FORMAL MODELING OF BEHAVIORAL PROPERTIES TO SUPPORT CORRECT BY DESIGN PUBLISH/SUBSCRIBE ARCHITECTURAL STYLES

Ikbel Krichen, Imen Loulou and Ahmed Hadj Kacem
University of Sfax, ENIS, Research Unit ReDCAD, B.P.W., 3038 Sfax, Tunisia

Keywords: Software architecture, Publish/subscribe style, Correct by design, Formal specification, Behavioral properties.

Abstract: We propose in this paper a formal modeling approach of behavioral properties of publish/subscribe architectural styles. We extend P/S-CoM formal approach that concentrates only on correctly design the structures of publish/subscribe styles and deals with dynamic views. We put the emphasis on the state notion for component types. Moreover, we focus on behavioral properties including lossless of exchanged events, delivery semantics and ordering of notifications. These properties are coded in Z notation.

1 INTRODUCTION

Publish/subscribe architectural styles (PSAS) are increasingly used in the large-scale distributed and cooperative applications. This is due to its dynamic nature, scalability and interaction model based on three-dimensional decoupling (Eugster et al., 2003). The interaction among the component types of this style (producer, consumer, prod/cons and event-service) depends on its structure and its behavior. The correct by design of PSAS is important in order to ensure that the architectures with which they are compliant, operate reliably. This design requires the use of formal methods as they provide exact semantics and eliminate ambiguities. Using the construction method is interesting since it promotes reuse by composing reusable modules and guides the designers to easily conceive correct architectural styles.

P/S-CoM formal approach (Loulou et al., 2010) helps designers build the structures of SPAS by composing reusable *communication schemas* but does not propose the modeling of behavioral properties. The obtaining of only correct structures is not sufficient for modeling PSAS and generating correct implementations. In this paper, we extend P/S-CoM approach by integrating the behavioral view. This extension includes the definition of the state of component types and behavioral properties: lossless of events, delivery semantics and ordering of notifications. Establishing a correct PSAS at design level will be done by composing the extended and new schemas.

The rest of this paper is structured as follows: Section 2 provides a general overview of the P/S-CoM formal approach. Section 3 describes our P/S-CoM extension approach. Section 4 presents the related work. Section 5 concludes this paper and presents future work directions.

2 P/S-COM APPROACH

P/S-CoM (Loulou et al., 2010) is a formal approach that provides the correct modeling of PSAS structures. It consists in composing reusable *communication schemas* and respecting a set of composition rules coded in Z notation and proved with the Z-Eves theorem prover. Z is a standard and formal language. It is based on the set of theory and the first-order predicate logic. It describes states and operations with *schemas*. A schema has a name, a declarative part which defines data, and a predicative part which enumerates constraints.

Among the provided communication schemas, D_C1 represents an example of communication between a distributed event-service and consumers where one link of communication *push* is considered. Its formal specification is described with the following generic schema D_C1[Consumer, EvDispatcher]. The declarative part contains a set ($\mathbb{F}$) of consumers and dispatchers, *PushD* and *PushDD* links as relations. In the predicative part, $[Pred-1]$ and $[Pred-2]$ describe inclusions of sets using domain (*dom*) and

286 Krichen I., Loulou I. and Hadj Kacem A..
FORMAL MODELING OF BEHAVIORAL PROPERTIES TO SUPPORT CORRECT BY DESIGN PUBLISH/SUBSCRIBE ARCHITECTURAL STYLES.
DOI: 10.5220/0003495402860289
In *Proceedings of the 6th International Conference on Software and Database Technologies* (ICSOFT-2011), pages 286-289
ISBN: 978-989-8425-77-5
Copyright © 2011 SCITEPRESS (Science and Technology Publications, Lda.)

range (*ran*) operators. $[Pred-3]$ expresses the inverse relationship ($\sim$). $[Pred-4]$ states that all the dispatchers have to be connected. $[Pred-5]$ requires that every consumer has a single access point in the network of dispatchers.

$D\_C1[Consumer, EvDispatcher]$

$C : \mathbb{F}\,Consumer$
$D : \mathbb{F}\,EvDispatcher$
$PushD : EvDispatcher \leftrightarrow Consumer$
$PushDD : EvDispatcher \leftrightarrow EvDispatcher$

$\operatorname{dom} PushD \subseteq D \wedge \operatorname{ran} PushD \subseteq C$ [Pred-1]
$\operatorname{dom} PushDD \subseteq D \wedge \operatorname{ran} PushDD \subseteq D$ [Pred-2]
$PushDD = PushDD^{\sim}$ [Pred-3]
$\forall x, y : D \mid x \neq y$
$\bullet \exists T : \operatorname{seq} EvDispatcher$
$\mid \operatorname{ran} T \subseteq D \wedge x \in \operatorname{ran} T \wedge y \in \operatorname{ran} T$
$\bullet \forall i : \mathbb{N} \mid 1 \leq i \wedge i+1 \leq \#T$
$\bullet (Ti, T(i+1)) \in PushDD$ [Pred-4]
$\forall c : C \bullet (\exists_1 d : D$
$\bullet d \in \operatorname{dom}(PushD \rhd \{c\}))$ [Pred-5]

3 P/S-COM EXTENSION

In this section, we describe the extension that we made on the P/S-CoM communication schemas and the behavioral properties. Obtained communication schemas (new ones and extended ones) can be composed in order to get PSAS considering structural and behavioral views.

3.1 Event Types and Matching Functions

Various events are exchanged between publishers (producer or prod/cons), subscribers (consumer or prod/cons) and event dispatchers (EvDispatcher) entities. We distinguish three types of events: *EVENT* for notification, *SUB_EVENT* for subscription and *ADV_EVENT* for advertisement (Figure 1). Every *EVENT* can be marked by a stamp. Every *SUB_EVENT* can be assigned with a priority by its subscriber. Since we are not interested, here, in the specification details of these event types, we introduce them as basic types.

$EVENT$: $stamp : \mathbb{N}$ $\quad$ SUB_EVENT: $priority : \mathbb{N}$ $\quad$ $[ADV\_EVENT]$

Figure 1: Event Types.

We represent *Advertising* and *Filtering* matching functions as relations. The first one is used to verify that a produced event corresponds to an advertisement $[R1]$. The second one is used to verify if an event notification matches a given subscription $[R2]$.

$Advertising : EVENT \leftrightarrow ADV\_EVENT$ [R1]
$Filtering : EVENT \leftrightarrow SUB\_EVENT$ [R2]

3.2 State of the Component Types

We characterize here the state of each publish/susbcribe component type. The state can be changed whenever an event is generated or received.

The state of the subscribers (resp. publishers) depends on the subscriptions (resp. advertisements and notifications) that he generated and the notifications that he received. He stores these event types. He cannot generate twice the same subscription (resp. notification and advertisement) but he can receive or not the notifications which match his subscriptions. In this paper, we are interested in modeling publish/subscribe architectures where notifications are not persistent. Thus, every event dispatcher stores only the exchanged advertisements and subscriptions in routing tables. It also maintains the source of each subscription. Given the lack of space, we present only the formal specification of the consumers state:

$State\_Cons[Consumer]$

$Sub\_EventsC : Consumer \leftrightarrow SUB\_EVENT$
$Notif\_EventsC : Consumer \leftrightarrow \operatorname{seq} EVENT$

$\forall c : Consumer \mid c \in \operatorname{dom} Notif\_EventsC$
$\bullet \forall T, H : \operatorname{seq} EVENT$
$\bullet (c, T) \in Notif\_EventsC$
$\wedge (c, H) \in Notif\_EventsC \Rightarrow T = H$ [Pred-1]
$\forall c : Consumer;\ J : \operatorname{seq} EVENT$
$\mid (c, J) \in Notif\_EventsC$
$\bullet \forall e : EVENT \mid e \in \operatorname{ran} J \bullet \exists s : SUB\_EVENT$
$\bullet s \in \operatorname{ran}(\{e\} \lhd Filtering)$
$\wedge s \in \operatorname{ran}(\{c\} \lhd Sub\_EventsC)$ [Pred-2]

In the declarative part, *Sub_EventsC* relation represents the subscriptions retained at corresponding consumer. *Notif_EventsC* relation states the sequences of notifications received by consumers. With *seq*, a consumer has a sequence of ordered notifications which their duplication is allowed. In the predicative part, we indicate that every consumer has a unique sequence of notifications $[Pred-1]$ and every notification received by a given consumer, corresponds to at least one of his subscriptions $[Pred-2]$.

3.3 Refinement of P/S-CoM Communication Schemas

In order to consider behavioral properties, we refine

P/S-CoM communication schemas and we get new reusable schemas as it is described in Figure 2. We put the name of the concerned communication schema and the state of the component types in the declarative part, and some constraints in the predicative part.

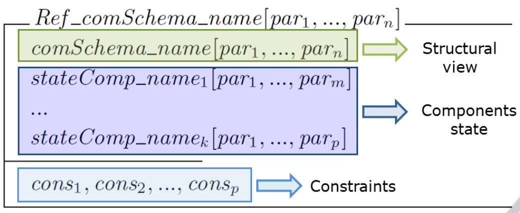
$Ref\_comSchema\_name[par_1, ..., par_n]$
$comSchema\_name[par_1, ..., par_n]$
Structural view
$stateComp\_name_1[par_1, ..., par_m]$
...
$stateComp\_name_k[par_1, ..., par_p]$
Components state
$cons_1, cons_2, ..., cons_p$
Constraints

Figure 2: Refinement method of P/S-CoM schemas.

Among the P/S-CoM communication schemas, we refine *D_C1* (section 2) and obtain the following Z specification:

$ref\_D\_C1[Consumer, EvDispatcher]$

$D\_C1[Consumer, EvDispatcher]$
$State\_Cons[Consumer]$
$D\_State\_EvD[EvDispatcher, Consumer, Prod\_Cons]$

$\mathrm{dom}\, Sub\_EventsC \subseteq C \land \mathrm{dom}\, Notif\_EventsC \subseteq C$
$\land\ \mathrm{dom}\, SubC\_EventsD \subseteq D$ [Pred-1]
$\{c : Consumer \mid \forall d : EvDispatcher;\ s : SUB\_EVENT$
$\bullet\ (d, (c, s)) \in SubC\_EventsD\} \subseteq \mathrm{ran}\, PushD$
[Pred-2]
$\forall d1, d2 : D;\ s : SUB\_EVENT$
$\mid (d1, (d2, s)) \in Sub\_EventsDD$
$\bullet\ (d1, d2) \in PushDD$ [Pred-3]
$\forall d : EvDispatcher$
$\mid d \in \mathrm{dom}\, PushDD \land d \notin \mathrm{dom}\, PushD$
$\bullet\ SubC\_EventsD = \varnothing$ [Pred-4]

In the declarative part, we invoke the *D_C1* schema, the state of the consumers and the distributed event-service specification. In the predicative part, $[Pred - 2]$ stipulates that the consumers from which an event dispatcher received their subscriptions are connected to this event dispatcher. $[Pred - 3]$ states that every two dispatchers which exchange subscriptions are directly connected. $[Pred - 4]$ defines that every event dispatcher that is not directly connected to a consumer, can receive only subscriptions issued by the neighbor dispatchers.

3.4 A Formal Approach for Designing Behavioral Properties

In this section, we present the adopted approach for specifying behavioral properties in Z notation. We distinguish local and global properties.

3.4.1 Local Properties

We aim at ensuring the lossless of exchanged advertisements (resp. subscriptions) among the publishers (resp. subscribers) and the event-service. Since these properties represent interactions between the event-service and the clients (publishers and subscribers), they can be integrated in the refined communication schemas. We extended them as it is described in Figure 3. We declare in the declarative part the name of the refined schema, and we define constraints in the predicative part.

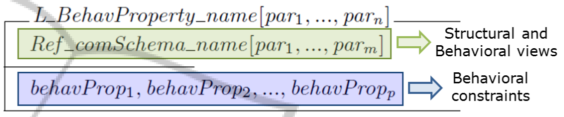
$L\_BehavProperty\_name[par_1, ..., par_n]$
$Ref\_comSchema\_name[par_1, ..., par_m]$
Structural and Behavioral views
$behavProp_1, behavProp_2, ..., behavProp_p$
Behavioral constraints

Figure 3: The local properties specification.

3.4.2 Global Properties

We model the delivery semantics and the ordering of notifications as global properties. They represent interactions between publishers and subscribers. We propose new communication schemas as it is described in Figure 4. We invoke the state of the concerned clients in the declarative part, and behavioral constraints in the predicative part.

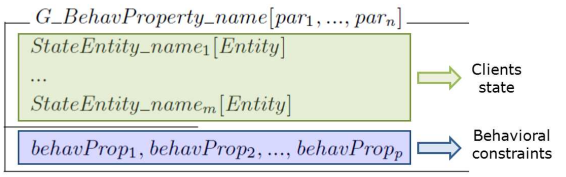
$G\_BehavProperty\_name[par_1, ..., par_n]$
$StateEntity\_name_1[Entity]$
...
$StateEntity\_name_m[Entity]$
Clients state
$behavProp_1, behavProp_2, ..., behavProp_p$
Behavioral constraints

Figure 4: The global properties specification.

The delivery process of notifications is ensured by the event-service following some semantics such as *at most once*, *at least once*, *exactly once* and *best effort* (Mahambre et al., 2007). These policies are used for setting the number of notification's instances received per interested subscriber. Concerning the ordering of notifications, it is interesting to establish a relationship between the order according to which event notifications are sent and the order in which they are received. We adopt three techniques: *pair-wise FIFO*, *priority* and *total* ordering (Mahambre et al., 2007).

Given the lack of space, we reveal the Z specification of the *pair-wise FIFO* ordering, among the local and global properties. Following this ordering, the notifications should be received in the same order of their generation. In the case of a style containing producers and consumers, we can ensure the *pair-wise FIFO* ordering by the use of the following schema:

$$
\begin{array}{ll}
\multicolumn{2}{l}{\mathit{FifoPairWiseOrder\_P\_C}[\mathit{Producer},\mathit{Consumer}]} \\
\mathit{State\_Prod}[\mathit{Producer}] & \\
\mathit{State\_Cons}[\mathit{Consumer}] & \\
\hline
\forall p : \mathit{Producer} \mid p \in \operatorname{dom} \mathit{Prod\_EventsP} & \\
\quad \bullet\ \exists S : \operatorname{iseq} \mathit{EVENT} \mid \{S\} = \mathit{Prod\_EventsP}(\!|\ \{p\}\ |\!) & \\
\quad\ \bullet\ \#S = 1 \Rightarrow (S1).\mathit{stamp} = 1 & \text{[Pred-1]} \\
\forall p : \mathit{Producer};\ F : \operatorname{iseq} \mathit{EVENT} & \\
\quad \mid p \in \operatorname{dom} \mathit{Prod\_EventsP} & \\
\quad\ \wedge \{F\} = \mathit{Prod\_EventsP}(\!|\ \{p\}\ |\!) \wedge \#F > 1 & \\
\quad\ \bullet\ \forall i,j : \operatorname{dom} F \mid j = i+1 & \\
\quad\quad \bullet\ (Fj).\mathit{stamp} = 1 + (Fi).\mathit{stamp} & \text{[Pred-2]} \\
\forall c : \mathit{Consumer};\ R : \operatorname{seq} \mathit{EVENT} & \\
\quad \mid c \in \operatorname{dom} \mathit{Notif\_EventsC} & \\
\quad\ \wedge \{R\} = \mathit{Notif\_EventsC}(\!|\ \{c\}\ |\!) \wedge \#R > 1 & \\
\quad\ \bullet\ \forall i,j : \operatorname{dom} R \mid j = i+1 & \\
\quad\quad \bullet\ (Ri).\mathit{stamp} \le (Rj).\mathit{stamp} & \text{[Pred-3]} \\
\hline
\end{array}
$$

The declarative part contains the state of producers and consumers. In the predicative part, $[Pred-1]$ states that the first notification issued by every producer has a stamp equal to *one*. $[Pred-2]$ expresses that the stamp is used to incrementally mark the notifications originating from a single producer. $[Pred-3]$ describes that the marked stamps will be used for ordering notifications on the consumer side.

3.5 Proving Consistency

We proved the consistency of the extended P/S-CoM schemas (state of components and behavioral properties). We instantiated the "initialization theorem" (Woodcock and Davies, 1996) for all the specifications and we implemented the proofs under the Z-EVES theorem prover.

4 RELATED WORK

In the literature, only the P/S-CoM formal approach focuses on the generic modeling of correct PSAS and uses the construction method. However, this approach is interested only in modeling the structural view. Concerning the formal modeling of the behavioral properties of the publish/subscribe systems, Mahambre et al. (Mahambre et al., 2007) and Baldoni et al. (Baldoni et al., 2003) propose the formal modeling of service guarantees with mathematical equations. The first work includes delivery semantics and message ordering whereas the second work stresses minimality and completeness properties considered as delivery semantics, too. However, these works do not promote the correct by design of PSAS and the use of the construction method. Some publish/subscribe middlewares support to behavioral properties as QoS guarantees. *Jedi* and *Ready* (Mahambre et al., 2007) provide mechanisms for the ordering of notifications and ensure the lossless of events. In addition, *Ready* adopt some delivery semantics for notifications. As standardized specification, we mention *DDS* (Data-Distribution Service) (OMG, 2007) that defines configurable QoS (priority, order of notifications, ...) and programming model for distributed systems. Though these works and specification provide behavioral properties as QoS, they are handled at runtime level. At design level, our work leverages on the P/S-CoM approach by augmenting it with behavioral view including the state of the component types and some behavioral properties.

5 CONCLUSIONS

In this paper, we proposed the integration of behavioral view into P/S-CoM approach. We formally model the state of publish/subscribe component types and behavioral properties (lossless of events, delivery semantics and ordering of notifications) in Z notation. For future work, we currently design a methodology for composing the new communication schemas and develop a GUI as an eclipse plugin. Several other issues require further investigations. First, we project to incorporate other behavioral properties and the temporal aspect. Second, we plan to integrate a process algebra language in order to check and validate these behavioral properties. Third, it is interesting to inspect the presented approach with applications.

REFERENCES

Baldoni, R., Contenti, M., Piergiovanni, S. T., and Virgillito, A. (2003). Modelling publish/subscribe communication systems: Towards a formal approach. *Object-Oriented Real-Time Dependable Systems, IEEE International Workshop on*, 0:304.

Eugster, P. T., Felber, P. A., Guerraoui, R., and Kermarrec, A.-M. (2003). The many faces of publish/subscribe. *ACM Computing Surveys*, 35(2):114–131.

Loulou, I., Jmaiel, M., Drira, K., and Kacem, A. H. (2010). P/S-CoM: Building correct by design publish/ subscribe architectural styles with safe reconfiguration. *Journal of Systems and Software*, 83(3):412–428.

Mahambre, S. P., Kumar S.D., M., and Bellur, U. (2007). A taxonomy of qos-aware, adaptive event-dissemination middleware. *IEEE Internet Computing*, 11(4):35–44.

OMG (2007). Data distribution service for real-time systems, version 1.2, 2007. http://www.omg.org/spec/DDS/1.2/PDF/.

Woodcock, J. and Davies, J. (1996). *Using Z: specification, refinement, and proof.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA.