SAP JVM Internals

Volker Simonis, SAP SE
October, 2015

Public

Agenda

- SAP JVM
- Supportability
- SAP JVM Debugger
- SAP JVM Profiler

SAP JVM in a Nutshell

- Certified Java Virtual Machine supporting Java 4, 5, 6, 7 and 8
- Outstanding Platform Support
- Supportability Features and Tools (Monitoring/Debugging/Profiling)
- Developed since ~10 years by a team of ~20 in Walldorf/Germany
- Derives from SUN/Oracle Hotspot VM code base:
  - Complete source code (commercially licensed)
  - TCK (Technology Compatibility Kit)
  - Java Trademark usage
- „HotSpot Extreme“ Model (one VM for Java 4,5,6,7,8 and 9?)

Platform Support

Operating System < > Processor Architecture Matrix | ∑ 15 Supported Platforms

X86_64 X86 IA64	X86_64	X86_64 X86 IA64 PowerPC zSeries	X86_64 SPARC	PARISC IA64	PowerPC	PowerPC

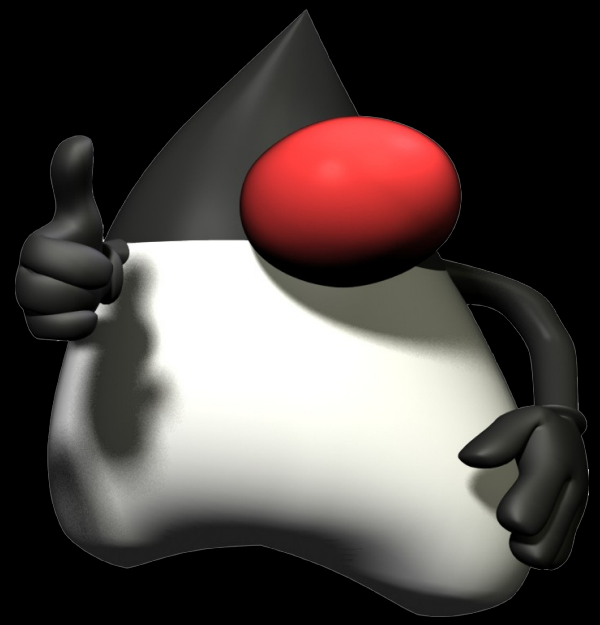
Why an own Java VM ?

- “Real” Platform Independence
  - Configuration (i.e. command line parameters)
  - JVM behavior (i.e. garbage collection, trace & error information)
  - JVM extensions, libraries (i.e. XML parsers, Security libraries)
  - Tools (i.e. Monitoring, Profiling, Debugging)
- Independence of other JVM vendors
  - Aligning maintenance periods of JVM and applications
  - Immediate bug fixing - no need for external support contracts
- Integrated offerings

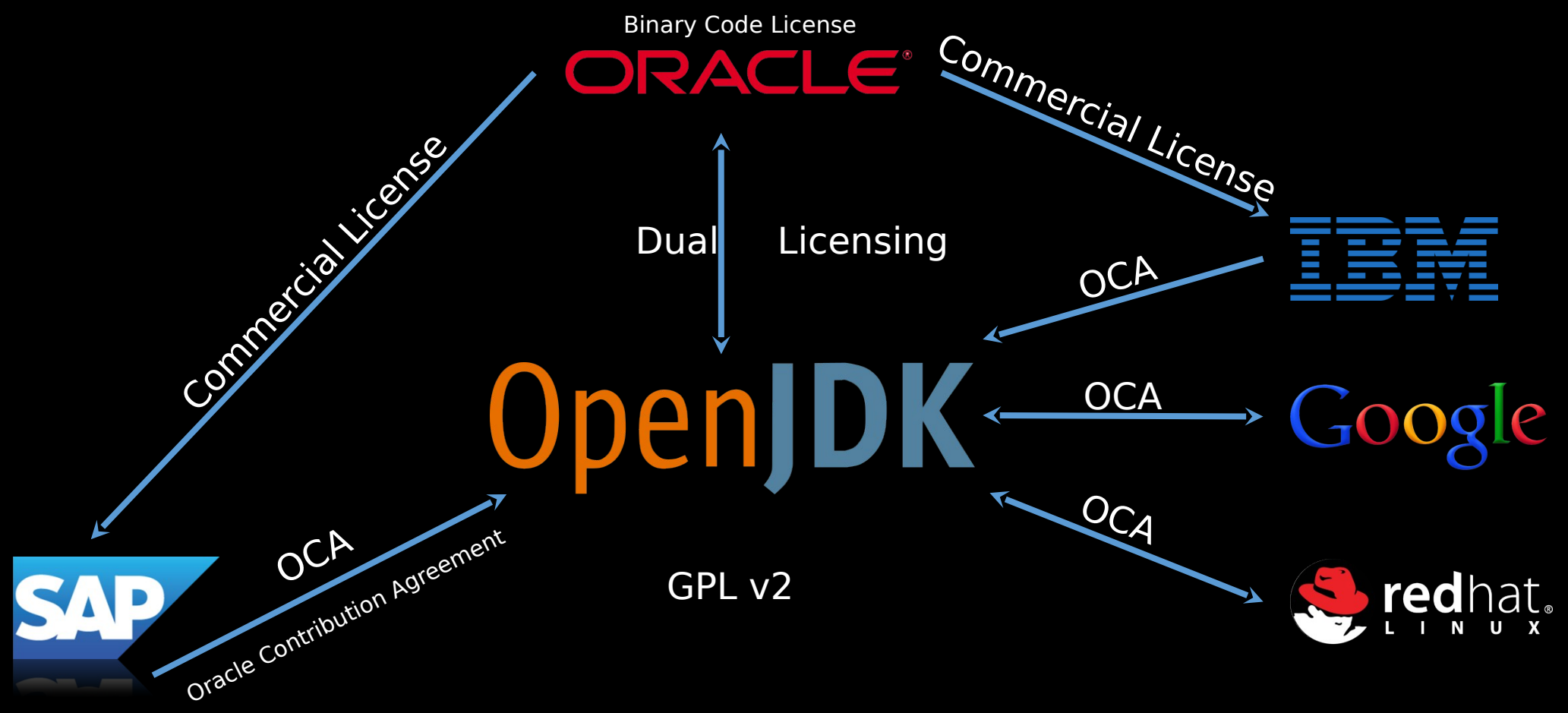
Product Support

- Standard Java VM for any Java Product shipped by SAP!

- SAP Hana Cloud Platform
- SAP NetWeaver
- SAP BusinessObjects
- SAP Business One
- Sybase
- Successfactors
- Hybris



OpenJDK / Oracle JDK / SAP JVM Licensing Model



Contributions to the OpenJDK

- PowerPC / AIX port
  - Together with IBM
  - Currently shipped by all major Linux distributions
- One Project Lead
- Two Reviewers
- One Committer
- Three Authors

} Hundreds of changes

- SAP is one of the biggest external OpenJDK contri

Power™

SAP JVM

SAP JVM

What is that?

SAP JVM is

- A certified Java Virtual Machine (JVM) for Java 1.4, 5, 6, 7, 8
- Available on all SAP-supported hardware and OS platforms
- Based on Oracle's Hotspot VM
- Enhanced with several supportability features

Team

~ 20 persons (developers & Q engineers), located in Walldorf, Germany

Used by

- All products based on SAP NetWeaver AS Java
- Business Objects XI
- Business One
- Business Connector
- HANA Cloud
- Sybase
- Success Factors
- Hybris

SAP JVM

Which OS platforms are supported?

SAP JVM runs on all 15 SAP-supported hardware / OS platforms

	Windows	Linux	SOLARIS	HP-UX 11i	AIX L	i5OS	MacOS
x86_32	**4 5 6 7 8**	**4 5 6 7 8**					
x86_64	**4 5 6 7 8**	**4 5 6 7 8**	**4 5 6 7 8**				**7 8**
SPARC			**4 5 6 7 8**				
PowerPC		**4 5 6 7 8**			**4 5 6 7 8**	**4 5 6 7 8**	
ia64	**4 5 6 7**	**4 5 6 7 8**		**4 5 6 7 8**			
z/Arch		**4 5 6 7 8**					
PA-RISC				**4 5 6**			

Java COMPATIBLE

7 platforms covered by Oracle

8 platforms added by SAP ℗ Oracle's HotSpot VM was ported by SAP

SAP JVM

Why an own Java VM?

Makes Java even more platform-independent, provides the same handling on all 15 platforms

- Configuration
- JVM behavior (i.e. garbage collection, trace & error information)
- JVM extensions, libraries (i.e. XML parsers)
- Analysis tools

Makes customers and SAP independent of other JVM vendors

- Aligns maintenance periods of JVM and applications
- Fix bugs immediately, no need for support contracts with JVM vendors

Integrated offering of applications, application server and JVM

- NetWeawer Application Server and SAP Hana Cloud both come pre-configured with the SAP JVM

Several enhancements provided by SAP

- Developer and supportability features
- Runtime improvements

Supportability Features

SAP JVM

Supportability Features

Informative thread dumps

- CPU, memory & I/O resource consumption information
- Detailed deadlock description
- Thread annotations – which user, application etc. is processed in a thread ?
- Thread dump redirection – store thread dumps in separate files

Supportability tool “jvmmon” & friends

Advanced monitoring API – thread & memory information for monitoring tools

Extensive garbage collection and thread dump traces with Graphical analysis

Traces for JVM experts

All the features are designed as interactive, on-demand facilities of the JVM. They can be switched on and off without restart.

SAP JVM Tools – jvmmon & Friends

Introduction

What is “jvmmon”?

It’s SAP JVM’s supportability tool, providing information about

- Command line parameters, JVM version, resource usage
- Thread stacks, deadlocks, resource consumption per thread
- Garbage collection runs
- Number & size of objects per class (class statistic)
- Text-based or GUI tool (“jvmmon-gui”)

Use it to

- *Print* information to the GUI / your console
- *Dump* information into the server’s trace files
- Start & stop debugging
- Trigger a garbage collection
- Write a heap dump file
- Switch JVM trace flags or JVM command line parameters (as far as technically possible)

Demo part I

The demos show a simple web application based on the Spring PetClinic

Here you will see:

- The application runs into an exception
- View the extended SAP JVM exception messages
- How to enable debugging in a running system on SAP JVM with jvmmon
- How the exception is fixed using Eclipse
- Generate deadlock in Vets list.
- Show the extended information in the thread dump.
- Show how to get a thread dump with jvmmon

SAP JVM Tools – jvmmon & Friends

How to use it? 1/2

Local scenario (JVM & jvmmon on the same machine)

You must use the same user account as the running JVM (security restriction) / or running as administrator

- On Windows you must be in the same Windows session

Start the jvmmon or jvmmon-gui executable, both located in sapjvm/bin

Select a JVM process (if there are multiple)

SAP JVM Tools – jvmmon & Friends

How to use it? 2/2

Remote scenario (JVM & jvmmon on different machines)

Start the jvmmon**d** executable (located in sapjvm/bin) on the same machine as the JVM(s)

- **Attention**: You open a security hole – everybody can connect to jvmmond
- You must use the same user account as the running JVM for jvmmond
- You may specify a port different than the default 1099 with the *–port* command line option

Start the jvmmon or jvmmon-gui executable on the remote machine

- Text-based jvmmon
  - Add the command line options *–hostname <host> –port <port>*
- GUI-based jvmmon
  - Press the *attach* button to attach to the remote machine
  - You can attach to multiple remote machines

SAP JVM – Supportability Features

Example: Additional resource information in Thread Dumps

```
"http-bio-9991-exec-10" daemon cpu=15.60 [reset 15.60] ms elapsed=10.02 [reset 10.02] s
          allocated=211152 B (206.20 KB) [reset 211152 B (206.20 KB)] defined_classes=0
io= file i/o: 0/0 B, net i/o: 891/12601 B, files opened:0, socks opened:0
          [reset file i/o: 0/0 B, net i/o: 891/12601 B, files opened:0, socks opened:0 ]
prio=6 tid=0x000000006255d800 nid=0x235c / 9052 runnable  [_thread_in_native (_at_safepoint),
stack(0x000000006b020000,0x000000006b120000)] [0x000000006b11f000]
   java.lang.Thread.State: RUNNABLE
   at java.net.SocketInputStream.socketRead0(Ljava/io/FileDescriptor;[BIII)I(Native Method)
   at java.net.SocketInputStream.read([BIII)I(SocketInputStream.java:150)
   - additional info (remote: WDFN00301656A/127.0.0.1:51609, local: localhost/127.0.0.1:9991)
   at java.net.SocketInputStream.read([BII)I(SocketInputStream.java:121)
   - additional info (remote: WDFN00301656A/127.0.0.1:51609, local: localhost/127.0.0.1:9991)
    at org.apache.coyote.http11.InternalInputBuffer.fill(Z)Z(InternalInputBuffer.java:516)
   at ...
```

SAP JVM – Supportability Features

Example: Deadlock info in thread dumps

```
 Found one Java-level deadlock:
"o1_before_o2" (tid=0x0000000062cb5000):
  waiting to lock monitor 0x000000006264fe10 (object 0x000000004df2ac88, a java.lang.String),
  which is held by "o2_before_o1" (tid=0x0000000062cb6000)
"o2_before_o1" (tid=0x0000000062cb6000):
  waiting to lock monitor 0x00000000632b81d8 (object 0x000000004df2acc0, a java.lang.String),
  which is held by "o1_before_o2" (tid=0x0000000062cb5000)

Java stack information for the threads listed above:
"o1_before_o2"(tid=0x0000000062cb5000):
  at org. … .web.VetController$1.run()V(VetController.java:100)
  - waiting to lock <0x000000004df2ac88> (a java.lang.String)
  - locked <0x000000004df2acc0> (a java.lang.String)
"o2_before_o1"(tid=0x0000000062cb6000):
  at org. … .web.VetController$2.run()V(VetController.java:111)
  - waiting to lock <0x000000004df2acc0> (a java.lang.String)
  - locked <0x000000004df2ac88> (a java.lang.String)
Found 1 deadlock

Threads (in)directly waiting on deadlocked thread "o1_before_o2" (tid=0x0000000062cb5000
"o1_only" (tid=0x0000000062cb6800
  waiting to lock monitor 0x00000000632b81d8 (object 0x000000004df2acc0, a java.lang.String),
  which is held by "o1_before_o2" (tid=0x0000000062cb5000)
```

SAP JVM

Developer Features

Integrated runtime & memory profiler backend

Debugging on demand

- turn on debugging on the fly (e.g. with the jvmmon tool)
- no performance impact if no debugging

Debuggable JDK classes

Detailed information for critical Java exceptions

- class cast
- no class definition found
- out of memory
- socket I/O

All the features are designed as interactive, on-demand facilities of the JVM. They can be switched on and off without restart.

SAP JVM – Developer Features

Example: Detailed NullPointerExceptions

Code

```
vets.getVetListN().addAll(this.clinicService.findVets());
```

HotSpot exception:

```
java.lang.NullPointerException: null

  at org.springframework.samples.petclinic.web.VetController.showVetListLast
                                                  (VetController.java:84) ~[classes/:na]
```

SAP JVM exception:

```
java.lang.NullPointerException: while trying to invoke the method
java.util.List.addAll(java.util.Collection) of a null object returned from
org.springframework.samples.petclinic.model.Vets.getVetListN()

  at org.springframework.samples.petclinic.web.VetController.showVetListLast
                                                  (VetController.java:84) ~[classes/:na]
```

SAP JVM – Developer Features

Example: Detailed NoClassDefFoundErrors

Oracle JDK

```
Exception in thread "main" java.lang.NoClassDefFoundError: Could not
initialize class UnloadableClass
  at java.lang.Class.forName0(Native Method)
  at java.lang.Class.forName(Unknown Source)
  at TryLoadUnloadableClass.main(TryLoadUnloadableClass.java:16)
```

SAP JVM

```
Exception in thread "main" java.lang.NoClassDefFoundError:
UnloadableClass : cannot initialize class because prior initialization
attempt failed
  at java.lang.Class.forName0(Native Method)
  at java.lang.Class.forName(Class.java:169)
  at TryLoadUnloadableClass.main(TryLoadUnloadableClass.java:16)
Caused by: java.lang.ExceptionInInitializerError
  at TryLoadUnloadableClass.doSomething(TryLoadUnloadableClass.java:6)
  at TryLoadUnloadableClass.main(TryLoadUnloadableClass.java:14)
Caused by: java.lang.ArrayIndexOutOfBoundsException: 3
  at UnloadableClass.<clinit>(UnloadableClass.java:7)
  ... 2 more
```

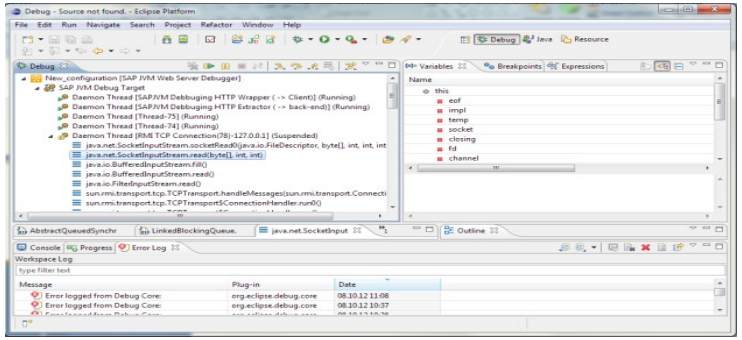
That‘s Cool in the SAP JVM

- **On Demand Debugging - No prerequisites, no restart required**
- **Detailed information for critical Java exceptions**
- **Powerful supportability tools**
- **68 times the very same VM**
  - **we take the “HotSpot Express” model to extremes!**

SAP JVM Debugger

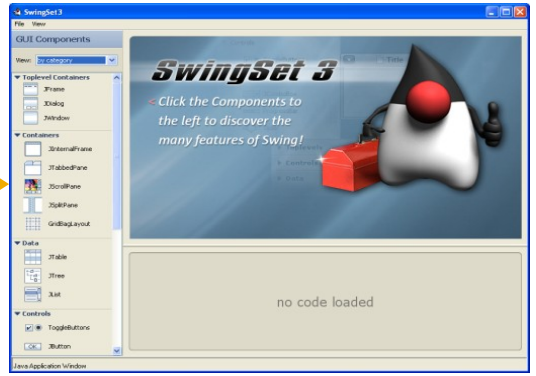
Common Java Debugging

Eclipse IDE



JDWP

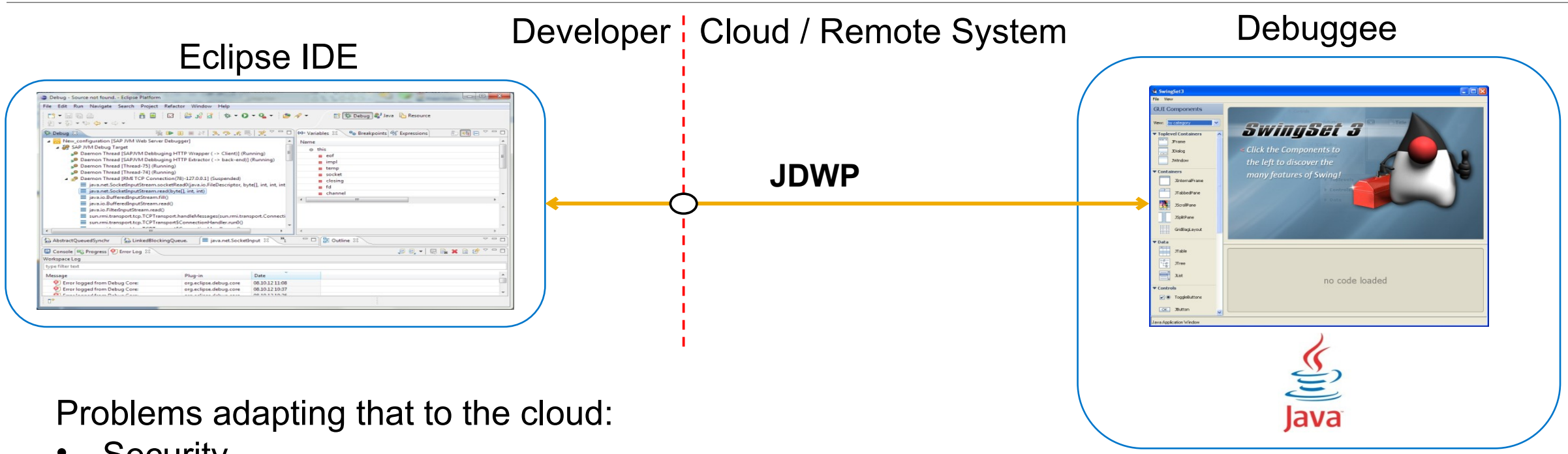
Debuggee



IDE local to Debuggee

- Debugger uses standard JDWP Protocol
- Simple tasks require many JDWP transactions

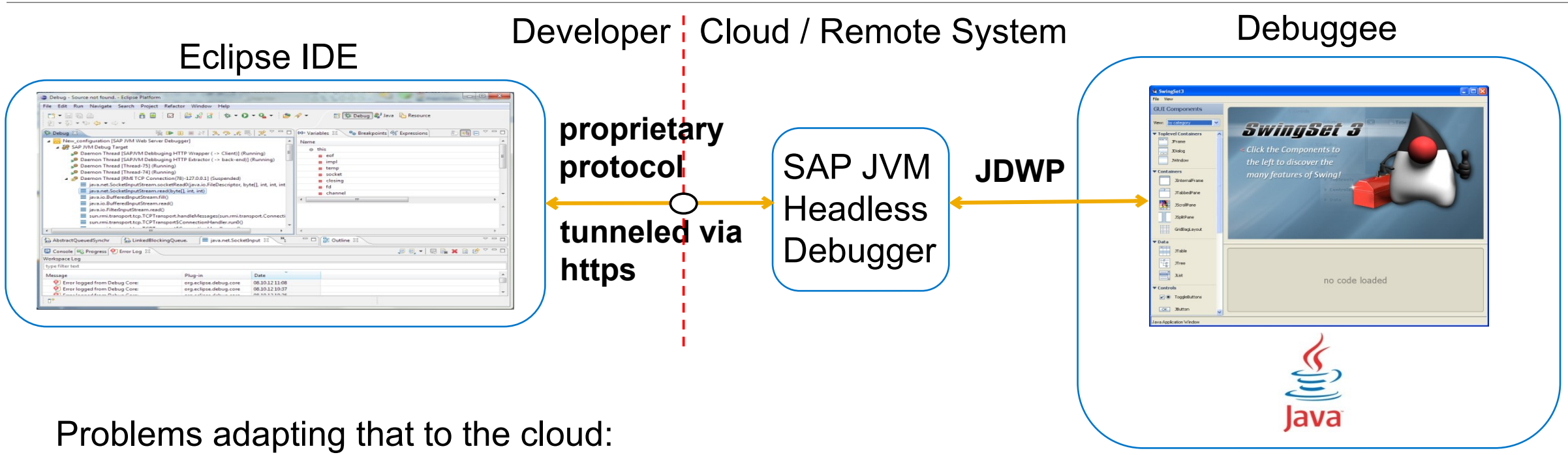
SAPJVM Headless Debugger



Problems adapting that to the cloud:

- Security
  - Unsecure port
  - Unencrypted communication
- Usability
  - Unbearable response time because of chatty JDWP protocol in conjunction with the latency

SAPJVM Headless Debugger



Problems adapting that to the cloud:

- Security
  - Solved by tunneling over https
- Usability
  - Low response times by aggregating information in proprietary protocol

That‘s Cool in the SAP JVM Debugger

- Enables SAP Hana Cloud Debugging
- Fully integrated into Eclipse
- As common for all our tools
  - On Demand Debugging
  - On Demand Profiling

SAP JVM Profiler

Introduction
Profiling Objectives

„Why should I profile?“

To avoid performance problems

Long-running code

- Non-linear algorithms, expensive framework or class library calls
- Redundant execution of code or unexpected program flow (exceptions)

Wait situations in parallel processing

- Linearization on hot locks, deadlocks and livelocks

I/O bottlenecks

- Extensive file or network I/O, waiting for network responses

To avoid memory bottlenecks

Memory thrashing

- Expensive framework or class library calls
- Sub-optimal application code

Memory leaks

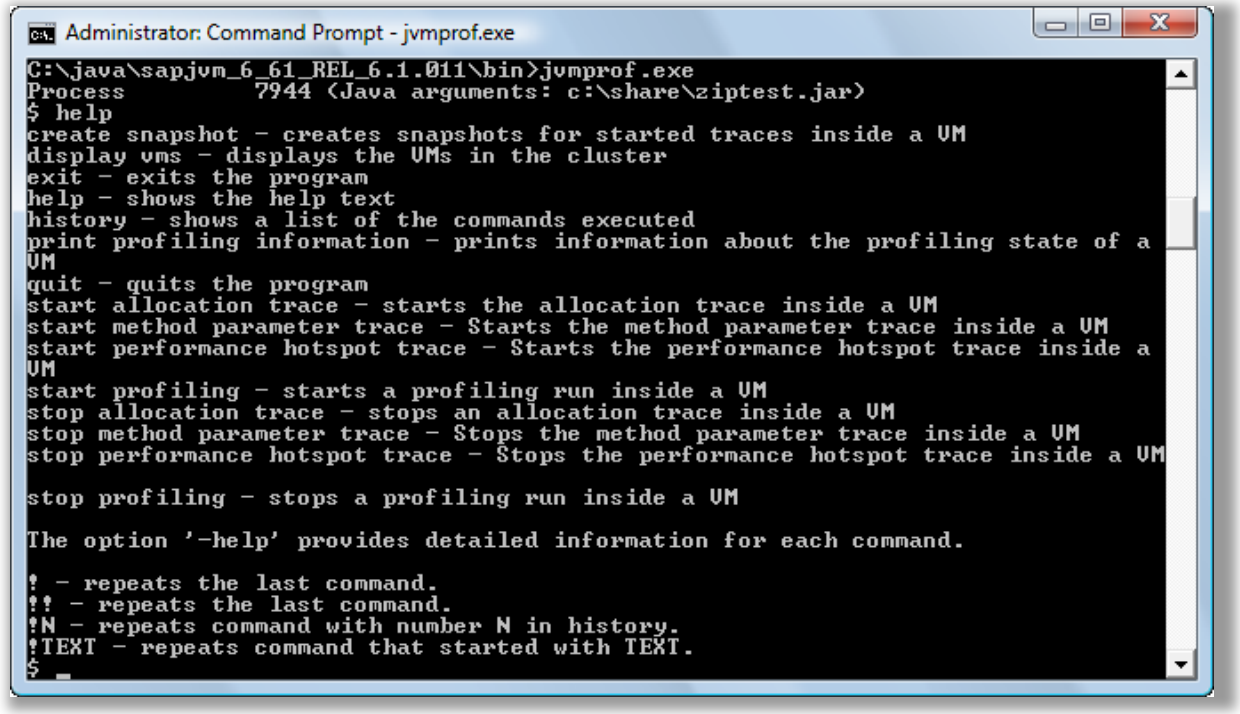
SAP JVM Profiler

Available Metrics

	Performance Hotspot Analysis	Allocation Analysis	Method Parameter Trace	File/Network I/O Analysis	Synchronization Analysis
Running time	✓				
Sleeping time	✓				
CPU time			✓		
Method call counts	with method parameter tracing	with method parameter tracing	✓		
Allocated memory		✓			
Allocated objects		✓			
I/O time				✓	
I/O data volume				✓	
Blocked time					✓
Blocking time					✓

Tools

jvmprof



Scriptable command line tool

Features

Start & stop of

- Performance analysis
- Memory allocation analysis
- Method parameter tracing
- File/Network I/O analysis
- Synchronization analysis

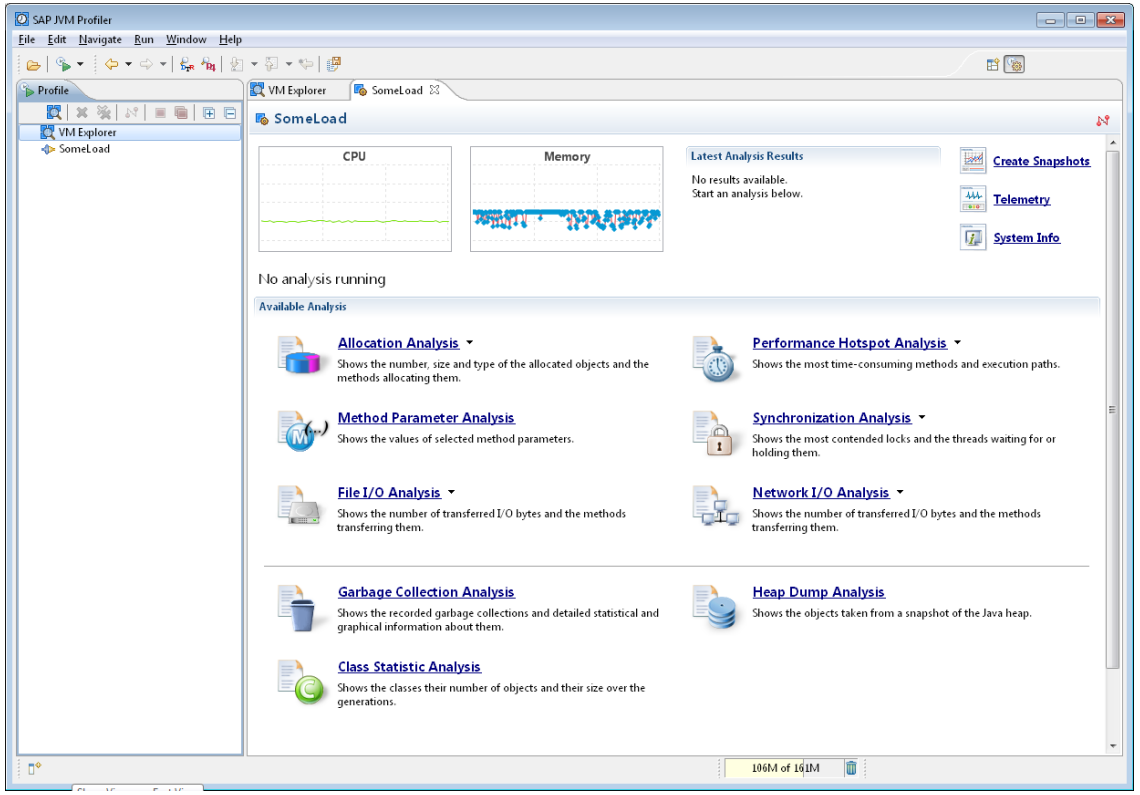
No result analysis, SAP JVM Profiler has to be used

Included in SAP JVM deliverable at **sapjvm/bin**

Tools

SAP JVM Profiler

Comprehensible tool to collect and analyze profiling data



Features

- Performance analysis
- Memory allocation analysis
- Method parameter tracing
- File/Network I/O analysis
- Synchronization (wait situation) analysis
- Comparison of multiple result sets
- Online monitoring of JVM resources (CPU, memory, GC activity)
- Eclipse/NetWeaver Developer Studio plug-in
- No setup on Java applications

Demo part II

Here you will see

- How to do profiling on demand with a running system

That‘s Cool in the SAP JVM Profiler

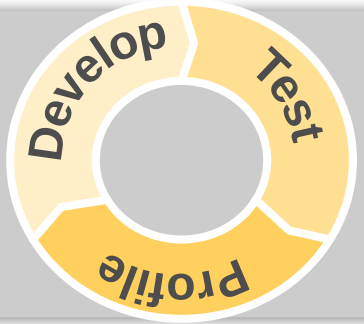
- On Demand Profiling; No prerequisites, configuration, or restart required – just start it on demand
- No overhead if profiling is switched off, moderate overhead when switched on
- Works reliably with large Java applications
- Time based sampling
- Integrated with SAP NetWeaver Developer Studio and Eclipse
- Profiling safepoint / JIT recompilation on demand
- Deeply integrated in the JVM
- Reliable GC Analyze
- Method Parameter Trace
- Fully integrated Memory Analyzer

Introduction

When to Profile ?

Development
Unit Testing

- Measure critical algorithms
- Investigate new frameworks used



Scenario Tests
Load Tests

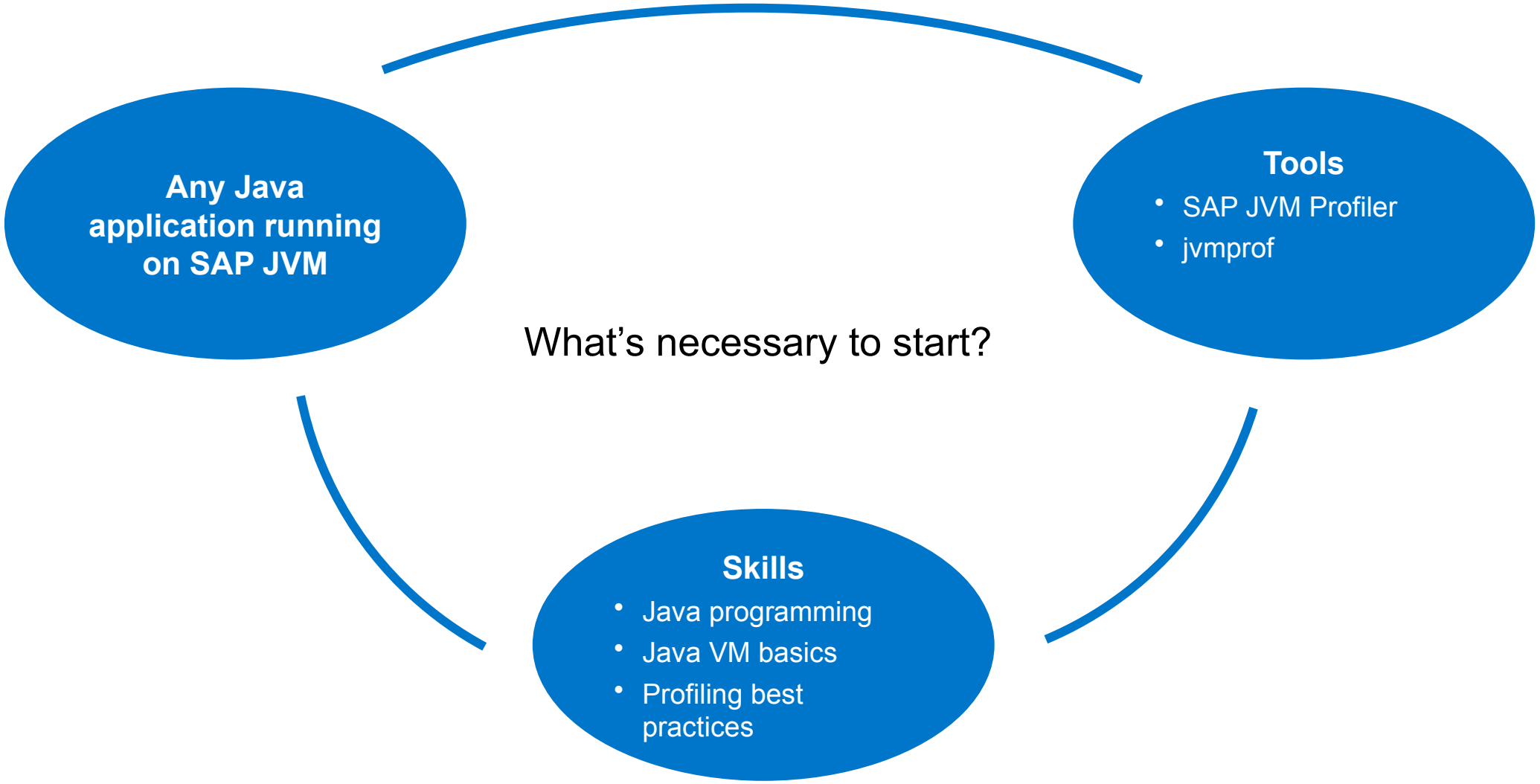
- Analyze performance of time-critical scenarios
- Investigate batch-type processes with non-linear runtime
- Check for wait situations
- Watch out for memory leaks and check memory consumption

Production

- Investigate performance or memory issues not reproducible in QA systems

Introduction

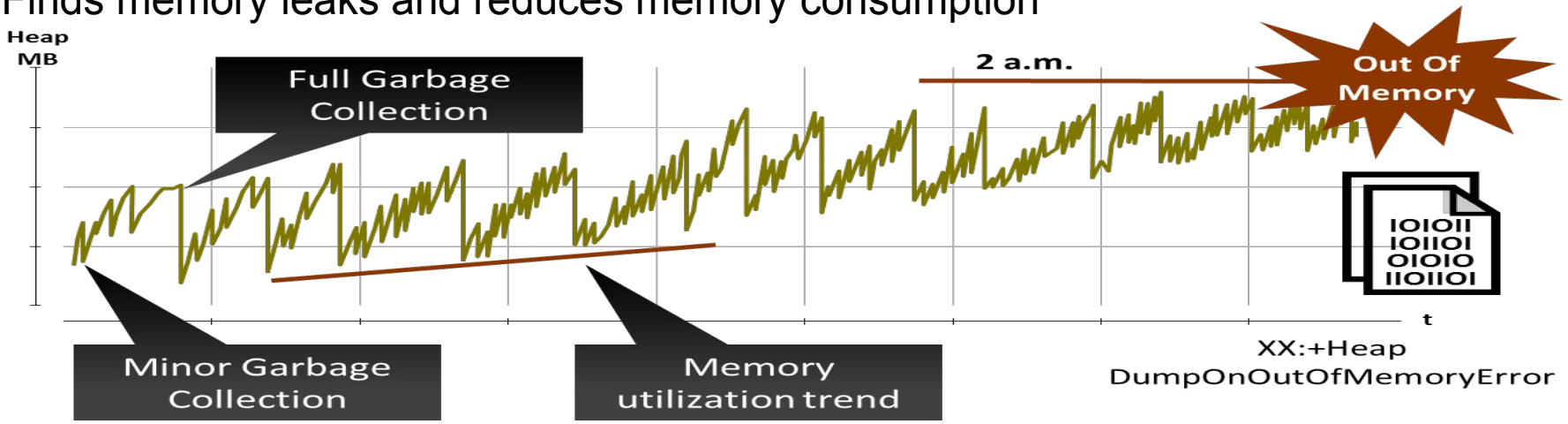
Profiling Setup



SAP Memory Analyzer

Introspects Java Heap Dumps

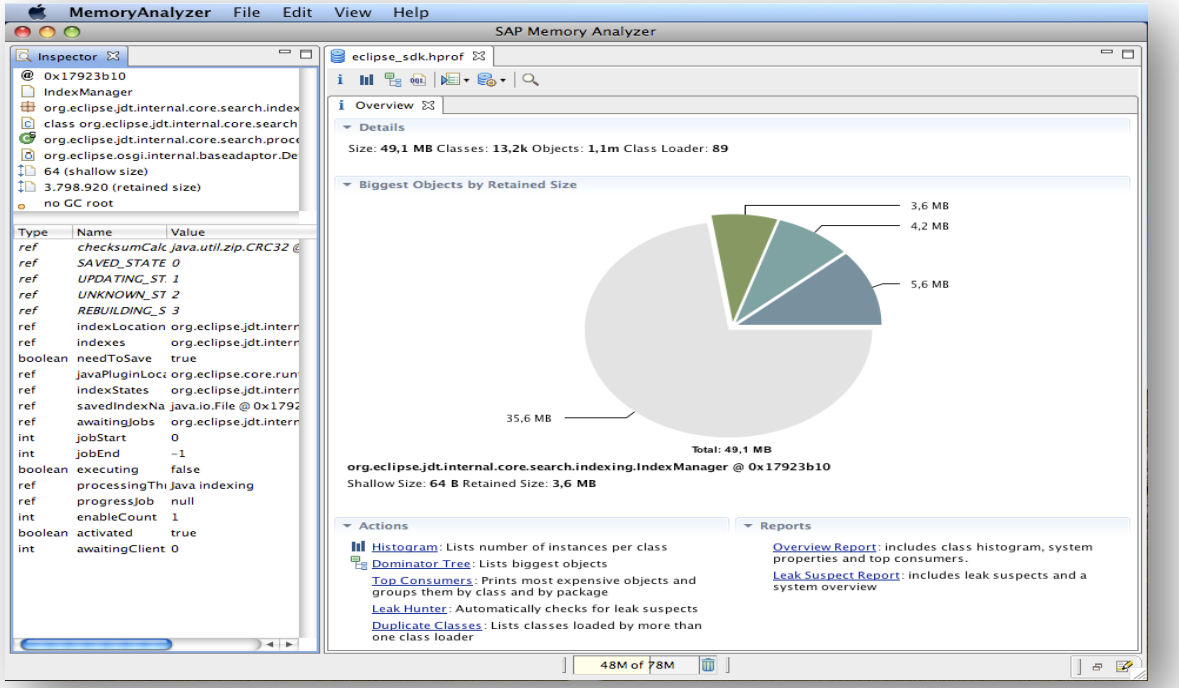
- Finds memory leaks and reduces memory consumption



Developed together with IBM as Eclipse Open Source Project

- http://www.eclipse.org/mat

Memory Analyzer Key Features



Reporting

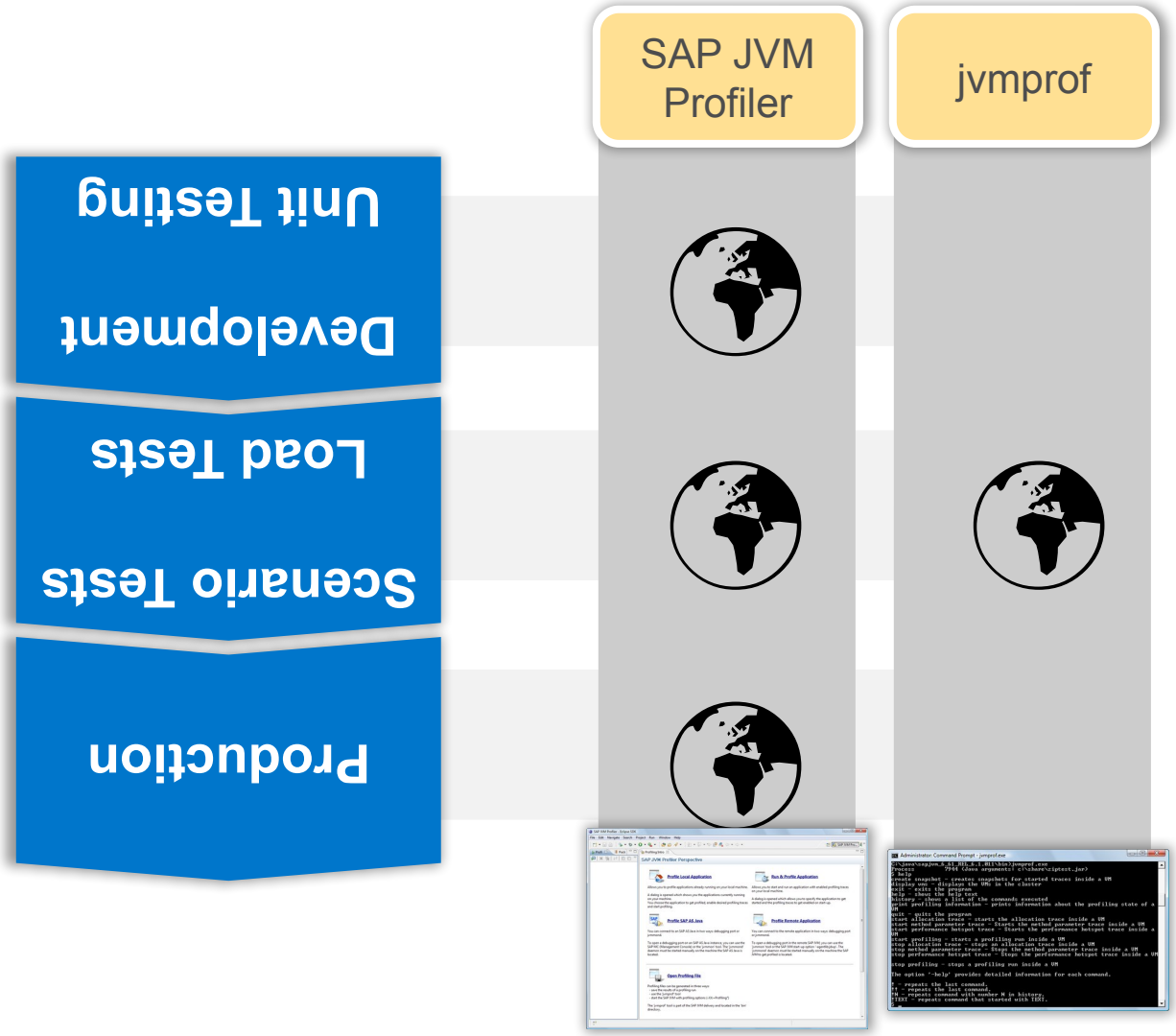
- Automation: Generates HTML Reports with charts and suggestions
- In depth follow up analyses with Eclipse RCP tool possible

Expert System Knowledge

- Millions of objects: Where is the „needle“?
- Complexity analysis and expert system knowledge
  - Reports on memory leak suspects and checks for known anti-patterns
- Application specific know-how
  - Meaningful names for class loaders and knowledge of programming models (OSGi & JRuby)
- Identifies regressions by comparing heap-dumps

Tools

What to use when?



SAP JVM Profiler and jvmprof

- Analyze memory footprint
- Identify runtime bottlenecks
- Find the culprits

Demo part III

Here you will see:
Evaluation of the collected profiling data: GC statistics

Further Information

SAP Community Network (SCN):

SAP JVM: http://wiki.scn.sap.com/wiki/display/ASJAVA/SAP+JVM
SAP JVM Profiler: http://wiki.scn.sap.com/wiki/display/ASJAVA/Java+Profiling

SAP Hana Cloud Platform (HCP):

SAP JVM download: https://tools.hana.ondemand.com/#cloud
SAP JVM Profiler & Tools (Eclipse download sites):
https://tools.hana.ondemand.com/mars
https://tools.hana.ondemand.com/luna

Thank you

Volker Simonis, SAP SE
volker.simonis@sap.com