FAKULTÄT FÜR !NFORMATIK

Diplomstudium:
Informatik

Diplomarbeitspräsentationen der Fakultät für Informatik

Benchmarking of Middleware Systems

Bernhard Löwenstein

Technische Universität Wien
Institut für Computersprachen
Arbeitsbereich: Programmiersprachen und Übersetzerbau
Betreuerin: Ao.Univ.Prof. Dipl.-Ing. Dr. eva Kühn

1. Challenge

Benchmarking of diverse middleware systems (MozartSpaces, GigaSpaces XAP and JBoss AS) with regard to performance and scalability

2. Performance vs. Scalability

Performance:

- Measuring the time how long the execution of a specified number of actions will take
- Returns a duration as objective indicator, also referred to as execution time

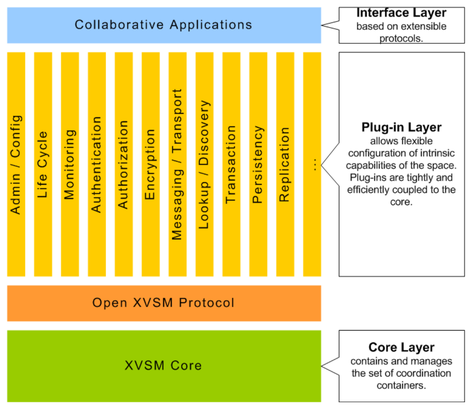
Scalability:

- Calculating the ratio between the time required for executing a problem on a machine and the time required for executing the quasi-same problem, but with n-fold problem size, on the quasi-same machine, but with n-fold resources
- Returns a number as objective indicator, also referred to as efficiency

3. Middleware Systems

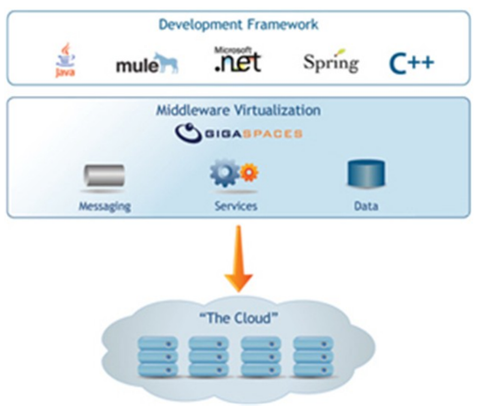
MozartSpaces – eXtensible Virtual Shared Memory (XVSM):

- Reference implementation of XVSM which allows the extension of the basic system through the usage of aspects
- Autonomous workers can share a container to interact among each other in a loosely-coupled way (peer-to-peer approach)
- XVSM API enables access to containers



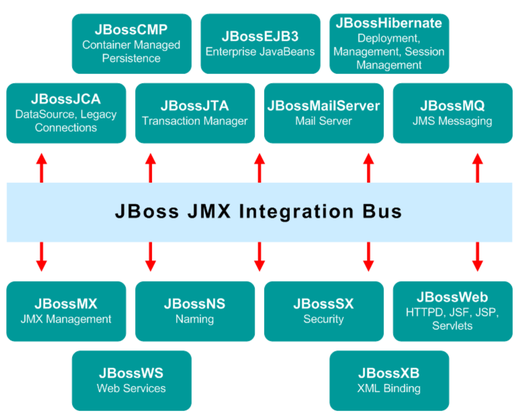
GigaSpaces eXtreme Application Platform (XAP) – JavaSpaces:

- Introduces the virtualization on middleware level by replacing the centralized messaging, data and service implementations with a virtual layer (“the cloud”)
- Autonomous workers can share a data space to interact among each other in a loosely-coupled way (peer-to-peer approach)
- JavaSpaces API enables access to data spaces

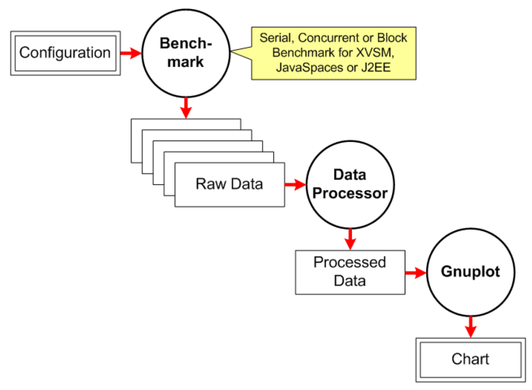


JBoss Application Server (AS) – Java 2 Platform, Enterprise Edition (J2EE):

- Application server which fully complies with J2EE specification and is made up of several JMX modules which allow the system's extension and adaption
- Clients can use various containers and services provided by the server for fulfilling their tasks (client-server approach)
- Miscellaneous Java APIs enables access to containers and services



4. Benchmark Procedure



1st Step:

- Definition of the benchmark scenario (serial, concurrent or block benchmark) based on the overlapping operations (write, shift, read, take and destroy)

Iteration	Entries (before)	Operations (measured)	Entries (after)
1	0	1K write	1K
2	1K	1K write	2K
...	...	...	...
n	1K · (n – 1)	1K write	1K · n
...	...	...	...
59	58K	1K write	59K
60	59K	1K write	60K
TOTAL		**60K write**	

2nd Step:

- Execution of the defined benchmark scenario using the benchmark suite which results in several data files

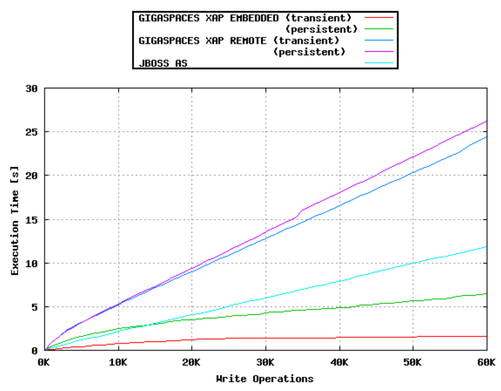
```
# ================================================
# BENCHMARKSERIAL - XVSM [MozartSpaces]
# ================================================
#
# ------------------------------------------------
# Data
# ------------------------------------------------
0 0
10000   1316
20000   2370
30000   3389
40000   4410
50000   5446
60000   6455
70000   7480
80000   8497
90000   9540
100000  10627
```

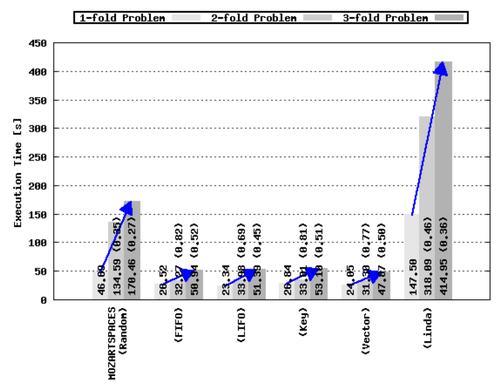
3rd Step:

- Processing of the data using the data processor

4th Step:

- Visualization of the data using Gnuplot which allows the generation of charts via scripts





5. Conclusion

MozartSpaces:

- Performance and scalability depends highly on the used coordinator and selector and can be rated in summary as poor

GigaSpaces XAP:

- Performance and scalability depends on the executed operations and can be rated in summary as mean
- Writing data into space performs and scales well, but reading or taking data from space do not

JBoss AS:

- Performance and scalability can be rated for all implemented benchmark scenarios as excellent
- Benefits most from additional computing resources of all benchmarked systems