SmartSlog 0.3x: New Ontology Library API and Optimization

Dmitry G. Korzun, Alexandr A. Lomov, Pavel I. Vanag
Department of Computer Science
Petrozavodsk State University, PetrSU
Petrozavodsk, Russia
Email: {dkorzun, lomov, vanag}@cs.karelia.ru

Abstract

A Smart-M3 semantic information broker (SIB) maintains its smart space in low-level terms of triples based on resource description framework (RDF). SmartSlog is a tool for constructing Smart-M3 knowledge processors (KPs) in high-level terms of OWL classes, properties, and individuals. SmartSlog maps OWL description to an ontology library (in ANSI C). Such a library provides API to communicate with SIB as well as data structures to represent ontology elements in KP code. In this extended abstract we describe the state of SmartSlog release 0.3x and what we plan to go beyond. In contrast to the previous releases, new SmartSlog API moves to the ontology subgraph concept to define patterns for matching between KP queries and the smart space content. Scribo architecture 0.3x becomes more capable for useful ontology manipulations in the code generation process as well as for various control and optimization features. As in the previous releases, SmartSlog keeps in focus the problem of device capacity and provides modest code in respect to performance and interoperability.

Extended Abstract

A smart space is a virtual, service-centric, multi-user, multi-device, dynamic interaction environment that applies a shared view of resources [1]. Information conforms to ontological representation with subject–relation(predicate)–object triples as in semantic web [2]. Triples are represented using Resource Description Framework (RDF). A number of devices may access information via semantic information brokers (SIBs), which also support information reasoning. Smart-M3 [3] (Multidomain, Multidevice, and Multivendor) is an open software platform [4] that implements the smart space concept.

A smart space application consists of one or more knowledge processors (KPs) running on various users' devices. KPs act cooperatively forming a publish/subscribe system [5]. Each KP can be thought as an agent using the smart space as a shared knowledge space. The KPs produce (insert, update, remove) and/or consume (query, subscribe, unsubscribe) information in a smart space. The smart space access protocol (SSAP) implements the SIB $\leftrightarrow$ KP communication, using operations with RDF content as parameters. A KP may provide information for the smart space and use information provided by other KPs. The information content is not restricted—it may be information relating to the physical environment, to the KPs themselves, or anything. Thus, multiple KPs from multiple users may share ad-hoc information across numerous domains, enabling cross-domain and cross-platform interoperability.

Real-life scenarios often involve a lot of information, which leads both to largish ontologies and complex instances that the KPs need to handle. Programming KPs on the level of SSAP operations and RDF triples bring unnecessary complexity for the developers, who have to divert effort for managing triples instead of concentrating on the application logic. The OWL representation of knowledge as classes, relations between classes, and properties maps quite

well to object-oriented paradigm in practice. Therefore, it is feasible to map OWL classes into OO classes and instances of OWL classes into objects in programming languages. This approach effectively binds the subgraph describing an instance of an OWL class to an object in a programming language.

We develop the SmartSlog ontology library generator tool [6], [7], our solution for efficient constructing Smart-M3 KPs by programming with domain concepts that encoded in the relevant ontology. SmartSlog is an ANSI C library generator for Smart Space ontology [7]. The generator maps an OWL ontology description to ANSI C library, abstracting the ontology and communication with SIB in KP code.

SmartSlog library simplifies construction of KP code. The code manipulates with ontology classes, relations, and individuals using predefined data structures and library API. The number of domain elements in KP code is reduced compared with the low-level triple-based scheme. The API are generic, hence does not depend on concrete ontology; all ontology entities appear as arguments in API functions. Search requests to SIB are written compactly by defining only what you know about the object to find (even if the object has many other properties).

The SmartSlog tool is constructed to take into account the limited resources available on small computers such as mobile and embedded devices, which are expected to play a central role in ubiquitous environments. KP code does not need to maintain the whole ontology as unused entities can be removed. Ontology library tries to keep the memory footprint minimal by freeing memory immediately after it is not needed anymore. Even if a high-level ontology entity consists of many triples, its synchronization with SIB transfers only a selected subset, saving on communication.

An ontology library generator uses a static templates/handlers scheme. Code templates are “pre-code” of data structures that implement ontology classes and their properties. The generator has a set of handlers; each handler transforms one or more templates into final code. Templates and their handlers are device-aware. The dependence is resolved on the level of a mediator library that implements triple-based ontology operations for RDF elements and SIB communication. SmartSlog uses KPI_low [8] as a mediator library, oriented to small embedded devices.

A SmartSlog library consists of two parts: dependent and independent on the given ontology (Figure 1). The ontology-independent part contains API: basic data structures (for generic ontology class, property, and individual) and functions for their manipulation. It implements all high-level ontology entity transformations to low-level triples and vice versa. The CodeGen produces ontology-dependent part that implements all ontology classes and properties as structures in C. Note that the generated code can be optimized by removing ontology entities unused in the KP.

Recently we develop SmartSlog release family 0.3x (release 0.30 appeared at SourceForge in August 2010). The key feature is new generic API based on “patterns” and related optimization of the code. The pattern mechanism allows efficient description of requests for ontology-based filtering and search. In the simplest case, a pattern can be thought as an abstract individual where only a subset of properties is set. Hence, it can be used to find all individuals that match the abstract description. In the general case, a pattern represents an ontology subgraph of individuals. A pattern can be applied locally or in queries to the smart space.

The further SmartSlog development is the following.

Although the high-level SmartSlog architecture is reasonable in general, we need to make more progress in internal technicalities. In particular, the strong dependence on KPI_low

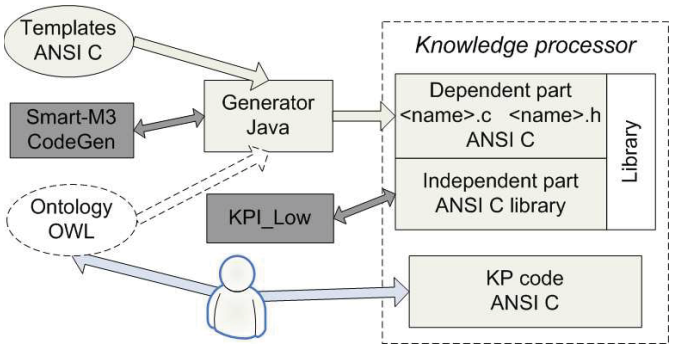

Fig. 1. The SmartSlog ontology library architecture: ontology-dependent and ontology-independent parts

should be eliminated at least partially. A uniform and compact way for supporting mediator triple-based libraries can be provided.

SmartSlog CodeGen needs improvement to support useful ontological properties. In particular, support for multiple ontologies, cardinality, and metainfo in ontology. For optimization purpose, various ontology manipulations can be introduced, e.g., ontology reduction when unneeded elements are eliminated from the code generation. Release 0.3x has some of these features already implemented but they are controlled manually by programmer. More automatic assistance could improve the usability.

Our pattern-based approach for API seems very promising for effective KP programming. We plan to make further research in this direction, e.g., applying name graphs technique [9] in patterns. Our focus is on presenting patterns in API and translating them into queries to SIB. We hope that appropriate processing mechanism will be integrated into upcoming Smart-M3 releases (e.g., SparQL).

Extensions for KPs to initialize and leave the smart space are needed. Recently KP either cleans or leaves all their data in the smart space. Nevertheless, some applications require KPs with the off/on-presence option: a part of data has leaved in the smart space, and each KP easier starts after off-periods.

Both CodeGen and ontology-independent part is the subject to optimization. The former should produce modes data structures for ontology elements. The latter should provide efficient transformations between high-level and low-level ontology elements. An important problem is the synchronization between KPs and SIBs. In some cases, KP can predict which data it needs from the smart space. We plan to develop a set of specific use case scenarios how a KP can participate in its smart space application. Then the performance for each scenario can be evaluated analytically and experimentally.

Since Smart-M3 is actively progressing the SmartSlog maintenance is essential. In particular, such features as SparQL support and a security mechanism can appear in Smart-M3 soon. We also keep in mind the importance of demos for SmartSlog and plan to develop the demo set further, see SmartSlog demos package at SourceForge [7].

ACKNOWLEDGMENT

Authors would like to thank Finnish-Russian University Cooperation in Telecommunications (FRUCT) program for the provided support and R&D infrastructure. We would also like to thank Sergey Balandin, Jukka Honkola, Vesa Luukkala and Ronald Brown from Nokia Research Center for providing feedback and guidance during the construction of the SmartSlog

tool. The special thanks to M3-Weather team (http://oss.fruct.org/wiki/M3-Weather), which actively used SmartSlog and provided many comments.

REFERENCES

[1] I. Oliver, J. Honkola, and J. Ziegler, "Dynamic, localised space based semantic webs," in *Proc. IADIS Int'l Conf. WWW/Internet 2008*. IADIS Press, Oct. 2008, pp. 426–431.

[2] D. Fensel, W. Wahlster, H. Lieberman, and J. Hendler, Eds., *Spinning the semantic web : bringing the World Wide Web to its full potential*. The MIT Press, 2005.

[3] J. Honkola, H. Laine, R. Brown, and O. Tyrkkö, "Smart-M3 information sharing platform," in *The 1st Int'l Workshop on Semantic Interoperability for Smart Spaces (SISS 2010) in conjunction with IEEE ISCC 2010*, Jun. 2010.

[4] "Download Smart-M3 software for free at SourceForge.net," Release 0.9.4beta, Sep. 2010. [Online]. Available: http://sourceforge.net/projects/smart-m3/

[5] R. Baldoni, M. Contenti, and A. Virgillito, "The evolution of publish/subscribe communication systems," in *Future Directions in Distributed Computing*, ser. Lecture Notes in Computer Science, vol. 2584. Springer, 2003, pp. 137–141.

[6] D. Korzun, A. Lomov, P. Vanag, J. Honkola, and S. Balandin, "Generating modest high-level ontology libraries for Smart-M3," in *Proc. 4th Int'l Conf. Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM 2010)*, Oct. 2010.

[7] "Download SmartSlog software for free at SourceForge.net," Release 0.30, Sep. 2010. [Online]. Available: http://sourceforge.net/projects/smartslog/

[8] "Download KPI_low software for free at SourceForge.net," Jun. 2010. [Online]. Available: http://sourceforge.net/projects/kpilow/

[9] J. J. Carroll, C. Bizer, P. Hayes, and P. Stickler, "Named graphs, provenance and trust," in *WWW '05: Proceedings of the 14th international conference on World Wide Web*. ACM, 2005, pp. 613–622.