Ing. Petr Aubrecht, Ph.D.

stringdata®

Real Deployments of JavaEE Applications

Introductory Question

- What technology would you choose to implement really big e-shop?
- How much can you bet on the reliability? SLA will include a fee per hour of not working system.
- How much are you sure it will not crash?
  - Out of memory
  - Unexpected behavior
  - Hardware error recovery
- Is it scaling?

Agenda

- What Is Enterprise Application? What to Consider? THE “Right” Technology...
- Development
- Deployment
- Production

Keep in Mind

- This whole presentation represents MY opinion, even in my company are people different view.
- If you don’t agree – DISCUSS!

Example – what do you think about SAP?

What is...

- What is enterprise application?

What Is Enterprise Application?

- Help people to do their business, they depend on it!
- SLA expresses the importance, the sw simply cannot stop working. How much bank looses per hour of not working home banking?
- Examples: ERP (manufacturing, hotels), management of anything, payments/billing processing, on-line marketplace...
- Most of the biggest enterprise applications
  - run on mainframes and
  - are done in COBOL.

What to Consider (I)

- How long will be supported enterprise application?
  - Enterprise application = implemented today, supported for many ears with small changes and small team.
  - Cannot use bleeding edge: Google, Youtube, FB, Twitter rewrites front page frequently!
    - Did you know, that in backend, FB has enterprise apps as well?
  - We need programmer for the technology in 10 year from now!
  - We need the technology to be supported in 10 year from now, maybe much longer!

stringdata®

What to Consider (II)

- What features we need? We ARE specialists in business logic, but not in these areas:
  - **Reliability** (transactions, recovery)
  - **Performance** (optimization, caching, pools)
  - **Scalability** (vertical, horizontal)
  - **Security** (authorization, authentication)

Reliability – multigeneration architecture in SQL dbs, 2-phase locking, prevention of deadlock
Preformance – optimization to the latest processor, branching optimization, why b-tree over binary tree,...
Scalability – theory of network computation, ...
Security – SQL injection, XSS, session stealing, rainbow tables, ...

stringdata®

What is THE “Right” Technology Forever?

- Win32 DEAD
- VBX DEAD
- Delphi DEAD
- MFC DEAD
- ActiveX DEAD
- Java Servlet
- JSP DEAD
- JSF
- GWT Sustaining
- JavaEE 3 DEAD, 4, ...

- COM/DCOM DEAD DEAD
- C# + .net
- Javascript
- HTML 5
- Angular 1 DEAD, 2 Not mature
- ReactJS
- Grid DEAD
- Cluster DEAD
- Cloud

- AJAX DEAD
- WS-SOAP DEAD
- REST
- Single Page
- node.js
- Struts 1,2 DEAD

- **COBOL!**

Few Myths – MySQL

- MySQL – fast db
  - **MyISAM** is fast, stupid
  - **InnoDB** – featureful (transactions, foreign keys), but slow
  - Must be good, FB uses it! Yes – they employ 40(!) people working ON mysql.
  - Either you can pay somebody to modify mysql or use PG/Oracle/MSSQL.

Predictability

- Why is predictability important?
- Only stable technologies have known limitations
  - There are projects rewritten from PHP to Java because of memory… predictability!
  - Example: xml-sql mapping library in metadata builder

Development

- OK, we chose the right technology, what to keep in mind during development?

Just Few Ideas For JavaEE

- Consider Remote stateless bean – it allows load balancing
- Learn EntityManager behavior, usual source of problems
- Learn from Clean Code, Effective Java, Adam Bien
- Some cool tools: JRebel, VisualVM for memory dumps

Usual Development Setup

- Unit test! TDD whenever possible.
- Simple setup (maven), newcomer must be productive from day 1.
- Automate
  - Continuous integration (Jenkins)
  - Continuous deployment
    - QA server with night build
    - Stable server with RC
    - Copy of production server(s) for performance of specific testing
  - Continuous verification of performance

Deployment

- Well, the application is developed, so we click in Jenkins to deploy to production and we are done!

Deployment – Servers

- 99.999 % reliability = mainframe
- Servers – choose one and stick with it
  - Tomcat, not JEE, but useful for Spring, simple
  - TomEE – lightweight, simple
  - Glassfish/Payara – full, reference implementation, nice GUI
  - IBM WebShere – full, “enterprise”, “IBM-way”
  - BEA Weblogic – very advanced and expensive

Deployment – (Virtual) Hardware

- SaaS – Software as a Service
- Virtualization
  - Docker
  - VMWare ESX, VirtualBox
- Paravirtualization
  - XEN – Citrix
- Cloud
  - Amazon - “THE” cloud
  - Either simply borrow a virtual machine with your server from last slide… → **VPS**
  - ...or use Amazon services (e.g. database) → **cloud app**
  - Others: Azure

Deployment of a New Version

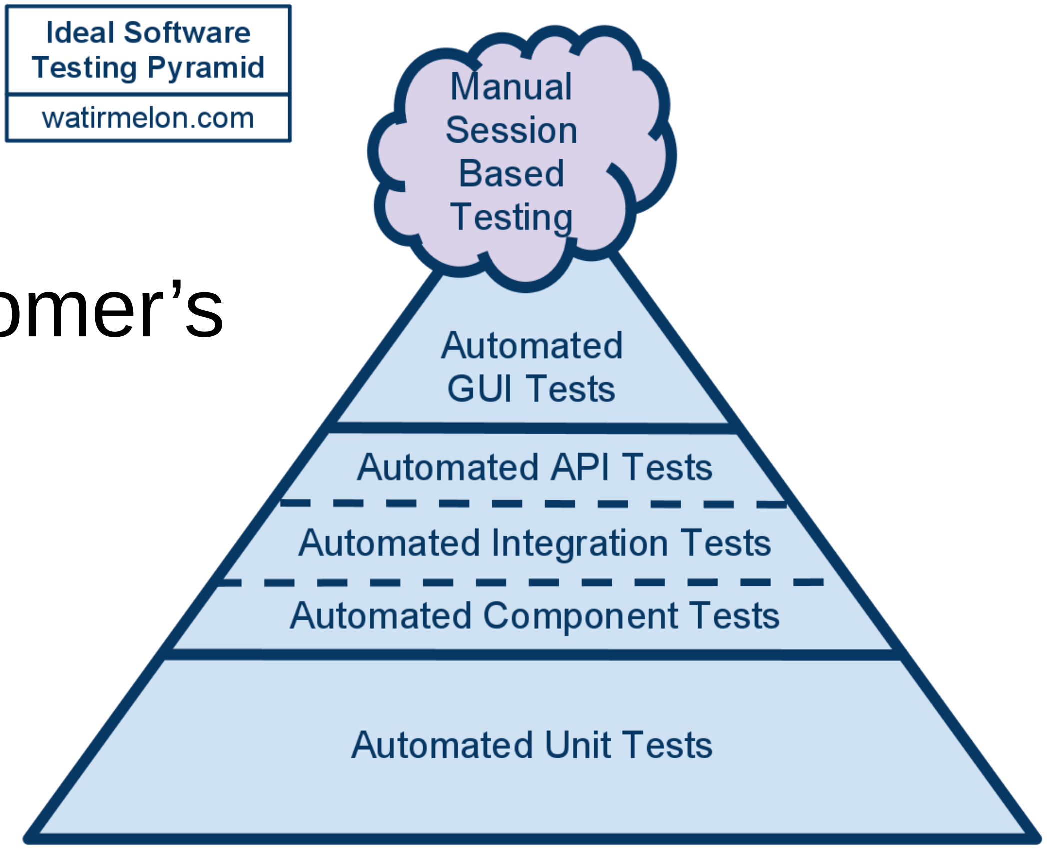
- Can we simply deploy a new version?
  - Database changes
  - What if it will not work?
  - Didn’t you forget backup?
  - What is the revert strategy?

New Version – Database Upgrade

- Usually we use library for DB upgrade
  - Liquibase (Flyway)
  - Keeps track of history of upgrades
  - Automates structure changes in all databases
  - Only forward and only step by step
    - Verifiable
    - Reliable

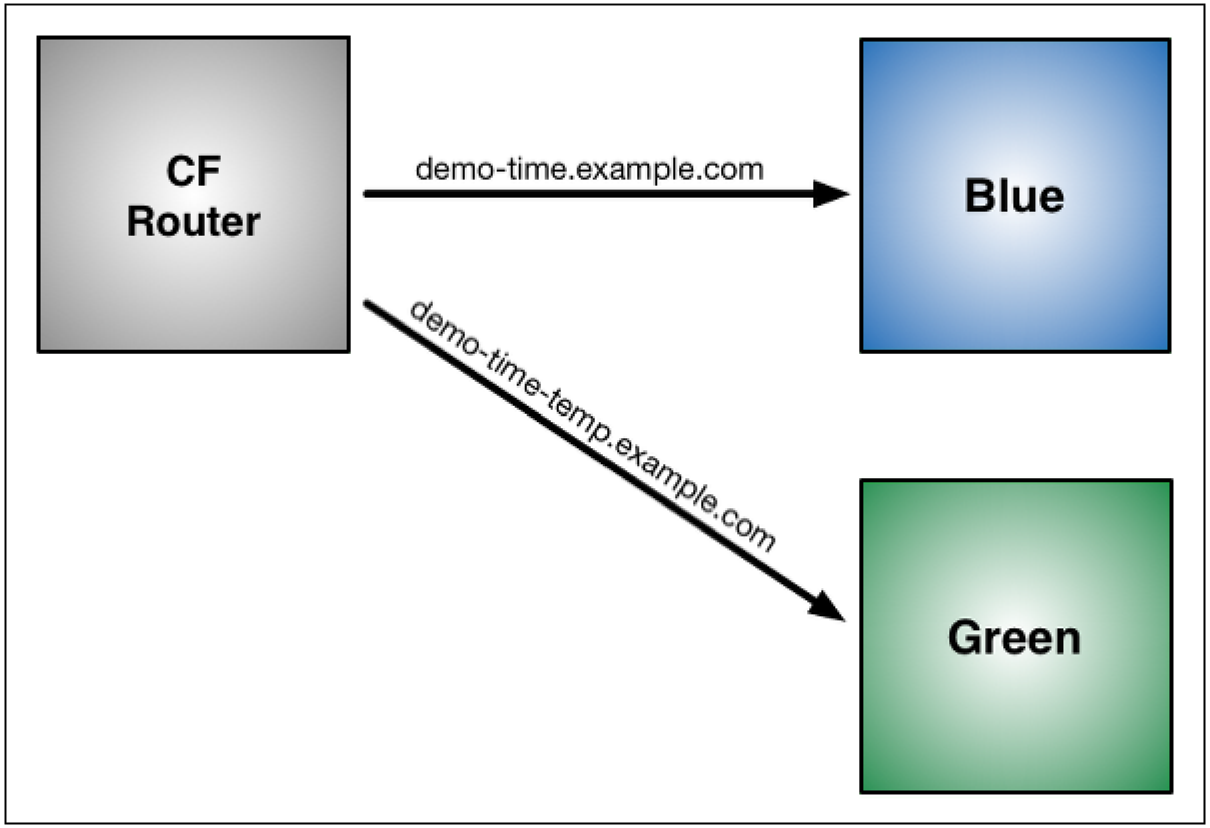
Deployment of a New Version

- Internal servers
  - QA – nightly, testers review
  - Stable version for demos, performance test, RC
  - Production copies
  - UAT
  - Production at customer's site


Ideal Software Testing Pyramid
watirmelon.com
Manual Session Based Testing
Automated GUI Tests
Automated API Tests
Automated Integration Tests
Automated Component Tests
Automated Unit Tests

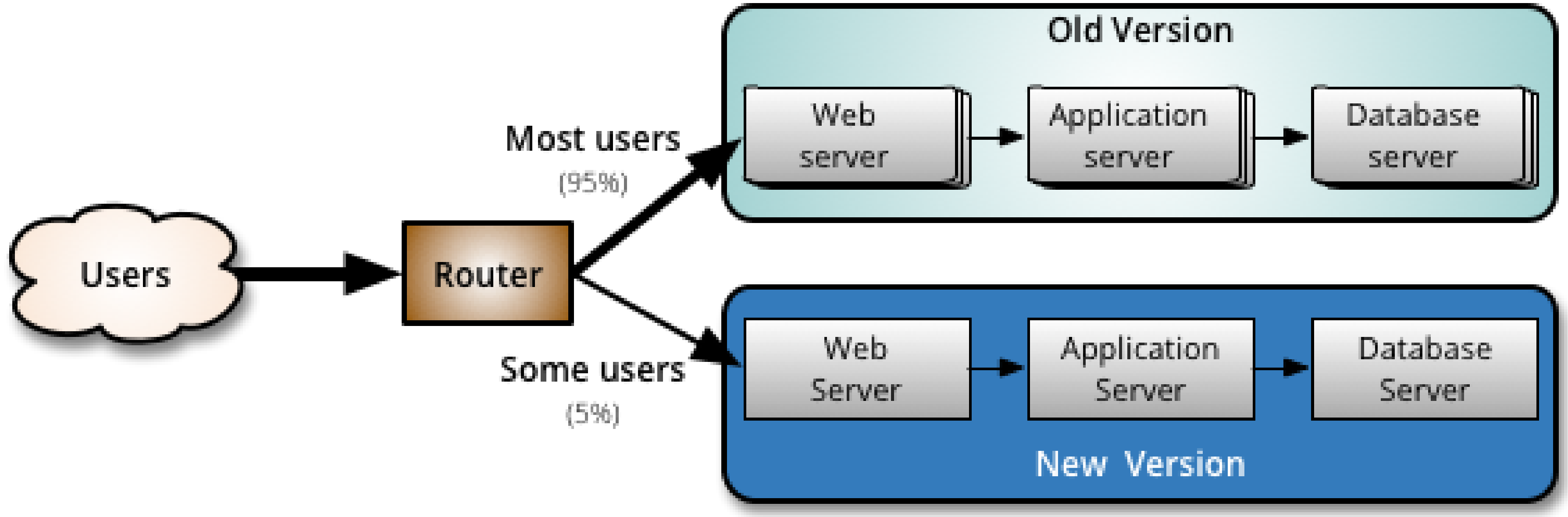
Deployment – Blue-Green

- Blue-Green
  - Copy of traffic to both servers during transition
  - Runs one or the other.



Deployment – Canary Deployment

- Canary Deployment
  - Sends only small amount of traffic to new version



Production

- The app is on the server, customers applaud, are we done?

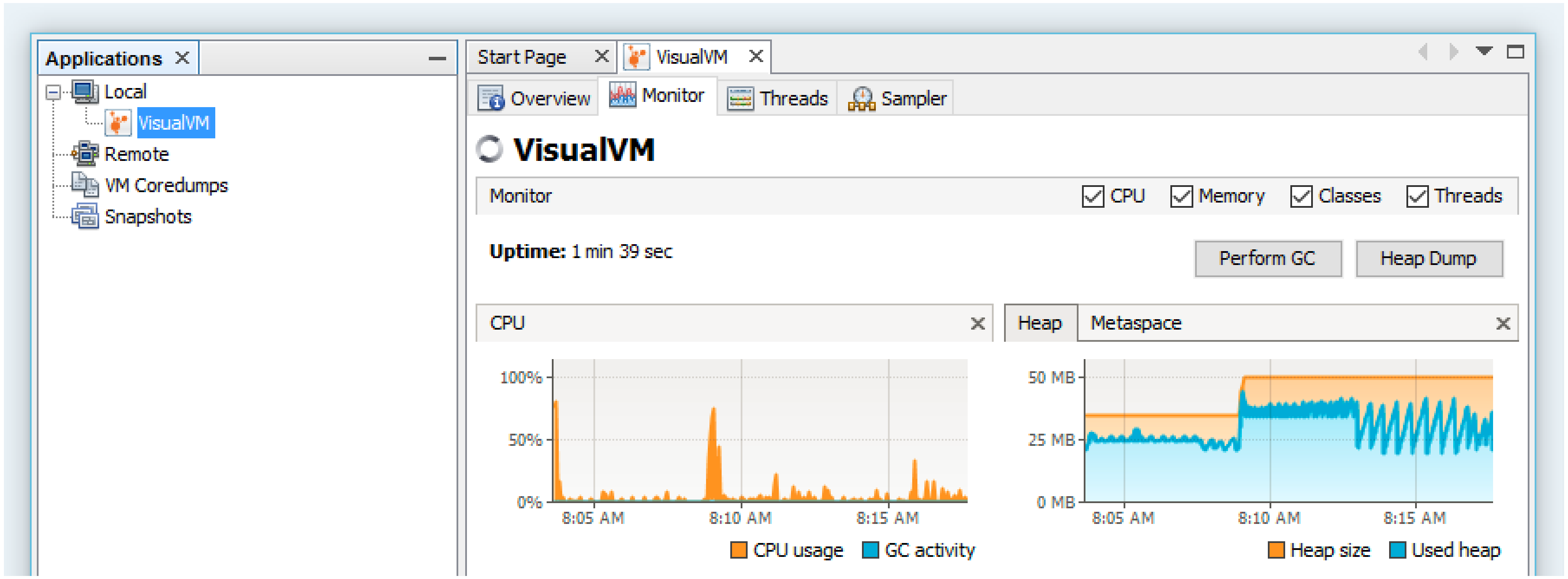
Production – Monitoring

- It’s important to monitor running application
  - Available memory
  - Exceptional states
  - Performance problems
- VisualVM – simple view of running JVM
- Wily – komplex system for JavaEE monitoring
- JProfiler, jhat

stringdata®

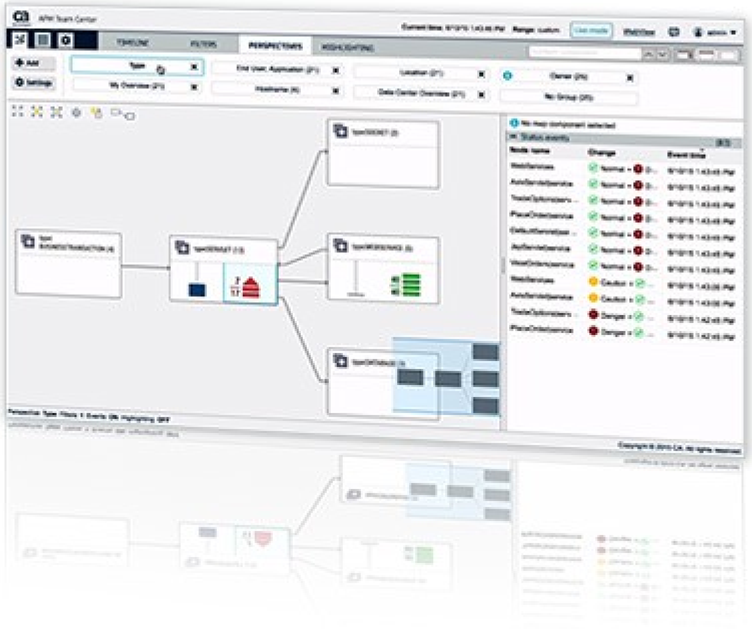
VisualVM

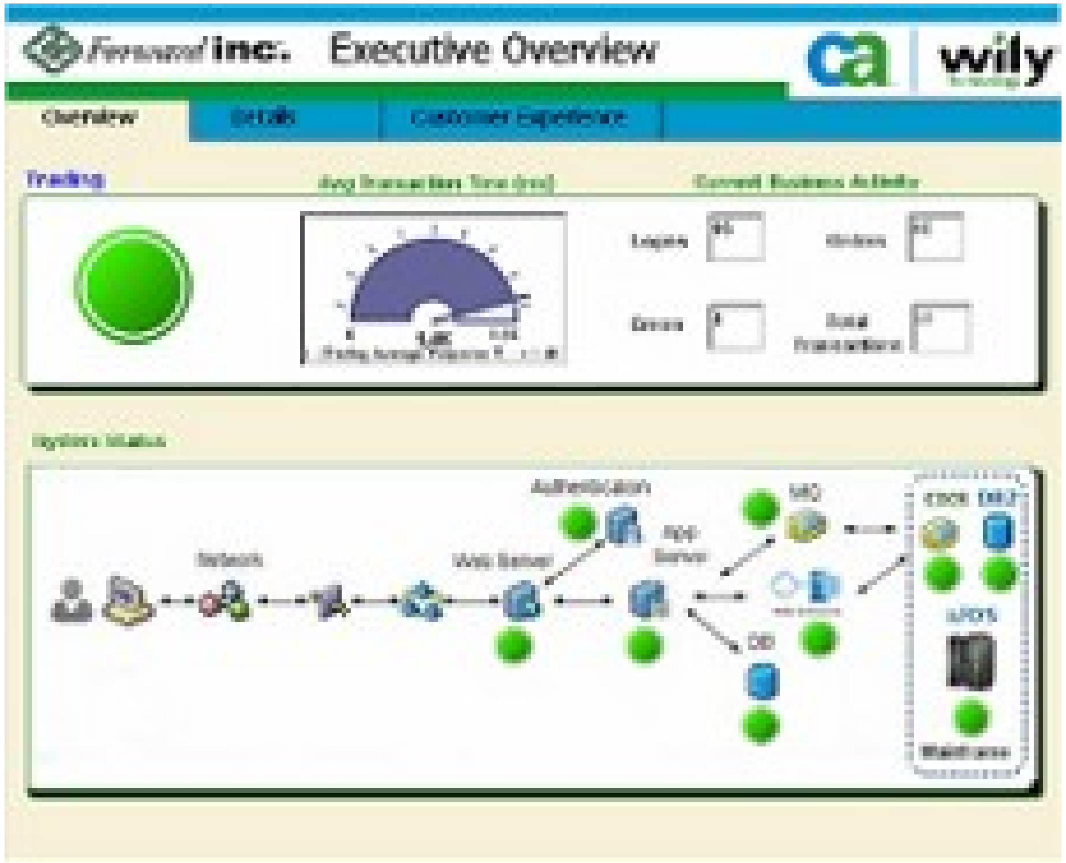
- VisualVM
  - Part of Java SE
  - Able to watch processes, memory, dumps



CA Vily Introscope

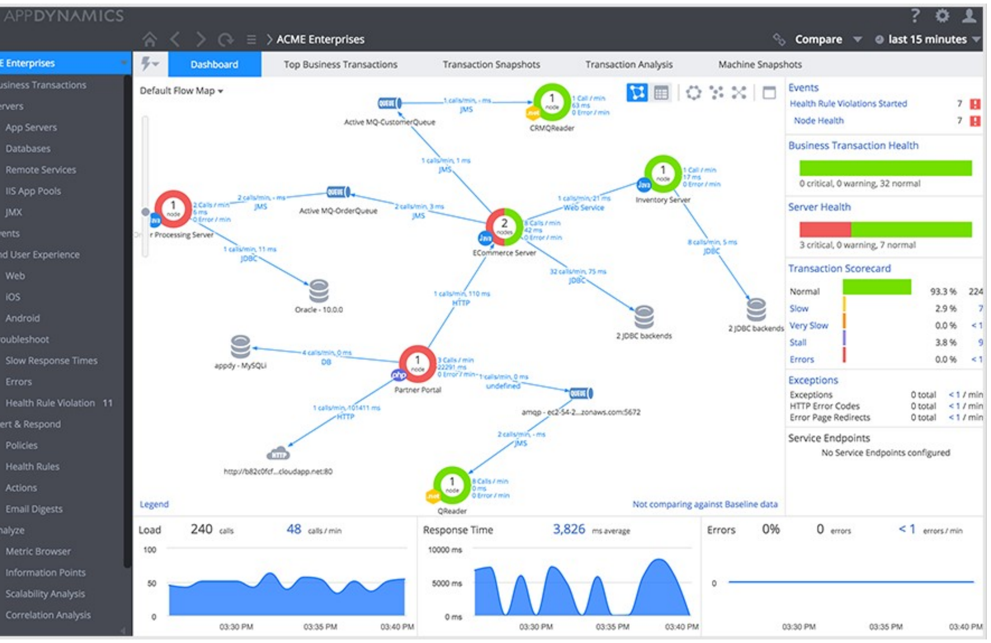
- Vily is very detail view into JavaEE
  – Video?

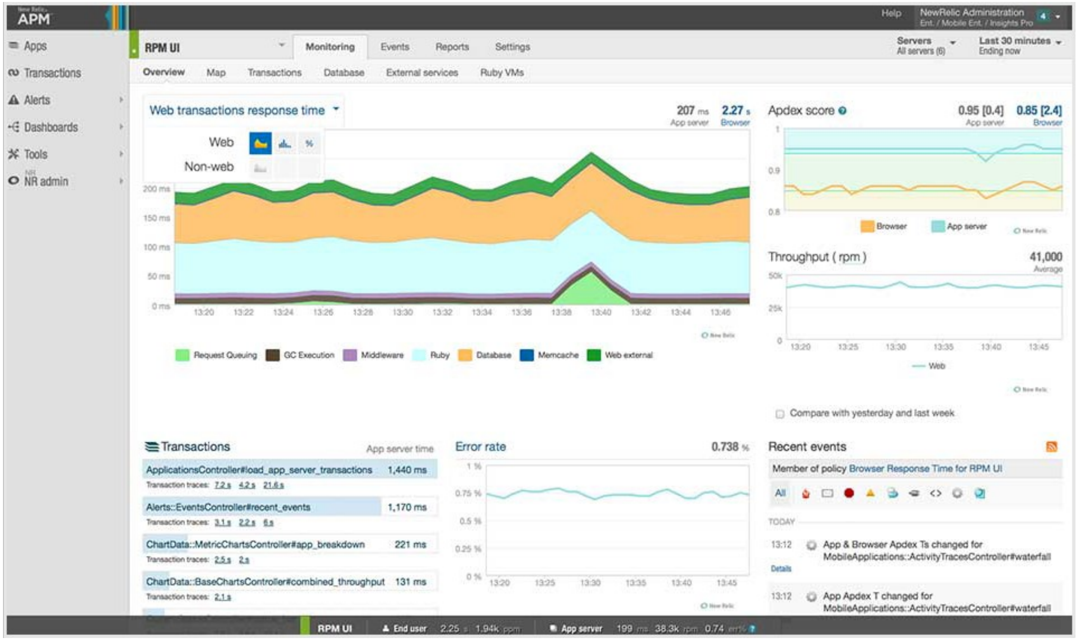




stringdata®

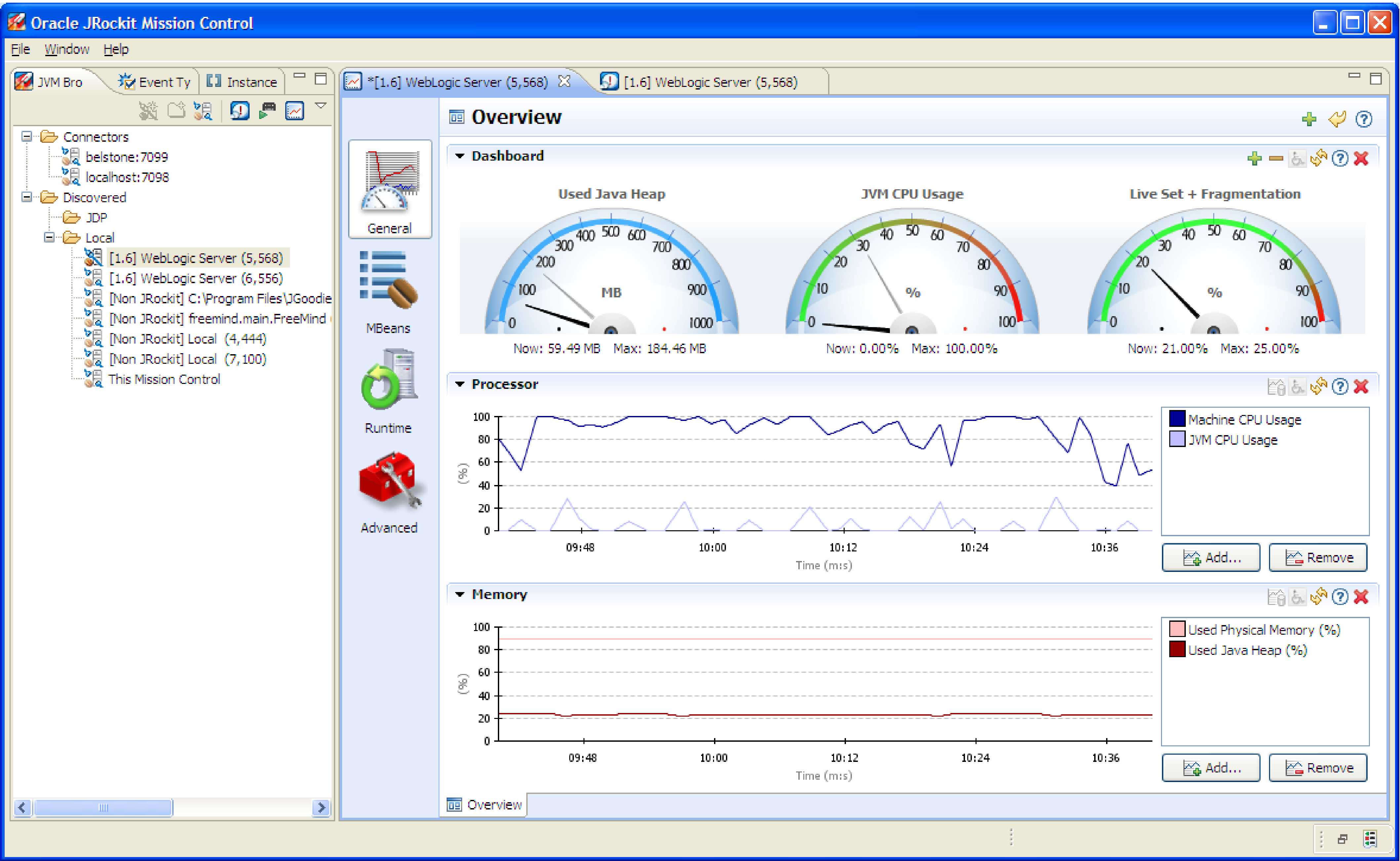
New Relic




APM
Apps
Transactions
Alerts
Dashboards
Tools
NR admin
RPM UI
Monitoring
Events
Reports
Settings
Overview
Map
Transactions
Database
External services
Ruby VMs
Web transactions response time
Web
Non-web
207 ms App server
2.27 s Browser
Request Queuing
GC Execution
Middleware
Ruby
Database
Memcache
Web external
Apdex score
0.95 [0.4] App server
0.85 [2.4] Browser
Throughput (rpm)
41,000 Average
Compare with yesterday and last week
Transactions
App server time
ApplicationsController#load_app_server_transactions 1,440 ms
Alerts::EventsController#recent_events 1,170 ms
ChartData::MetricChartsController#app_breakdown 221 ms
ChartData::BaseChartsController#combined_throughput 131 ms
Error rate 0.738 %
Recent events
Member of policy Browser Response Time for RPM UI
App & Browser Apdex Ts changed for MobileApplications::ActivityTracesController#waterfall
App Apdex T changed for MobileApplications::ActivityTracesController#waterfall
End user 2.25 s 1.94k ppm
App server 199 ms 38.3k rpm 0.74 err%

stringdata®

JRocket Mission Control



Performance Measurement

- Jmeter – the easy to use load generator
- At the end – performance is not always a priority. Why?

Citation from interview: “I prefer readable code over performance.”

Want performing code? Write it simple, readable.

Future Development

- Keep meaningful architecture, it makes sense
  - Direct access to database from multiple points is simple and tempting
    - In the future, synchronization will be huge problem
    - Intermediate layer keeping model and doing messaging
    - Example: EQUAL, manager of tests

My Own Experience

- MSM, Vantage
- Hotel planning support
- KNBox

Conclusion Question

- What technology would you choose NOW?
  Are you still confident with your favorite? Can you fulfill all requirements?
- Did you support any app for > 10 years? 15 years?
  Not very funny :-)
  My own example: not using Windows anymore, JDBC doesn’t support ODBC, lack of continuity.

Conclusion

- Review
  - TDD, jUnit as a part of build
  - Continuous deployment, functional tests
  - Careful deployment
  - Monitor in production

Thank you

Petr.Aubrecht@stringdata.cz