# PRISM: A Knowledge Engineering Tool to Model Collective Behaviors of Real-time IoT Systems

Maryam Rahmani, Junsup Song and Moonkun Lee

Chonbuk National University
567 Beakje-daero Deokjin-gu
Jeonju-si Jeonbuk 54896, Republic of Korea
e-mail:moonkun@jbnu.ac.kr

**Abstract.** This paper presents a knowledge engineering tool, called *PRISM*, to model collective behaviors of real-time IoT systems. PRISM is developed on the ADOxx Meta-Modeling Platform, in order to implement the new notion of a domain engineering method, known as *behavior ontology*. In PRISM, the ontology can be constructed as follows: 1) All the collective behaviors for a domain are defined from *active ontology*, 2) the behaviors are formed in a quantifiably abstracted lattice, called *n:2-Lattice*, and 3) a behavior ontology for the domain can be constructed by merging the n:2-lattices into an integrated lattice. Once the ontology is constructed, each system in the domain can be interpreted with respect to the ontology or lattice. In the paper, the *Emergency Medical Service* (EMS) domain and a smart IoT example for EMS are selected for modeling and interpretation in PRISM. PRISM shows an innovative approach for meta-modeling of domain knowledge as a tool.

**Keywords:** Collective Behavior; Behavior Ontology; PRISM; ADOxx; Lattice

## 1 Introduction

There are strong needs to represent system *behavior*s for each knowledge domain in some collective patterns, especially using tools. However the needs are not easily satisfied due to the following reasons:

1) There were the structural limitations caused by the size of system components and the complexity of their interactions, as well as their composition, causing *state explosion* [1].
2) There were no effective tools to model collective behaviors of systems with the supporting meta-modeling platforms [2].

In order to satisfy the needs partially, this paper presents a tool, called PRISM, with the following method and the meta-modeling support:

1) The method is based on a concept of *behavior ontology* [3]. The approach in this paper extended the previous research [3] for implementation in PRISM. The method solved the state explosion problem with abstracting behaviors.
2) PRSIM is developed on the ADOxx Meta-Modeling Platform [4] to implement the method. The implementation demonstrates the feasibility of the method by the meta-modeling tool.

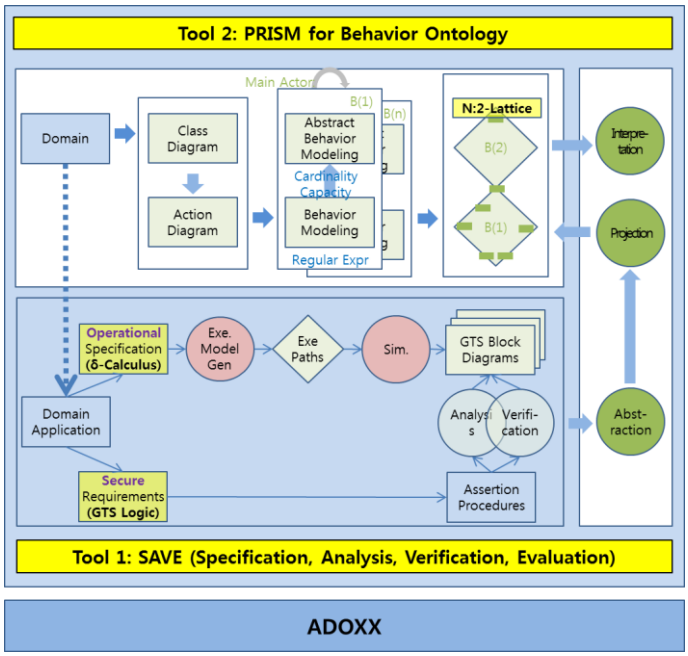


**Fig.** 1 Overview of the Meta-Modeling Method in PRISM

The overview of the approach in PRISM is shown in the top white box of Fig. 1. It consists of the following steps:

1) A class hierarchy of a domain is constructed based on *Active Ontology* [5], where all the actors of a domain and their interactions are defined as classes and relations, respectively.
2) All the collective behaviors of the domain are defined in *regular expression*, where each behavior is defined as a sequence of interactions among actors. The behaviors can be organized in a hierarchical order based on their inclusion relations, forming a special lattice, called *n*:2-*Lattice* [6].
3) All the behaviors are quantifiably abstracted with a notion of cardinality and capacity for actors.
4) A behavior ontology for the domain is constructed by merging the n:2-lattices into an integrated lattice, based on quantifiably common actors.

Once a final lattice for a domain is constructed, it can be used to interpret the collective behaviors of systems in the domain as follows:

1) Behavior extraction from SAVE [7]: The shaded box below the white box just mentioned in Fig. 1 shows the method for specification and verification of operational and secure requirements for a system with called δ-calculus [8] and GTS Logic [9]. The method is also implemented on the ADOxx Meta-Modeling Platform as a tool, called SAVE. As a result of the simulation from the specification, the behavior of the system can be extracted. However we don't know yet what the collective pattern of each behavior is.

2) Behavior projection and interpretation to PRISM: As shown in the right white box of the figure, we can project the behaviors of the system, after restructuring the raw behaviors into the abstract ones, and interpret the behaviors in the patterns of the lattice as the behavior ontology for the system.

In order to demonstrate the feasibility of the approach, the *Emergency Medical Service* (EMS) domain is selected for modeling its collective behaviors in PRISM. Further a smart IoT example for EMS is selected to interpret its collective behaviors on PRISM by projection and interpretation. The EMS example shows that the method is very effective and efficient to construct a hierarchy of collective behaviors in the lattice as the behavior ontology, as well as projection and interpretation. Compare to other approaches for modelling behaviors and analyzing patterns of the behaviors [2], our method can be considered to be innovative in representing the behaviors with collective patterns by the n:2-Lattice. Further PRISM demonstrates the efficiency and effectiveness of feasibility of the method as a tool.

This paper is organized as follows. Section 2 overviews the PRISM tool. Section 3 shows the approach in steps with the EMS domain in PRISM. Section 4 shows behaviors for the smart IoT example for EMS simulated in SAVE and their projection to PRISM for interpretation. Finally, conclusions and future research will be made in Section 5.

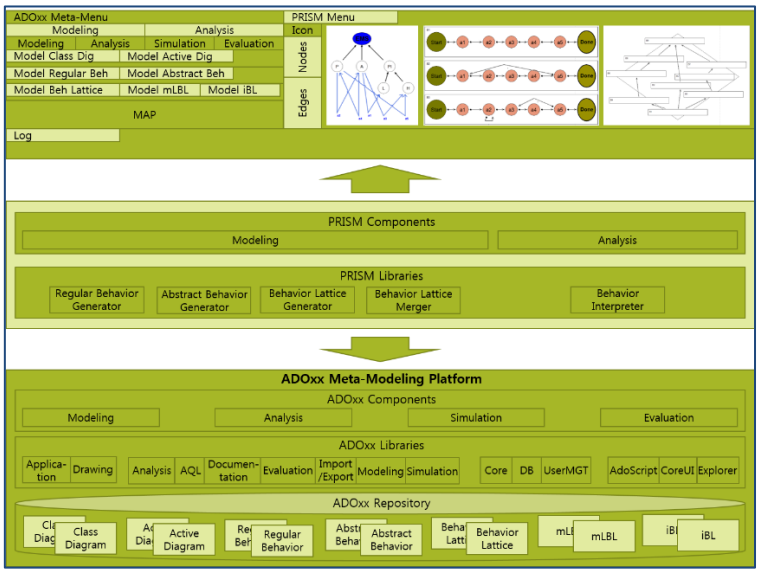


**Fig. 2** The Views and Architecture of PRISM on ADOxx

## 2 PRISM

As stated, the PRISM tool is developed on the ADOxx Meta-Modeling Platform [5]. ADOxx was originally developed and released by the OMiLAB of the University of Vienna, and is known as one of the most innovative meta-modeling tools to model

many modeling methods. There are total 42 open models developed on ADOxx and open to the public for non-profit applications [9].

The architecture and modeling views of PRISM is shown in Fig. 2. The graphical representations of the models in PRISM are designed by the ADOxx Development Tool, and the procedures of its components are built from the ADOxx libraries. The detailed algorithms of the procedures are programmed in the ADOScript language.

The figure shows three system layers of PRISM implemented on ADOxx as follows:

1) ADOxx Platform: ADOxx provides three layers to implement mechanisms and algorithms for PRISM as follows:
   i) First Layer: Beside the pre-defined functionality, which is a basic set of features most commonly used by modeling tools, it is possible to configure the basic features for modeling definitions.
   ii) Second Layer: To implement scripts like the stated ADOScripts, it provides approximately 400 APIs for the ADOxx components. Those APIs enable the generation of objects, editing of their properties, etc.
   iii) Third Layer: There are three ways of interactions with ADOxx from outside. The simple interaction is by exporting and importing XML files.
2) PRISM Components: PRISM uses those of the second layer of ADOxx to implement the basic components of PRISM as follows:
   i) Regular Behavior Generator (RBG): This is an engine to generate a set of regular, that is, basic behaviors from Active Ontology.
   ii) Abstract Behavior Generator (ABG): This is an engine to abstract a set of the regular behaviors from i) with respect to cardinality and capacity.
   iii) Behavior Lattice Generator (BLG): This is an engine to generate a behavior lattice from the abstract behaviors from ii).
   iv) Behavior Lattice Merger (BLM): This is an engine to merge two behavior lattices into an integrated lattice with respect to the same main actors with different cardinalities. It forms a lattice of lattices.
   v) Behavior Interpreter (BI): This is an engine to input behaviors from SAVE and project them onto the final lattice of its domain for interpretation in the collective behavior patterns of the domain in the form of the lattice.
3) PRISM Modelers: PRISM uses the functionalities of the first layer of ADOxx to implement the graphical elements and attributes of its graphic models as follows:
   i) Class Diagram (CD): The model to define the architecture of classes for a domain.
   ii) Active Diagram (AD): The model to define the active ontology of the domain from i).
   iii) Regular Behavior (BB): The model to define a set of regular behaviors from Active Ontology from ii).
   iv) Abstract Behavior (AB): The model for a set of abstract behaviors generated automatically from that of basic behaviors by ABG from ii) of 2).
   v) Behavior Lattice (BL): The model for a behavior lattice generated automatically from a set of abstract behaviors by BLG from iii) of 2).

vi) Merged Lattice of Behavior Lattices (mLBL): The model for a lattice of behavior lattices merged automatically from a set of behavior lattices by BLM from iv) of 2).
vii) Interpreted Behavior Lattice (iBL): The model for a behavior lattice with interpretation of behaviors of a system in the domain of the lattice projected automatically by BI from v) of 2).

Section 3 shows how PRISM works in steps, with the EMS domain example.

# 3 Approach

This section presents each step of modeling in PRISM with the EMS domain. EMS is the system where, in case of traffic and car accidents, the drivers or patients from the accidents are transported to proper medical institutes under control of the 911.

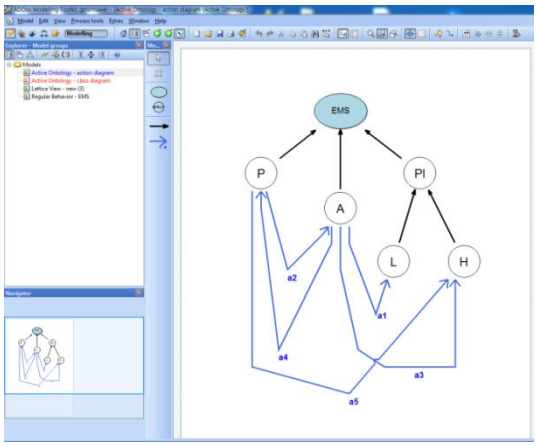


**Fig. 3** Active Ontology for EMS Domain

## 3.1 Step 1: Active Ontology

The first step is to design active ontology for the EMS example. Active ontology consists of classes and subclasses in the domain, including their interactions.

The EMS domain example contains four classes: *Ambulance* (A), *Patient* (P), and *Place* (PL). Note that *Place* contains *Location* (L) and *Hospital* (H) as subclasses. Fig. 3 shows the active diagram for the active ontology as follows:

1) *Actors*: There are 4 different kinds of actors:
   i) Patient (P): Person to be transported.
   ii) Ambulance (A): Actor to deliver Patient.
   iii) Location (L): Place for Patient to be delivered from.
   iv) Hospital (H): Place for Patient to be delivered to.
2) *Interactions*: There are 6 kinds of interactions:
   i) $a_1$ = <A, L>: Ambulance goes to Location
   ii) $a_2$ = <P, A>: Patient gets on Ambulance.

iii) $a_3$ = <A, H>: Ambulance goes to Hospital.
iv) $a_4$ = <A, P>: Patient gets off Ambulance.
v) $a_5$ = <P, H>: Patient goes to Hospital.

### 3.2 Step 2: Regular Behaviors

In this step, all the collective behaviors are defined as a sequence of interactions from Step 1. In order to quantify the behaviors, all behaviors are divided into two kinds of behaviors: the one with one main actor and the others with other actors. In the other words, there are different views by different actors. For example, in EMS there are four kind of actors, represented as B(L, A, H, P). Then, there are two types of behaviors for Ambulance as a main actor, represented as B(n, 1, n, n) for 1 Ambulance and B(n, *n*, n, n) for *n* Ambulances.

There are total 9 behaviors possible for EMS, defined in regular expression as follows:

1) $B_1 = \langle a1, a2, a3, a4, a5 \rangle$: An Ambulance goes to a Place, gets a Patient on, goes to a Hospital, and gets the Patient off, who goes into the Hospital.
2) $B_2 = \langle a1, a2, a3, a4, a5 \rangle^+$: A repeating behavior of $B_1$.
3) $B_3 = \left\langle a1, \langle a2 \rangle^+, a3, \langle a4, a5 \rangle^+ \right\rangle^+$: An Ambulance goes to a Place, gets Patients on, goes to a Hospital, and gets the Patients off, who go into the Hospital. And it repeats itself.
4) $B_4 = \left\langle a1, \langle a2 \rangle^+, \langle a3, a4, a5 \rangle^+ \right\rangle^+$: An Ambulance goes to a Place, gets Patients on, and goes to Hospitals to get some of the Patients off until all the Patients off, each group of who goes into the Hospital. And it repeats itself.
5) $B_5 = \left\langle a1, \langle a2 \rangle^+, \left\langle a3, \langle a4, a5 \rangle^+ \mid a3, a4, a5 \right\rangle^+ \right\rangle^+$: A repeating behavior of $B_3$ and $B_4$, that is, $B_1$ through $B_4$.
6) $B_6 = \left\langle \langle a1, a2 \rangle^+, a3, \langle a4, a5 \rangle^+ \right\rangle^+$: An Ambulance goes to Places to get Patients on, goes to a Hospital, and gets the Patients off, who go to the Hospital. And it repeats itself.
7) $B_7 = \left\langle \langle a1, a2 \rangle^+, \langle a3, a4, a5 \rangle^+ \right\rangle^+$: A repeating behavior that an Ambulance goes to Places to get Patients on, goes to Hospitals, and gets some of the Patients off to each Hospital until all the Patients get off, each group of who goes to the Hospital. And it repeats itself.
8) $B_8 = \left\langle \langle a1, a2 \rangle^+, \left\langle a3, \langle a4, a5 \rangle^+ \mid a3, a4, a5 \right\rangle^+ \right\rangle^+$: A repeating behavior of $B_6$ and $B_7$, that is, $B_1$ through $B_7$ except $B_3$, $B_4$ and $B_5$.
9) $B_9 = \left( \left\langle a1, \langle a2 \rangle^+, \left\langle \begin{matrix} a3, \langle a4, a5 \rangle^+ \mid \\ a3, a4, a5 \end{matrix} \right\rangle^+ \right\rangle \mid \left\langle \langle a1, a2 \rangle^+, \left\langle \begin{matrix} a3, \langle a4, a5 \rangle^+ \mid \\ a3, a4, a5 \end{matrix} \right\rangle^+ \right\rangle \right)^+$: A repeating behavior of $B_2$, $B_5$ or $B_8$.

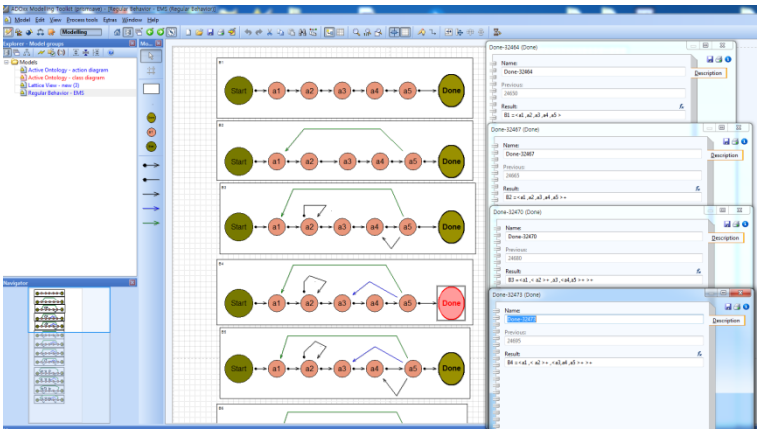

**Fig. 4** A Part of Basic Behavior Specifications in PRISM

Fig. 4 shows a part of the BB model for EMS specified in PRISM.

### 3.3 Step 3: Abstract Behaviors

The next step is to abstract the regular behaviors from Step 2. The abstraction is based on the number of main actors and the numbers of actors collaborating with their degree of interactions. Formally *Abstract Behavior* is the behavior that has been quantitatively abstracted with respect to *cardinality* and *capacity* of actors. The cardinality implies the number of actors involved in behavior, and the capacity does the number of possible interactions among the actors. The behavior is denoted by $B(c_1,\cdots,c_n)$, where each $c$ is an actor, $c^{x}_{\langle p_1,\cdots,p_x\rangle}$, where $x$ and $\langle p_1,\cdots,p_x\rangle$ are the cardinality and capacity of $c$.

For EMS, the behaviors for 1 Ambulance from Step 2 can be abstracted as follows:

1) $B_1 = B_1\left(P^1_{\langle 1\rangle}, A^1_{\langle 1\rangle}, H^1_{\langle 1\rangle}\right)$
2) $B_2 = B_2\left(P^i_{\langle x_1,\cdots,x_i\rangle}, A^1_{\langle 1\rangle}, H^k_{\langle z_1,\cdots,z_k\rangle}\right)$
3) $B_3 = B_3\left(P^1_{\langle x\rangle}, A^1_{\langle y\rangle}, H^1_{\langle z\rangle}\right)$
4) $B_4 = B_4\left(P^1_{\langle x\rangle}, A^1_{\langle y\rangle}, H^k_{\langle 1_1,\cdots,1_k\rangle}\right)$
5) $B_5 = B_5\left(P^1_{\langle x\rangle}, A^1_{\langle y\rangle}, H^k_{\langle z_1,\cdots,z_k\rangle}\right)$
6) $B_6 = B_6\left(P^i_{\langle 1_1,\cdots,1_i\rangle}, A^1_{\langle y\rangle}, H^1_{\langle k\rangle}\right)$
7) $B_7 = B_6\left(P^i_{\langle 1_1,\cdots,1_i\rangle}, A^1_{\langle 1\rangle}, H^k_{\langle 1_1,\cdots,1_k\rangle}\right)$
8) $B_8 = B_8\left(P^i_{\langle 1_1,\cdots,1_i\rangle}, A^1_{\langle y\rangle}, H^k_{\langle z_1,\cdots,z_k\rangle}\right)$
9) $B_9 = B_8\left(P^i_{\langle x_1,\cdots,x_i\rangle}, A^1_{\langle y\rangle}, H^k_{\langle z_1,\cdots,z_k\rangle}\right)$

Further, abstract behaviors for $n$ Ambulances can be defined as follows:

1) $B_{11} = B_{11}\left(P^1_{\langle x\rangle}, A^j_{\langle y_1,\cdots,y_j\rangle}, H^1_{\langle z\rangle}\right)$

2) $B_{12} = B_{12}\left(P^1_{\langle x\rangle}, A^j_{\langle y_1,\cdots,y_j\rangle}, H^k_{\langle z_1,\cdots,z_k\rangle}\right)$

3) $B_{13} = B_{13}\left(P^1_{\langle x_1,\cdots,x_i\rangle}, A^j_{\langle y_1,\cdots,y_j\rangle}, H^k_{\langle z\rangle}\right)$

4) $B_{14} = B_{14}\left(P^1_{\langle x_1,\cdots,x_i\rangle}, A^j_{\langle y_1,\cdots,y_j\rangle}, H^k_{\langle z_1,\cdots,z_k\rangle}\right)$

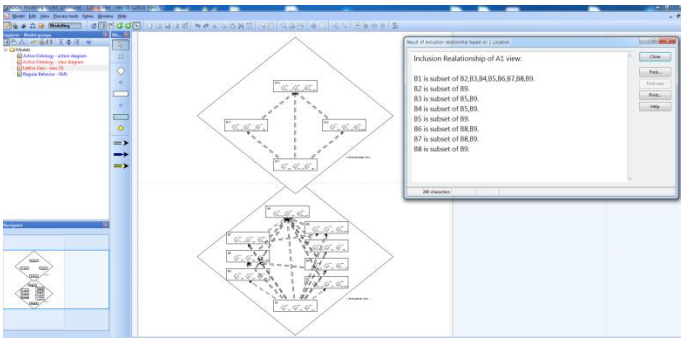

**Fig. 5** Behavior Lattices for EMS B(n, 1, n, n) & B(n, *n*, n, n)

### 3.4 Step 4: Behavior Lattice (BL)

Lattice can be constructed from Step 3, based on the inclusion relations among behaviors. Formal definitions for the lattice are reported in [5].

Figure 5 shows two lattices, where the bottom is for 1 Ambulance and the top one is for *n* Ambulances. Note that the inclusion relations among the lattices are generated automatically from the regular behaviors from Step 2 by RLG of PRISM.

### 3.5 Step 5: Behavior Ontology (BO)

Last step is to merge the lattices from Step 4 into one integrated lattice of lattices, known as Behavior Ontology. Figure 5 shows the final output of Step 4: the lattice, known as mLBL, consisting of the lattice for 1 Ambulance and the lattice for *n* Ambulances. As stated, it is generated by BLM of PRISM.

$$
\begin{aligned}
&CS = ((CALL(HAHBP).ORDER(\overline{HAHBP}).CALL(HDHD).ORDER(\overline{HDHD}))\\
&\quad +(CALL(HDHD).ORDER(\overline{HDHD}).CALL(HAHBP).ORDER(\overline{HAHBP}))).\\
&\quad CALL(SCFP1).CALL(SCFP2).CALL(SCHD).CALL(SCHBP).ORDER(\overline{ALL1}).ORDER(\overline{ALL2}).\\
&\quad ((CALL(HBHBP).ORDER(\overline{HBHBP}).CALL(HCHD).ORDER(\overline{HCHD}))\\
&\quad +(CALL(HCHD).ORDER(\overline{HCHD}).CALL(HBHBP).ORDER(\overline{HBHBP}))).\emptyset^{\infty};\\
&911 = ORDER(HDHD).((AmbA(\overline{HD}).AmbAout.ORDER(ALL1).AmbB(\overline{SC}).AmbBout.ORDER(HCHD).AmbA(\overline{HC}))\\
&\oplus^1(AmbB(\overline{HD}).AmbBout.ORDER(ALL1).AmbA(\overline{SC}).AmbAout.ORDER(HCHD).AmbA(\overline{HC}))).AmbAin.AmbBin.\emptyset^{\infty};\\
&HospitalA =\\
&ORDER(HAHBP).AmbC(\overline{HA}).AmbCout.CALL(Arrive1).HA(\overline{Ready1}).AmbCin.ORDER(ALL2).AmbC(\overline{SC}).\\
&\quad AmbCout.ORDER(HBHBP).AmbC(\overline{HB}).CALL(Arrive2).HA(\overline{Ready2}).HA(\overline{Ready3}).AmbCin.\emptyset^{\infty};\\
&HospitalB = CALL(Arrive3).HB(\overline{Ready1}).((AmbAin.AmbAout)\oplus^1(AmbBin.AmbBout)).\\
&\quad CALL(Arrive4).HB(\overline{Ready2}).((AmbBin.AmbBout)\oplus^1(AmbAin.AmbAout)).
\end{aligned}
$$

$$CALL(Arrive5).HB(\overline{Ready3}).\Big(\big(AmbAin.AmbAout\big)\oplus^1\big(AmbBin.AmbBout\big)\Big).\emptyset^{\infty};$$

$$HospitalC = CALL(Arrive6).HC(\overline{Ready1}).HC(\overline{Ready2}).\Big(\big(AmbBin.AmbBout\big)\oplus^1\big(AmbAin.AmbAout\big)\Big).\emptyset^{\infty};$$

$$AmbA = ((AmbA(HD).out911.inHouseD.getPHD2.outHouseD.CALL(\overline{Arrive3}).inHospitalB.putPHD2.outHospitalB.AmbA(HC).inHouseC.getPHD1.outHouseC.CALL(\overline{Arrive3}).inHospitalB.putPHD1 outHospitalB)\oplus^1(AmbA(SC).out911.inSchool.getPFP1.getPFP2.getPHD3.outSchool.((CALL(\overline{Arrive4}).inHospitalB.putPHD3.outHospitalB.CALL(\overline{Arrive6}).inHospitalC.putPFP1.putPFP2.outHospitalC)+(CALL(\overline{Arrive6}).inHospitalC.putPFP1.putPFP2.outHospitalC.CALL(\overline{Arrive4}).inHospitalB.putPHD3.outHospitalB))).in911.\emptyset^{\infty};$$

$$AmbB = ((AmbB(SC).out911.inSchool.getPFP1.getPFP2.getPHD3.outSchool.((CALL(\overline{Arrive4}).inHospitalB.putPHD3.outHospitalB.CALL(\overline{Arrive6}).inHospitalC.putPFP1.putPFP2.outHospitalC)+(CALL(\overline{Arrive6}).inHospitalC.putPFP1.putPFP2.outHospitalC.CALL(\overline{Arrive4}).inHospitalB.putPHD3.outHospitalB)))\oplus^1(AmbB(HD).out911.inHouseD.getPHD2.outHouseD.CALL(\overline{Arrive3}).inHospitalB.putPHD2.outHospitalB.AmbB(HC).inHouseC.getPHD1.outHouseC.CALL(\overline{Arrive3}).inHospitalB.putPHD1 outHospitalB)).in911.\emptyset^{\infty};$$

$$AmbC = AmbC(HA).outHospitlA.inHouseA.getPHBP1.outHouseA.CALL(\overline{Arrive1}).inHospitalA.putPHBP1.AmbC(SC).outHospitalA.inSchool.getPHBP3.outSchool.AmbC(HB).inHouseB.getPHBP2.outHouseB.CALL(\overline{Arrive2}).inHospitalA.putPHBP2.putPHBP3.\emptyset^{\infty};$$

$$HouseA = AmbC\,in.AmbC\,out.\emptyset^{\infty};$$

$$HouseB = AmbC\,in.AmbC\,out.\emptyset^{\infty};$$

$$HouseC = \Big(\big(AmbA\,in.AmbA\,out\big)\oplus^1\big(AmbB\,in.AmbB\,out\big)\Big).\emptyset^{\infty};$$

$$HouseD = \Big(\big(AmbA\,in.AmbA\,out\big)\oplus^1\big(AmbB\,in.AmbB\,out\big)\Big).\emptyset^{\infty};$$

$$School = \Big(\big(AmbB\,in.AmbB\,out\big)\oplus^1\big(AmbA\,in.AmbA\,out\big)\Big).AmbC\,in.AmbC\,out.\emptyset^{\infty};$$

$$PHBP1 = CALL(\overline{HAHBP}).AmbC\,get.AmbC\,put.SurgeryA\,get.DoctorA1(Surgery).\emptyset^{\infty};$$

$$PHBP2 = CALL(\overline{HBHBP}).AmbC\,get.AmbC\,put.SurgeryA\,get.DoctorA2(Surgery).\emptyset^{\infty};$$

$$PHBP3 = CALL(\overline{SCHBP}).AmbC\,get.AmbC\,put.SurgeryA\,get.DoctorA3(Surgery).\emptyset^{\infty};$$

$$PHD1 = CALL(\overline{HCHD}).\Big(\big(AmbA\,get.AmbA\,put\big)\oplus^1\big(AmbB\,get.AmbB\,put\big)\Big).SurgeryB\,get.DoctorB1(Surgery).\emptyset^{\infty};$$

$$PHD2 = CALL(\overline{HDHD}).\Big(\big(AmbA\,get.AmbA\,put\big)\oplus^1\big(AmbB\,get.AmbB\,put\big)\Big).SurgeryB\,get.DoctorB2(Surgery).\emptyset^{\infty};$$

$$PHD3 = CALL(\overline{SCHD}).\Big(\big(AmbB\,get.AmbB\,put\big)\oplus^1\big(AmbA\,get.AmbA\,put\big)\Big).SurgeryB\,get.DoctorB3(Surgery).\emptyset^{\infty};$$

$$PFP1 = CALL(\overline{SCHD}).\Big(\big(AmbB\,get.AmbB\,put\big)\oplus^1\big(AmbA\,get.AmbA\,put\big)\Big).SurgeryC\,get.DoctorC1(Surgery).\emptyset^{\infty};$$

$$PFP2 = CALL(\overline{SCHD}).\Big(\big(AmbB\,get.AmbB\,put\big)\oplus^1\big(AmbA\,get.AmbA\,put\big)\Big).SurgeryC\,get.DoctorC2(Surgery).\emptyset^{\infty};$$

**Fig. 6 δ-Calculus Code for a Smart IoT for EMS**

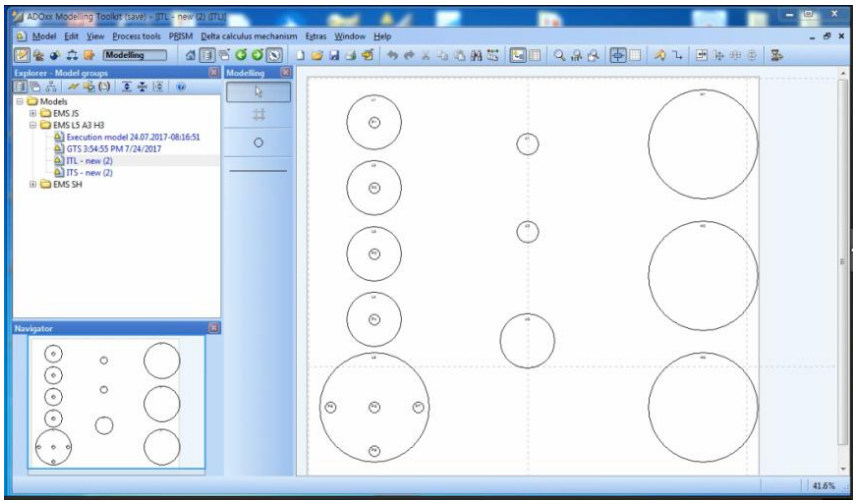


**Fig. 7** ITL View of a Smart IoT for EMS

## 4 Interpretation by Projection

This section shows how a system in a domain can be interpreted for its collective behavior with respect to the behavior ontology of the domain. Here a smart IoT for EMS has been designed as an example, as Fig. 6 shows with a code for the example in δ-Calculus [7], which is a process algebra to specify the movements of business processes in real-time environments with the special notion of movements over a conceptual geographical space. All the interactions among processes in the system are automatically handled by the notion of IoT.

### 4.1 SAVE

Fig. 7 shows the ITL view of the example in SAVE [7], which consists of the following actors:

1) Patient ( $P^{5}_{\langle 1,1,1,1,4\rangle}$ ): 8 Patient Objects in 5 Places with the capacity of 1, 1, 1, 1, 4.
2) Ambulance ( $A^{3}_{\langle 1,1,3\rangle}$ ): 3 Ambulance with capacity of 1, 1, 3.
3) Hospital ( $H^{3}_{\langle 3,3,5\rangle}$ ): 3 Hospitals with capacity of 3, 3, 5.

Here, $B(P^{5}_{\langle 1,1,1,1,4\rangle}, A^{3}_{\langle 1,1,3\rangle}, H^{5}_{\langle 3,3,5\rangle})$ can be interpreted as an abstract behavior that 5 groups of Patients with capacities of 1, 1, 1, 1 and 4 are to be delivered by 3 Ambulances with capacities of 1, 1 and 3 to 3 Hospitals with capacity of 3, 3, and 5.

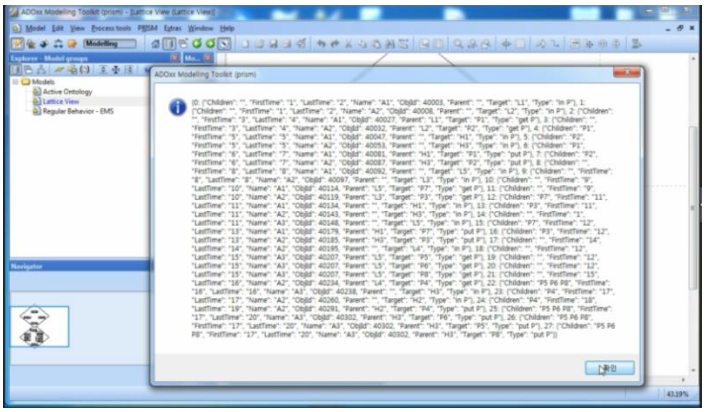

**Fig. 8** The Raw Data for Behaviors from Simulation in SAVE

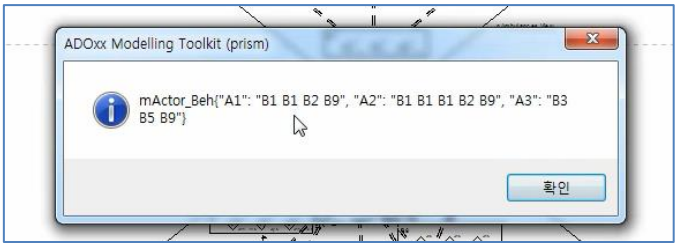


**Fig. 9** The Abstract Behaviors for Those in Fig. 8

### 4.2 Behaviors from Simulation

Fig. 8 shows the raw data of behaviors for the example from the simulation in SAVE. These behaviors are abstracted by PRISM for SAVE, as shown in Fig. 10 and Fig. 11.

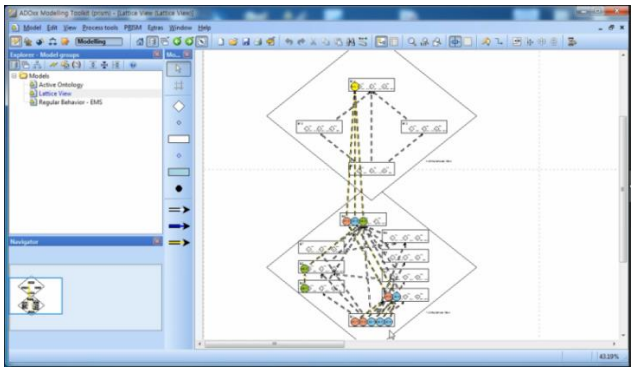

**Fig. 10** The Projected Behaviors from Fig. 9 to the Lattice of the EMS Domain in PRISM

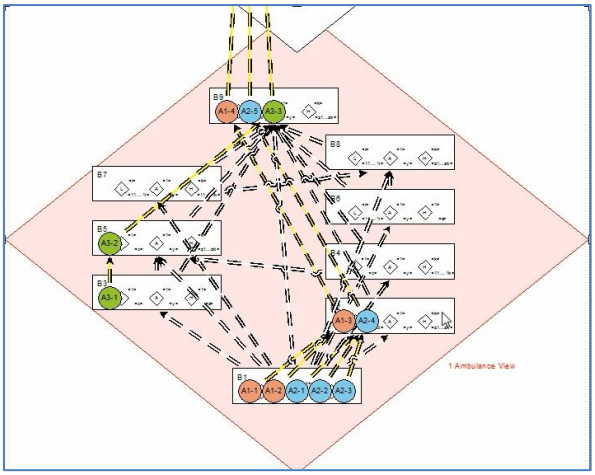


**Fig. 11** The Projected Behaviors in the Bottom Lattice from Fig. 10

### 4.3 Projection to PRISM

Fig. 11 shows the projection of the abstract behaviors on the lattice for EMS in PRISM. From the top to the bottom, the behaviors are organized as follows::

$$B_{16}(P^5_{\langle 1,1,1,1,4\rangle}, A^3_{\langle 1,1,3\rangle}, H^5_{\langle 3,3,5\rangle}) = \left\{B_{9.1}\left(P^2_{\langle 1,1\rangle}, A^A_{\langle 1\rangle}, H^2_{\langle 1,1\rangle}\right), B_{9.2}\left(P^3_{\langle 1,1,1\rangle}, A^B_{\langle 1\rangle}, H^3_{\langle 1,1,1\rangle}\right), B_{9.3}\left(P^1_{\langle 3\rangle}, A^C_{\langle 3\rangle}, H^1_{\langle 3\rangle}\right)\right\}$$

where,

$$B_{9.1}\left(P^2_{\langle 1,1\rangle}, A^A_{\langle 1\rangle}, H^2_{\langle 1,1\rangle}\right) = \left\{B_{2.1}\left(P^2_{\langle 1,1\rangle}, A^A_{\langle 1\rangle}, H^2_{\langle 1,1\rangle}\right)\right\},$$

$$B_{9.2}\left(P^3_{\langle 1,1,1\rangle}, A^B_{\langle 1\rangle}, H^3_{\langle 1,1,1\rangle}\right) = \left\{B_{2.2}\left(P^3_{\langle 1,1,1\rangle}, A^B_{\langle 1\rangle}, H^3_{\langle 1,1,1\rangle}\right)\right\}$$

$$B_{9.3}\left(P^1_{\langle 3\rangle}, A^C_{\langle 3\rangle}, H^1_{\langle 3\rangle}\right) = \left\{B_{5.1}\left(P^1_{\langle 3\rangle}, A^C_{\langle 3\rangle}, H^1_{\langle 3\rangle}\right)\right\}$$

where,

$$B_{2.1}\left(P^{2}_{\langle 1,1\rangle}, A^{A}_{\langle 1\rangle}, H^{2}_{\langle 1,1\rangle}\right) = \left\{B_{1.1}\left(P^{1}_{\langle 1\rangle}, A^{A}_{\langle 1\rangle}, H^{1}_{\langle 1\rangle}\right), B_{1.2}\left(P^{1}_{\langle 1\rangle}, A^{A}_{\langle 1\rangle}, H^{1}_{\langle 1\rangle}\right)\right\}$$

$$B_{2.2}\left(P^{3}_{\langle 1,1,1\rangle}, A^{B}_{\langle 1\rangle}, H^{3}_{\langle 1,1,1\rangle}\right) = \left\{B_{1.3}\left(P^{1}_{\langle 1\rangle}, A^{B}_{\langle 1\rangle}, H^{1}_{\langle 1\rangle}\right), B_{1.4}\left(P^{1}_{\langle 1\rangle}, A^{B}_{\langle 1\rangle}, H^{1}_{\langle 1\rangle}\right), B_{1.5}\left(P^{1}_{\langle 1\rangle}, A^{B}_{\langle 1\rangle}, H^{1}_{\langle 1\rangle}\right)\right\}$$

$$B_{5.1}\left(P^{1}_{\langle 3\rangle}, A^{C}_{\langle 3\rangle}, H^{1}_{\langle 3\rangle}\right) = \left\{B_{3.1}\left(P^{1}_{\langle 3\rangle}, A^{C}_{\langle 3\rangle}, H^{1}_{\langle 3\rangle}\right)\right\}$$

## 5 Conclusion and Future research

This paper presented the PRISM tool, developed on the ADOxx Meta-Modeling Platform, for a new method for knowledge engineering to model collective behaviours of systems, based on Behavior Ontology. PRISM showed an innovative approach for meta-modeling of domain knowledge and demonstrated the efficiency and effectiveness of the approach for implementation and feasibility as a tool. The future research includes developing other domain knowledge and their composition, as well as developing an open model tool for PRISM on ADOxx.

## Acknowledgment

This work was supported by Basic Science Research Programs through Space Core Technology Development Program through the NRF (National Research Foundation of Korea) funded by the Ministry of Science, ICT and Future Planning (NRF-2014M1A3A3A02034792), and Basic Science Research Program through NRF funded by the Ministry of Education (NRF-2015R1D1A3A01019282).

## References

1. W. Choi, Y. Choe, and M. Lee. A Reduction Method for Process and System Complexity with Conjunctive and Complement Choices in a Process Algebra. Proceedings of 39th IEEE COMPSAC/MVDM. July 2015.
2. Longbing Cao, Philip S.Yu. Edited, *Behavior Computing: Modeling, Analysis, Mining and Decision*. Springer, 2012.
3. S. Woo, J. On, and M. Lee. An Abstraction Method for Mobility, and Interaction in Process Algebra Using Behavior Ontology. 37th IEEE COMPSAC. July 2013.
4. W. Xing, O. Corcho, C. Goble, and M. Dikaiakos. Active Ontology: An Information Integration Approach for Highly Dynamic Information Sources. Europe Semantic Web Conference 2007 (ESWC-2007), Innsbruck, Austria. 2007.
5. Y. Choe and M. Lee. A Lattice Model to Verify Behavioral Equivalence. UKSim-AMSS 8th European Modelling Symposium. Oct 2014.
6. Y. Choe, W. Choi, G. Jeon and M. Lee. A Tool for Visual Specification and Verification for Secure Process Movements. eChallenges e-2015. November 2015.
7. Y. Choe and M. Lee. δ-Calculus: Process Algebra to Model Secure Movements of Distributed Mobile Processes in Real-Time Business Application. 23rd European Conference on Information Systems. April 2015.
8. S. Lee, Y. Choe, M. Lee. A Dual Method to Model IoT Systems. International Journal of Mathematical Models and Methods in Applied Sciences. May 2016.
9. http://austria.omilab.org/psm/exploreprojects?param=explore