

# xOperator

## Interconnecting the Semantic Web and Instant Messaging Networks

Sebastian Dietzold

Jörg Unbehauen

Sören Auer

UNIVERSITÄT LEIPZIG



**LEUPHANA**  
UNIVERSITÄT LÜNEBURG

# Contents

- Background and Motivation
- Application Context and Idea
- Communication Scenarios
- Technical Architecture
- Query Methods
- Evaluation
- Conclusion

# Semantic Web: Current Problems

- A clear lack of simple to use end user applications
  - (1) Difficult to **hide technical details**
- Huge amount of resources
  - (2) **Provenance and trust** of resources?
  - (3) **Context awareness** of queries?

# Instant Messaging (IM)

- > **1000 Mio users** (among different providers)
- One of the **most popular internet service**
- An **omnipresent** interface
- A trusted social **network**



# xOperator = IM + Semantic Web

- Interconnect Instant Messaging Networks with the Semantic Web
- Use Instant Messaging Clients as a widely used interface to query the users resources
- Use the inherent social network to achieve a trust in resources to achieve context awareness of the queries

# Talk to the Semantic Web

The image shows two overlapping windows from the Gajim chat client. The window on the left is titled "xOperator - Gajim" and contains a chat window for "xOperator". The chat window has a yellow star icon and the text "xOperator". Below the chat area, there is a text input field containing the message "tell me the phone of sören". The window on the right is titled "Gajim" and shows the contact list. The contact list includes several groups and individual contacts, each with a yellow star icon. The groups are: "aksw.org (10/23)", "AKSW (2/5)", "Allgemein (1/2)", "Bots (1/1)", "IfI (3/3)", and "Studenten (3/11)". The individual contacts are: "Jens Lehmann", "Thomas Riechert", "Andreas Thor", "Frank Loebe", "Hannes Michalek", "Jörg Unbehauen (MA)", "Martin Peklo (MA)", and "Norman Heino (SHK)". At the bottom of the contact list, there is a status bar showing "Angemeldet".

**xOperator - Gajim**

**xOperator**

tell me the phone of sören

**Gajim**

Aktionen Ändern Ansicht Hilfe

- ★ aksw.org (10/23)
- ▼ AKSW (2/5)
- 📁 Jens Lehmann
- ★ Thomas Riechert
- ▶ Allgemein (1/2)
- ▼ Bots (1/1)
- ★ xOperator
- ▼ IfI (3/3)
- ★ Andreas Thor
- ★ Frank Loebe
- ★ Hannes Michalek
- ▼ Studenten (3/11)
- ★ Jörg Unbehauen (MA)
- 📁 Martin Peklo (MA)
- ★ Norman Heino (SHK)
- ★ [dietzold.de] (10/44)

★ Angemeldet

# Talk to the Semantic Web

The image shows two overlapping windows from the Gajim chat client. The window on the left is titled "xOperator - Gajim" and contains a chat message: "[21:48:31] Seebi: tell me the phone of sören". The window on the right is the main Gajim interface, showing a list of contacts and groups. The contacts list includes "aksw.org (10/23)", "AKSW (2/5)", "Jens Lehmann", "Thomas Riechert", "Allgemein (1/2)", "Bots (1/1)", "xOperator", "IfI (3/3)", "Andreas Thor", "Frank Loebe", "Hannes Michalek", "Studenten (3/11)", "Jörg Unbehauen (MA)", "Martin Peklo (MA)", "Norman Heino (SHK)", and "[dietzold.de] (10/44)". The status bar at the bottom of the Gajim window shows "Angemeldet".

**xOperator - Gajim**

**xOperator**

[21:48:31] Seebi: tell me the phone of sören

**Gajim**

Aktionen Ändern Ansicht Hilfe

- ★ aksw.org (10/23)
- ▼ AKSW (2/5)
- 👤 Jens Lehmann
- ★ Thomas Riechert
- ▶ Allgemein (1/2)
- ▼ Bots (1/1)
- ★ xOperator
- ▼ IfI (3/3)
- ★ Andreas Thor
- ★ Frank Loebe
- ★ Hannes Michalek
- ▼ Studenten (3/11)
- ★ Jörg Unbehauen (MA)
- 👤 Martin Peklo (MA)
- ★ Norman Heino (SHK)
- ★ [dietzold.de] (10/44)

★ Angemeldet

# Talk to the Semantic Web

xOperator - Gajim

**xOperator**

[21:48:31] Seebi: tell me the phone of sören  
[21:48:33] xOperator: From SPARQL endpoint (<http://demo.ontowiki.net/service/sparql>)  
\* +49 341 97-32367

**Aktionen** **Senden**



# Talk to the Semantic Web

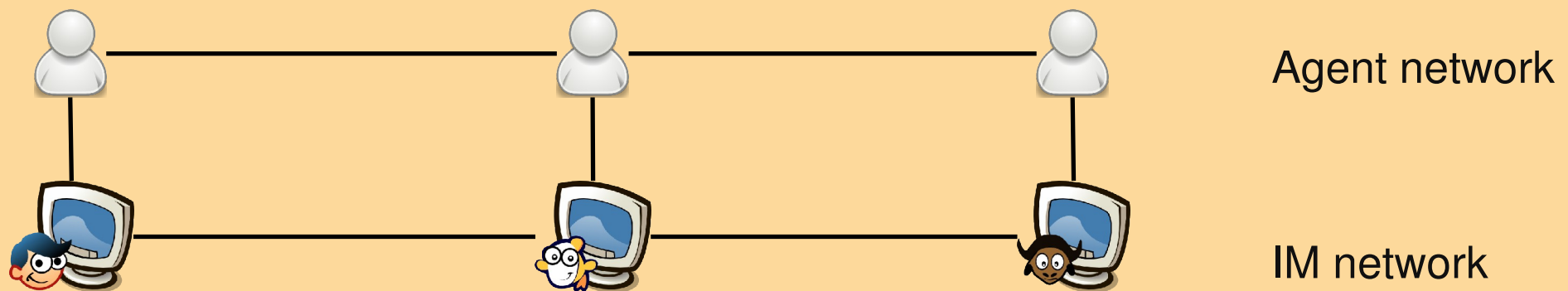
xOperator - Gajim

**xOperator**

[21:48:31] Seebi: tell me the phone of sören  
[21:48:33] xOperator: From SPARQL endpoint (<http://demo.ontowiki.net/service/sparql>)  
\* +49 341 97-32367  
[21:48:36] xOperator: From agent ([soerenauer@jabber.ccc.de](mailto:soerenauer@jabber.ccc.de))  
\* +49(341)97-32323

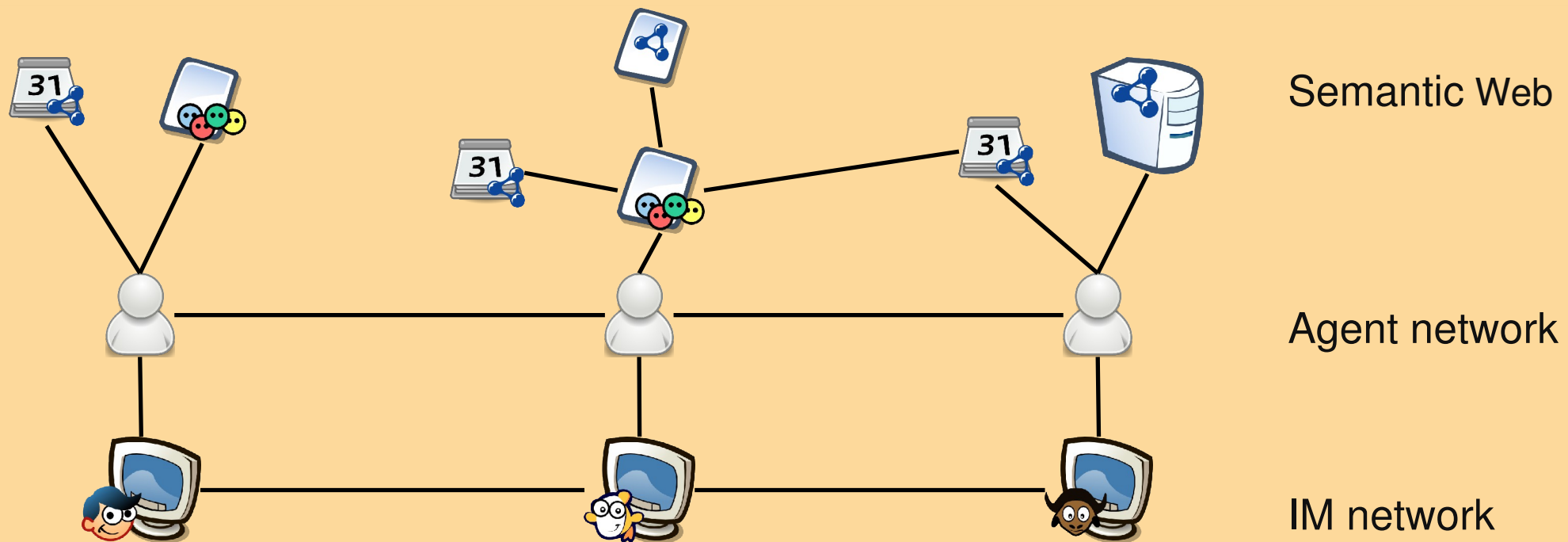
# Application Context and Idea

- Build an agent **overlay network** inside the IM network



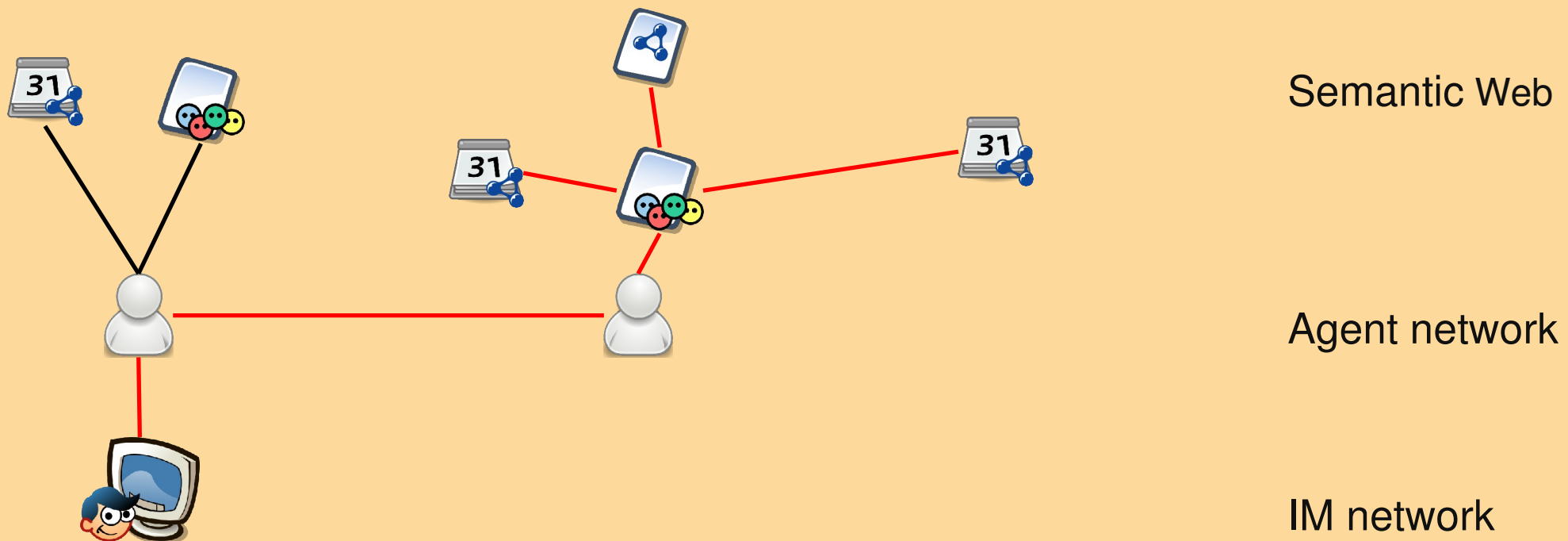
# Application Context and Idea

- Build an agent **overlay network** inside the IM network
- Agents are equipped with **trusted resources**



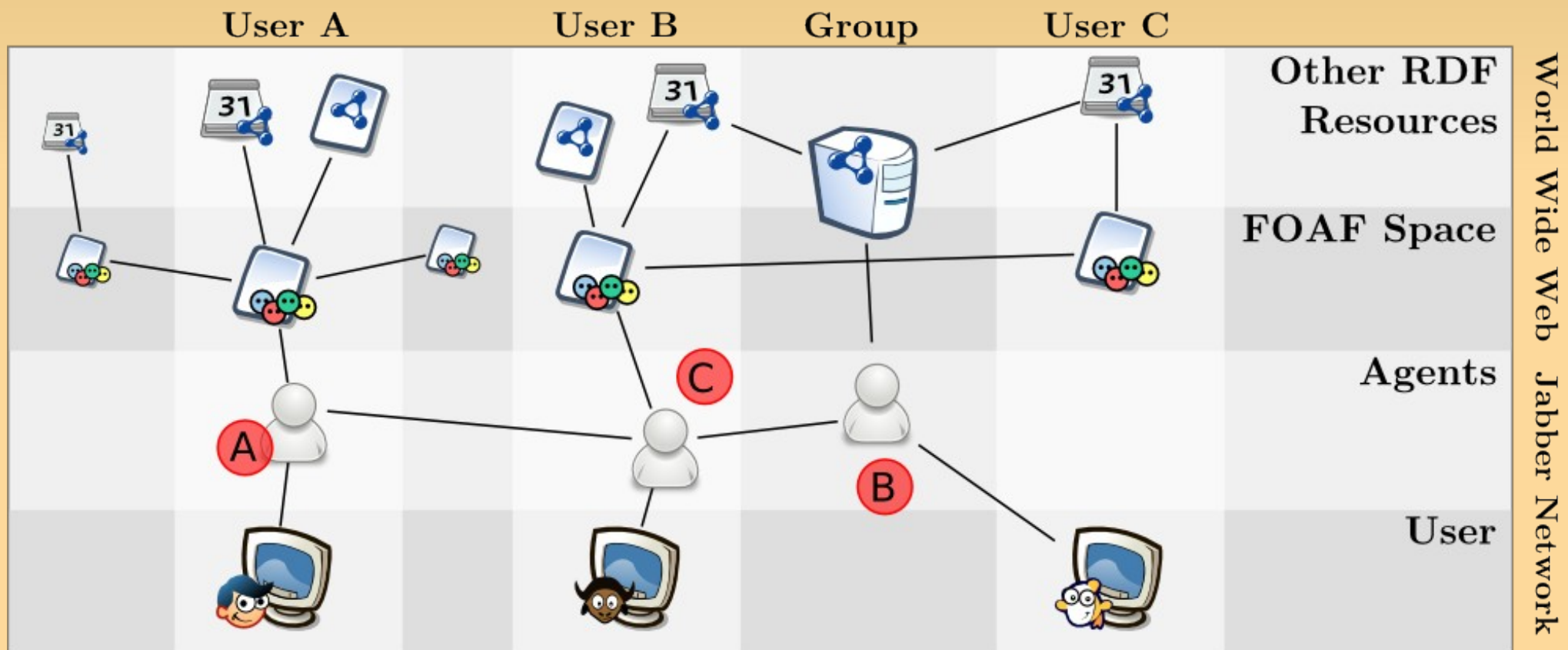
# Application Context and Idea

- Build an agent **overlay network** inside the IM network
- Agents are equipped with **trusted resources**
- Queries are routed to **friend agents**



# Communication Scenarios

(A) Personal Agent - (B) Group Agent - (C) Agent Network



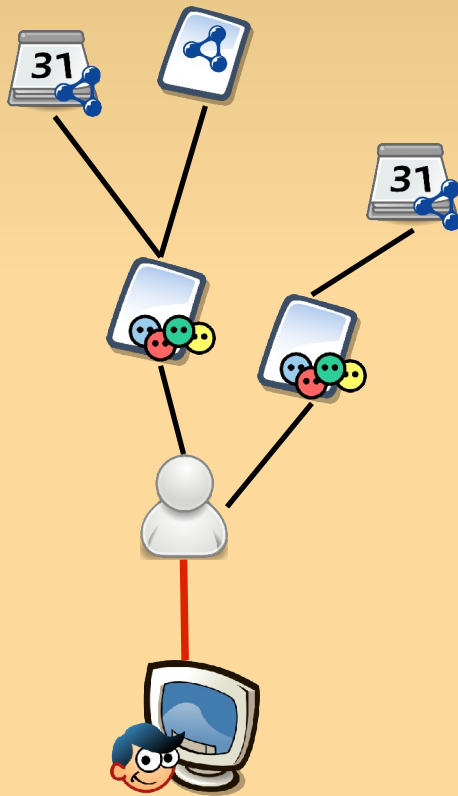
# Personal Agent

- Access to your trusted RDF data

Your own FOAF profile

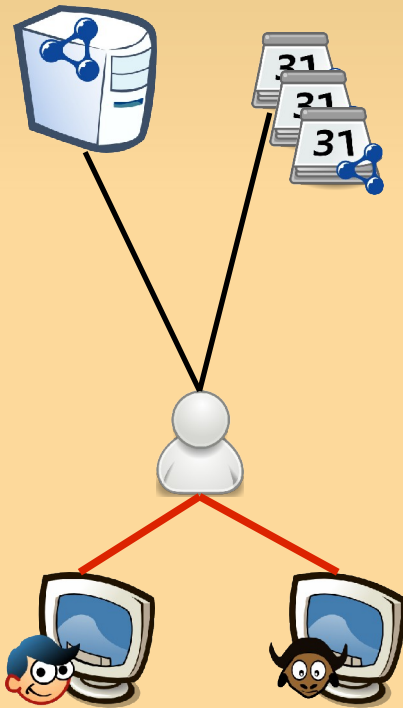
Your friends FOAF profile

RDF calendars

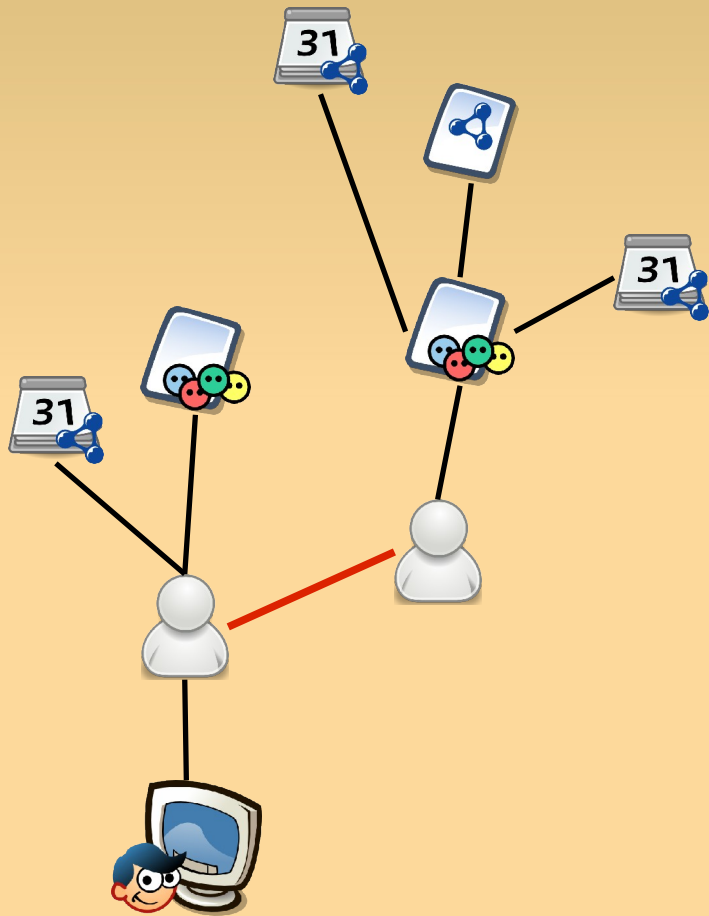


# Group Agent

- Agent permits more than one user
- Access to the group resources
  - Company directory
  - Group calendar
  - Semantic wiki



# Agent Network



- Agents communicate with their **neighboring agents**

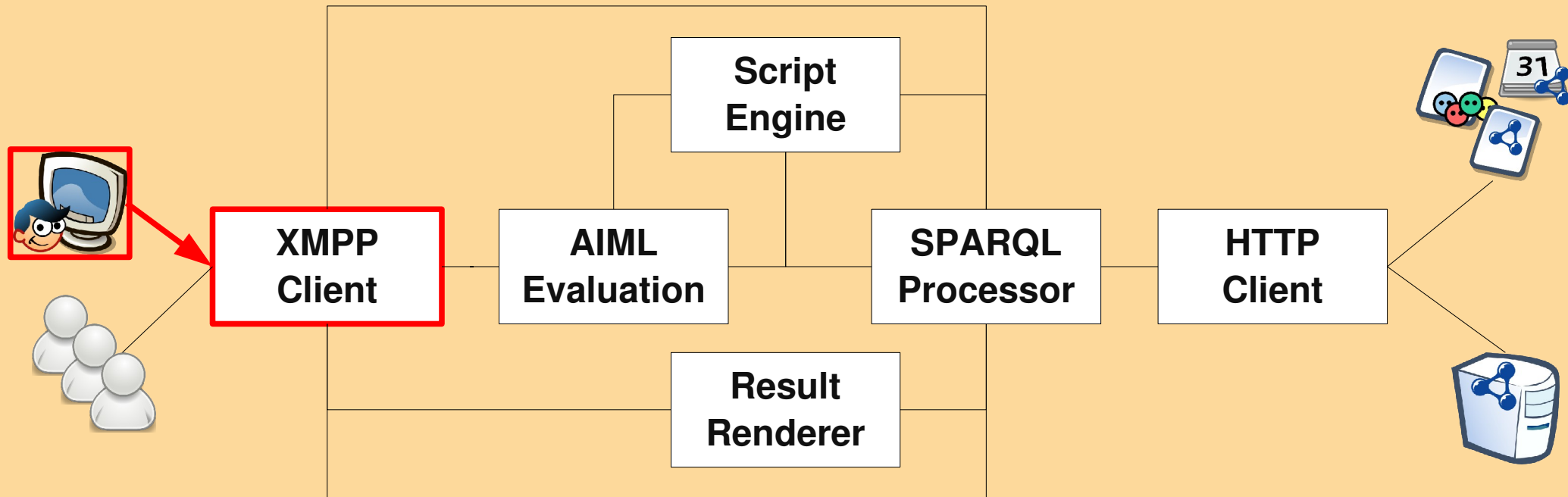
SPARQL over XMPP

Neighbourhood = Agents which are owned by a friend (using the **IM roster**)



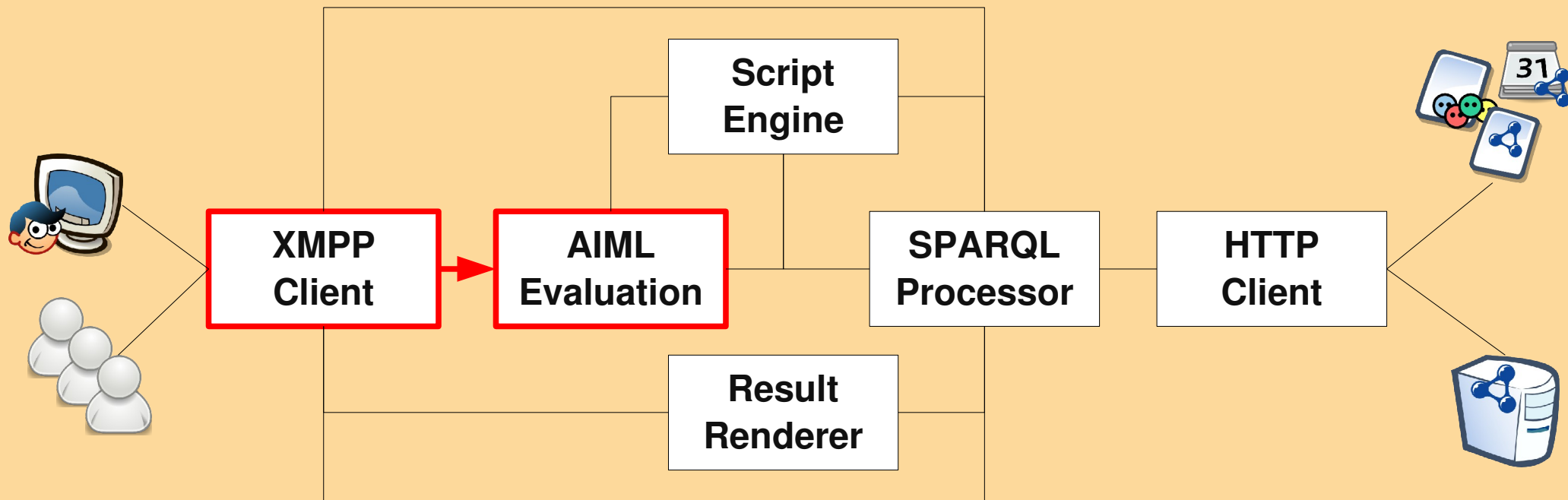
# Technical Architecture

- User types a chat message to her agent



# Technical Architecture

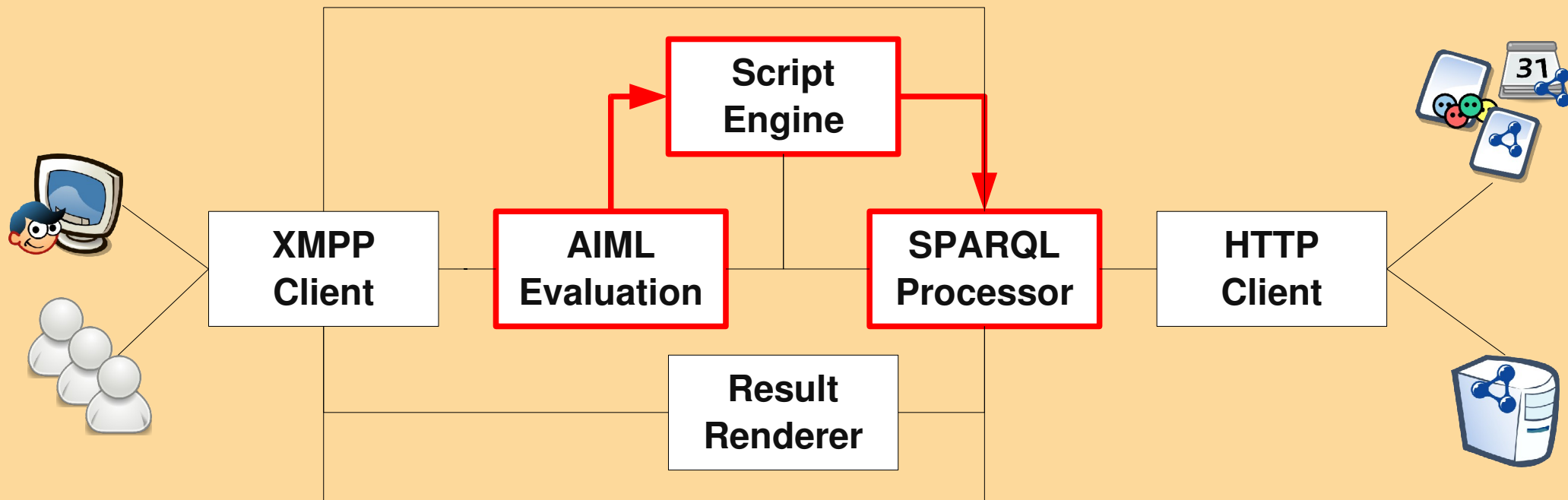
- Message processing  
string substitutions, stop word filtering



# Technical Architecture

- Query script execution

Parameter manipulation, endpoint filtering, query chaining, ...

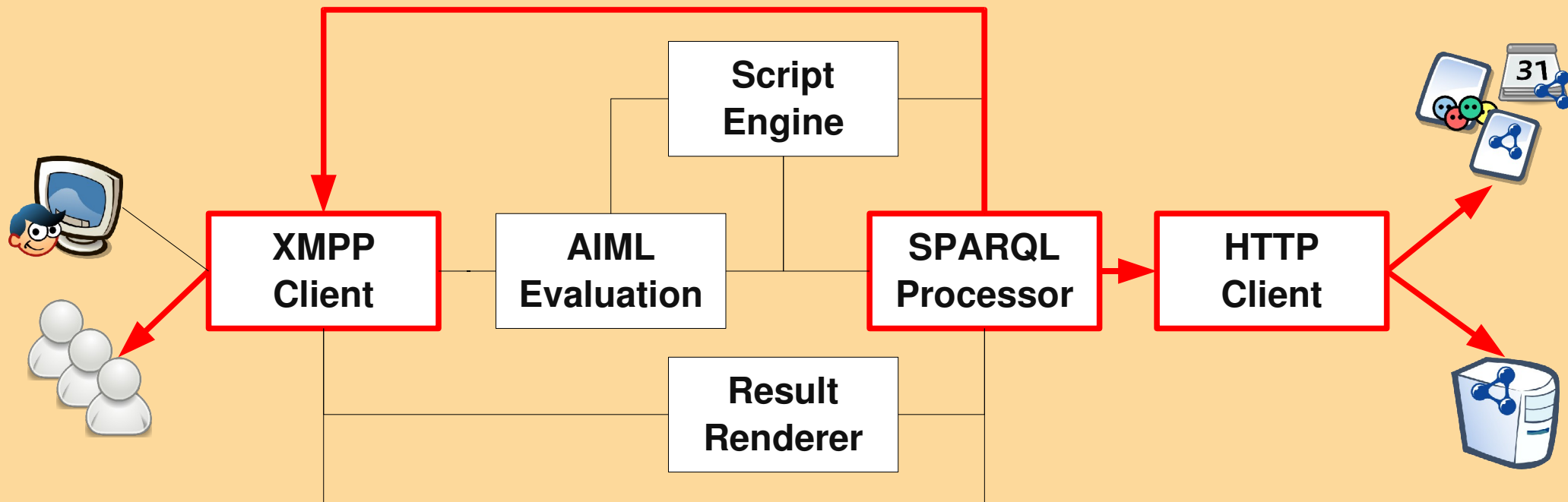


# Technical Architecture

- Query distribution and execution

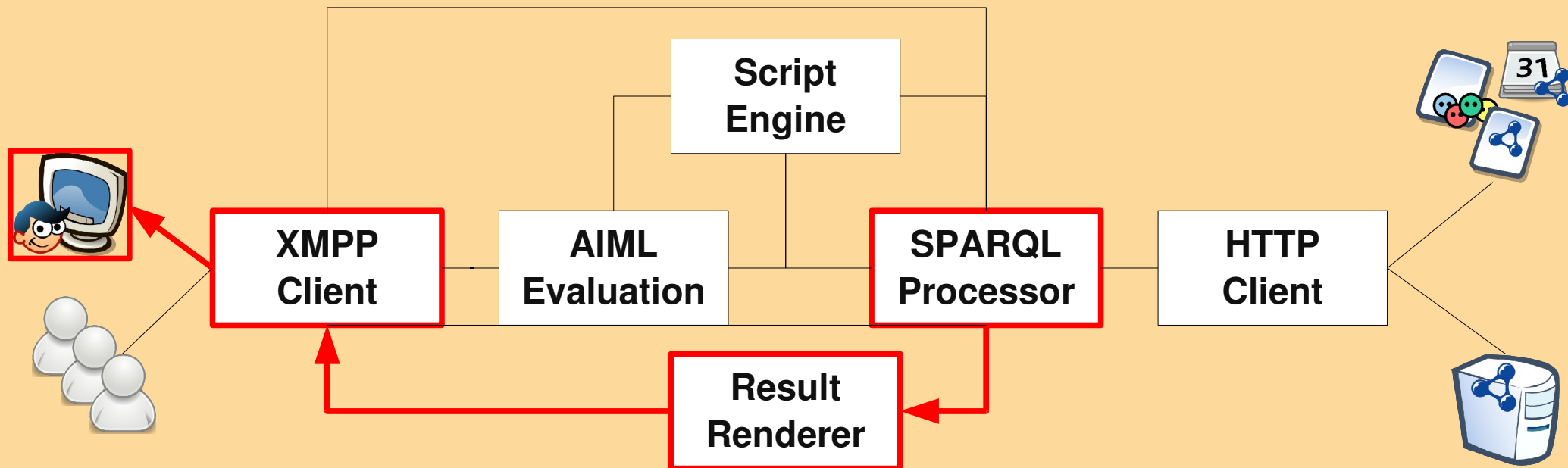
Local endpoint, distributed endpoints, documents

Agent communication as special XMPP queries



# Technical Architecture

- Rendering of SPARQL results
- Return message to the user



# Query Method: Template Based

User input is matched with a **single** SPARQL query

Tell me the **phone** of **sören**

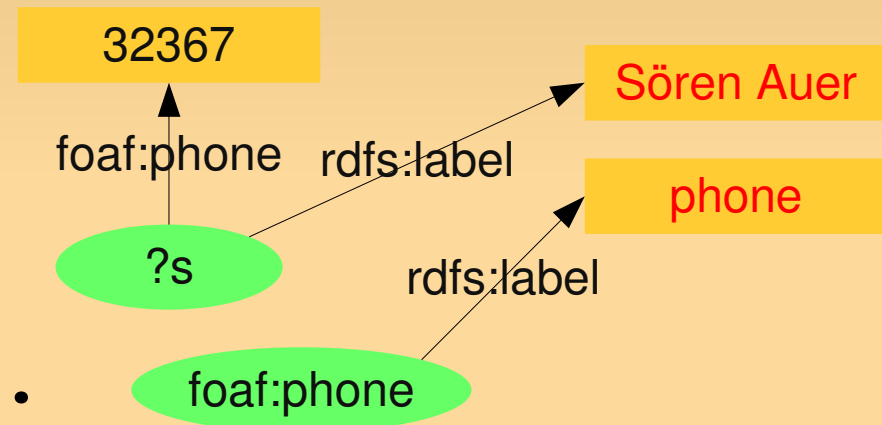
```
<category>
  <pattern>TELL ME THE * OF *</pattern>
  <template>
    <external name="query" param="SELECT ..." />
  </template>
</category>
```

# Query Method: Template Based

User input is matched with a **single** SPARQL query

Tell me the **phone** of **sören**

```
SELECT ?value WHERE {
  ?s      rdfs:label ?spattern.
  ?s      ?prop      ?value.
  ?prop   rdfs:label ?ppattern.
  FILTER regex(?spattern, '.*sören.*', 'i')
  FILTER regex(?ppattern, '.*phone.*', 'i')}
```



# Query Method: Script Based

User input is matched with a **Groovy Script**

where is **sören** now

```
<category>
  <pattern>WHERE IS * NOW</pattern>
  <template>
    <external name="groovy"
      param="iCalWhereNow.groovy" />
  </template>
</category>
```



# Query Method: Script Based

where is **sören** now

```
// where can we find infos about the subject
```

```
documents = context.query(NAMESPACES + "SELECT DISTINCT ?cal "+  
    "WHERE{?cal rdf:type ical:Vcalendar " + ...)
```

```
// build the next query based on the first one
```

```
if(documents.counter < 1) context.sendMessageToUser("Sorry, ...")
```

```
else{ documents.each(){
```

```
    res.getResultRows().each(){ GRAPHS += "FROM NAMED <${it["cal"]} > " } }
```

```
events = context.query(NAMESPACES + "SELECT DISTINCT ?loc ?sum"+
```

```
    GRAPHS +
```

```
    "WHERE { GRAPH ?g { "+
```

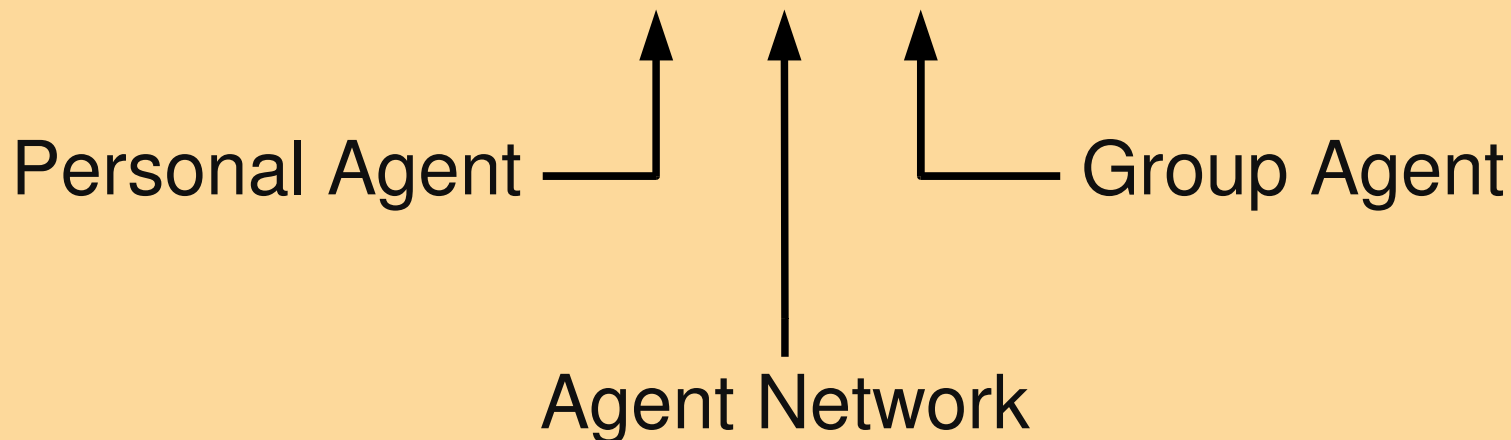
```
    "?s ical:dtstart ?sbn. ?sbn ical:dateTime ?stime."+
```

```
    "?s ical:dtend ?ebn. ?ebn ical:dateTime ?etime."+
```

```
    "OPTIONAL {?s ical:location ?loc.} " + ... }
```

# Evaluation

Use case / template	runtime	Sources
What is / Tell me (the) * of *	2.3 / 3.9 / 1.5	FOAF + Jabber Network
Who is member of *	3.5 / 4.3 / 1.6	Group endpoint
Tell me more about *	3.2 / 5.6 / 1.1	DBpedia
Where is * now	5.1 / 6.7 / 4.2	FOAF + Google calendar
Free dates * between * and *	5.1 / 6.8 / 4.7	Google calendar + OntoWiki
Which airports are near *	- / - / 3.4	DBpedia



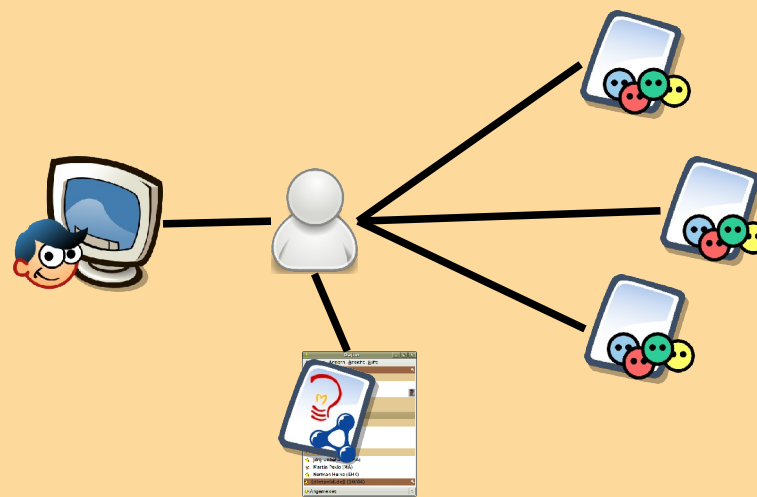
# Evaluation

Use case / template	runtime	Sources
What is / Tell me (the) * of *	2.3 / 3.9 / 1.5	FOAF + Jabber Network
Who is member of *	3.5 / 4.3 / 1.6	Group endpoint
Tell me more about *	3.2 / 5.6 / 1.1	DBpedia
Where is * now	5.1 / 6.7 / 4.2	FOAF + Google calendar
Free dates * between * and *	5.1 / 6.8 / 4.7	Google calendar + OntoWiki
Which airports are near *	- / - / 3.4	DBpedia

Tell me the room of Thomas

From OntoWiki

\* 5-09 (Thomas Riechert)



# Evaluation

## Use case / template

## runtime

## Sources

What is / Tell me (the) \* of \*

2.3 / 3.9 / 1.5

FOAF + Jabber Network

Who is member of \*

3.5 / 4.3 / 1.6

Group endpoint

Tell me more about \*

3.2 / 5.6 / 1.1

DBpedia

Where is \* now

5.1 / 6.7 / 4.2

FOAF + Google calendar

Free dates \* between \* and \*

5.1 / 6.8 / 4.7

Google calendar + OntoWiki

Which airports are near \*

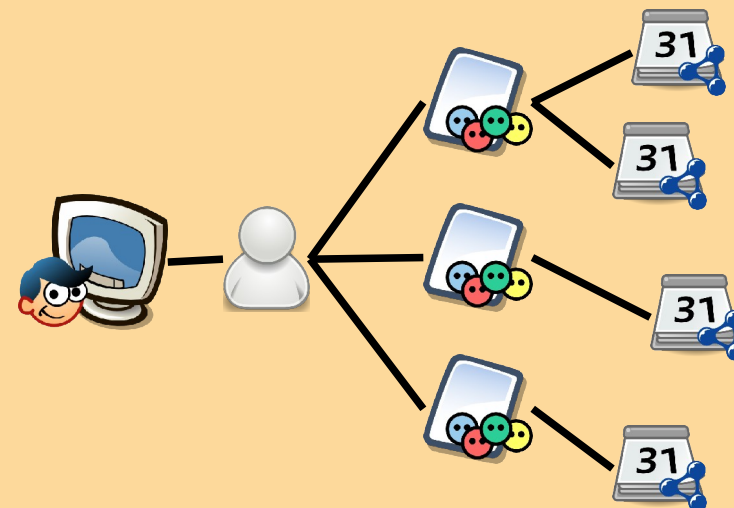
- / - / 3.4

DBpedia

Where is Thomas now?

I have found 1 Calender ...

Cafeteria (Summary: Lunch)



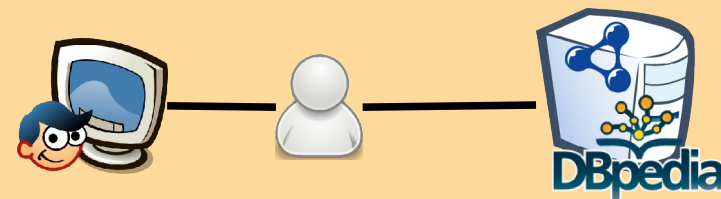
# Evaluation

Use case / template	runtime	Sources
What is / Tell me (the) * of *	2.3 / 3.9 / 1.5	FOAF + Jabber Network
Who is member of *	3.5 / 4.3 / 1.6	Group endpoint
Tell me more about *	3.2 / 5.6 / 1.1	DBpedia
Where is * now	5.1 / 6.7 / 4.2	FOAF + Google calendar
Free dates * between * and *	5.1 / 6.8 / 4.7	Google calendar + OntoWiki
Which airports are near *	- / - / 3.4	DBpedia

Tell me more about Tenerife

**Tenerife can be a: Island,  
City, ...**

Tell me more about Tenerife  
the Island



# Conclusion

## xOperator

- Interconnects the Semantic Web with IM
- Uses IM clients to query RDF data
- Utilizes the users IM neighbourhood  
to obtain a trusted subset of the Semantic Web  
to hold the context of semantic queries

# Thank You

- More information:  
<http://aksw.org/Projects/xOperator/>
- XMPP / Jabber demo agent:  
[xoperator-demo@aksw.org](mailto:xoperator-demo@aksw.org)

