FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Diplomarbeitspräsentation

Design and Implementation of the next Generation XVSM Framework

Operations, Coordination and Transactions

Masterstudium:
Computational Intelligence

Martin-Stefan Barisits

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

Context

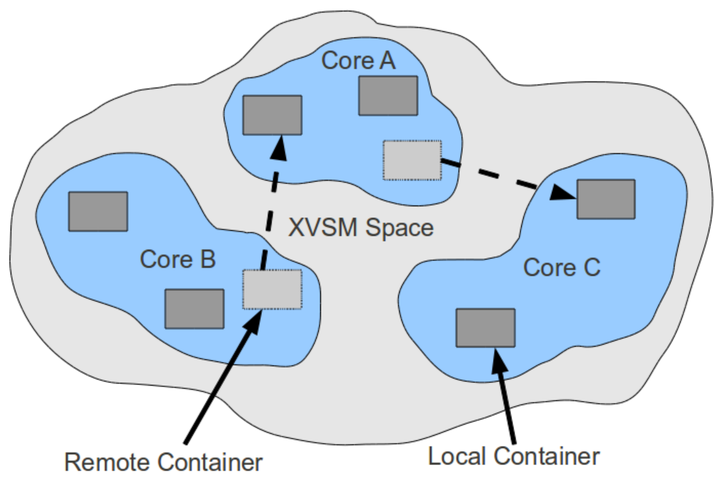
Developing distributed systems is a difficult tasks as many aspects have to be considered by the application programmer:

- Concurrency
- Transactional safety
- Scalability
- Collaboration of peers
- Coordination of processes
- Network communication

Solved by

eXtensible Virtual Shared Memory

The XVSM (eXtensible Virtual Shared Memory) concept is a framework based on the Space-Based Computing paradigm, which supports the developer to solve this architectural style. XVSM enables partners in a peer-to-peer infrastructure to collaborate with each other in an intuitive and efficient way. To this purpose it introduces different coordination patterns which dissolve most tasks of modern distributed applications.



Represented by

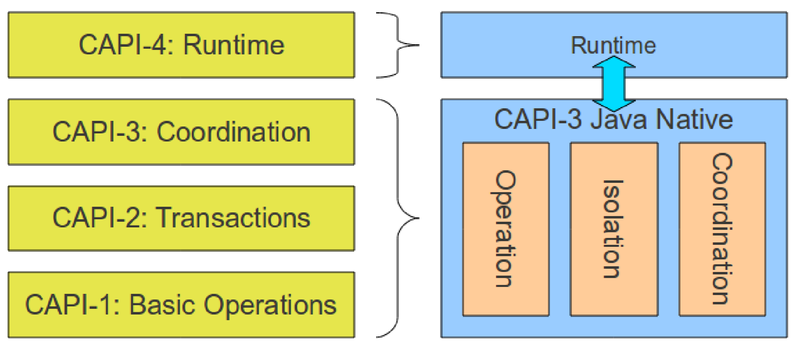
With XVSM so called coordinators can be associated to containers which are responsible for the specialized behavior of how data is stored and accessed by the user. For example, a coordinator which tracks the entries in a FIFO like queue and a coordinator which assigns labels to entries can both be active at one container.

MozartSpaces

This thesis introduces MozartSpaces, the Java-Native architecture of XVSM, which had to solve the following challenges:

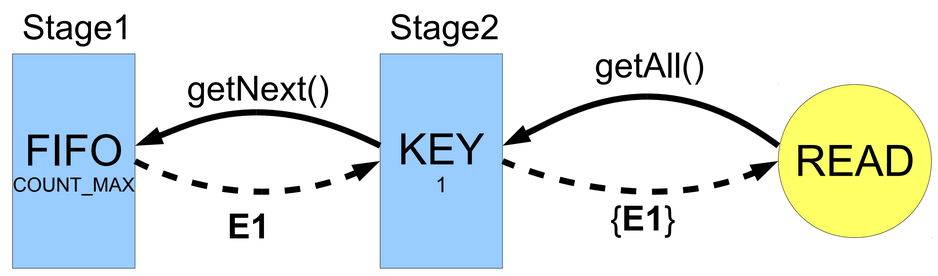
- Compliance to the XVSM formal model
- A modular architecture for high flexibility
- A highly concurrent data structure
- An agile locking mechanic
- A Stream-driven coordination concept for high performance

Modular Architecture



- **Operation Module:** Provides container and space operations
- **Isolation Module:** Provides transactional isolation with multiple isolation levels
- **Coordination Module:** Stream-driven coordination patterns for containers

Stream-Driven Coordination



The proposed coordination concept is based on streams, where each stage passes intermediate results on to the next stage, right at the moment when they are available, instead of waiting for the computation of a complete result set. This strategy enables the core to finish operations at the earliest moment, instead of executing unnecessary computations which are not relevant for the operations result set.

Results

	GigaSpaces	Previous MozartSpaces	MozartSpaces CAPI-3 H2	new MozartSpaces CAPI-3
FIFO	-	6.9s	1140s	2.4s
LIFO	-	6.9s	1131s	2.4s
RANDOM	-	6.8s	>1000s	2.3s
KEY	-	7.3s	>1000s	2.4s
LABEL	-	7.1s	-	2.3s
VECTOR	-	7.1s	-	2.4s
LINDA	1.4s	6.9s	-	2.4s
QUERY	-	-	-	2.4s
FIFO + LABEL	-	7.9s	-	3.8s
LIFO + LINDA + VECTOR	-	14.3s	-	7.6s
NOOP	-	6.1s	-	2.3s

Write benchmark

	GigaSpaces	Previous MozartSpaces	MozartSpaces CAPI-3 H2	new MozartSpaces CAPI-3
FIFO	-	0.003s	136s	0.005s
LIFO	-	2.2s	135s	0.005s
RANDOM	-	4.2s	82s	1.9s
KEY	-	0.006s	150s	0.005s
LABEL	-	0.009s	-	0.005s
VECTOR	-	0.026s	-	0.006s
LINDA	34s	7.9s	-	2.03s
QUERY	-	-	-	490s[1]
FIFO + LABEL[2]	-	71s	-	0.015s
LIFO + LINDA + VECTOR[3]	-	>500s	-	0.014s
NOOP	-	0.003s	-	0.003s

Read benchmark

The execution-time evaluation in contrast to other middle-ware systems shows, that the proposed isolation concept is the critical property responsible for these performance results.

In respect to more complex staged selection queries, the stream-driven coordination concept also shows excellent results.

Conclusion

- Modular architecture
- Stream-driven coordination
- High concurrency
- Multiple isolation levels
- Decrease of complexity
- Robust system design

Kontakt: mb@complang.tuwien.ac.at