Assessing the Applicability of Use Case Maps for Business Process and Workflow Description

Gunter Mussbacher, Daniel Amyot
SITE, University of Ottawa, 800 King Edward, Ottawa, ON, K1N 6N5, Canada
{gunterm | damyot}@site.uottawa.ca

Abstract

Use Case Maps (UCMs) have already been used to describe business processes and workflows at a high level of abstraction. The semantics of UCMs, however, require further clarification and enhancement. An initial assessment based on 27 workflow and communication patterns (a) highlighted some of the semantic variation points of UCMs, (b) introduced small extensions to the UCM language in order to more precisely define scenarios, high-level business processes, and workflows, and (c) compared UCMs with other business process and workflow languages. This short paper summarizes the continuation of the assessment with a larger set of workflow patterns recently made available. The assessment concludes that the UCM notation including the proposed extensions is a competitive language to describe high-level business processes and workflows, while providing additional benefits over the other languages.

Keywords—Business Process and Workflow Modeling Languages, Requirements Engineering, Use Case Maps, User Requirements Notation.

1. Introduction

Use Case Maps (UCMs) [7] are one of the two pillars of the International Telecommunication Union's (ITU) effort to standardize the User Requirements Notation (URN). Although UCMs have been used for business process and workflow modeling for a while [4][9], only a recent assessment [3] based on workflow and communication patterns evaluated more formally the applicability of UCMs to this type of modeling.

It is now necessary to revisit the initial assessment, because the set of workflow patterns at the basis of that assessment [8] was itself recently updated and extended [6]. The revised assessment also includes an updated comparison to the same three standards used in the initial assessment: the Business Process Modeling Notation (BPMN) 1.0, the Business Process Execution Language for Web Services (BPEL4WS) 1.1, and UML 2.0 Activity Diagrams (AD).

The goal of this assessment is to further understand the strengths and weaknesses of UCMs compared to other notations, as this is important for the future of UCMs as a business process and workflow modeling language as well as a general scenario description language. The assessment must also provide insight into how to evolve UCMs and tool support for UCMs.

In the remainder of this paper, section 2 provides background on workflow patterns and UCMs. Section 3 updates the initial assessment, re-comparing UCMs to BPMN 1.0, BPEL4WS 1.1, and UML 2.0 AD. Section 4 gives conclusions and identifies future work.

2. Background

Recently, a report [6] was published that revisits the initial set of *workflow patterns*, published a few years ago [8], which describes typical control flow dependencies in workflow models. The research led to the YAWL (Yet Another Workflow Language) initiative with its goal to create a workflow language with direct support for all of the discovered workflow patterns. The new report defines the workflow pattern more formally with Colored Petri-Nets and introduces further patterns, increasing the number of workflow patterns to 43. Due to space constraints, the individual patterns cannot be described here. The interested reader is referred to [10]. The new report also re-evaluates 14 workflow management systems and standards for business process and workflow modeling. It is important to note that although the workflow patterns were initially observed for workflow systems, they have a much broader scope and can be used to evaluate general scenario description languages [6].

Use Case Maps (UCMs) [7] are a visual notation for the description of functional requirements and

high-level architecture expressed as scenarios. UCMs abstract from the details of message exchange and communication infrastructures while still showing the interaction between architectural entities. UCMs are integrated with goal models described with the *Goal-oriented Requirement Language* (GRL), allowing for the seamless capture of stakeholders' goals, rationale, and alternative solutions to achieve such goals. The solutions are reasoned about with GRL and their behavior and structure described in more detail with UCMs. UCMs and GRL form the *User Requirements Notation* (URN), a standardization effort of the International Telecommunication Union's (ITU). The semi-formal nature of UCMs enables reasoning about undesired interactions between scenarios, analysis of performance and architectural alternatives, and model-driven test generation. Weiss and Amyot [9] show how URN can be used to model business processes, exemplified by a supply chain management case study. Pourshahid *et al*. [4] apply URN to online business process monitoring and alignment. Due to space constraints, the reader is referred to [1][7] for a more detailed description of the UCM notation.

The Eclipse-based jUCMNav [5] tool offers support for URN. One of jUCMNav's features is a *traversal mechanism*. Given a UCM scenario definition specifying one path through the UCM model with the help of global *scenario variables*, the traversal mechanism highlights the scenario or translates it into message sequence charts (MSC). With the help of the traversal mechanism, the scenario definitions effectively become a test suite for the UCM model. Furthermore, the traversal mechanism defines more precisely the semantics of UCMs. As there is no standard traversal mechanism for UCMs, jUCMNav's traversal mechanism [2] is based on a simple but intuitive interpretation of the notational elements for sequences, alternatives, and concurrency in UCMs. An OR-fork is an XOR, there is no synchronization on an OR-join, an AND-fork denotes strict concurrency, an AND-join requires all incoming branches to arrive before continuation, and the selection policy of a dynamic stub is also an XOR.

The initial assessment of UCMs [3] suggested a number of small changes to the UCM notation to be able to support more workflow patterns. The *synchronizing stub* is a dynamic stub which allows its plug-in maps to be executed in parallel. By default, a synchronizing stub expects all plug-in maps chosen by its selection policy (i.e., not necessarily all plug-in maps defined for the stub) to finish before the path traversal is allowed to continue. A *synchronization threshold* N (with $1 \leq N \leq$ number of the stub's plug-in maps) may override the default and can be defined for each out-path of a stub. Furthermore, a *replication factor* can be defined for each plug-in map, specifying how many instances of the plug-in map are required. The initial assessment also suggested improvements to the current traversal mechanism to support multiple choice and deferred choice OR-forks, replication factors of components for multiple instances (MI) that do not need to be synchronized, and interleaved activities.

3. Pattern-Based Assessment of UCMs

The first 20 patterns as well as pattern WCP-30 (Structured Partial Join) were discussed in [3]. All of them are therefore only summarized in Table 2 except for WCP-18 (Milestone) which is discussed in more detail as the level of support for this pattern had to be lowered. The first assessment of WCP-18 did not cover the case where a parallel branch influences another parallel branch. The example given in [3] only shows that UCMs can model WCP-18 for one branch with the help of a deferred choice. With two parallel branches (see Figure 1), scenario variables in addition to the deferred choice are required to indicate to the second branch (e.g., *sendBill* and *receivePayment*) when the milestone has been reached in the first branch (e.g., *shipGoods*). Essentially, only after goods were shipped (i.e., the variable *Shipped* is set to true) and between sending a bill and receiving payment is it possible to resend the bill. The need to use variables, however, reduces the assessment score from + to +/-.

Of the new patterns, WCP-21 (Structured Loop), WCP-22 (Recursion), and WCP-33 (Generalized AND-Join) are supported by basic UCM constructs. WCP-23 (Transient Trigger) and WCP-24 (Persistent Trigger) can be modeled using the waiting place and timer constructs but the current traversal mechanism persists the first triggers only. This needs to be improved so that persistent or transient behavior can be chosen for a waiting place or timer. WCP-40 (Interleaved Routing) is supported through the same mechanism as WCP-17 (Interleaved Parallel Routing).

WCP-28 (Blocking Discriminator), WCP-31 (Blocking Partial Join), WCP-34 (Static Partial Join for MI), WCP-37 (Acyclic Synchronizing Merge) are all modeled with the help of the synchronizing stub. WCP-28 and WCP-31 require a new attribute for the synchronizing stub indicating whether the stub is

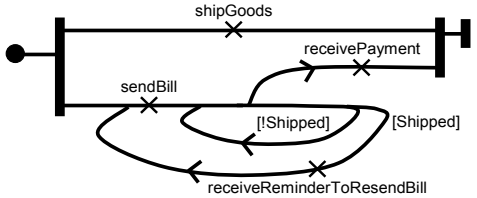

Figure 1. WCP-18 – Parallel branches

blocking. Additionally, the synchronization threshold for the out-path of the synchronizing stub must be set. For WCP-28, the synchronization threshold is set to 1 whereas for WCP-31 it is set to N with 1 < N < number of stub's plug-in maps. For WCP-34, the replication factor is set to M (the number of desired multiple instances, aka MI) and the synchronization threshold is set again to N with 1 < N < M. WCP-37 is supported by the default behavior of the synchronizing stub (the path traversal is allowed to continue when as many plug-in map instances finished as were created). The synchronization threshold is therefore empty. However, the traversal mechanism expects all plug-in maps to arrive at the synchronization point and does not allow the path traversal to continue if one of the plug-ins does not arrive (e.g., because an alternative branch was followed on the plug-in map). Therefore, the score for this pattern is only +/-.

WCP-39 (Critical Section) is supported in the UCM metamodel by the attribute *protected* of a *Component*. The current traversal mechanism, however, does not use this attribute and therefore has to be extended in order to ensure that another path cannot enter a protected component if one path is already being traversed in the component.

The remaining patterns are not supported. WCP-25 (Cancel Region), WCP-26 (Cancel MI Activity), WCP-27 (Complete MI Activity), WCP-29 (Cancelling Discriminator), WCP-32 (Cancelling Partial Join), and WCP-35 (Cancelling Partial Join for MI) are not supported because they deal with cancellation scenarios. Note, however, that WCP-29, WCP-32, and WCP-35 could be supported simply by adding another attribute to synchronizing stubs, indicating whether a stub is a cancelling stub. Since cancellation needs to be considered more generally for UCMs, it was decided to wait with this extension until it is clear how cancellation scenarios will be supported by UCMs in general.

WCP-36 (Dynamic Partial Join for MI) is similar to WCP-15 and therefore not supported. WCP-38 (General Synchronizing Merge) is not supported because the traversal mechanism is not meant to perform an evaluation of possible future states to decide whether the path traversal is allowed to continue.

WCP-41 (Thread Merge) is not supported as the UCM notation currently does not allow synchronization of multiple instances on one execution branch. WCP-42 (Thread Split) is not supported as it is the complement of WCP-41. WCP-41 and WCP-42 could be supported with specialized AND-fork and AND-

Table 1. Summary of assessment

Assessment of all patterns (43) [excl. cancellation (35)]				
Score	UCMs	BPMN	AD	BPEL4WS
+	27 [27]	24 [20]	25 [19]	17 [15]
+/-	2 [2]	9 [6]	5 [5]	4 [3]
-	14 [6]	10 [9]	13 [11]	22 [17]

Table 2. Comparison of UCMs, BPMN, UML 2.0 Activity Diagrams, and BPEL4WS

Workflow Pattern	UCMs	BP MN	UML AD	BPEL 4WS	Workflow Pattern	UCMs	BP MN	UML AD	BPEL 4WS
WCP-01 Sequence	+	+	+	+	WCP-23 Transient Trigger	+**	-	+	-
WCP-02 Parallel Split	+	+	+	+	WCP-24 Persistent Trigger	+**	+	+	+
WCP-03 Synchronization	+	+	+	+	WCP-25 Cancel Region	-	+/-	+	+/-
WCP-04 Exclusive Choice	+	+	+	+	WCP-26 Cancel MI Act.	-	+	+	-
WCP-05 Simple Merge	+	+	+	+	WCP-27 Complete MI Act.	-	-	-	-
WCP-06 Multi-Choice	+*	+	+	+	WCP-28 Bl. Discriminator	+**	+/-	+/-	-
WCP-07 Str. Syn. Merge	+*	+	-	+	WCP-29 Ca. Discriminator	-	+	+	-
WCP-08 Multi-Merge	+	+	+	-	WCP-30 Str. Partial Join	+*	+/-	+/-	-
WCP-09 Str. Discriminator	+*	+/-	+/-	-	WCP-31 Bl. Partial Join	+**	+/-	+/-	-
WCP-10 Arbitrary Cycles	+	+	+	-	WCP-32 Ca. Partial Join	-	+/-	+	-
WCP-11 Implicit Term.	+	+	+	+	WCP-33 Gen. AND-Join	+	+	-	-
WCP-12 MI without Syn.	+*	+	+	+	WCP-34 Static P. J. for MI	+*	+/-	-	-
WCP-13 MI with ADK	+*	+	+	-	WCP-35 Ca. P. Join for MI	-	+/-	-	-
WCP-14 MI with ARK	+*	+	+	-	WCP-36 Dyn. P. J. for MI	-	-	-	-
WCP-15 MI without ARK	-	-	-	-	WCP-37 Acy. Syn. Merge	+/-*	-	+/-	+
WCP-16 Deferred Choice	+*	+	+	+	WCP-38 Gen. Syn. Merge	-	-	-	-
WCP-17 Interleaved P. R.	+	-	-	+/-	WCP-39 Critical Section	+**	-	-	+
WCP-18 Milestone	+/-**	-	-	-	WCP-40 Interleaved R.	+*	+/-	-	+
WCP-19 Cancel Activity	-	+	+	+	WCP-41 Thread Merge	-	+	+	+/-
WCP-20 Cancel Case	-	+	+	+	WCP-42 Thread Split	-	+	+	+/-
WCP-21 Structured Loop	+	+	+	+	WCP-43 Explicit Term.	-	+	+	-
WCP-22 Recursion	+	-	-	-					

+…supported, +/-…partially supported, -…not supported, *…improved by previous assessment, **… improved by updated assessment

join constructs, allowing only one outgoing or incoming branch. Furthermore, a new attribute of AND-forks must be able to specify how many threads should be created (a replication factor) and a new attribute for AND-joins must be able to specify how many threads have to arrive before the traversal can continue (a synchronization threshold). Note that the supported WCP-12 is also a form of thread split but with the assumption that the threads never have to be synchronized.

Finally, WCP-43 (Explicit Termination) is not supported as it is indirectly related to the cancellation patterns and since the UCM notation does not contain special nodes for terminating a scenario. Possible support for this pattern would require (a) the current end point construct to be extended to include an attribute indicating whether explicit termination is desired and (b) a way to cancel active branches of the traversal.

Table 1 summarizes the results of the assessment whereas Table 2 gives a detailed comparison of UCMs with BPMN 1.0, UML 2.0 Activity Diagrams (AD), and BPEL4WS 1.1. The assessments of these three standards are taken from [6].

The main figures of Table 1 show the result for all workflow patterns whereas the figures in brackets show the result for all patterns except cancellation patterns (WCP-19, WCP-20, WCP-25, WCP-26, WCP-27, WCP-29, WCP-32, and WCP-35). The performance of the UCM notation is already competitive when all patterns are taken into account and excellent when cancellation patterns are not considered. The assessment, however, highlights the need to introduce constructs for cancellation and exception handling to the UCM notation. If such constructs are added to the notation so that all cancellation patterns are supported, then almost all patterns can be supported. Only three patterns (WCP-15, WCP-36, and WCP-38) remain unsupported and two partially supported (WCP-18 and WCP-37), as WCP-41, WCP-42, and WCP-43 require only small changes to the UCM notation or traversal mechanism to be supported.

In summary, the traversal mechanism now has to (a) be able to differentiate between persistent and transient triggers, (b) prevent the execution of multiple paths in a protected component, and (c) prevent the creation of a new set of plug-in maps for a blocking synchronizing stub if the current set is still being traversed.

4. Conclusion

This short paper presents a summary of a continued assessment of UCMs based on a recently updated set of workflow patterns. Based on this assessment, UCMs are a competitive business process and workflow description language compared to BPMN 1.0, BPEL4WS 1.1, and UML 2.0 Activity Diagrams. This is particularly interesting because UCMs offer additional benefits compared to these three standards. Foremost, UCMs are integrated with GRL goal models in URN. Secondly, UCMs offer greater flexibility for connecting sub-diagrams and for modeling component containment. UCMs also integrate a simple data model, performance annotations, and a simple action language used for analysis. On the other hand, the assessment also highlights the urgent need to support cancellation patterns in the UCM notation. Future work will have to address cancellation patterns and will have to implement the proposed extensions to UCMs and the traversal mechanism in the jUCMNav tool.

Acknowledgement—This research was supported by NSERC, through its programs of Discovery Grants and Postgraduate Scholarships, and by ORNEC.

5. References

[1] Amyot, D.: "Introduction to the User Requirements Notation: Learning by Example". *Computer Networks*. Vol. 42(3), 21 June 2003, pp 285-301.

[2] Kealey, J. and Amyot, D.: "Enhanced Use Case Map Traversal Semantics". 13th SDL Forum (SDL'07), Paris, France, September 2007, pp. 133-149.

[3] Mussbacher, G.: "Evolving Use Case Maps as a Scenario and Workflow Description Language". *10th Workshop on Requirements Eng.*. Toronto, Canada, May 2007, pp. 56-67.

[4] Pourshahid, A., Chen, P., Amyot, D., Weiss, M., and Forster, A.: "Business Process Monitoring and Alignment: An Approach Based on the User Requirements Notation and Business Intelligence Tools". *10th Wksh. on Requirements Engineering*. Toronto, Canada, May 2007, pp. 149-159.

[5] Roy, J-F.: *Requirements Engineering with URN: Integrating Goals and Scenarios*. M.Sc. thesis, OCICS, University of Ottawa, Canada, 2007; accessed November 2007: http://www.softwareengineering.ca/jucmnav.

[6] Russell, N., ter Hofstede, A.H.M., van der Aalst, W.M.P., and Mulyar, N.: "Workflow Control-Flow Patterns: A Revised View". *BPM Center Report BPM-06-22*. BPMcenter.org, 2006; acc. Nov. 2007: http://workflowpatterns.com/documentation/documents/BPM-06-22.pdf.

[7] *URN - Use Case Map Notation (UCM), ITU-T Draft Recommendation Z.152*. Geneva, Switzerland, Sep. 2003; accessed Nov. 2007: http://www.UseCaseMaps.org/urn.

[8] van der Aalst, W.M.P., ter Hofstede, A.H.M., Kiepuszewski, B., and Barros, A.P.: "Workflow Patterns". *Dist. and Parallel Databases*. Vol 14(3), July 2003, pp 5-51.

[9] Weiss, M. and Amyot, D.: "Business Process Modeling with URN". *International Journal of E-Business Research*. Vol. 1(3), July-September 2005, pp 63-90.

[10] Workflow Patterns website; accessed November 2007: http://www.workflowpatterns.com.