Evaluation of Program Code of Smart-M3 Knowledge Processors Developed Using the SmartSlog Tool

Aleksandr A. Lomov, Dmitry G. Korzun

The work is supported by project 1481
from the basic part of state research assignment # 2014/15
Of the Ministry of Education and Science of the Russian Federation
and by project # 14-07-00252

16th FRUCT conference
October 30, Oulu, Finland

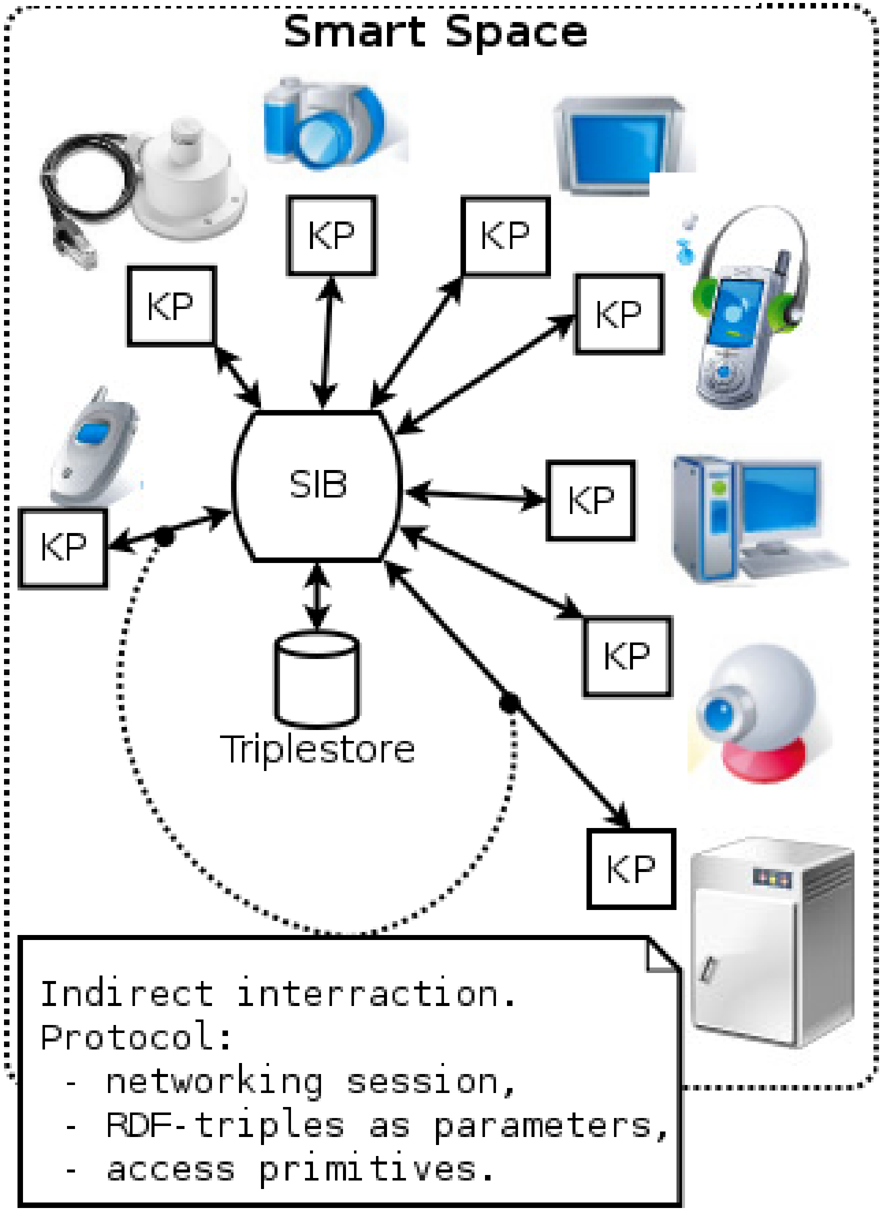
Smart-M3 platform and SmartSlog development tool

Smart-M3 platform:

- Smart Space deployment in IoT.
- Agents – knowledge processors (KP)
- KP Interface (KPI) for KPIs to access SmartSpace.
- Semantic Information Broker (SIB) – access to shared information (RDF-triplestore)

Software SmartSlog tool:

- High-level KP agent development.
- Generation of ontology library (SmartSlog library).
- Programming mechanisms for agent logic.



```
Indirect interraction.
Protocol:
  - networking session,
  - RDF-triples as parameters,
  - access primitives.
```

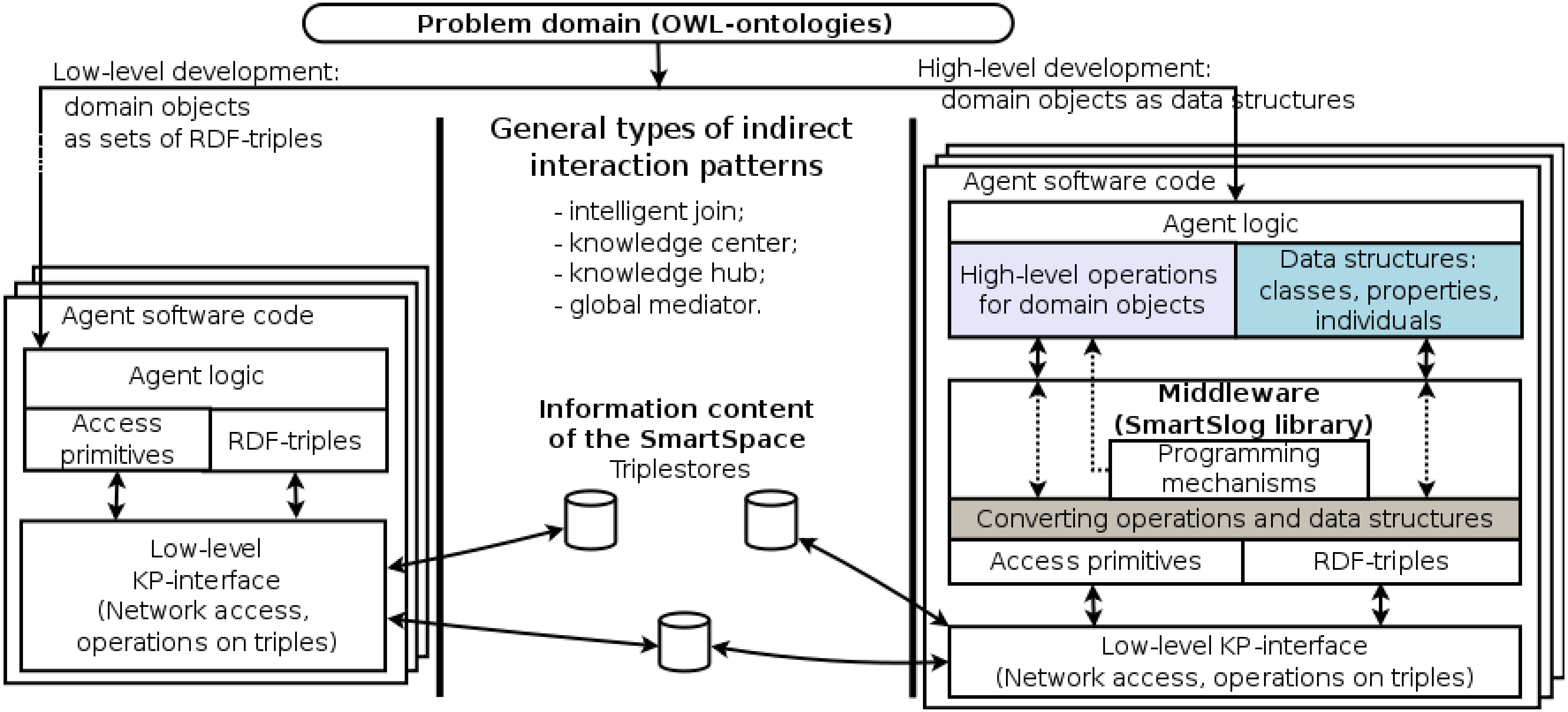
Programming of indirect interaction of agents

Approaches of KP development:

Low-level (RDF-triple), High-level (OWL ontologies)

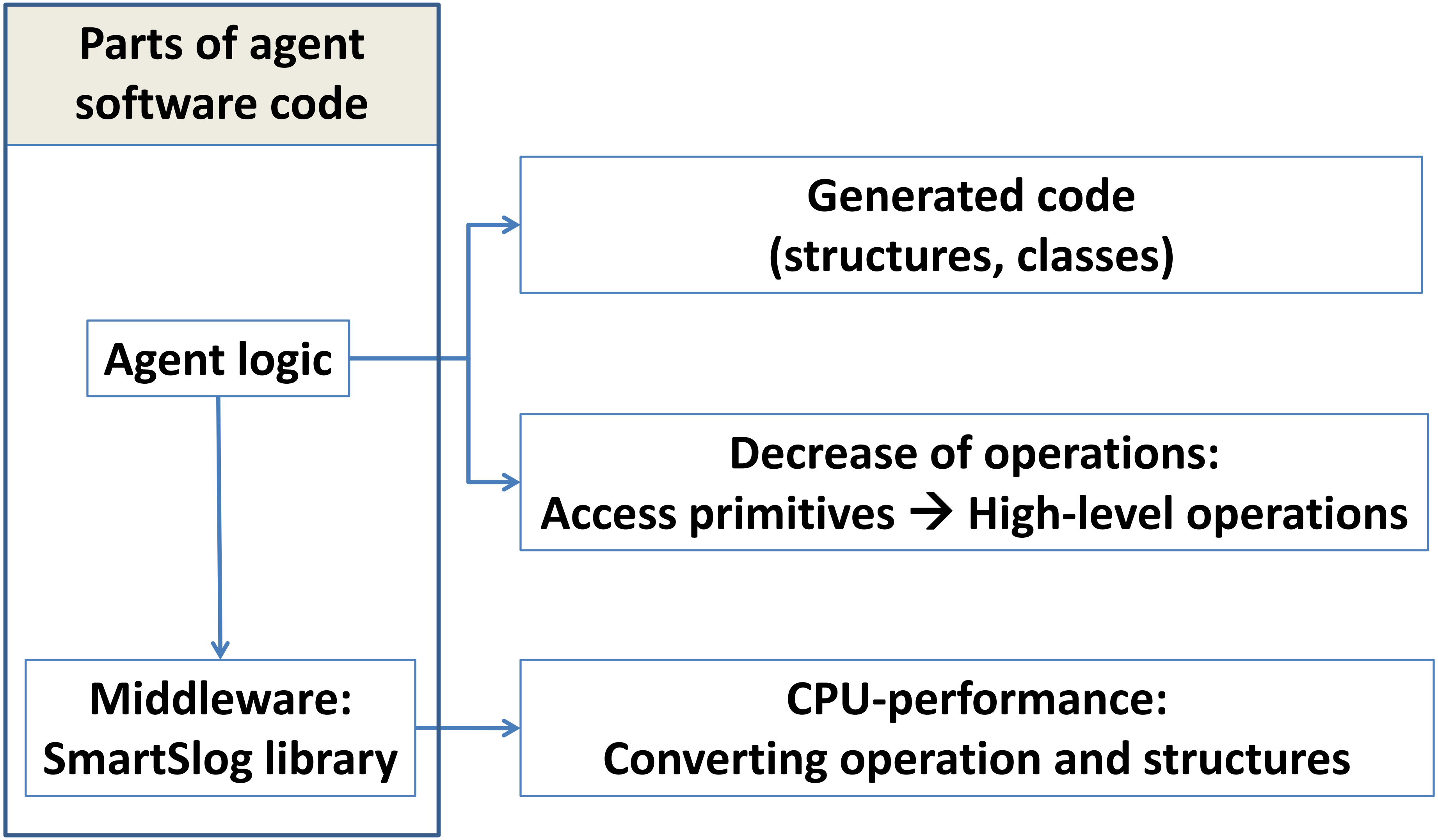
Evaluation:

generated code, operation in the agent logic and CPU-performance



Problem domain (OWL-ontologies)
Low-level development:
domain objects
as sets of RDF-triples
High-level development:
domain objects as data structures
General types of indirect
interaction patterns
- intelligent join;
- knowledge center;
- knowledge hub;
- global mediator.
Agent software code
Agent logic
Access primitives
RDF-triples
Low-level KP-interface (Network access, operations on triples)
Information content
of the SmartSpace
Triplestores
Agent software code
Agent logic
High-level operations for domain objects
Data structures: classes, properties, individuals
Middleware (SmartSlog library)
Programming mechanisms
Converting operations and data structures
Access primitives
RDF-triples
Low-level KP-interface (Network access, operations on triples)

Evaluation ways



Parts of agent software code

Agent logic

Middleware: SmartSlog library

Generated code (structures, classes)

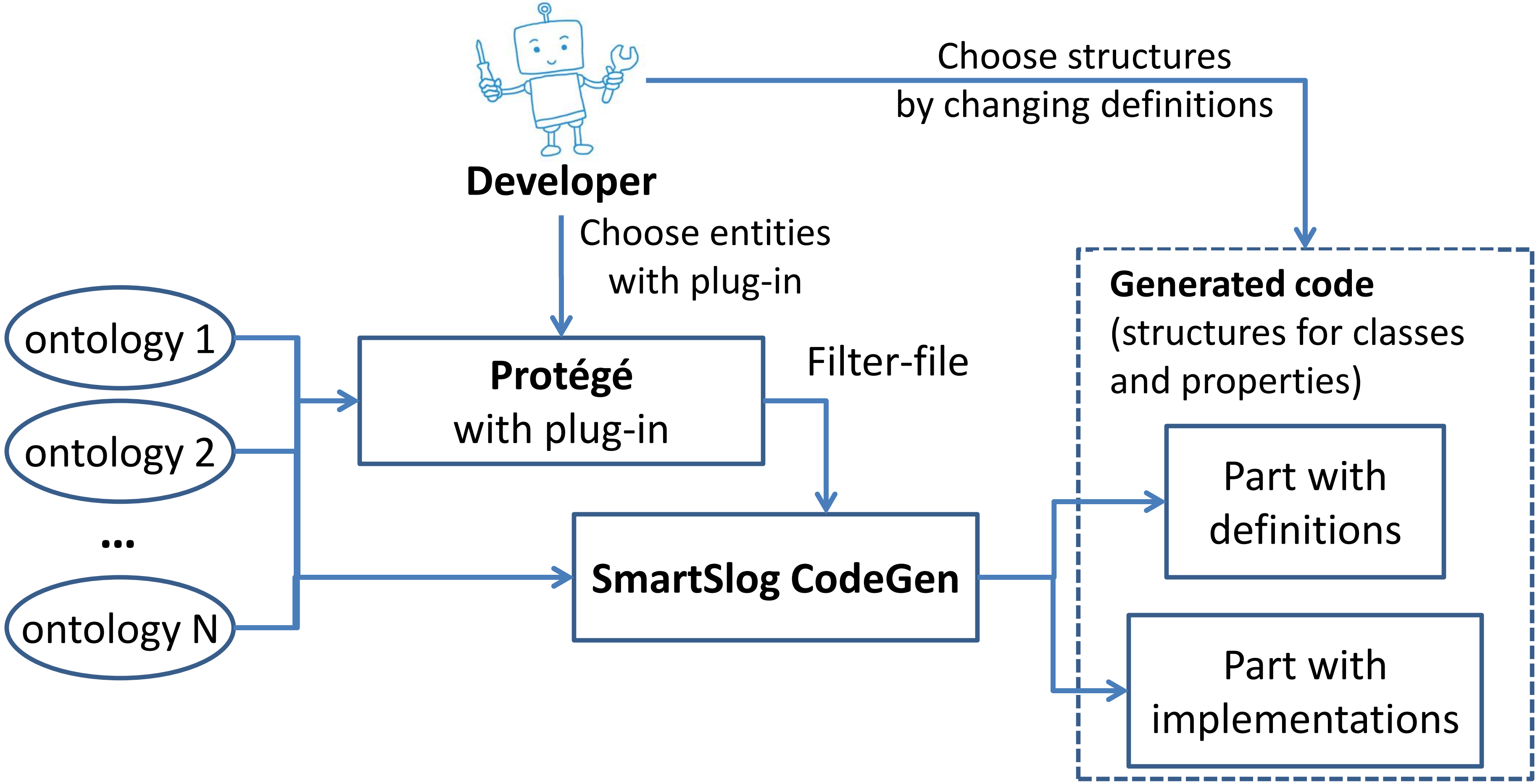
Decrease of operations: Access primitives → High-level operations

CPU-performance: Converting operation and structures

SmartSlog: Generated Code for Developer

Choice of ontologies entities for KP's interaction:

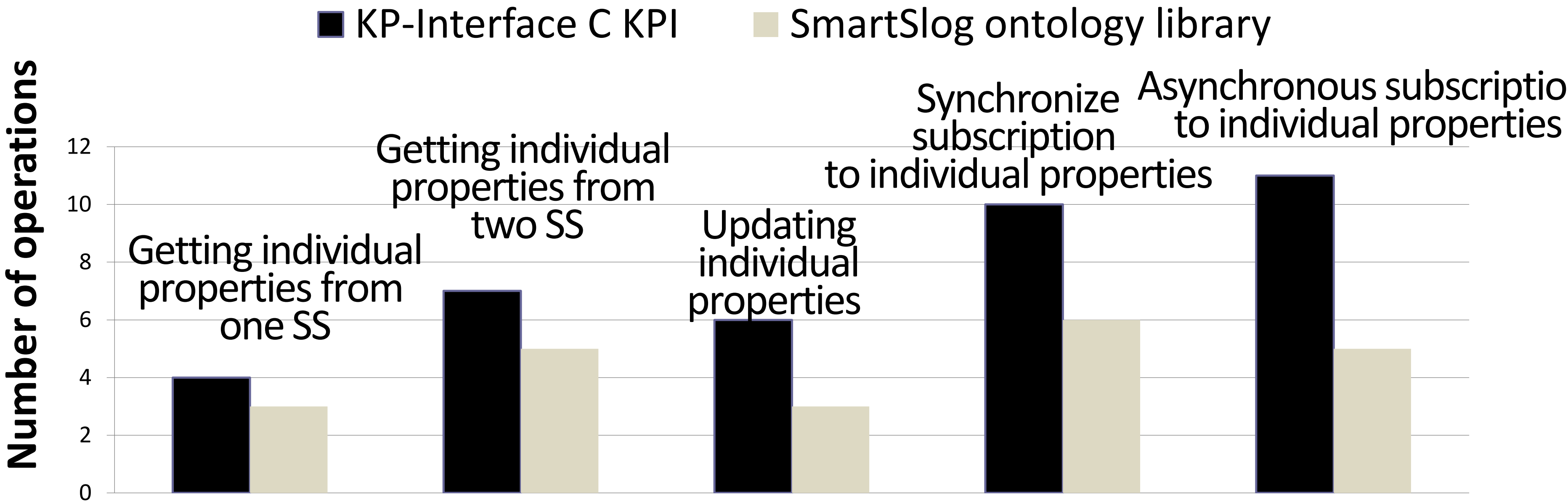
- with Protégé while modeling ontologies
- in the generated code with definitions of structures



Measure the amount of programming operations

Operation - one complete action (creating individual or set of triples)

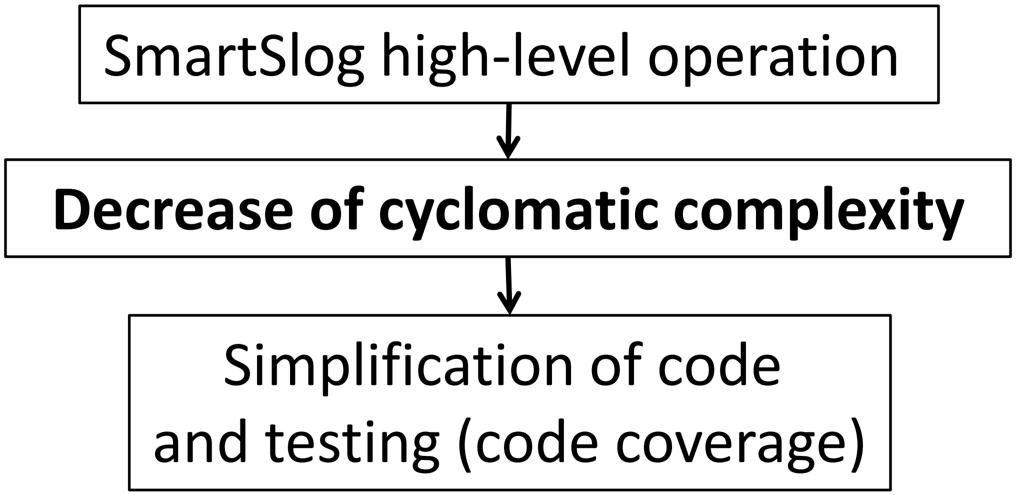
The average reduction in the number of operations to program is about 39%.



Template sets of operation for indirect agets interaction.

Decrease of cyclomatic complexity

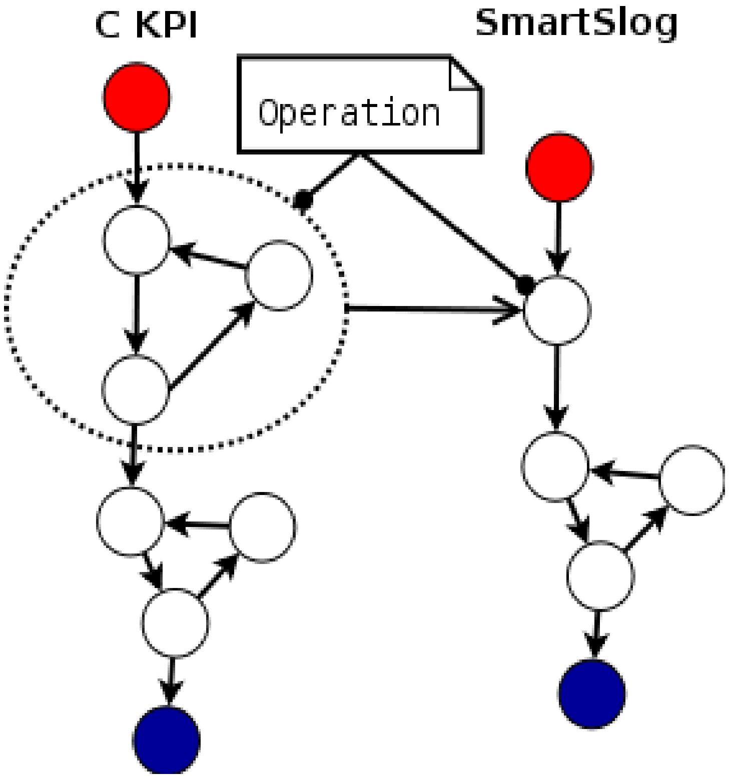
Measures the number of linearly independent paths through a program's source code.



SmartSlog high-level operation

Decrease of cyclomatic complexity

Simplification of code and testing (code coverage)

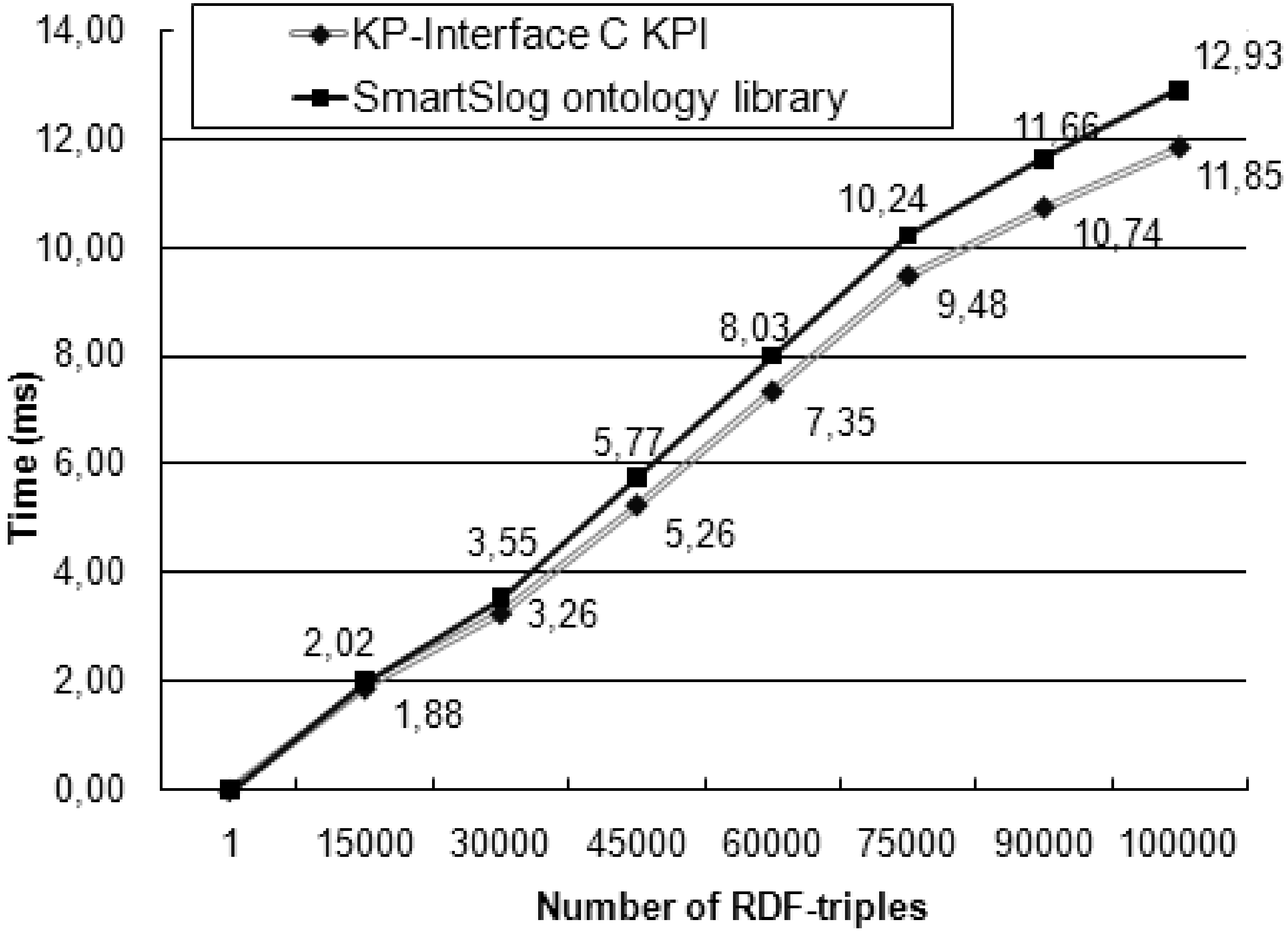


Example implemetation	C KPI		SmartSlog ontology library		
	Hello World		Hello World	GPS	
	Without subscription	Synchronous subscription	Asynchronous subscription	Asynchronous subscription	Connection reconnect
Lines of code	49	144	68	173	194
Cyclomatic complexity	13	37	19	38	40

CPU-performance for middleware: low decrease

Extra CPU resources: High-level operations and structures ↔ **convert** ↔ Access primitives and RDF-triples for KPI

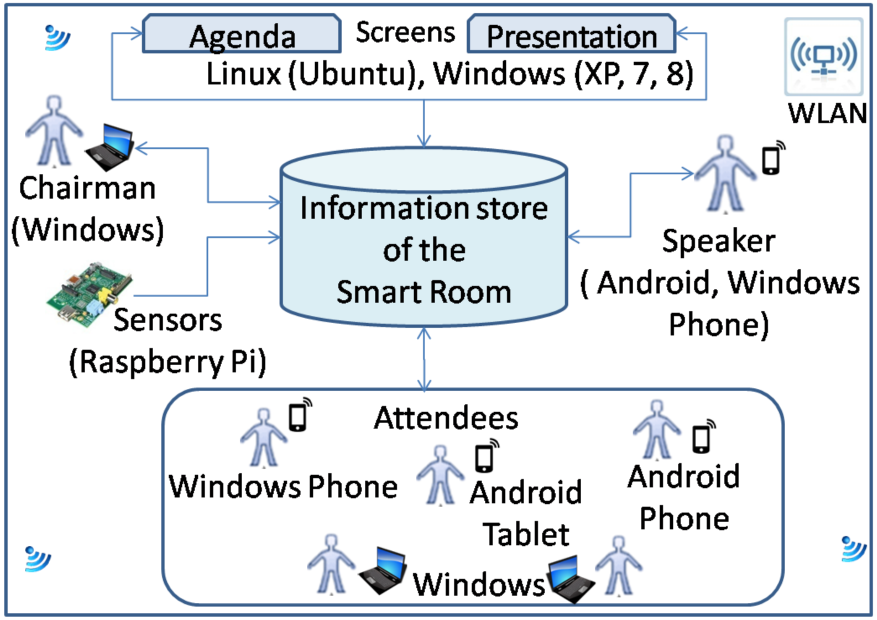
Average decrease of performance is 7% (0.15 ms).



Example of development: SmartRoom system

Specifics of the system:

- Several ontologies
- Many devices
- Different subscriptions
- Network state checking



Platform / Language	Windows	Windows Phone	Linux systems	Mac OS	Android
ANSI C (C KPI)	+	–	+	+	+ (Java code calls ANSI C code)
C# (C KPI)	+ (C KPI adapter)	+ (C KPI adapter)	Using Mono framework		–
C# (C# KPI)	+	+			–

Further experiments with SmartSlog

Metrics:

- **Halstead** metric counts operators, keywords (return, if, continue), identifiers, and constants
- **Jibs** metric is defined as saturation of the program code with such expressions as IF-THEN-ELSE

KP comparison:

- Different languages of low-level KPs
- More complex KPs for experiments with the same functionality based on low-level KPIs and SmartSlog

Thank you!