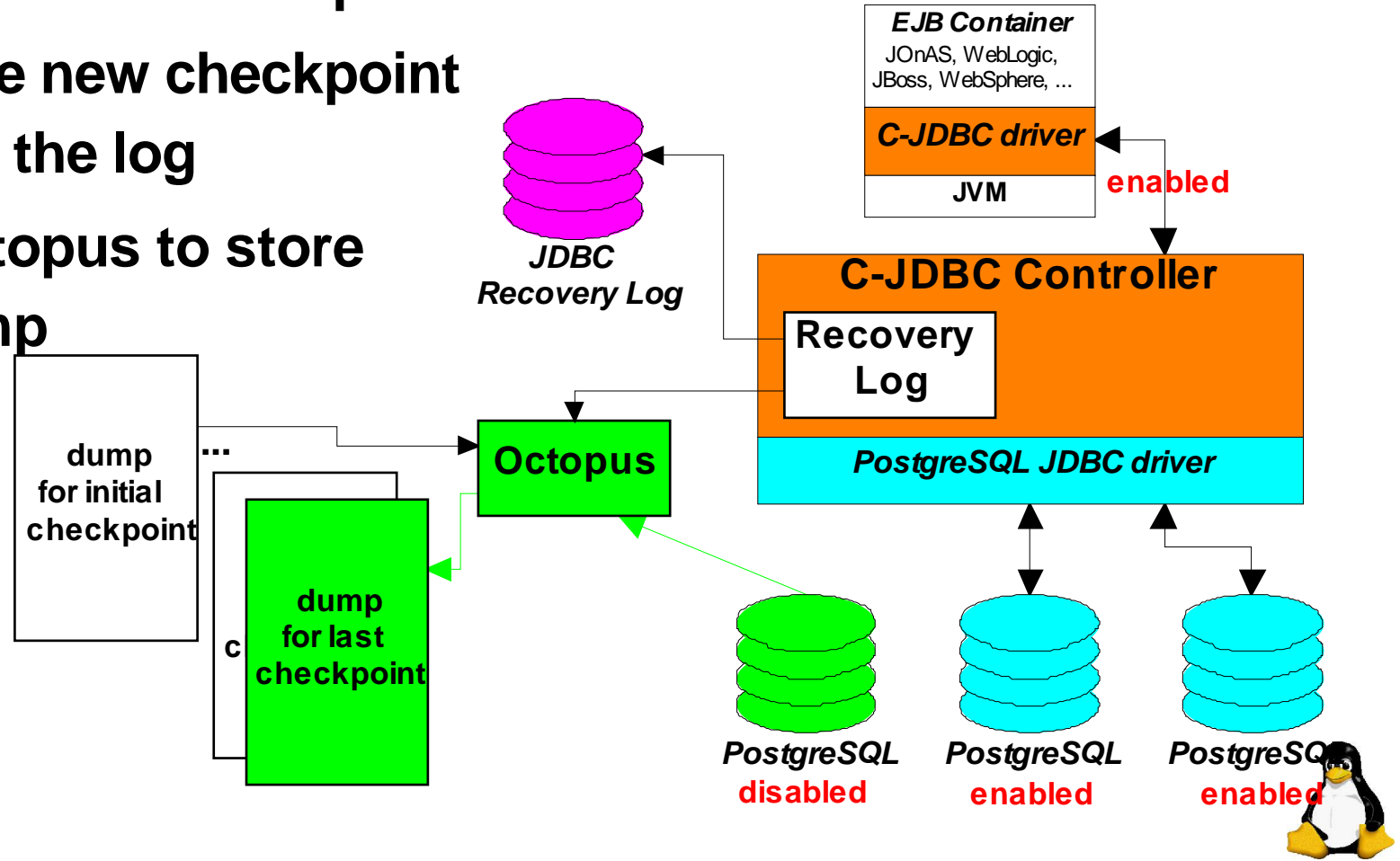
## JDBCRecoveryLog

- store information in a database
- can be re-injected in a C-JDBC cluster for fault tolerance



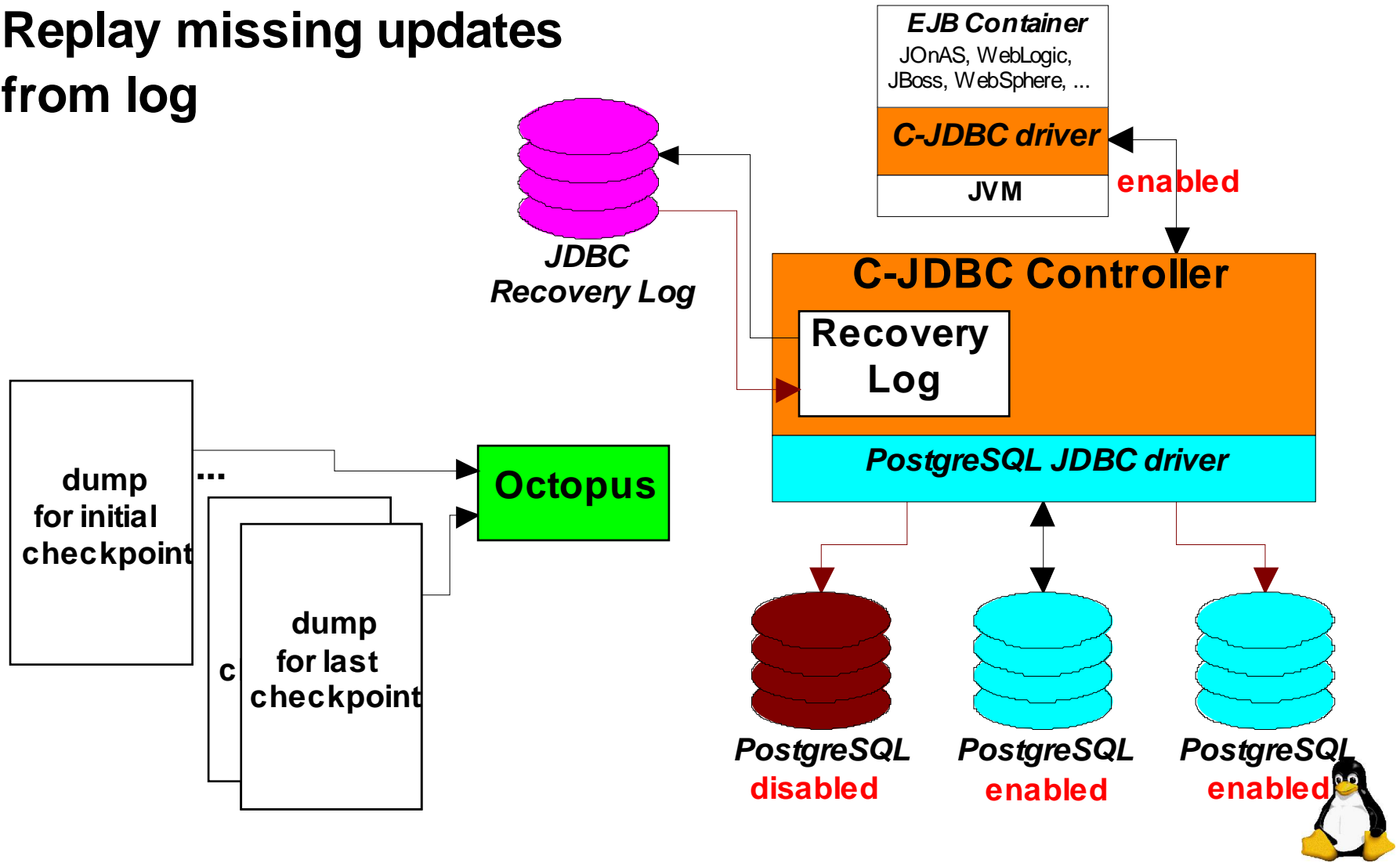
# Making new checkpoints

- ➔ Disable one backend to have a coherent snapshot
- ➔ Mark the new checkpoint entry in the log
- ➔ Use Octopus to store the dump



# Making new checkpoints

## ➔ Replay missing updates from log



# Making new checkpoints

➔ **Re-enable backend when done**

