Challenge Problem: Subject-Observer Specification with Component-Interaction Automata

Pavlína Vařeková*, Barbora Zimmerova*
Faculty of Informatics
Masaryk University
602 00 Brno, Czech Republic
{xvareko1, zimmerova}@fi.muni.cz

ABSTRACT

This paper presents our solution to the *Subject-Observer Specification* problem announced as the challenge problem of the *SAVCBS 2007* workshop. The text consists of two parts. In the first part, we present the model of the Subject-Observer system in terms of Component-interaction automata. In the second part, we present our approach to verification of the system model with respect to unlimited number of Observers.

Categories and Subject Descriptors

D.2.4 [**Software Engineering**]: Software/Program Verification

General Terms

Component-based systems, verification, specification

Keywords

Component-based systems, dynamic number of components, finite-state systems, verification, software modelling

1. INTRODUCTION

The following solution to the *Subject-Observer Specification* challenge problem is based on the paper [3] that is going to be presented at the workshop. For this reason we do not repeat the definitions given in the paper and reference the reader to the paper. The model is created using the *Component-interaction automata* modelling language (first presented in [1]). For more information on the language please see [4] or the coming detailed case study [5], our result in the *CoCoME (Common Component Modelling Example) Contest*[1], where we have first experienced the Subject-Observer modelling problem.

*The authors have been supported by the grant No. 1ET400300504.

[1] `http://www.cocome.org/`

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Sixth International Workshop on Specification and Verification of Component-Based Systems (SAVCBS 2007), September 3-4, 2007, Cavtat near Dubrovnik, Croatia.
Copyright 2007 ACM ISBN 978-1-59593-721-6/07/0009 ...$5.00.

2. SPECIFICATION

Consider the assignment of the challenge problem. It states that there may be many Observers for a Subject and an Observer may be registered with more than one Subject. But it does not discuss whether the numbers of Subjects and Observers are fixed or change at run-time. While working on the *CoCoME*, we have observed that in practical applications, the number of Subjects is usually fixed, but the number of Observers can grow dynamically. In this example, we suppose the same. In addition, as distinct to the official assignment, we add a possibility of the Observer to deregister from Subjects to make the solution more interesting.

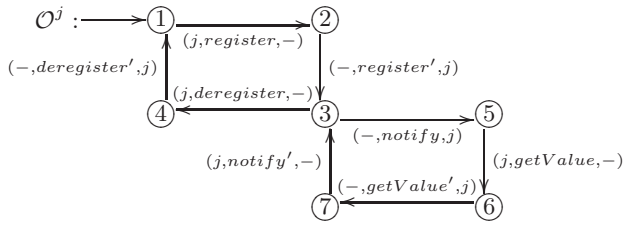
Now we present our model for the example created using Component-interaction automata. For clearness, we start with the model of the system with one Subject only. Then we generalize the model to multiple Subjects.

2.1 Model with one Subject

The model of the system with one Subject and multiple Observers is the following.

2.1.1 Observer

The Observer (in this case with a component name $j \in \mathbb{N}$) first needs to register to get to the state 3 where it can accept notifications and ask for the value managed by the Subject. In the model, each method, e.g. `register()`, is assigned a tuple of action names: `register` denotes the *call* of the method, and `register'` the *return* from the method. These two determine the start and the end of the method's execution.

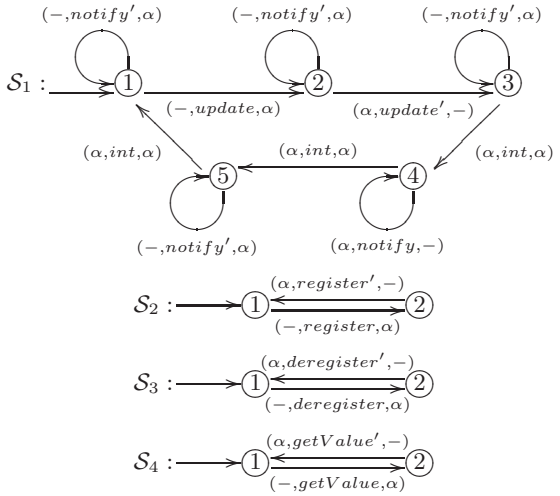


A hierarchy of component names: (j)

Figure 1: A CI model of the Observer $\mathcal{O}^j$

2.1.2 Subject

The Subject $\mathcal{S}$ (component name α) implements four methods, `update()`, `register()`, `deregister()`, `getValue()`, and hence its model consists of four parts connected via the full composition operator $\otimes$ (no transitions removed) as $\mathcal{S} = \otimes\{\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4\}$. Models of the parts are in figure 2.



A hierarchy of component names: (α)

Figure 2: A CI model of the Subject $\mathcal{S}$ parts

On the `update()` method (automaton $\mathcal{S}_1$), the Subject first receives the method call, confirms its return (to allow the updater to continue its execution while the notifications are delivered, which is common in Subject-Observer communicational models) and then takes care about notifying the Observers. This proceeds in two loops separated by internal actions. The first loop distributes the notification to the Observers, the second confirms termination of notifications. This allows the Observers to execute bodies of their methods in parallel. More, the confirmation $(-, notify', \alpha)$ is allowed also in other states than 5. This protects the system from deadlock of the Observers that do not manage to synchronize with the Subject before it leaves the state 5. Note that the composition with Observers using a handshake-like composition (required synchronization of complementary labels, which are removed and only synchronized internal labels remain) includes paths representing that $0, 1, 2, ...$,all *registered* Observers are notified. However no Observer can be notified twice. This confirms to the *at most once* constraint, which will be verified later in this text.

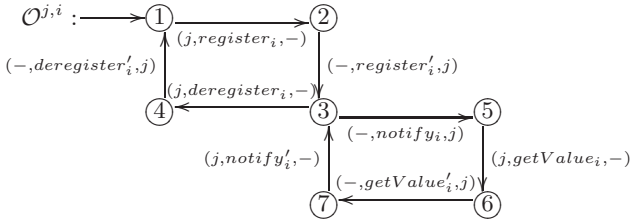
On the remaining methods `register()`, `deregister()`, `getValue()` (automata $\mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4$), the Subject only receives the call and then returns. More interesting behaviour could be inserted on the place of state 2.

2.2 Model with several Subjects

The presented model can be readily extended to the multi-Subject case. Suppose the number of Subjects is n, then we have Subjects $\mathcal{S}^i$ for $i \in \{1, 2, ..., n\}$ where i represents the *id* of the Subjects. We add this *id* also to the names of methods to distinguish which Subject an Observer wants to communicate with.

2.2.1 Observer

The Observer now consists of n parts identical up to indexes in actions, each one for communication with one Subject. A model of one part is in figure 3. The parts are again connected via the full composition operator $\otimes$, hence the model of the Observer (component name j) is $\mathcal{O}^{j,\{1,...,n\}} = \otimes\{\mathcal{O}^{j,i}\}_{i\in\{1,...,n\}}$.

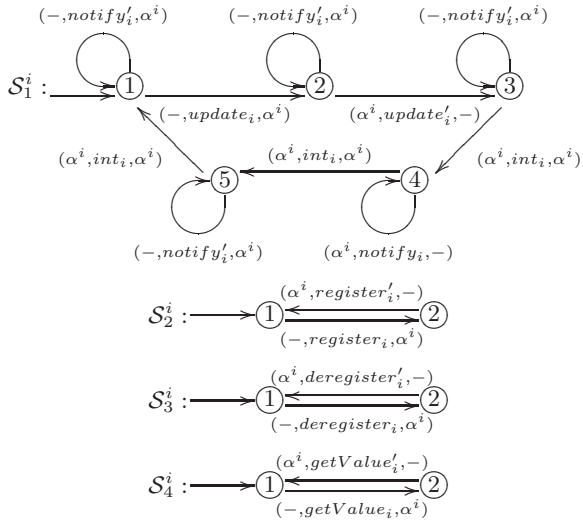


A hierarchy of component names: (j)

Figure 3: A CI model of the Observer part $\mathcal{O}^{j,i}$

2.2.2 Subject

The model of a Subject $\mathcal{S}^i$ (component name α^i where $\alpha^i, i \in \mathbb{N}$, denotes the sequence of i symbols α) is analogical to the model of Subject $\mathcal{S}$ (figure 2). It again consists of four parts $\mathcal{S}^i = \otimes\{\mathcal{S}^i_1, \mathcal{S}^i_2, \mathcal{S}^i_3, \mathcal{S}^i_4\}$. The models of the parts are in figure 4.



A hierarchy of component names: (α^i)

Figure 4: A CI model of the Subject $\mathcal{S}^i$ parts

2.2.3 The composite model

Now we can fix the number of Subjects to n, and the number of Observers to m, hence get automata $\mathcal{S}^1, \mathcal{S}^2, \ldots, \mathcal{S}^n$ and $\mathcal{O}^{1,\{1,2,...,n\}}, \mathcal{O}^{2,\{1,2,...,n\}}, \ldots, \mathcal{O}^{m,\{1,2,...,n\}}$ given by description above, and compose them together using handshake-like composition. The handshake-like composition requires synchronization of those labels that have a counterpart at other component. Such labels are afterwards removed from the composition and only the internally synchronized labels are left.

This type of composition can be realized via composition operator $\otimes^{\mathcal{F}}$ (see [3]) where $\mathcal{F} =$
$\bigcup_{i,j\in\mathbb{N}}\{_{(\alpha^i,notify_i,j),(j,notify'_i,\alpha^i),(j,register_i,\alpha^i),(\alpha^i,register'_i,j),}$
$_{(j,deregister_i,\alpha^i),(\alpha^i,deregister_i,j),(j,getValue_i,\alpha^i),(\alpha^i,getValue'_i,j)}\}$
$\cup \bigcup_{i\in\mathbb{N}}\{_{(-,update_i,\alpha^i),(\alpha^i,update'_i,-),(\alpha^i,int_i,\alpha^i)}\}$
The resulting model is then:
$\otimes^{\mathcal{F}}\{\mathcal{S}^1,\ldots,\mathcal{S}^n,\mathcal{O}^{1,\{1,2,\ldots,n\}},\ldots,\mathcal{O}^{m,\{1,2,\ldots,n\}}\}$.

Such a model is suitable for verification of properties fixed to the selected numbers of Subjects and Observers. However we are interested also in verification of the properties for an arbitrary number of Observers, because it is usual that the Observers can be added to and removed from the system dynamically. To this issue, we may apply the solution [3] presented at this workshop. However before we can do so, we need to adjust the system to the structure that is awaited by our approach. That is that the system consists of one stable component (called *provider*) and a number of dynamic components of one type (called *clients*). To meet this constraint, we only compose all Subjects into a composite Subject managing all values $\mathcal{S}^{\{1,\ldots,n\}} = \otimes\{\mathcal{S}^i\}_{i\in\{1,\ldots,n\}}$ and define the dynamic system model as $\mathcal{S}^n\&\mathcal{O} = (\mathcal{S}^{\{1,\ldots,n\}},\{\mathcal{O}^{j,\{1,\ldots,n\}}\}_{j\in\mathbb{N}},\mathcal{F})$ and a composite system with m Observers as $\mathcal{S}^n\&\mathcal{O}_m = \otimes^{\mathcal{F}}\{\mathcal{S}^{\{1,\ldots,n\}},\mathcal{O}^{1,\{1,\ldots,n\}},\ldots,\mathcal{O}^{m,\{1,\ldots,n\}}\}$.

Now the composite Subject represents the *provider* and Observers the *clients* of the dynamic system we have just defined.

3. VERIFICATION

In this section, we describe the verification of a dynamic system $\mathcal{S}^n\&\mathcal{O}$ for arbitrary fixed number of Subjects n. We present the application of the approach introduced in [3] for effective verification of properties expressed as sequences $Property(\mathcal{S}^n\&\mathcal{O},m)$. Then we illustrate the technique on the properties of the systems $\mathcal{S}^1\&\mathcal{O}$ and $\mathcal{S}^2\&\mathcal{O}$.

3.1 Properties for verification

In formal verification techniques, like *model checking* [2], the properties for verification are specified in temporal logics. In our approach, we use the logic CI-LTL. CI-LTL is an extension of the action-based LTL, which is in addition to expressing that an action (label) l is proceeding $\mathcal{P}(l)$ able to express that a given label l is enabled $\mathcal{E}(l)$ in a state of a path (one-step branching). See [3] for description of CI-LTL.

Assume a dynamic system $\mathcal{D}$. The properties that we aim to verify, can be specified with a sequence of formulas $\{\varphi_i\}_{i\in\mathbb{N}_0}$ over $\mathcal{L}_{\mathcal{D}}$ such that a property is satisfied iff for each $i\in\mathbb{N}_0$ it holds that $\varphi_i \models \mathcal{D}_i$. Note that not every sequence of formulas $\{\varphi_i\}_{i\in\mathbb{N}_0}$ represents a meaningful property of the system. Thus we concentrate on the formulas satisfying the following.

(1) The property makes no distinctions among clients.
(2) If the property is violated by a path in a system $\mathcal{D}_{i+j}$ where j components do not perform any steps, the property is violated by the same sequence of labels also in the system with i clients only.

Moreover, we pose the following two restrictions. See [3] for proper description of these.

(3) We focus only on the formula sequences that represent properties whose violation involves a finite number m of observed components only.
(4) We consider only the properties that are invariant under stuttering according to CI-LTL.

The set of properties that fulfills these conditions for model $\mathcal{D}$ and the finite number of components m is denoted $Property(\mathcal{D},m)$.

3.2 Verification technique

The core idea of the verification process is based on finding a number $k\in\mathbb{N}$ such that if the property is violated on a system then it must be violated for the system with maximally k clients deployed. If such k exists, it allows us to reduce verification of the infinite system to a finite one. In particular, to a verification of a finite number of finite systems – with $0,1,\ldots,k$ clients deployed. The value k for a dynamic system $\mathcal{D}$ and a set of observable actions X can be estimated as a sum of two measures. The first one is a measure of complexity of a dynamic system reflecting the maximal number of clients that are regarded by the provider. The second one is a similar measure on properties that reflects the minimal number of clients necessary to exhibit a path violating studied property.

3.3 Optimizations

The model of the dynamic system for the Subject-Observer problem $\mathcal{S}^n\&\mathcal{O}$ does not exactly follow the pattern client-provider supposed by our approach [3]. In [3] we require that the provider of a dynamic system can in any time regard at most n clients for a constant $n\in\mathbb{N}_0$ [2]. In the case of a dynamic system $\mathcal{S}^n\&\mathcal{O}$, the client (Observer) is regarded (served) if and only if it is registered. Note that the number of registered Observers in the system $\mathcal{S}^n\&\mathcal{O}$ can be arbitrary. Hence $|\mathcal{S}^n\&\mathcal{O}|_{\mathcal{L}_{\mathcal{C}_0}} = \infty$. Therefore we need to use two optimizations that lead to $|\mathcal{S}^n\&\mathcal{O}|_X$ finite.

3.3.1 Optimization 1.

With regard to the presence of sub-formulas $\mathcal{P}(l)$ and $\mathcal{E}(l)$ in the verified sequence of formulas, we minimize the set X used in the computation of $|\mathcal{D}|_X$. This decrease of the number of observed labels increases the probability that w.r.t. these labels the provider (combined Subject) regards at most a fixed number of clients (Observers) at any time, which makes $|\mathcal{D}|_X$ finite.

Example 3.1. Suppose a dynamic system $\mathcal{S}^1\&\mathcal{O}$ and two sets of observable labels X for which we compute the value $|\mathcal{S}^1\&\mathcal{O}|_X$:

• $X = \{(\alpha,notify_1,-),(-,notify'_1,\alpha)\}$
In this case, the Observer is in a cycle of service if and only if it started and did not finish the notification. The number

[2]It could be said that a client is regarded (or served) by the provider if it synchronized with the provider on an observable label that started client's execution (cycle of service) and has not yet synchronized on an observable label representing the end of this execution. It holds that the value $|\mathcal{D}|_X$ is always grater than the maximal number of served clients.

of such Observers is not bound. It holds that the maximal number of served Observers is lower or equal to $|\mathcal{S}^1\&\mathcal{O}|_X$, hence $|\mathcal{S}^1\&\mathcal{O}|_X = \infty$.

• $X = \{(-, notify'_1, \alpha)\}$
In this case no Observer can be in a cycle of service. Roughly speaking this is because X implies only one observable transition and hence no Observer can get in between of two observable transitions that bound a cycle of service (for definition of a cycle of service see [3]). Therefore it is possible that the value $|\mathcal{S}^1\&\mathcal{O}|_X$ is finite. After computing this value we get $|\mathcal{S}^1\&\mathcal{O}|_X = 1$.

3.3.2 Optimization 2.

This optimization is based on a modification of the dynamic system and the property in a way that thanks to them we may verify the original property on the original system, and we increase the probability that the measure of complexity of the modified model with respect to the modified set of properties and the set of observable labels is finite. Let $\{\varphi_i\}_{i\in\mathbb{N}_0} \in Property(\mathcal{D}, m)$. Suppose:
• A dynamic system $\overline{\mathcal{D}}$ created from system $\mathcal{D}$ by modification of its provider – modelling the provider of the system $\mathcal{D}$ composed with m clients (see figure 5). The remaining items of the dynamic system are identical to the system $\mathcal{D}$.
• A sequence of properties $\{\overline{\varphi}_i\}_{i\in\mathbb{N}_0}$ such that the formula $\overline{\varphi}_n$ captures for the automaton $\overline{\mathcal{D}}_n$ the same property as the formula φ_{n+m} narrowed down to the clients $1, ..., m$ for the automaton $\mathcal{D}_{n+m}$.

Example 3.2. The formulas $\{\varphi_i\}_{i\in\mathbb{N}_0} \in Property(\mathcal{D}, 1)$ for instance capture the property:
After every start of a notification (sent to any Observer) there follows the finish of this notification.
Then the formulas $\{\overline{\varphi}_i\}_{i\in\mathbb{N}_0} \in Property(\overline{\mathcal{D}}, 0)$ capture the property:
After every start of a notification sent to an Observer that is modelled as a part of the provider, there follows the finish of this notification.

As $\{\varphi_i\}_{i\in\mathbb{N}_0} \in Property(\mathcal{D}, m)$ and these formulas make no distinction among clients, it holds that:

$$\overline{\mathcal{D}}_n \models \overline{\varphi}_n \text{ iff } \mathcal{D}_{n+m} \models \varphi_{n+m}.$$

As $\{\overline{\varphi}_i\}_{i\in\mathbb{N}_0} \in Property(\overline{\mathcal{D}}, 0)$ then if $\overline{\mathcal{D}}_0 \models \overline{\varphi}_0, \ldots, \overline{\mathcal{D}}_{|\overline{\mathcal{D}}|_{X'}} \models \overline{\varphi}_{|\overline{\mathcal{D}}|_{X'}}$ (for appropriate X') it follows from the intuition presented in Optimization 2 that it must hold that $\overline{\mathcal{D}}_n \models \overline{\varphi}_n$ for every $n \in \mathbb{N}_0$. From lemmas in [3] we get that for every $n \in \mathbb{N}_0$ it holds that $\mathcal{D}_{n+m} \models \varphi_{n+m}$.
Moreover, the set X' contains less external labels (input and output labels) than the set X, and hence there is higher probability that $|\overline{\mathcal{D}}|_{X'}$ is finite.

Example 3.3. Suppose that for the system $\mathcal{S}^1\&\mathcal{O}$ we are interested in the verification of the properties from example 3.2. These properties can be captured as formulas $Property(\mathcal{S}^1\&\mathcal{O}, m)$ and for their verification, it suffices to use the set of observable labels $X = \{(\alpha, notify_1, -), (-, notify'_1, \alpha)\}$. From the reasoning above, we know that $|\mathcal{S}^1\&\mathcal{O}|_X = \infty$. However if we consider the model $\overline{\mathcal{S}^1\&\mathcal{O}}$ where the provider contains not only n components for the Subjects, $n = 1$ here, but also m components representing Observers, $m = 1$ here (modelled with component name β), the minimal set of observable actions is $X' = \{(\alpha, notify_1, \beta), (\beta, notify'_1, \alpha)\}$ and it holds that $\overline{|\mathcal{S}^1\&\mathcal{O}|}_{X'} \leq 1$.

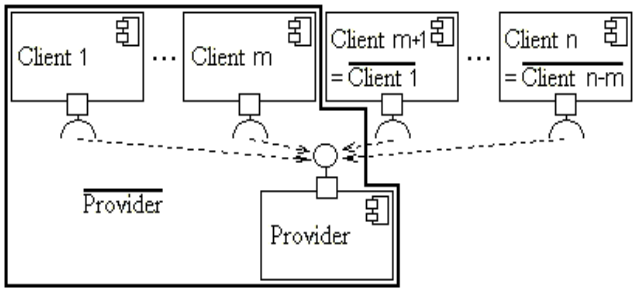


Figure 5: A dynamic system $\mathcal{D}$ with n clients and dynamic system $\overline{\mathcal{D}}$ with $n - m$ clients

3.4 Algorithm for verification

The next section presents examples of the properties of the Subject-Observer system that are interesting for verification. Validity of these properties can be showed by intuitive reasoning. However for more complex properties, the automatic verification technique is necessary. In this section we present such a technique based on the results from [3]. This verification will take advantage of the optimizations introduced above. It consists of the following three steps.

(1) Creation of a harmonized sequence of formulas $\{\varphi_i\}_{i\in\mathbb{N}_0}$ that corresponds to a given property for the dynamic system $\mathcal{S}^n\&\mathcal{O}$. Detecting whether there is a finite m such that the sequence of formulas $\{\varphi_i\}_{i\in\mathbb{N}_0}$ is part of the set $Property(\mathcal{S}^n\&\mathcal{O}, m)$. If there is no such m, it is not possible to verify $\{\varphi_i\}_{i\in\mathbb{N}_0}$ using the approach from [3]. It is clear that a property can be captured with various formulas, some of them can be verified using our approach, some of them cannot (see the property (c) in section 4.1).

(2) Modification of the dynamic system $\mathcal{S}^n\&\mathcal{O}$ and the sequence of formulas $\{\varphi_i\}_{i\in\mathbb{N}_0}$ using Optimization 2 to the dynamic system $\overline{\mathcal{S}^n\&\mathcal{O}}$ (the Observers that become part of the provider will be referred to as $\beta, \beta\beta, \ldots$) and the sequence of formulas $\{\overline{\varphi}_i\}_{i\in\mathbb{N}_0}$. With respect to the Optimization 1, we select the set X of observable labels and we compute $|\overline{\mathcal{S}^n\&\mathcal{O}}|_X$.

(3) The dynamic system $\overline{\mathcal{S}^n\&\mathcal{O}}$ and formulas $\{\overline{\varphi}_i\}_{i\in\mathbb{N}_0}$ agree with the prerequisites of Lemma 5.4 from [3]. Hence for the verification of the given property, it suffices to check the validity of the according formulas on the systems $\overline{\mathcal{S}^n\&\mathcal{O}}_0$, $\overline{\mathcal{S}^n\&\mathcal{O}}_1$, ..., $\overline{\mathcal{S}^n\&\mathcal{O}}_{|\overline{\mathcal{S}^n\&\mathcal{O}}|_X}$. If the verification succeeds, it is verified that the property holds for all systems $\overline{\mathcal{S}^n\&\mathcal{O}}_0$, $\overline{\mathcal{S}^n\&\mathcal{O}}_1$, ... Therefore from the discussion in Optimization 2 it follows that the original property is verified on models $\mathcal{S}^n\&\mathcal{O}_m$, $\mathcal{S}^n\&\mathcal{O}_{m+1}$, ... For verification of the whole dynamic system, we more need to verify the original property on $\mathcal{S}^n\&\mathcal{O}_0$, $\mathcal{S}^n\&\mathcal{O}_1$, ..., $\mathcal{S}^n\&\mathcal{O}_{m-1}$.

Only the first step (1) needs to be supported manually, the steps (2) and (3) can be done automatically.

The intuition for getting the over-approximation of the value $|\mathcal{D}|_X$, which we will use together with Lemma 5.4 from [3] for automatization of the steps (2) and (3) from the algorithm above is based on a simple observation:

"If for a dynamic system $\mathcal{D}$ the automaton $\mathcal{D}_n$ generates all possible runs with respect to the observable labels X, then for every $i \in \mathbb{N}$ the automaton $\mathcal{D}_{n+i}$ generates again the same runs. Hence it holds that $|\mathcal{D}|_X \leq n$."

4. EXAMPLES

Now we present several examples to demonstrate the verification of the Subject-Observer system model.

4.1 Verification of the syst. with one Subject

In this section, we illustrate the verification technique presented above on the dynamic system $\overline{\mathcal{S}^1\&\mathcal{O}}$. In the examples, we discuss the first step in detail because it is the manual part of the verification. For the remaining two steps, which can be done automatically, we just present the results without further discussion.

a) If a run contains infinitely many steps concerning some Observer, then the Observer is infinitely many times enabled to receive notifications.

- This property can be expressed by a harmonized set of formulas $\{\varphi_i\}_{i\in\mathbb{N}_0}$:
  $\varphi_i = \bigwedge_{j\leq i} \varphi(\alpha, j)$,
  where
  $\varphi(\alpha, j) = [\mathcal{G}\,\mathcal{F}\,\bigvee_{l\in Lab_j} \mathcal{P}(l)] \Rightarrow [\mathcal{G}\,\mathcal{F}\,\mathcal{E}(\alpha, notify_1, j)]$,
  $Lab_j = \{_{(j,register_1,\alpha),(\alpha,register'_1,j),(j,deregister_1,\alpha),}$
  $_{(\alpha,deregister'_1,j),(\alpha,notify_1,j),(j,notify'_1,\alpha),(j,getValue_1,\alpha),}$
  $_{(\alpha,getValue'_1,j)}\}$
  (Lab_j is a set of all the labels that are present in some of the automata $\mathcal{S}^1\&\mathcal{O}_j$, $\mathcal{S}^1\&\mathcal{O}_{j+1}$, ... concerning the Observer $\mathcal{O}^j$).
  For any $i \in \mathbb{N}_0$ and an infinite run π starting in an initial state of the automaton $\mathcal{S}^1\&\mathcal{O}$ satisfying $\pi \not\models \varphi_i$, there exists a number $j \in \mathbb{N}$ such that on this run the formula $\varphi(\alpha, j)$ is not valid. For confirming this violation it suffices to observe the Subject and the Observer with the numerical name j. Hence in general, it always suffices to observe one Observer to confirm the violation of the property. It holds that:
  $\{\varphi_i\}_{i\in\mathbb{N}_0} \in Property(\mathcal{S}^1\&\mathcal{O}, 1)$

- The modified sequence of formulas $\{\overline{\varphi}_i\}_{i\in\mathbb{N}_0}$ is then
  $\overline{\varphi}_i = \varphi(\alpha, \beta)\ \ \forall i \in \mathbb{N}_0$.
  In this case, it suffices to compute the value $|\overline{\mathcal{S}^1\&\mathcal{O}}_{Lab_\beta}|$ (see Lemma 5.4 from [3])
  and from the algorithm discussed above it follows that:
  $|\overline{\mathcal{S}^1\&\mathcal{O}}_{Lab_\beta}| \leq 0$.

- Then after verifying the models $\mathcal{S}^1\&\mathcal{O}_0$ and $\overline{\mathcal{S}^1\&\mathcal{O}_0}$ we can conclude that the property always holds.

b) If one of the registered Observers receives a notification and some other Observer is also ready to receive one (is registered and has not receive it yet), it will receive the notification too.

- This property can be expressed by a harmonized set of formulas $\{\varphi_i\}_{i\in\mathbb{N}_0}$:
  $\varphi_i = \bigwedge_{j_1,j_2\leq i, j_1\neq j_2} \varphi(\alpha, j_1, j_2)$,
  where
  $\varphi(\alpha, j_1, j_2) = \mathcal{G}\ [(\mathcal{P}(\alpha, notify_1, j_1) \wedge \mathcal{E}(\alpha, notify_1, j_2)) \Rightarrow (true\ \mathcal{U}\ \mathcal{P}(\alpha, notify_1, j_2))]$
  For any $i \in \mathbb{N}_0$ and an infinite run π starting in an initial state of the automaton $\mathcal{S}^1\&\mathcal{O}$ satisfying $\pi \not\models \varphi_i$, there exists distinct numbers $j_1, j_2 \in \mathbb{N}$ such that on this run the formula $\varphi(\alpha, j_1, j_2)$ is not valid. For confirming this violation it suffices to observe the Subject and the Observers with the numerical names j_1 and j_2. Hence in general, it always suffices to observe two Observer to confirm the violation of the property. It holds that:
  $\{\varphi_i\}_{i\in\mathbb{N}_0} \in Property(\mathcal{S}^1\&\mathcal{O}, 2)$.

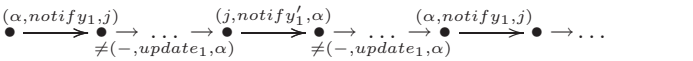
- $\{\overline{\varphi}_i\}_{i\in\mathbb{N}_0}$ satisfies $\forall i \in \mathbb{N}_0\ \overline{\varphi}_i = \varphi(\alpha, \beta, \beta\beta)$.
  In this case, it suffices to compute the value $|\overline{\mathcal{S}^1\&\mathcal{O}}_X|$ for $X = \{_{(\alpha,notify_1,\beta^k),(\beta^k,notify'_1,\alpha),(\alpha,register'_1,\beta^k),}$
  $_{(\beta^k,deregister_1,\alpha)\ \mid\ k\in\{1,2\}}\}$
  (see Lemma 5.4 from [3]) and from the algorithm discussed above it follows that: $|\overline{\mathcal{S}^1\&\mathcal{O}}|_X \leq 0$.

- Then after verifying the models $\mathcal{S}^1\&\mathcal{O}_0$, $\mathcal{S}^1\&\mathcal{O}_1$ and $\overline{\mathcal{S}^1\&\mathcal{O}_0}$ we can conclude on the validity of the formula. In this case the formula does not hold which is confirmed by the model $\overline{\mathcal{S}^1\&\mathcal{O}_0}$.
  The counterexample is the run of the automaton $\overline{\mathcal{S}^1\&\mathcal{O}_0}$ where first the Observers β and $\beta\beta$ register for receiving notifications, then the Subject is updated, it sends the notification to Observer β, but never deliverers the notification to Observer $\beta\beta$ because this gets locked in a loop of registering and deregistering.

c) After any update, each Observer receives at most one notification about value change. This reflects that *"each Observer is called at most once per state change"* **from the assignment of the problem.**

- This property can be expressed by a set of formulas $\{\varphi_i\}_{i\in\mathbb{N}_0}$:
  $\varphi_i = \bigwedge_{j\leq i} \neg\mathcal{F}\ \varphi(\alpha, j)$,
  where
  $\varphi(\alpha, j) = \mathcal{P}(\alpha, notify_1, j) \wedge \mathcal{X}\ [\neg\mathcal{P}(-, update_1, \alpha)\ \mathcal{U}\ \mathcal{P}(\alpha, notify_1, j)]$
  These formulas contain operator $\mathcal{X}$, therefore they are not invariant under stuttering and that is why $\{\varphi_i\}_{i\in\mathbb{N}_0} \notin Property(\mathcal{S}^1\&\mathcal{O}, m)$ holds for any $m \in \mathbb{N}_0$.
  We take advantage of the fact that each Observer after getting the notification must first confirm the end of the notification before it is able to receive another one. Hence we can express the property with the following harmonized set of formulas:
  $\varphi_i = \bigwedge_{j\leq i} \neg\mathcal{F}\ \varphi(\alpha, j)$,
  where
  $\varphi(\alpha, j) = \mathcal{P}(\alpha, notify_1, j) \wedge [\neg\mathcal{P}(-, update_1, \alpha)\ \mathcal{U}\ \{\mathcal{P}(j, notify'_1, \alpha) \wedge [\neg\mathcal{P}(-, update_1, \alpha)\ \mathcal{U}\ \mathcal{P}(\alpha, notify_1, j)]\}]$.

  Example of a run violating the formula $\varphi(\alpha, j)$ is for instance (without names of states):



Using the reasoning analogical to the one above, we can conclude that $\{\varphi_i\}_{i\in\mathbb{N}_0} \in Property(\mathcal{S}^1\&\mathcal{O}, 1)$

- $\overline{\varphi}_i = \neg\mathcal{F}\ \varphi(\alpha, \beta)$.
  We need to compute the value $|\overline{\mathcal{S}^1\&\mathcal{O}}|_{\{(-,update_1,\alpha),(\alpha,notify_1,\beta),(\beta,notify'_1,\alpha)\}}$.
  $|\overline{\mathcal{S}^1\&\mathcal{O}}|_{\{(-,update_1,\alpha),(\alpha,notify_1,\beta),(\beta,notify'_1,\alpha)\}} \leq 1$.

- Then after verifying the models $\mathcal{S}^1\&\mathcal{O}_0$, $\overline{\mathcal{S}^1\&\mathcal{O}}_0$ and $\overline{\mathcal{S}^1\&\mathcal{O}}_1$ we can conclude that the property always holds.

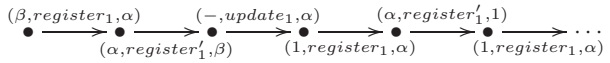
d) Anytime, the Subject is in future enabled to execute `update()`.

- This property can be expressed by a harmonized set of formulas $\{\varphi_i\}_{i\in\mathbb{N}_0}$:
  $\varphi_i = \mathcal{G}\,\mathcal{F}\,\mathcal{E}(-, update_1, \alpha)$.
  As the formulas do not contain a name of any Observer, it holds that $\{\varphi_i\}_{i\in\mathbb{N}_0} \in Property(\mathcal{S}^1\&\mathcal{O}, 0)$.

- $\overline{\varphi}_i = \varphi_i = \mathcal{G}\,\mathcal{F}\,\mathcal{E}(-, update_1, \alpha)$.
  We need to compute the value $|\overline{\mathcal{S}^1\&\mathcal{O}}|_{\{(-,update_1,\alpha),(\alpha,int_1,\alpha)\}}$.
  It holds that $|\overline{\mathcal{S}^1\&\mathcal{O}}|_{\{(-,update_1,\alpha),(\alpha,int_1,\alpha)\}} \leq 1$.

- The verification of the models $\overline{\mathcal{S}^1\&\mathcal{O}}_0$, $\overline{\mathcal{S}^1\&\mathcal{O}}_1$ shows that the property holds on the model $\overline{\mathcal{S}^1\&\mathcal{O}}_0$ but does not hold on the model $\overline{\mathcal{S}^1\&\mathcal{O}}_1$.

  The counterexample is the run of the automaton $\overline{\mathcal{S}^1\&\mathcal{O}}_1$ (for space reasons we do not include the names of the states):



e) If it holds for a run that every update of the Subject is followed by starting the notification of all the Observers, then each update will be also followed by finishing of the notifications by all the Observers.

- This property is an example of a property that can be expressed as a sequence of formulas, but there exists no $m \in \mathbb{N}_0$ such that the sequence of formulas is in $Property(\mathcal{S}^1\&\mathcal{O}, m)$. This follows from the fact that if a property is violated by some run, we are not able to bound the number of clients that always suffices to confirm the faultiness of the run.

4.2 Verification of the syst. with two Subjects

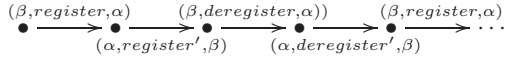
In this section we present examples of two properties discussed above on the dynamic system Subject-Observer with two Subjects $\mathcal{S}^2\&\mathcal{O}$.

f) If a run contains infinitely many steps concerning some Observer, then the Observer is infinitely many times enabled to receive notifications from both Subjects.

- This property can be expressed by a harmonized set of formulas $\{\varphi_i\}_{i\in\mathbb{N}_0}$:
  $\varphi_i = \bigwedge_{j\leq i} \varphi(\alpha, \alpha\alpha, j)$,
  where
  $\varphi(\alpha, \alpha\alpha, j) = [\mathcal{G}\,\mathcal{F}\ \bigvee_{l\in Lab_j} \mathcal{P}(l)] \Rightarrow$
  $[\mathcal{G}\,(\mathcal{F}\,\mathcal{E}(\alpha, notify_1, j)) \wedge (\mathcal{F}\,\mathcal{E}(\alpha\alpha, notify_2, j))]$,
  and
  $Lab_j = \{_{(j,register_k,\alpha^k),(\alpha^k,register'_k,j),(j,deregister_k,\alpha^k),}$
  $_{(\alpha^k,deregister'_k,j),(\alpha^k,notify_k,j),(j,notify'_k,\alpha^k),}$
  $_{(j,getValue_k,\alpha^k),(\alpha^k,getValue'_k,j)}\ |_{k\in\{1,2\}}\}$
  (Lab_j is a set of all the labels that are present in some of the automata $\mathcal{S}^2\&\mathcal{O}_0$, $\mathcal{S}^2\&\mathcal{O}_1$, ... concerning the Observer $\mathcal{O}^j$).
  From the reasons analogical to the ones discussed above, it follows that:
  $\{\varphi_i\}_{i\in\mathbb{N}_0} \in Property(\mathcal{S}^2\&\mathcal{O}, 1)$

- The modified sequence of formulas $\{\overline{\varphi}_i\}_{i\in\mathbb{N}_0}$ is then $\overline{\varphi}_i = \varphi(\alpha, \alpha\alpha, \beta)\ \ \forall i \in \mathbb{N}_0$.
  Hence it suffices to compute the value $|\overline{\mathcal{S}^2\&\mathcal{O}}_{Lab_\beta}|$ and it follows that: $|\overline{\mathcal{S}^2\&\mathcal{O}}_{Lab_\beta}| \leq 0$.

- From the verification of the models $\overline{\mathcal{S}^2\&\mathcal{O}}_0$ we learn that the property is not satisfied. The counterexample is the run containing an infinite number of the transitions concerning the Observer β who can never accept the notification about the update of the value managed by the Subject $\alpha\alpha$ (for space reasons we again do not include the names of the states):



g) After any update, each Observer receives at most one notification about value change. This reflects that *"each Observer is called at most once per state change"* **from the assignment of the problem.**

- Analogically to the property (c) in section 4.1, we take advantage of the fact that each Observer after getting the notification must first confirm the end of the notification before it is able to receive another one. Hence we can express the property with the following harmonized set of formulas (without the operator $\mathcal{X}$):
  $\varphi_i = [\bigwedge_{j\leq i} \neg\mathcal{F}\ \varphi(1, j)] \wedge [\bigwedge_{j\leq i} \neg\mathcal{F}\ \varphi(2, j)]$,
  where
  $\varphi(k, j) = \mathcal{P}(\alpha^k, notify_k, j) \wedge [\neg\mathcal{P}(-, update_k, \alpha^k)\ \mathcal{U}$
  $\{\mathcal{P}(j, notify'_k, \alpha^k) \wedge [\neg\mathcal{P}(-, update_k, \alpha^k)\ \mathcal{U}$
  $\mathcal{P}(\alpha^k, notify_k, j)]\}]$.

  Using the reasoning analogical to the one above, we can conclude that $\{\varphi_i\}_{i\in\mathbb{N}_0} \in Property(\mathcal{S}^2\&\mathcal{O}, 1)$

- $\overline{\varphi}_i = \neg\mathcal{F}\ \varphi(\alpha, \beta) \wedge \neg\mathcal{F}\ \varphi(\alpha\alpha, \beta)$.
  We need to compute the value $|\overline{\mathcal{S}^2\&\mathcal{O}}|_X$ for
  $X = \{(-, update_1, \alpha), (\alpha, notify_1, \beta), (\beta, notify'_1, \alpha),$
  $(-, update_2, \alpha\alpha), (\alpha\alpha, notify_2, \beta), (\beta, notify'_2, \alpha\alpha)\}$
  and it holds that
  $|\overline{\mathcal{S}^2\&\mathcal{O}}|_X \leq 1$.

- Then after verifying the models $\mathcal{S}^2\&\mathcal{O}_0$, $\overline{\mathcal{S}^2\&\mathcal{O}}_0$ and $\overline{\mathcal{S}^2\&\mathcal{O}}_1$ we can conclude that the property always holds.

4.3 Closing remarks

We could also study the general issue of whether for a certain property (parametrized with a number of Subjects and

Observers) there exists n_{max} and n'_{max} such that the validity of the property for systems with at most n'_{max} Subjects and at most n_{max} Observers implies the validity of the property on all systems. The reasoning would follow the same intuition as the verification of the properties from the set $Property(\mathcal{D}, m)$ on the dynamic system $\mathcal{D}$. However from space reasons, we do not study this general case here.

5. REFERENCES

[1] L. Brim, I. Černá, P. Vařeková, and B. Zimmerova. Component-Interaction Automata as a Verification-Oriented Component-Based System Specification. In *Proceedings of SAVCBS'05*, pages 31–38, Lisbon, Portugal, September 2005.

[2] E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking*. The MIT Press, USA, January 2000.

[3] P. Vařeková, P. Moravec, I. Černá, and B. Zimmerova. Effective-Verification of Systems with a Dynamic-Number of Components. In *Proceedings of SAVCBS'07*, Dubrovnik, Croatia, September 2007.

[4] I. Černá, P. Vařeková, and B. Zimmerova. Component-interaction automata modelling language. Technical Report FIMU-RS-2006-08, Masaryk University, Faculty of Informatics, Brno, Czech Republic, October 2006.

[5] B. Zimmerova, P. Vařeková, N. Beneš, I. Černá, L. Brim, and J. Sochor. *The Common Component Modeling Example: Comparing Software Component Models*, chapter Component-Interaction Automata Approach (CoIn). To appear in LNCS, 2007.