On Semantics in Onto-DIY

Yan Tang Demey[1] and Zhenzhen Zhao[2]

[1] Semantic Technology and Application Research Laboratory (STARLab),
Department of Computer Science,
Vrije Universiteit Brussel, Pleinlaan 2 B-1050 Brussels, Belgium
yan.tang@vub.ac.be
[2] Institut Mines-Télécom, Télécom SudParis
Service RS2M (Département Réseaux et Services Mutimédia Mobiles) – salle D 108 02
9, Rue Charles Fourier, 91000 Evry, France
zhenzhen.zhao@it-sudparis.eu

Abstract. This paper illustrates how semantics is modeled and used in the flexible and idea inspiring ontology-based Do-It-Yourself architecture (Onto-DIY). The semantics part contains three divisions - 1) semantics from the domain ontologies, which are used as the semantically rich knowledge for describing Internet of Things (IoT) components, 2) semantics stored in semantic decision tables, which contain decision rules on top of domain ontologies, and, 3) semantics from user-centric services, which define the software composition for the deployment.

Keywords: ontology, semantic decision table, decision table, user-centric service, Do-It-Yourself, description logic.

1 Introduction

In the past EU ITEA2 Do-it-Yourself Smart Experiences Project[1], we have developed a use case scenario as follows. Mary has a naughty boy called James. He treats Mary's iPhone as his favorite toy. Mary wants to create a small tool called "naughty boy protector" (NBP) using a smart rabbit (e.g., a Nabaztag rabbit[2]), which takes web feeds and speaks. What NBP does is as follows. When James shakes Mary's iPhone, iPhone will send a signal to NBP. When NBP gets this signal, it checks the rules stored in its rule base and fires the relevant rules. As the action of this rule, NBP sends a message to the server of this smart rabbit. After the rabbit gets the web feed, it speaks out loudly. At the end, Mary will reset the rabbit and take back her iPhone.

This use case has been used to design an architecture called Ontology-based Do-it-Yourself architecture (Onto-DIY[3]), which allows end users to create their own

[1] `http://dyse.org:8080`. The authors' work is also supported by OSCB (`www.oscb.be`)
[2] `http://www.nabaztag.com`; its APIs can be found at `http://api.nabaztag.com`
[3] As a result, the demo can be viewed at `http://www.youtube.com/watch?v=O0hbB162HIY`

applications using their own evolving semantics. In this paper, we want to focus on how semantics is modeled and used in Onto-DIY.

2 Semantics in Onto-DIY

There are mainly three semantic divisions while designing Onto-DIY. The first one is the domain ontologies, which are used as the knowledge base for describing Internet-of-Things components, such as iPhone and smart rabbit. Fig. 1 shows how we model the concepts "iPhone" and "Smart Rabbit" in Semantic Decision Rule Language (SDRule-L, [3]), which is an extension to Object Role Modeling language (ORM/ORM2, [1]).

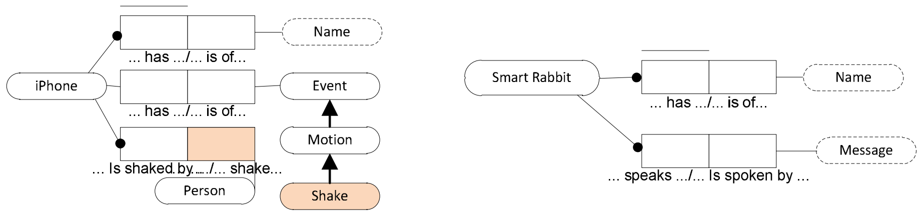

Fig. 1. The concepts defined in the IoT domain ontologies

Following the methodology called "an information analysis method" (NIAM, [5]), the ancestor of ORM, we use the idea of identifying an object type as a LOT (lexical object type) or a NOLOT (non-lexical object type). A LOT, modeled graphically in an eclipse with a dotted line, e.g. "Name" in Fig. 1, is an object that can be uttered and represented in a certain reality or context. A NOLOT, modeled graphically in an eclipse with a solid line, e.g. "iPhone" Fig. 1, is an object that cannot be represented. A NOLOT must be named, presented or referred to using a LOT. A binary fact type containing both a LOT and a NOLOT is called a "bridge type". An example of bridge type is <iPhone, has, is of, Name> (see Fig. 1). Furthermore, we call "has/is of" is the role pair that the objects "iPhone" and "Name" can play with each other.

Note that in Fig. 1, the role "shake" and the object "Shake" are shaded. When we highlight a role and an object in a same color, we call this object is an *objectification* of a role. It means that their meanings have a certain similarity. We use this mechanism to identify and discover tasks and events of a physical object for context-aware computing.

SDRule-L models can be translated into a text that is easy to read. This process is called "verbalization" in [1]. The verbalization of Fig. 1 is shown as follows.

Each iPhone has exactly one Name. An iPhone has an Event, a subtype of which is Motion. Shake is a subtype of Motion. Each iPhone is shaked by at least one Person. Each Smart Rabbit has exactly one Name. Each Smart Rabbit speaks at least one Message.

Fig. 1 is formalized in $\mathcal{SOIQ}$ – a Description Logic (DL [1]) dialect – as follows.

$$IPhone \sqsubseteq \leq 1 has.Name \sqcap \exists has.Name \qquad People \sqsubseteq shake.IPhone$$
$$IPhone \sqsubseteq has.Event \qquad SmartRabbit \sqsubseteq \leq 1 has.Name \sqcap \exists has.Name$$
$$Shake \sqsubseteq Motion \sqsubseteq Event \qquad SmartRabbit \sqsubseteq \exists speaks.Message$$

The second semantic division is stored in semantic decision tables (SDT, [1]), with which we model decision rules using the semantics defined and formalized in domain ontologies. Table 1 is an example. The semantics from an SDT is considered as ontological commitments viewed by a particular business application. In this example, the semantics comes from the annotation using the domain ontologies (see the model in Fig. 1) and instantiation (see C1~6 in Table 1).

Table 1. An SDT of deciding on activate a smart rabbit

Condition	**1**	**2**	**…**
People shake iPhone	Yes	No	…
…			
Action			…
Bunny speaks	Message1		…
…			
C1	$People \equiv \{james, mary\}$	**C4**	$shake(james, marysiPhone)$
C2	$IPhone \equiv \{marysiPhone\}$	**C5**	$Messaqe \equiv \{message1\}$
C3	$SmartRabbit \equiv \{marysRabbit\}$	**C6**	$speak(marysRabbit, message1)$

A non-technical user uses SDTs to model business/application rules that are consistent, complete and ontology-based.

The third semantic division is from user-centric services. User centric service is to apply user-centered design (UCD) process in designing a useful and easy-to-use service, which takes into account user motivation, user requirements, user behaviors, user interactions etc. End-user driven service creation aims to foster the evolution from the absolutely successful user-generated content (UGC) to the user-generated service (UGS) area, to empower end-users' creativity in service generation to achieve better service personalization [6]. The metrics, often termed service composition (back-end) and service mashups (front-end), allow users to play with various service building blocks to create enhanced services. To achieve a better UGS, an intuitive and easy-to-use platform, a use scenario which can promote greater user's motivation, and a sustainable service database, are desired. In return, the semantics, which is extracted from user-centric services, allows end-user to do goal-driven service creation in a technology-agnostic way. Separating technology of a service from its meaning could improve the process of assembling new services.

In order to deploy the SDT illustrated in Table 1, we propose to develop a semantic service creation assistant, e.g., whenever the user picks an existing service, the system should be able to suggest a set of syntactically or semantically related services that can be connected to the existing service (a kind of inter-service communication); or the user can locate a specific service based on a description of all (or part of) its functionality with a near natural language request. For example, we can use it to discover the web service of asking a smart rabbit to speak as shown in Fig. 2.

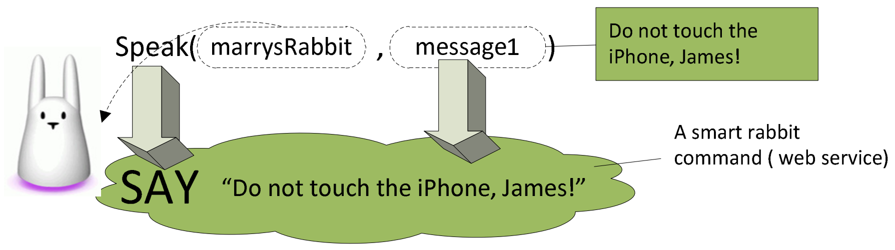
Speak(marrysRabbit , message1)
Do not touch the iPhone, James!
A smart rabbit command (web service)
SAY "Do not touch the iPhone, James!"

Fig. 2. The concepts defined in the IoT domain ontologies

3 Discussion, Conclusion and Future Work

We use SDRule-L and DL to formalize the semantics in Onto-DIY in this paper. Although we do not use other languages, readers should keep in mind that, for IoT applications, it is recommended to use any kinds of conceptual modeling means, as long as they meet the request and technically/conceptually sound.

With regard to the user-centric service creation in this paper, it is interesting to study the following aspects in the future:

- *Context-aware service creation* – tracing end-users' behaviors and help them to organize and filter information in order to provide personalized service
- *Service creation with trust* – assisting end-users to control privacy policies to protect their sensitive data in a nonintrusive manner
- *Social service co-creation* – allowing users to share their existing "Do-It-Yourself" solutions with others in order to "Do-It-Together"
- *Social service venue sharing* – bringing a business concern to Onto-DIY in order to promote software innovation and encourage users to create their solutions and provide to the market

A future refinement could also be done is the aspect of creating an automatic mapping between the semantics in the three semantic divisions illustrated in this paper.

References

[1] Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications, 2nd edn. Cambridge University Press (2007)

[2] Halpin, T., Morgan, T.: Information Modeling and Relational Databases. The Morgan Kaufmann Series in Data Management Systems (2008) ISBN-10: 0123735688

[3] Tang, Y., Meersman, R.: SDRule Markup Language: Towards Modeling and Interchanging Ontological Commitments for Semantic Decision Making. In: Handbook of Research on Emerging Rule-Based Languages and Technologies: Open Solutions and Approaches, USA, ch. V (sec. I). IGI Publishing (2009) ISBN: 1-60566-402-2

[4] Tang, Y.: Semantic Decision Tables - A New, Promising and Practical Way of Organizing Your Business Semantics with Existing Decision Making Tools. LAP LAMBERT Academic Publishing AG & Co. KG, Saarbrücken (2010) ISBN 978-3-8383-3791-3
[5] Wintraecken, J.J.V.R.: The NIAM information analysis method: theory and practice, p. 469. Kluwer Academic Publishers, The University of California (1990)
[6] Zhao, Z., Laga, N., Crespi, N.: The Incoming Trends of End-User Driven Service Creation. In: Telesca, L., Stanoevska-Slabeva, K., Rakocevic, V. (eds.) DigiBiz 2009. LNICST, vol. 21, pp. 98–108. Springer, Heidelberg (2010)