Design of Semantic Information Broker for Localized Computing Environments in the Internet of Things

Ivan V. Galov, Aleksandr A. Lomov, Dmitry G. Korzun

Petrozavodsk State University
Department of Computer Science

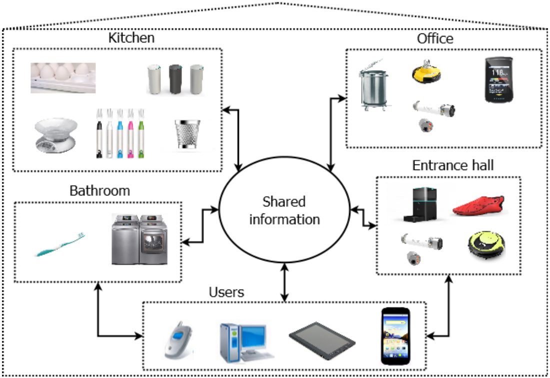
Supported by the Ministry of Education and Science of the Russian Federation and by the Russian Fund for Basic Research

AMICT'2015 Conference

May 14, 2015, Petrozavodsk, Russia

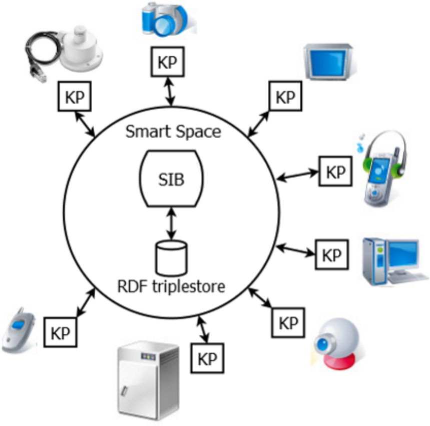
Introduction: Internet of Things (IoT)

- Multitude of physical and digital objects in our daily life
- Localized IoT environments appear everywhere
- Environment inhabitants perceive “smart services”



Smart Spaces: The M3 Architecture

- Multidevice, Multidomain, Multivendor
- Infrastructure: Semantic Information Broker (SIB) maintains smart space content in RDF triples
- Application: Knowledge Processors (KPs, agents) run on IoT devices
- Interaction: Blackboard and Pub/Sub
- Smart space: KPs share ad-hoc knowledge and reason over it to construct services



Existing SIB Implementations

- Smart-M3 SIB: the first official prototype

  J. Honkola, H. Laine, R. Brown, and O. Tyrkkö, “Smart-M3 information sharing platform” (2010)

- RIBS: targets resource limited devices

  J. Suomalainen, P. Hyttinen, and P. Tarvainen, “Secure information sharing between heterogeneous embedded devices” (2010)

- OSGi SIB: higher level of modularity and portability (Java-based)

  D. Manzaroli, L. Roffia, T. S. Cinotti, E. Ovaska, P. Azzoni, V. Nannini, and S. Mattarozzi, “Smart-M3 and OSGi: The interoperability platform” (2010)

- RedSIB: evolution of Smart-M3 SIB with Redland triplestore

  F. Morandi, L. Roffia, A. DElia, F. Vergari, and T. S. Cinotti, “RedSib: a Smart-M3 semantic information broker implementation” (2012)

SIB Implementations: Properties

	Smart-M3 SIB	**RIBS**	**OSGi SIB**	**RedSIB**
Language	C	ANSI C	Java	C
Triplestore	Piglet	Bitcube	Jena	Redland
Features	glib library, SSAP, WQL	lightweight, KSP, constant access latency	SPARQL and reasoning support	improved subscription, SPARQL support
Drawbacks	no SPARQL support, performance problems	cubical memory consumption	resource-demanding, incompatible with other SIBs	performance problems

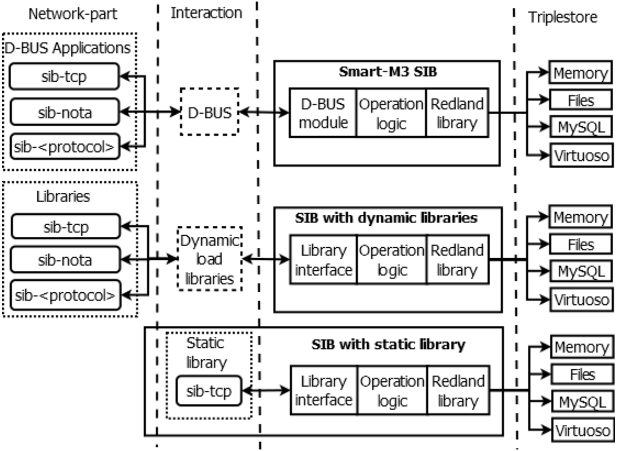
Research prototypes, unsuitable for localized IoT environments

Crucial SIB Properties

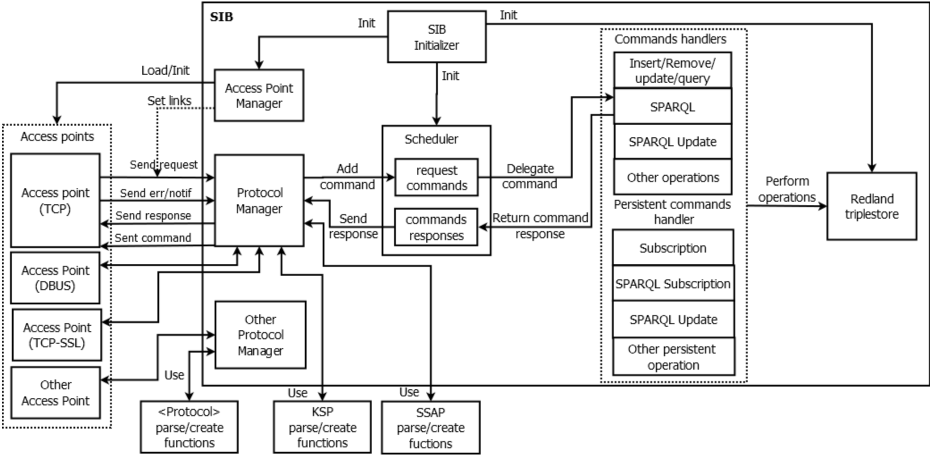
- *Simplicity*: SIB architecture is easy to elaborate, evolve and understand by third-party developers.
- *Extensibility*: SIB architecture provides a modular way of enhancing the functionality.
- *Dependability*: SIB operation is resilient. SIB runs continuously for lengthy time periods. In case of failures, SIB recovers its working state.
- *Portability*: Host devices for SIB are diverse. Traditional Linux and Windows based systems as well as embedded systems (e.g., OpenWrt on routers).

Redesigning: Our Approach

- Based on RedSIB
- Eliminated D-BUS
- Plug-in approach: dynamic libraries
- Modular architecture
- Qt framework



Renewed SIB Architecture



Smart Space Access Protocols

SSAP: join, leave, insert, remove, update, (un)subscribe

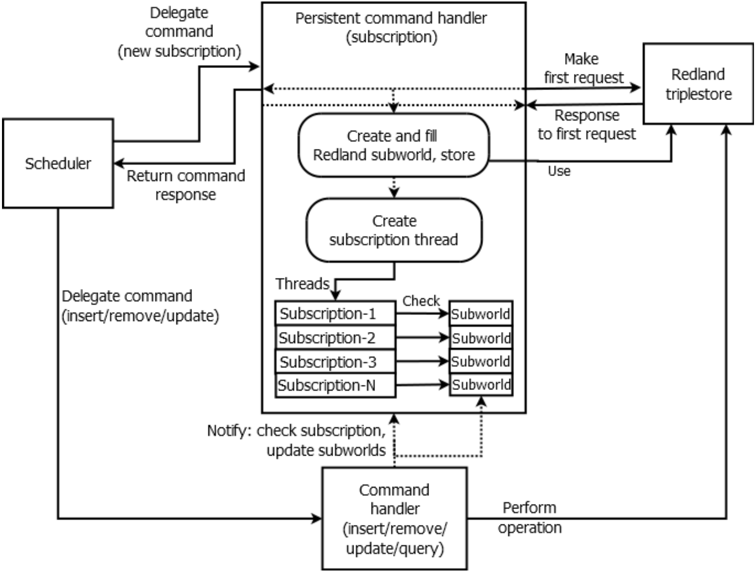
KSP differences:

- compact binary format;
- transactions are based on the SPARQL 1.1 (and SPARQL UPDATE) only;
- no join and leave operations;
- possibility to define the maximum size for SIB response;
- additional persistent operations, which continuously change the smart space content.

Spectrum of Supported Access Operations

Type		Operations
Basic operations (SSAP)	Session management	Join, Leave
	Content access and management	Instant Query, Insert, Remove, Update
	Persistent operations	Subscribe, Unsubscribe
Extended operations		Persistent Insert, Remove, Update
		SIB configuration rules
SPARQL operations	SPARQL	SELECT, CONSTRUCT, ASK, DESCRIBE
	SPARQL Update	INSERT, DELETE, INSERT DATA, DELETE DATA

Subscription Mechanism (as in RedSIB)


Delegate command (new subscription)
Persistent command handler (subscription)
Make first request
Redland triplestore
Response to first request
Scheduler
Return command response
Create and fill Redland subworld, store
Use
Create subscription thread
Threads
Delegate command (insert/remove/update)
Check
Subscription-1
Subscription-2
Subscription-3
Subscription-N
Subworld
Subworld
Subworld
Subworld
Notify: check subscription, update subworlds
Command handler (insert/remove/ update/query)
Perform operation

Properties of renewed SIB

- *Simplicity*: functional allocation into modules, D-BUS is eliminated.
- *Extensibility*: modular architecture allows to extend SIB functionality (new protocols, operations, rules).
- *Dependability*: SIB implementation takes into account problems of other SIBs. Code is based on Qt framework which contributes dependability.
- *Portability*: D-BUS removal and cross-platform Qt framework allows to run SIB on Windows and Linux machines as well as various embedded devices.

Conclusion

- Renewed SIB design for the smart space applications development
- Simplicity, extensibility, dependability and portability of SIB
- Compatibility with previous Smart-M3 applications for Smart-M3 SIB and RedSIB
- New opportunities for application development due to advanced smart space access operations

Thank you for attention

E-mail: galov@cs.karelia.ru