A portable implementation of Semantic Information Broker in OSGi technology

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Alfredo D’Elia

ARCES

DISI

adelia@arces.unibo.it

www.unibo.it

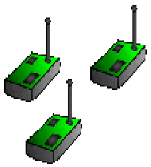
Innovation House – Otaniemi
12th November 2013

Overview

- Reference scenario
- Motivations
- Features
- Internal architecture
- Performance evaluation
- Conclusion

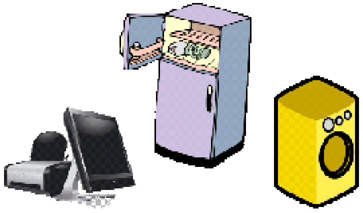
Reference scenario

Sensors



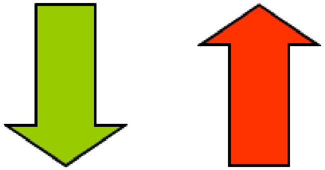
People and smart devices

Appliances



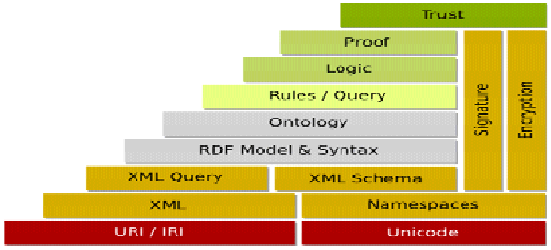
PHYSICAL
WORLD

Context: Raw sensor data, profiles, location, requests geometries, preferences, goals…



Services: health, privacy, integrated localization, personalized behaviour, proactive suggestions, social solutions, …

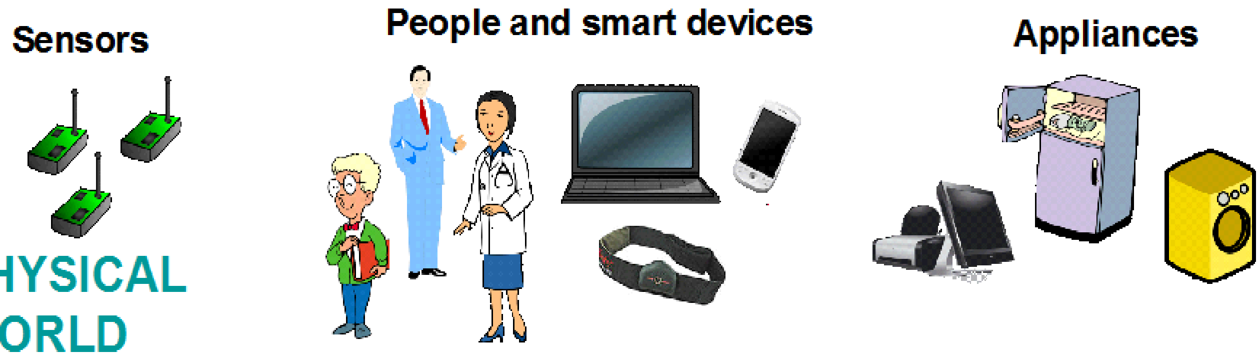
SEMANTIC
AGENTS



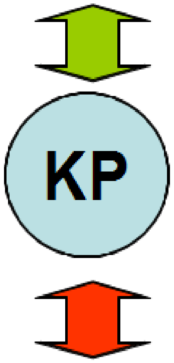
Trust
Proof
Logic
Rules / Query
Ontology
RDF Model & Syntax
XML Query | XML Schema
XML | Namespaces
URI / IRI | Unicode
Signature
Encryption

INFORMATION
WORLD

Reference scenario



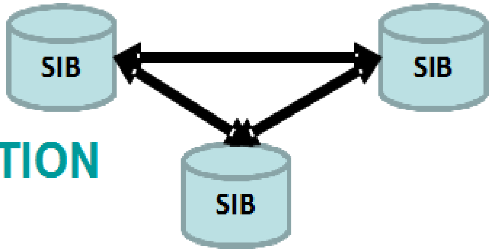
PHYSICAL WORLD



Knowledge processor (KP):

- Interacts in R/W mode with the information world
- Adapts the legacy technology to the software architecture to share all relevant information
- Multiplatform: libraries (KPI) and tools for the most common languages and platforms augment developers productivity

SEMANTIC AGENTS



Semantic Information Brokers

- Information as a semantic graph made of RDF triples
- Notification mechanism to promptly react to context changes

INFORMATION WORLD

Motivations

- Meet the requirements of evolving scenarios
- Portability
- Compatibility
- Flexibility
- Meet the technological expectations of dynamic communities (Semantic Web, Context awareness, etc.)

Combining M3 and OSGi

- Smart M3:
  - General purpose
  - Information interoperability support
  - Semantic software agents

- OSGi
  - Java
  - Portable
  - Modular
  - Maintainable
  - Reliable

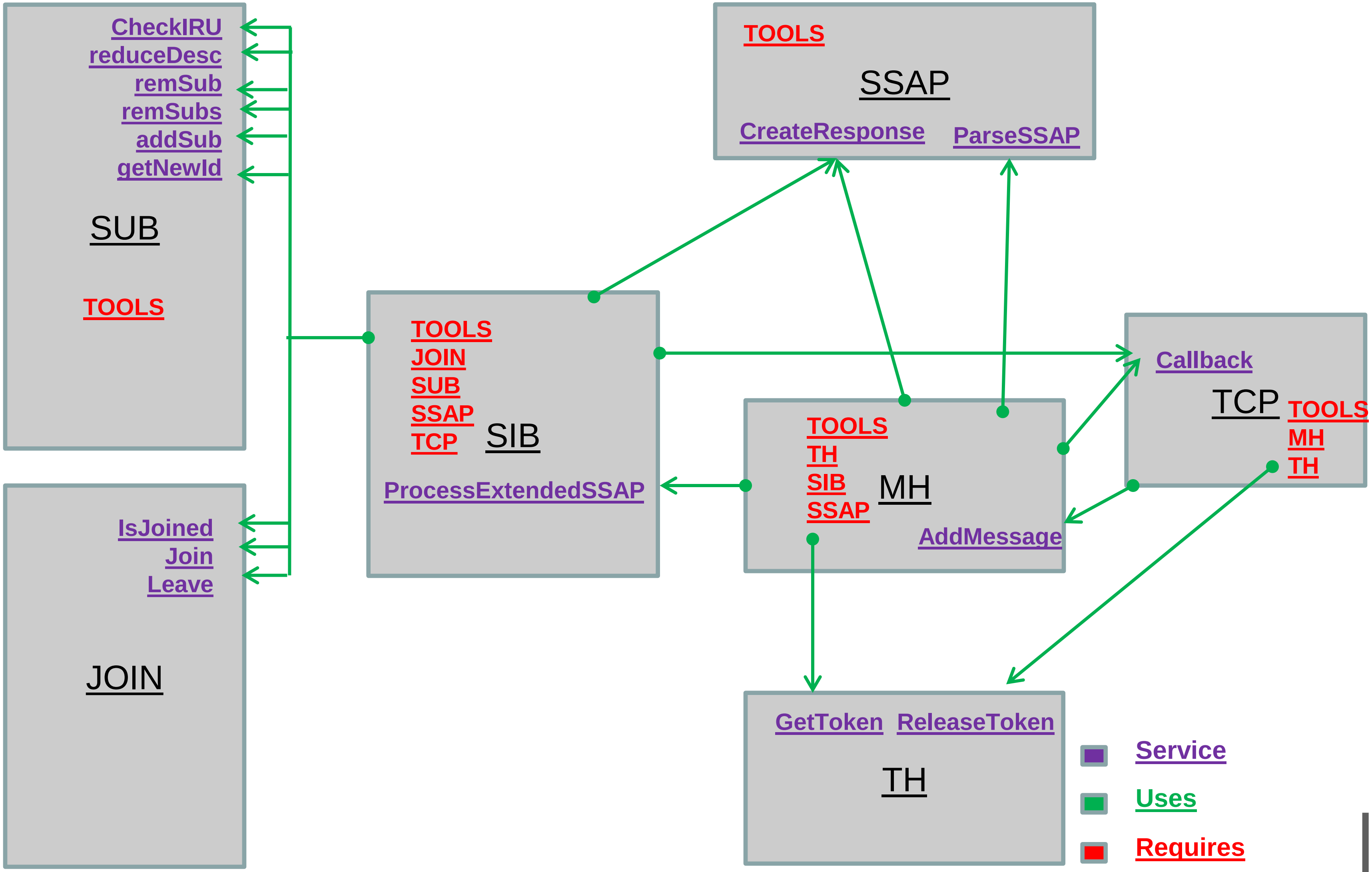
- The OSGi SIB provides the advantages of both smart-M3 and OSGi in a single core component for semantic context aware applications.

- The OSGi SIB is more appropriate than a monolithic implementation as long as the scenario and the requirements are in rapid evolution.

Features

- All basic SIB primitives
- Completely compatible with the current version of SSAP and existing libraries
- SPARQL query and update
- SPARQL subscription
- Persistent SPARQL UPDATE (similar to a rule) *new*
- DL reasoning at SROIQ level (using Pellet) *experimental prototype*

Internal architecture



CheckIRU
reduceDesc
remSub
remSubs
addSub
getNewId
SUB
TOOLS

IsJoined
Join
Leave
JOIN

TOOLS
SSAP
CreateResponse
ParseSSAP

TOOLS
JOIN
SUB
SSAP
TCP
SIB
ProcessExtendedSSAP

TOOLS
TH
SIB
SSAP
MH
AddMessage

Callback
TCP
TOOLS
MH
TH

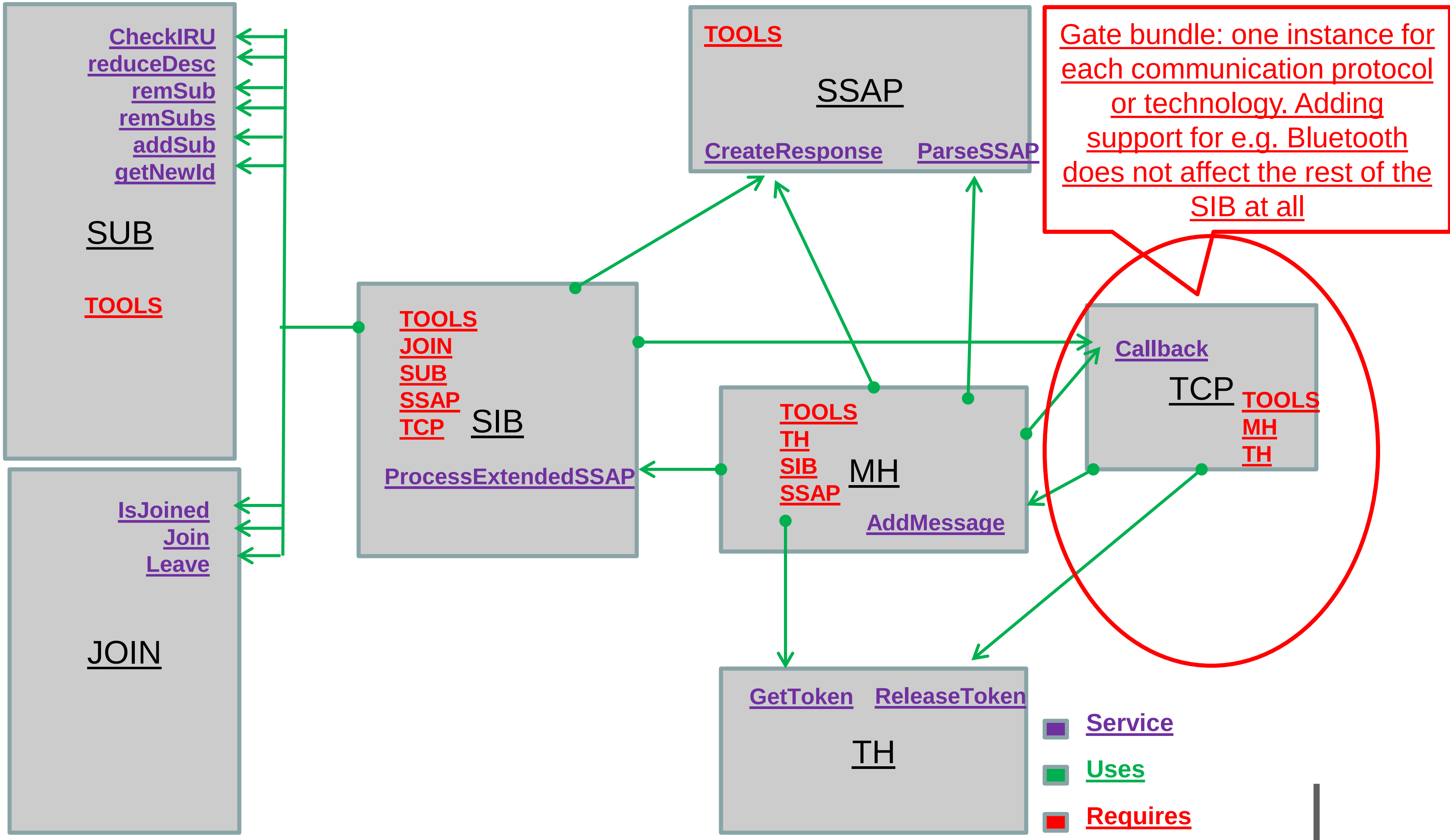
GetToken
ReleaseToken
TH

🟪 Service

🟩 Uses

🟥 Requires

Internal architecture



SUB

CheckIRU
reduceDesc
remSub
remSubs
addSub
getNewId

TOOLS

JOIN

IsJoined
Join
Leave

SIB

TOOLS
JOIN
SUB
SSAP
TCP

ProcessExtendedSSAP

SSAP

TOOLS

CreateResponse
ParseSSAP

MH

TOOLS
TH
SIB
SSAP

AddMessage

TCP

Callback

TOOLS
MH
TH

TH

GetToken
ReleaseToken

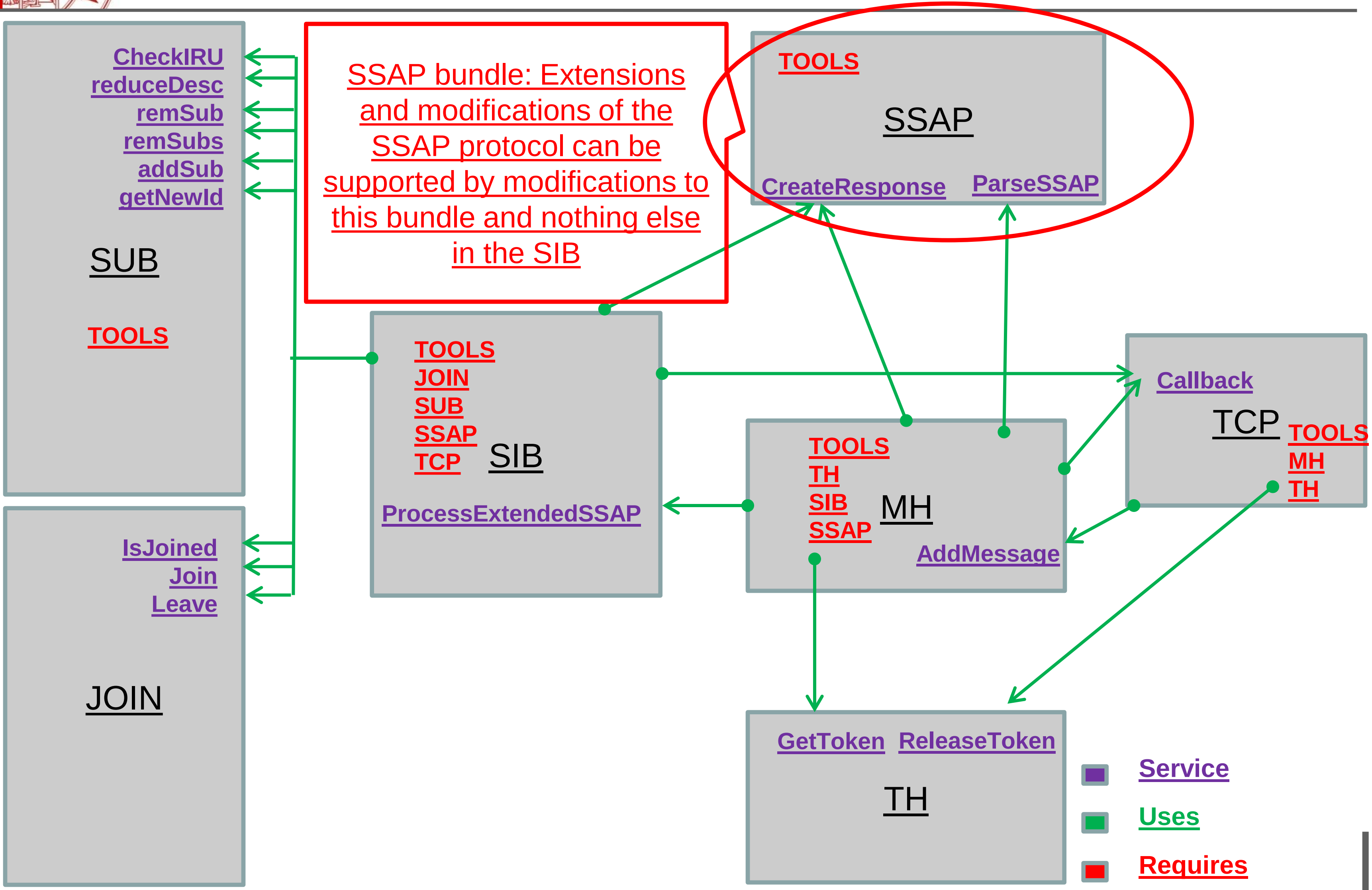
Gate bundle: one instance for each communication protocol or technology. Adding support for e.g. Bluetooth does not affect the rest of the SIB at all

🟪 Service

🟩 Uses

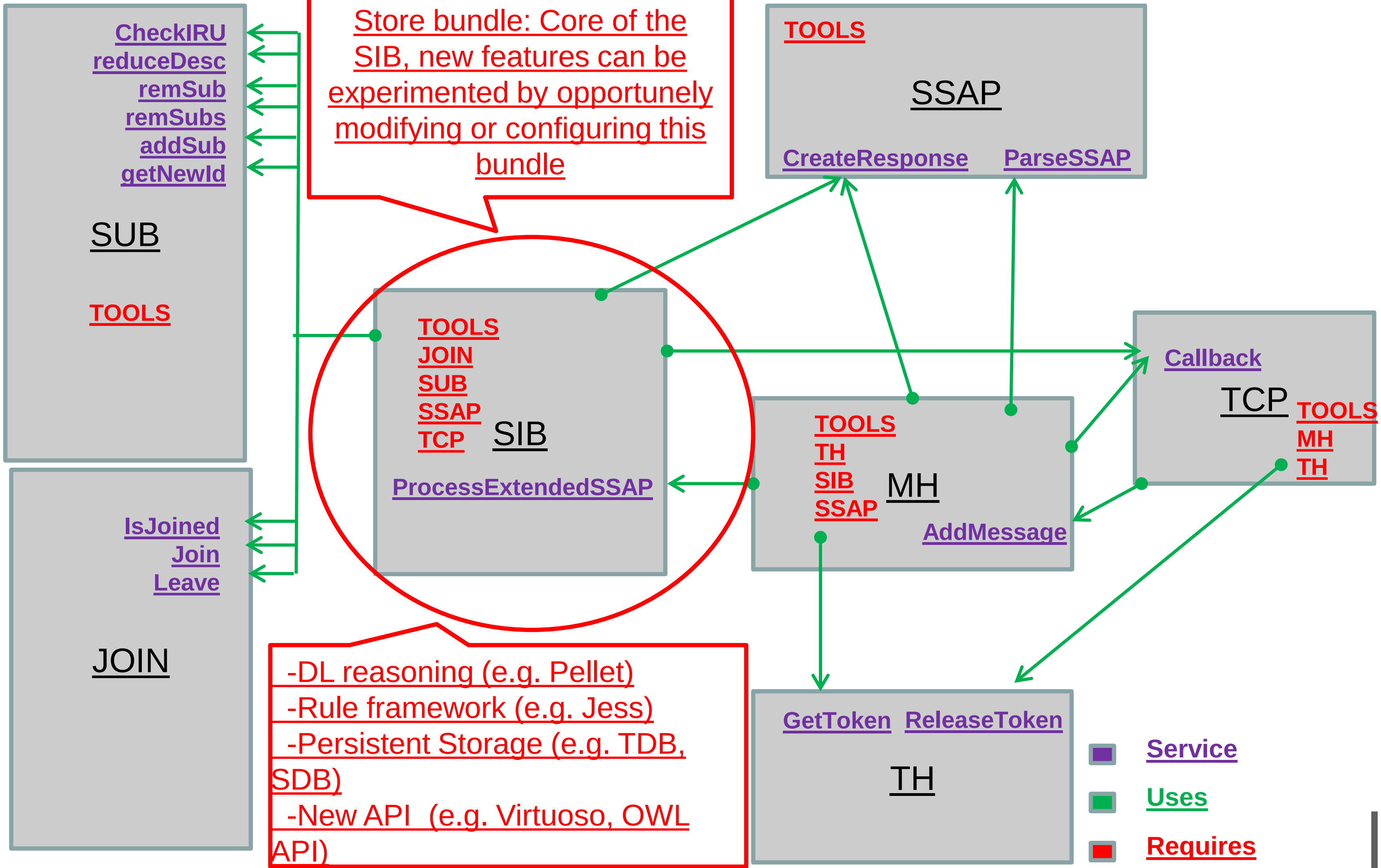
🟥 Requires

Internal architecture



SSAP bundle: Extensions and modifications of the SSAP protocol can be supported by modifications to this bundle and nothing else in the SIB

Internal architecture



Store bundle: Core of the SIB, new features can be experimented by opportunely modifying or configuring this bundle

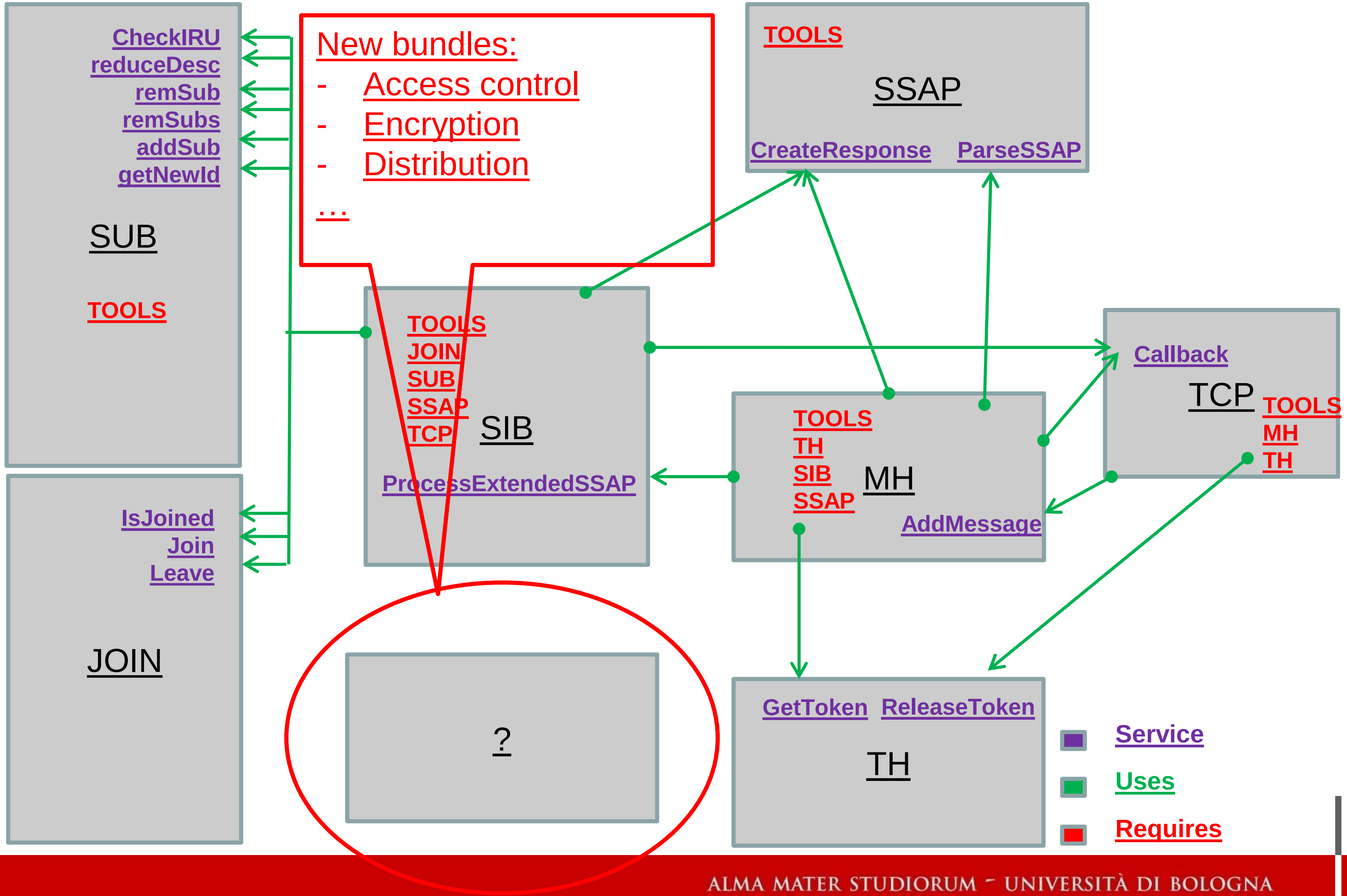
-DL reasoning (e.g. Pellet)
-Rule framework (e.g. Jess)
-Persistent Storage (e.g. TDB, SDB)
-New API (e.g. Virtuoso, OWL API)

🟪 Service

🟩 Uses

🟥 Requires

Internal architecture



SUB

CheckIRU
reduceDesc
remSub
remSubs
addSub
getNewId

TOOLS

JOIN

IsJoined
Join
Leave

New bundles:

- Access control
- Encryption
- Distribution

…

SIB

TOOLS
JOIN
SUB
SSAP
TCP

ProcessExtendedSSAP

?

SSAP

TOOLS

CreateResponse
ParseSSAP

MH

TOOLS
TH
SIB
SSAP

AddMessage

TCP

Callback

TOOLS
MH
TH

TH

GetToken
ReleaseToken

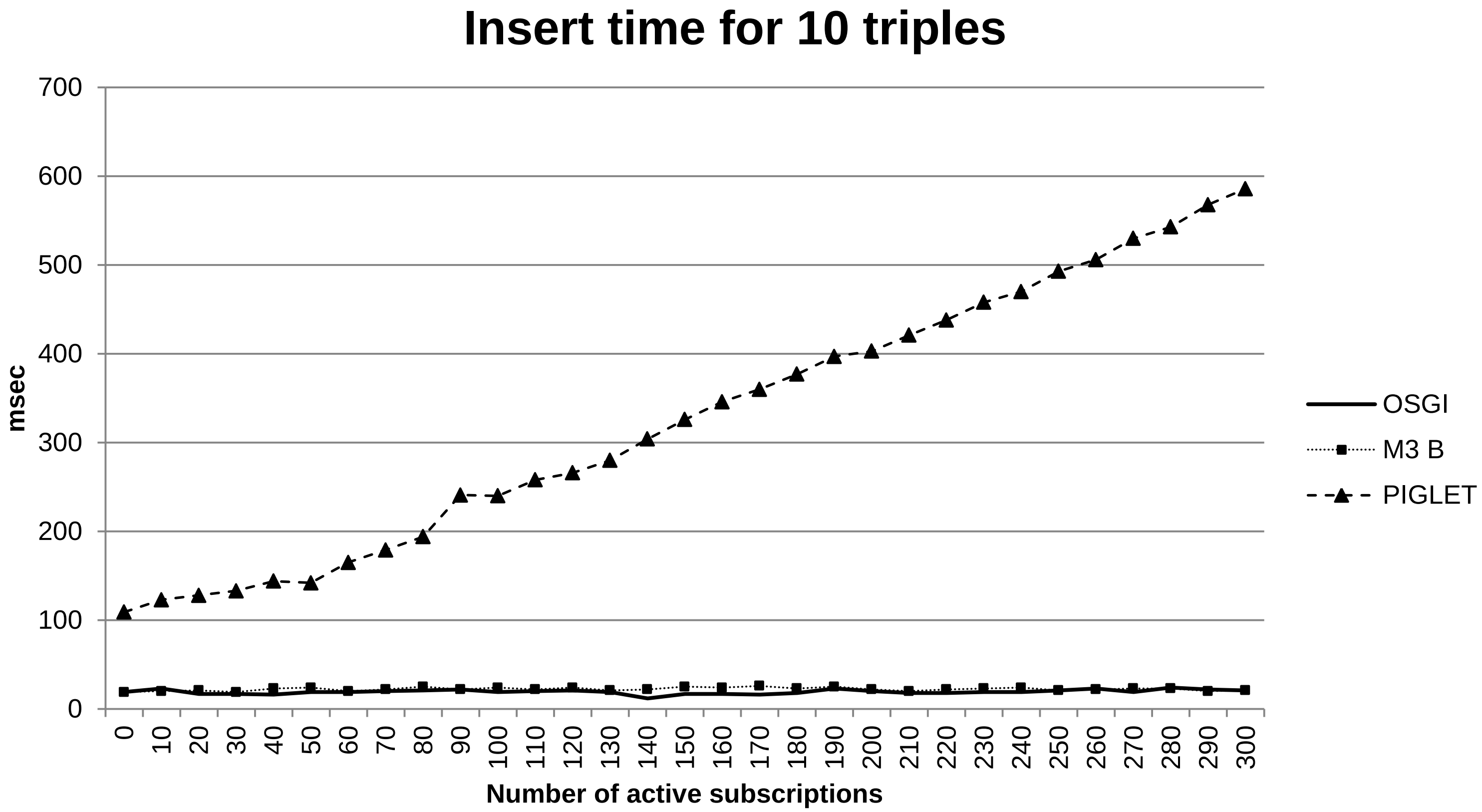
🟪 Service
🟩 Uses
🟥 Requires

Performance evaluation

- Performance of primitives supported by the internal RDF store match in optimal conditions the performance of the store itself (i.e. Jena Framework for RDF graphs)

- Interesting topic is to study SIB specific functionalities (i.e. not originally supported by the internal store) and how they affect the other primitives

Performance with respect to subscription patterns



Conclusion

- New SIB implementation running on most popular operating systems

- New features implemented and tested in prototype versions

- Performance comparable to the reference implementation which is more optimized, but based on a monolithic approach

- Simple adaptation to possible future requirements among which:
  - new SSAP version
  - new primitives
  - security