FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Diplomarbeitspräsentation

XVSM Persistence - Developing a functional profile for the eXtensible Virtual Shared Memory

Thomas Meindl

Masterstudium:
Software Engineering & Internet Computing

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

Context

The eXtensible Virtual Shared Memory (XVSM) is a coordination-focused peer-to-peer middleware based on the distributed shared memory paradigm. XVSM also offers an easy extension mechanisms.

Motivation

- **Develop an orthogonal functional profile for XVSM.** A functional profile is an extension to XVSM, that enables peers to inject code into the XVSM space and therefore is loosely coupled to the XVSM core. XVSM Persistence should be a **proof of concept** of this design pattern.
- The main feature of XVSM Persistence is to **enrich XVSM with persistency** as a functional profile. The persistency feature addresses two issues: persistence semantics and recovery:
- Several **persistence semantics** are valid within a space based middleware, when client peers want to make the effects of space operations durable. These semantics need to be defined and research needs to be done on other space based architectures (SBAs), that too feature persistency.
- **Recovery** is done by restore routines, that are used to retrieve the persisted data after a system crash.
- The implementation of XVSM Persistence should offer **pluggable persistence providers**, i.e. the storage type, that XVSM Persistence uses is interchangeable.

Persistency characteristics

- Fulfill Reliability, Security, Modularity and ACID properties.
- **Persistence models and policies:** A persistence model can be seen as an abstract definition, that decides what to persist from the space. Whereas a persistence policy is a concrete implementation of a persistence model.
- **Synchronous and asynchronous persistency:** With asynchronous persistency, the persistable data is given to an external component, that persists this data at some point in the future asynchronously. Synchronous persistency persists data immediately after every data modifying operation.
- **Explicit and implicit persistence:** Explicit persistence lets the user decide when to use certain persistence features. Whereas implicit persistence hides the persistence configuration from the user - it is defined by the space itself.

Related Work

	P.Models	Sync.	Async.	Explicit	Implicit
Persistent Linda		X			X
Outrigger		X			X
Blitz		X	X		X
GigaSpaces	partly	X	X		X
SQLSpaces		X			X
RDBSpace		X			X
TSpaces		X			X
TuCSoN	partly	X		X	X

Table: Persistence support in space based frameworks

Persistable entities and space operations

- Persistable entities: containers, entries and aspects
- Space modifying operations: container-related (create, destroy), entry related (read, write, take, destroy), aspect related (add, remove)

Persistence Models

A persistence model is a concept, that defines which entities are persisted inside a persistent space with the use of which space operations. **Four different persistence models** have been identified, where persistence is tied to

(A) the whole space, i.e. every entity and every operation.
(B) containers, i.e. all entities of a persistent container.
(C) transactions, i.e. all entities within a persistent transaction.
(D) operations only, i.e. all entities of a persistent operation.

Design and Architecture

XVSM Persistence installs aspects in the space, that synchronize XVSM entities with a database. Those **persistence aspects** include the following XVSM Persistence components:

- The **TransactionMapping** component synchronizes an XVSM transaction with a database transaction.
- The persistable entries are transformed into **HolderObjects** with the help of the **ModelMapper**.
- The **PersistenceProvider** is designed as a database independent component that processes and stores **HolderObjects** in a persistent storage.

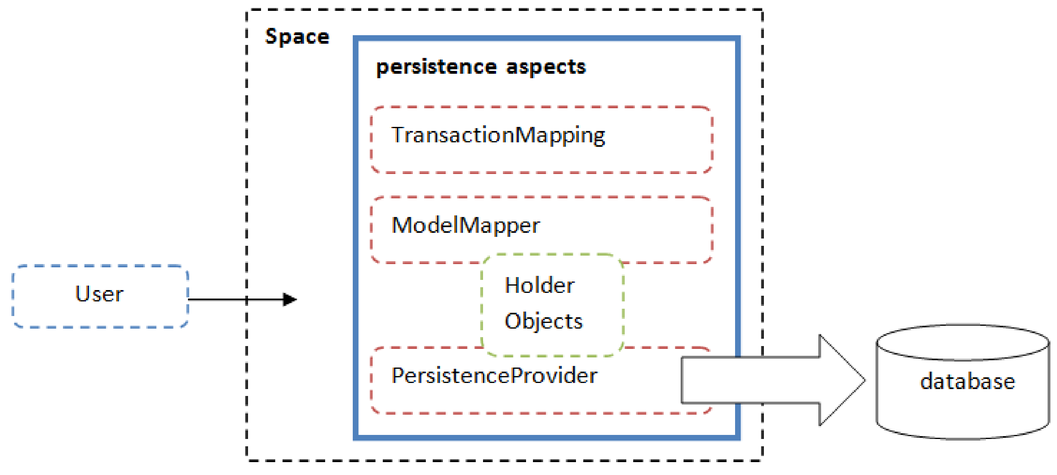

Figure: Basic architecture of XVSM Persistence

Conclusion

- XVSM Persistence was **implemented for MozartSpaces 1.0** - the Java reference implementation of XVSM.
- Suggestions on how to improve XVSM to support flexible plugging of persistency extensions and were propagated to the development team of the next MozartSpaces version 2.0.
- The design of XVSM Persistence realizes an architecture that can be used to implement persistency for the 2.0 version of MozartSpaces.
- The definition of the different **persistency models** and the enhanced **persistence semantics** distinguishes XVSM Persistence from any other SBA.

Kontakt: t.meindl@gmail.com