Coordinating Web-based Systems with Documents in XMLSpaces

Dr.-Ing. Robert Tolksdorf / Dirk Glaubitz
Technische Universität Berlin
Fakultät IV - Informatik
Formale Modelle, Logik, Programmierung
KIT
tolk@cs.tu-berlin.de

Motivation: Coordination and the Web

- Web has become *the* universal information system
- Not many distributed applications utilizing the Web for universal access

- Core question: What is the concept applied for the coordination of independent activities in a cooperative whole?

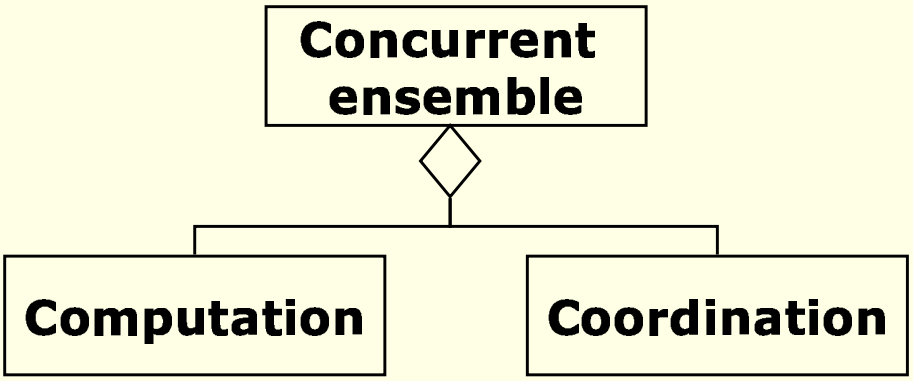
- XMLSpaces follows idea of separate coordination language that deals exclusively with the aspects of the interplay of entities and provides concepts orthogonal to computation.

Coordination Languages

- Coordination theory [Malone, others]:

  Coordination is the management of dependencies between activities

- Coordination technology [Gelernter/Carriero]:



Linda

- Coordination language with tuplespace accessed with few operations: out, in/rd, inp/rdp, eval

- `out(<2000, "Eilat">)`
  `out(<2001, "Trento">)`
- `in(<2001,?town)`
  Matching relation determines result
- `in(<2002,?town)`
  Blocks until *someone* out´s a match *sometimes*

- Linda is good for Web based systems:
  Uncoupled in space and time, asynchronous, anonymous...

Motivation: Coordination and the Web

- Tupels are weak in expressibilty
  - Fixed typing
  - No higher order structures
  - „small"
- Extensible Markup Language XML has become the format to exchange data markup following application specific syntaxis
- XMLSpaces =
  common communication format XML +
  coordination language Linda

Motivation: Coordination and the Web

- Features:

① XML documents serve as field-data

② Multitude of relations amongst XML docs for matching

③ XMLSpaces is distributed, servers at different locations form one logic dataspace.

④ Distributed events supported so that clients are notified when a tuple is added or removed somewhere

① XML documents serve as field-data

```
<?xml version="1.0"?>
<!DOCTYPE address SYSTEM
                    "address.dtd">
<location>
  <city name="Trento"/>
  <host>FACOLTÀ DI ECONOMIA</host>
  <street name="Via Vigilio Inama"
          no="5"/>
</location>
```

- <2001, __>

How to match?

- A formal is an object describing an XML document
- Can be:
  - Another document:

```
<?xml version="1.0"?>
<!DOCTYPE address SYSTEM
                    "address.dtd">
<location>
  <city name="Trento"/>
  <host>FACOLTÀ DI ECONOMIA</host>
  <street name="Via Vigilio Inama"
          no="5"/>
</location>
```

```
in(2001, ?__)
```

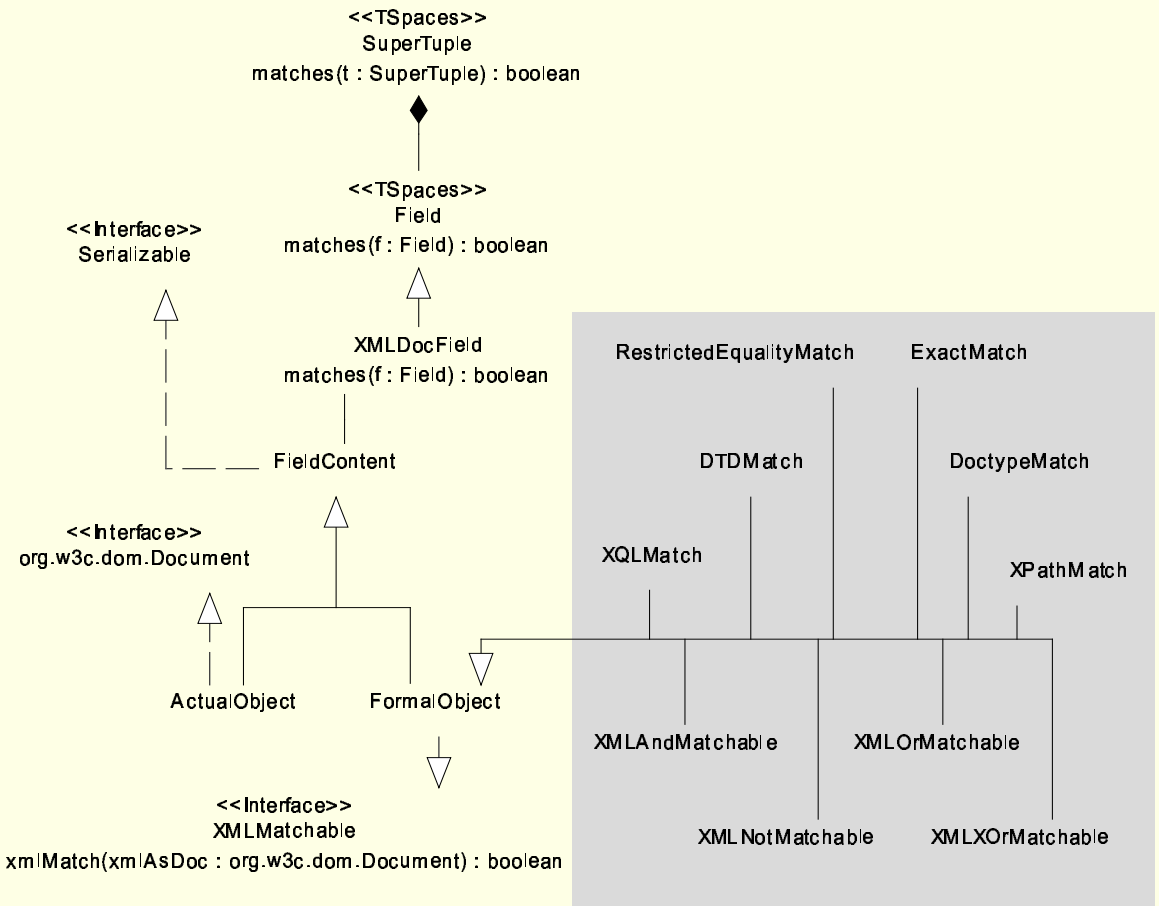
  - Something else
    `in(2001, DOCTYPE="address.dtd")`

- There is a variety of relations that identify XML documents

② Multiple matching relations in XMLSpaces

Relation	*Meaning*	*Tool used*
Exact equality	Exact textual equality	DOM interfaces
Restricted equality	Textual equality ignoring comments, PIs, etc.	DOM interfaces
DTD	Valid towards a DTD	XML4J
DOCTYPE	Uses specific Doctype name	DOM
XPath	Fulfills an XPath expression	Xalan-Java
XQL	Fulfills an XQL expression	GMD-IPSI XQL
AND	Fulfills two matching relations	–
NOT	Does not fulfill matching relation	–
OR	Fulfills one or two matching relations	–
XOR	Fulfills one matching relation	–

Class hierarchy for XML tuplefields



Openness

- Further relations can be added easily
- Example from XQL integration:

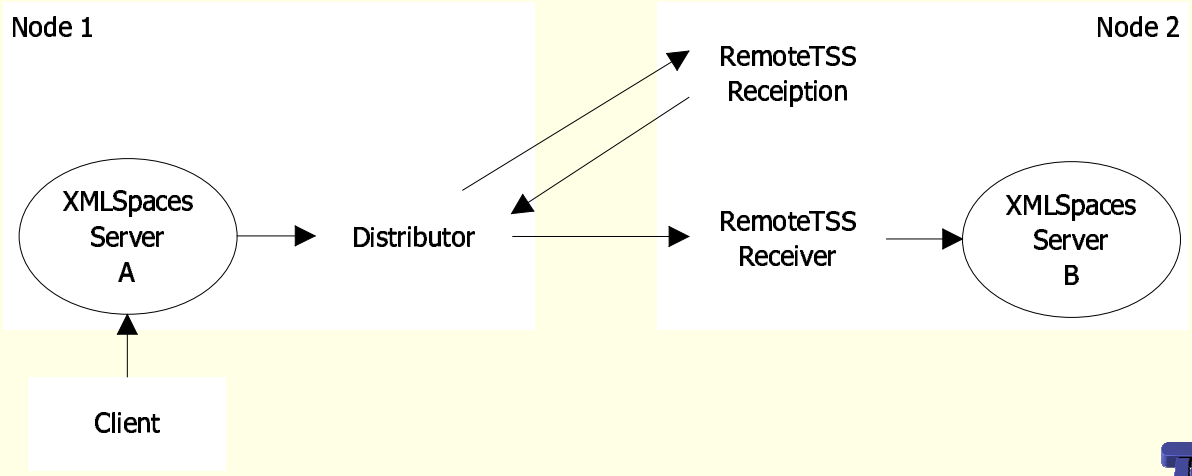
```
package matchingrelation;
import xmlspaces.XMLMatchable;
import java.io.*;
import org.w3c.dom.Document;
import de.gmd.ipsi.xql.*;
public class XQLMatch implements XMLMatchable{
  String query;
  public XQLMatch(String xqlQuery){
    query = xqlQuery;
  }
  public boolean xmlMatch(Document xmlAsDoc){
    return XQL.match(query, xmlAsDoc);
  }
}
```


③ Distribution

- Options known from work on Linda:
  - *Centralized*: One server holds the complete dataspace.
  - *Distributed*: All servers hold distinct subsets of the complete dataspace
  - *Full replication*: All servers hold consistent copies of the complete dataspace
  - *Partial replication*: Subsets of servers hold consistent copies of subsets of the dataspace
  - *Hashing*: Matching tuples and templates are stored at the same server selected by some hashing function
- XMLSpaces supports all schemas and is extensible

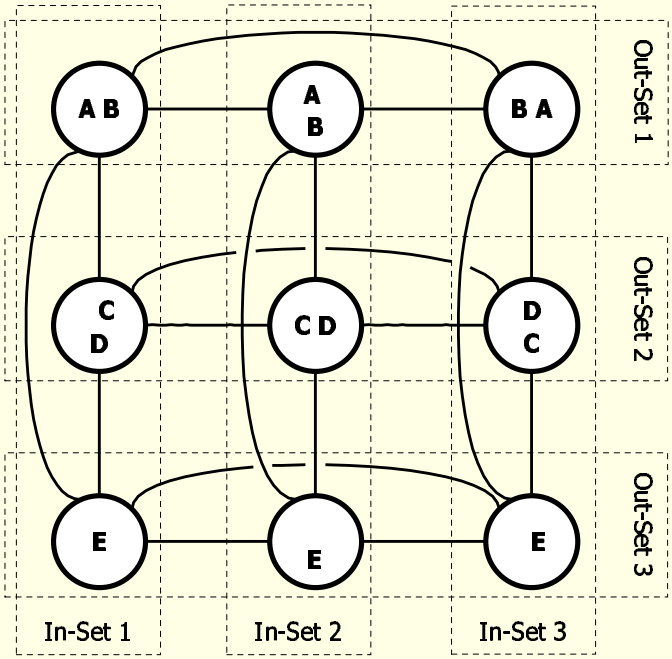
Distributed servers

- Receiption tells node which peers to use
- Each peer is initially contacted by a RemoteTSSReceiption
- Further communication is by a RemoteTSSReceiver
- Distributor object encapsulates distribution schema



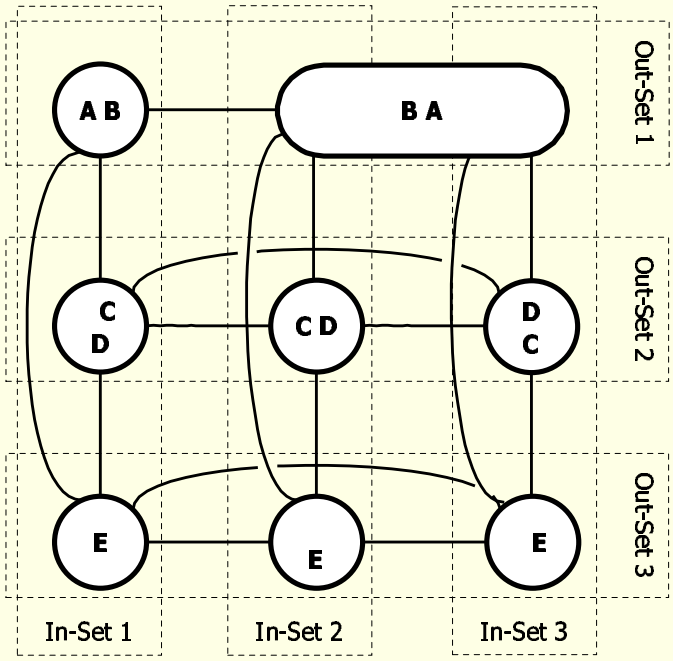
Partial replication scheme

- Nodes in out-set contain replica
- Union of contents in nodes in in-set is whole content of space



Partial replication scheme

- Simulated nodes to keep grid
- Protocols for join and leave of nodes
- Centralized, distributed and fully replicated schemes are special cases of partial replication



KIT
FLP

④ Distributed events

- Register for addition or removal of data in the distributed space
- Distributed along in-sets
- Joining and leaving protocols accordingly
- Encapsulated in Distributor object

KIT
FLP

Implementation

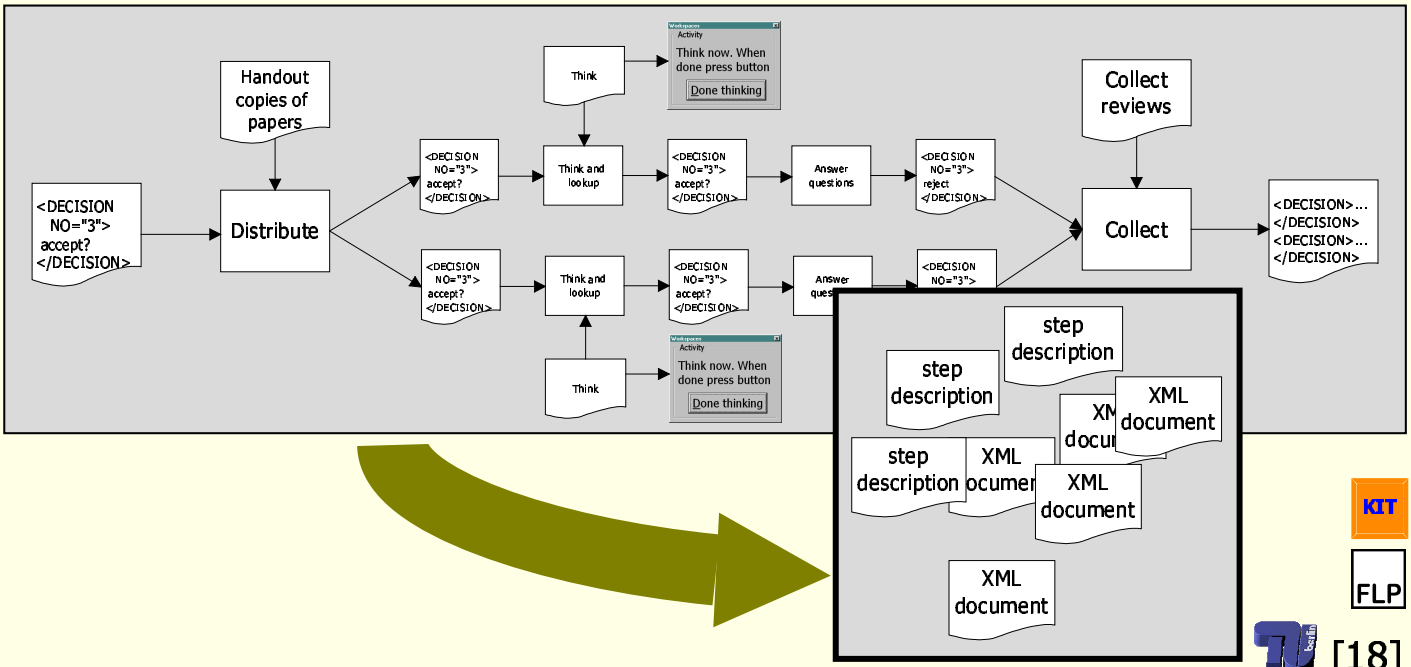
- DOM level 1 as internal representation
- TSpaces for local spaces (source licence granted by IBM Almaden)
- Various XML software
- JVM based

KIT

FLP

Application: Workspaces

- An XML/XSL based workflow system
- Workflow graph is split into single steps that are represented as XSL documents

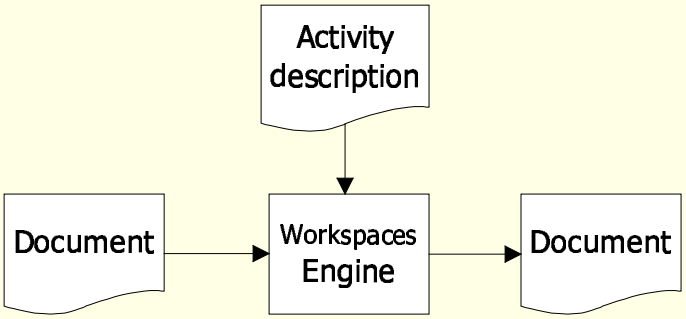


KIT

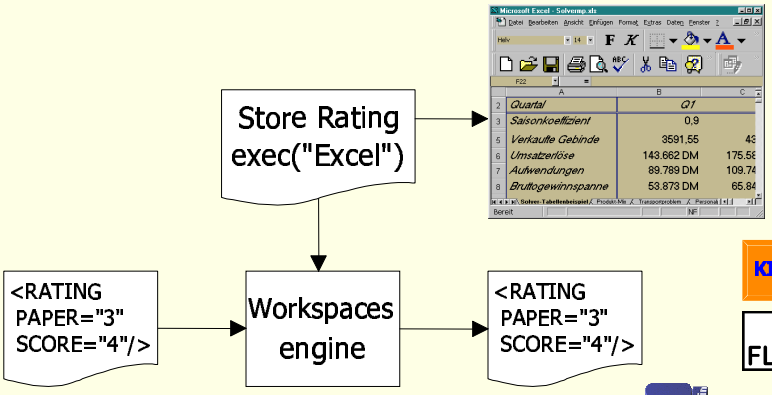
FLP

Application: Workspaces

- Documents and worksteps as XML documents

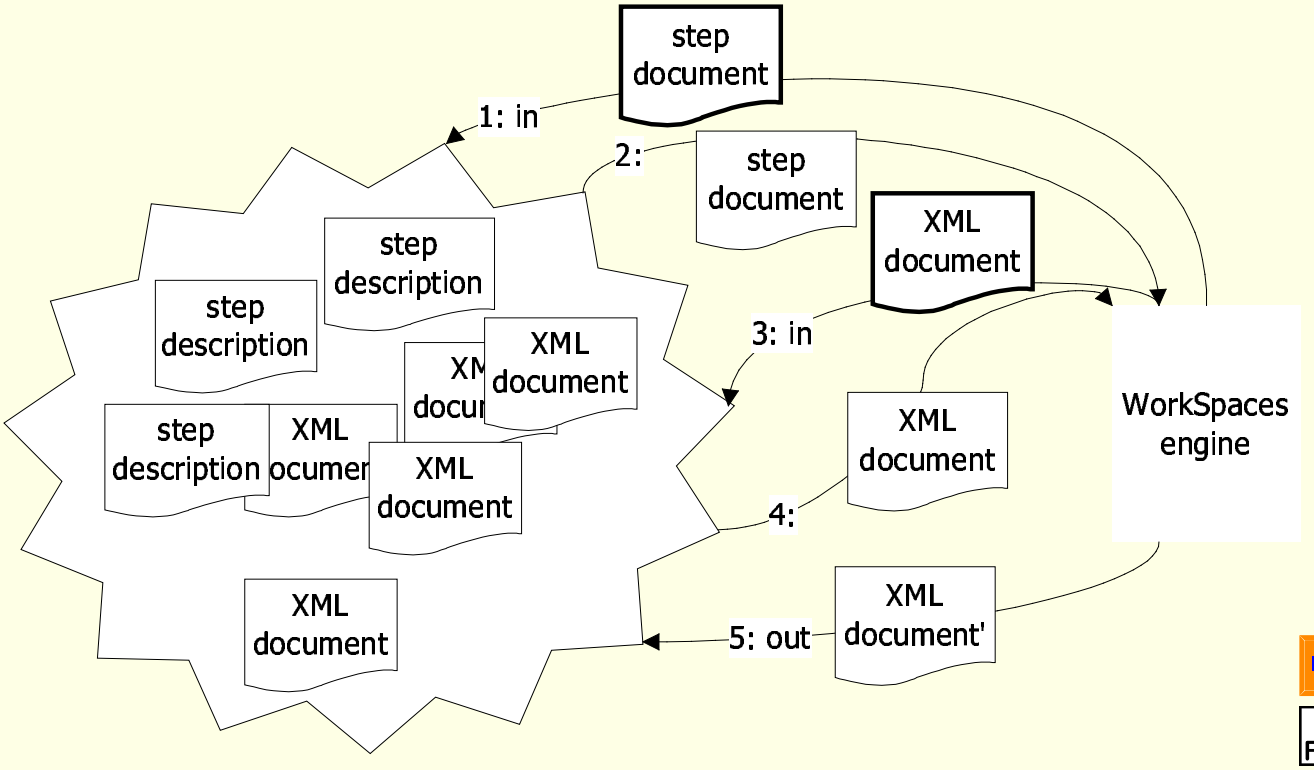


- Workspaces engine is an extended XSL engine



KIT
FLP

XML access in Workspaces



KIT
FLP

Conclusion and outlook

- Using XML as tuple-fields adds necessary structure for data
- Relations on XML documents are manifold
- XMLSpaces =
  common communication format XML
  \+ coordination language Linda
- \+ open set of matching relations
- \+ open set of distribution strategies

- Further info (and software) at
  http://www.cs.tu-berlin.de/~tolk/xmlspaces
  (http://www.cs.tu-berlin.de/~tolk/workspaces)

KIT

FLP