An Overview of a Mapping from Processes to Agents
(Extended Abstract)

Tobias Küster
Technische Universität Berlin, DAI-Labor
Ernst-Reuter-Platz 7, 10587 Berlin, Germany
tobias.kuester@dai-labor.de

Marco Lützenberger
Technische Universität Berlin, DAI-Labor
Ernst-Reuter-Platz 7, 10587 Berlin, Germany
marco.luetzenberger@dai-labor.de

ABSTRACT

Business processes are well suited for modelling agents and their interrelations, but vague semantics and structural differences make a mapping from business processes to multi-agent systems difficult. In this paper, we outline such a mapping that can be applied to different agent frameworks and languages. Using this mapping, we created three implementations suiting different areas of application.

Categories and Subject Descriptors

I.2.11 [**Computing Methodologies**]: Distributed Artificial Intelligence—*Multiagent systems*

General Terms

Algorithms, Design, Languages

Keywords

Agent engineering; Process modelling; Model-driven engineering; BPMN; JIAC

1. INTRODUCTION

Business process modelling has many notions in common with agents-oriented programming, and has been adopted for the modelling of multi-agent systems in a number of approaches (see [1] for an overview of some related work). One common problem with translating processes to programs is the mapping of free-form process graphs to more restricted block-structured programming languages. Also, often the mapping is informal and ambiguous, or it covers just a part of the language, particularly for more expressive (and thus interesting) notations like BPMN [5].

In this paper, we outline a mapping from BPMN processes to multi-agent systems, covering diverse notions common to both, such as actors/roles, reaction rules, events, behaviours, services and message-based communication, and particularly the different process structures. The mapping has been implemented in three different fashions for the JIAC V agent framework [3], being suited for different areas of application.

Appears in: *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015), Bordini, Elkind, Weiss, Yolum (eds.), May 4–8, 2015, Istanbul, Turkey.*

2. MAPPING PROCESSES TO AGENTS

For our mapping, a multi-agent system is assumed to consist of agents implementing different *roles*, defined by *plans*, describing individual behaviours and capabilities, and *rules* and *goals* for when to perform those behaviours. For the plans, we assume common operations such as sending and receiving messages, and invoking other plans, as well as basic structural constructs, like sequential and parallel execution, conditions, and loops. For the processes, we are assuming BPMN [5], including elements such as participants and pools, activities, subprocesses, events, and messages.

The mapping works as follows: For each participant, one agent role is created, with one plan for each process (i.e. each pool) that participant is involved in. Those pools' start events are mapped to rules, triggering the respective plans, e.g., on the receipt of a particular message, or at a certain time. The start and end events also influence the plan's input and output, if any. The content of the plan is determined by the workflow within that particular pool [2].

The mapping of the workflow to equivalent program code is not trivial, as it involves the identification and transformation of structures in the process graph [4], corresponding to, e.g., loops and conditions, but also more complex structures involving event handling. The different process structures covered by this mapping are shown in Figure 1. Here, the nodes in white colour are actual BPMN nodes, such as activities and events, while the shaded regions, labelled z_i, correspond to self-contained regions of the process graph (complex or atomic) that have previously been mapped to agent script elements, which are then incorporated 'bottom-up' into the encompassing structures.

At the lowest level, there are individual activities and events, which are mapped to different script elements depending on their respective type. Most of those are straightforward, e.g., a *send* activity is mapped to sending a message, and a *message* event to receiving one. The *script* task can be used to inject arbitrary code into the agent's plan.

3. IMPLEMENTATION

The mapping has been implemented in three different ways for the JIAC V multi-agent framework: By generating either *JADL++* services [2] or JIAC *agent beans* [1], or by direct interpretation. As JIAC combines agents with ideas from service-oriented architectures [3], the business process metaphor lends itself well to it.

Each implementation has its strengths and weaknesses, and they also differ in their exact coverage of the mapping (see Table 1, including the mapping from BPMN to

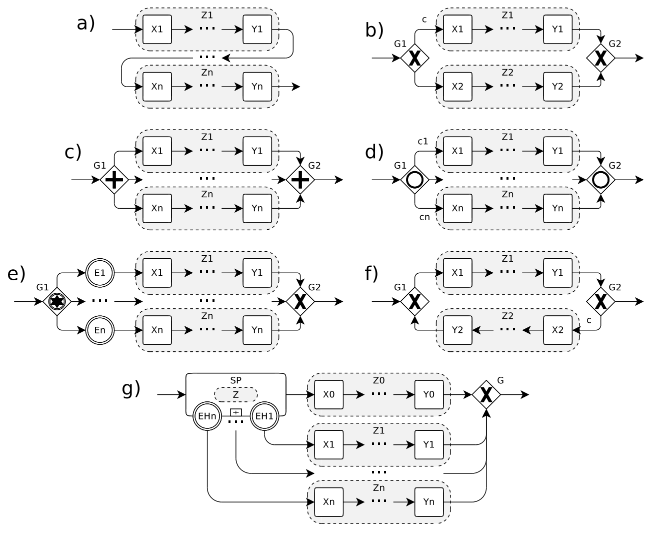

Figure 1: Mapping of Structures. a) Sequence, b) Condition, c) Parallel, d) Parallel-Conditional, e) Event-based Condition, f) While-Loop, g) Sub-process with Event-Handler. Shaded regions correspond to previously matched structures.

BPEL [5] for comparison): i) JADL code is very 'high-level' and compact, and it can be deployed at runtime, but its expressiveness is limited; ii) being pure Java, agent beans can support any process feature and are highly extensible, but the created Java classes are also more complex; iii) with direct interpretation, the processes do not have to follow a block-structure, but without generated code, there is no way to extend or adapt the process prior to execution. At the same time, they are all compatible with each other, e.g., a message sent by a generated agent bean can be received by the interpreter or a JADL service, and vice versa. Thus, it is possible to export one business process diagram to a heterogeneous system, mapping one pool to, e.g., a JADL service and another to an agent bean.

Business process modelling can best be applied either at an early system design stage, to visually model the interaction protocols in the core system [2], or at a later stage, for modelling individual high-level services. Both stages are supported by the mapping and its implementations.

4. CONCLUDING REMARKS

The presented mapping covers most important aspects of processes and agents, such as roles and rules, activities and events, messages and services. It also supports many different process control-flow structures, translating them to equivalent block-structures, like loops and conditions.

The mapping has been implemented in three different ways for the JIAC V multi-agent framework, having individual strengths and weaknesses: Agent beans are fast and versatile, making them the best choice for the core processes of the multi-agent application, while JADL scripts and interpreted processes are more flexible and thus best suited for dynamic and adaptable behaviours.

For future work, we plan to extend the mapping to also cover the creation of agent goals. The BPMN *ad-hoc* sub-process appears to be a good candidate for this. Also, this will require the extension of BPMN with service semantics, both of which are goals of our ongoing research projects.

Table 1: Comparing mappings from BPMN to *X*. -/o/x means no/partial/full support.

	Element	BPEL	JADL	Ag.Beans	Interpr.
Workflow	XOR, AND, OR Gtw.	x	x	x	x
	Event-based XOR Gtw.	x	x	x	x
	Complex Gateway	-	-	-	-
	Event Handler, Error	x	x	x	x
	Event Handler, Other	x	-	x	x
Activities	Send, Receive Task	o	x	x	x
	Service Task	x	x	x	x
	User Task	o	-	o	o
	Manual Task	-	-	-	-
	Script Task	-	x	x	o
	Subprocess	o	o	x	x
	Transaction	-	-	-	-
	Call Activity	o	o	o	o
Events	Message	o	x	x	x
	Timer	x	x	x	x
	Rule	-	o	x	o
	Signal	-	-	-	o
	Escalate	-	-	-	-
	Error	x	-	x	x
	Compensate	x	-	-	-
	Cancel	-	-	-	-
	Terminate	x	-	x	x
Misc.	Properties, Assignments	x	x	x	x
	Lanes, Artefacts	-	-	-	-
	Participants / Roles	-	x	x	x
	Service Starter	o	x	x	x

REFERENCES

[1] T. Küster, A. Heßler, and S. Albayrak. Towards process-oriented modelling and creation of multi-agent systems. In F. Dalpaiz, J. Dix, and B. van Riemsdijk, editors, *LNAI post-proceedings of 2nd Int. Workshop on Engineering Multi-Agent Systems*, volume 8758 of *LNAI*, pages 163–180. Springer, 2014.

[2] T. Küster, M. Lützenberger, A. Heßler, and B. Hirsch. Integrating process modelling into multi-agent system engineering. *Multiagent and Grid Systems*, 8(1):105–124, January 2012.

[3] M. Lützenberger, T. Küster, T. Konnerth, A. Thiele, N. Masuch, A. Heßler, J. Keiser, M. Burkhardt, S. Kaiser, J. Tonn, M. Kaisers, and S. Albayrak. A multi-agent approach to professional software engineering. In M. Cossentino, A. E. F. Seghrouchni, and M. Winikoff, editors, *Engineering Multi-Agent Systems – 1st Int. Workshop, EMAS 2013, Revised Selected Papers*, volume 8245 of *LNAI*, pages 158–177. Springer, St. Paul, MN, USA, May 6–7 2013.

[4] J. Mendling, K. B. Lassen, and U. Zdun. Transformation strategies between blockoriented and graph-oriented process modelling languages, 2005.

[5] OMG. Business process model and notation (BPMN) version 2.0. Specification formal/2011-01-03, Object Management Group, August 2011.