Agent-based development for business processes (Early Innovation)

Hoa Khanh Dam and Aditya Ghose

School of Computer Science and Software Engineering
University of Wollongong
Northfields Av, Wollongong, NSW 2522, Australia
hoa@uow.edu.au, aditya@uow.edu.au

Abstract. Due to the ever-changing business environment, the supporting IT-systems that execute business processes within organisations must be increasingly flexible and adaptable if those organisations are to remain competitive in today's environment. On the other hand, despite offering promising solutions to autonomy, flexibility and adaptability, intelligent agent technology still faces many challenges in being adopted by the industry. Due to their distinct properties, agent-based systems provide a powerful platform for business process execution. Our work focuses in this area with the aim to bridge the gap between business process modelling and agent-oriented development, and consequently contributes to bring benefits to both communities. More specifically, we propose a method for a seamless transition from business process models in Business Process Modelling Notation (BPMN) to agent-oriented models in the Prometheus methodology.

1 Introduction

Business process management (BPM) refers to all activities that support the design, modelling, execution, monitoring and optimisation of business processes. In recent years, the ever-changing business environment demands constant and rapid evolution of an organisation. The flexibility in process execution through IT-systems has significant impact on the success of an organisation's business operations. Existing BPM systems, which require a priori representation of a business process and all potential deviations from that process, however, do not provide adequate support to achieve these requirements in a satisfactory way [1]. On the other hand, despite its popularity and attractiveness as a research area, agent technology still faces many challenges in being adopted by the industry [3]. Multi-agent systems (MAS) provide powerful and flexible execution platform for business processes. Therefore, closing the gap between the business community, BPM in particular, and agent technology can bring substantial benefits to both sides: agents gaining better industry traction whilst BPM having a powerful solution to deal with its current challenges.

In this paper we will propose a mapping between business process models specified in BPMN to concepts and artefacts of the Prometheus agent-oriented

methodology [2]. We have chosen BPMN since it is a standard for business process modelling and has been widely used and supported in numerous modelling tools. Figure 1 shows a BPMN diagram describing a typical process of a conference management system (CMS). This translation is a starting point towards the development of an agent system based on business processes and the use of new, alternative, behaviours discovered from such an agent-based system to enrich the original business processes.

2 BPMN to Prometheus mapping

We now propose how details contained in BPMN models can be directly translated to Prometheus concepts and/or be used to help develop Prometheus artefacts.

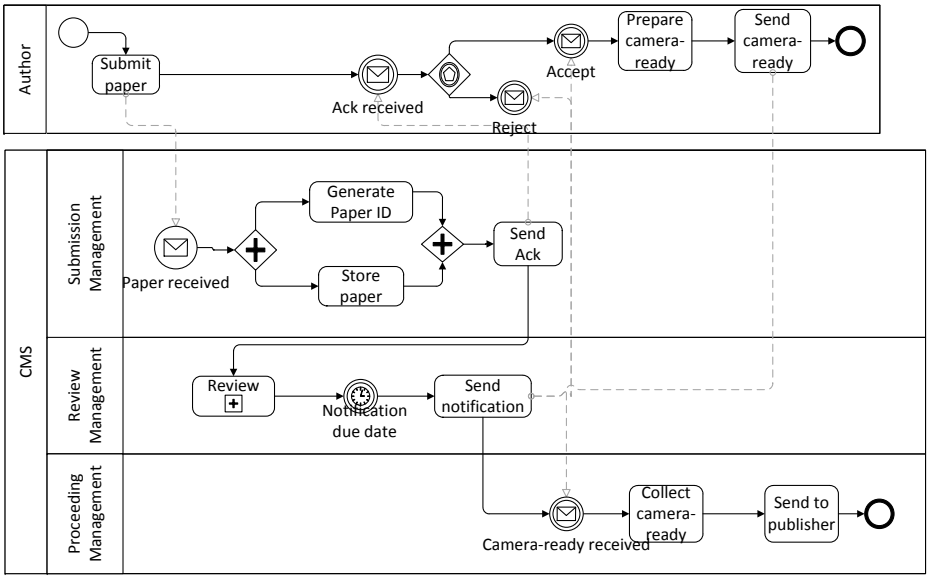

Fig. 1. A process between an author and the conference management system (CMS)

Non-system Pool to Actor: A non-system pool in BPMN represents a business entity or a participant of a process. In this sense, a pool can represent either a human, an organisation or another software system. Non-system pools that interact (e.g. having message exchanged) with the pool representing the system (i.e. the system pool) are candidates for an actor in Prometheus. For example, in Figure 1 the pool "CMS" is a system pool while "Author" is a non-system pool which can be mapped to an actor in Prometheus.

Lanes situated in System Pool to Roles: a lane is situated in a pool and also stands for a process participant. Therefore, a lane represents the two-tier

hierarchy within a process participant. A lane can be mapped to a system role in Prometheus. For example, the three swimlanes in the CMS pool, namely "Submission Management", "Review Management" and "Proceeding Management", can be translated into three equivalent roles.

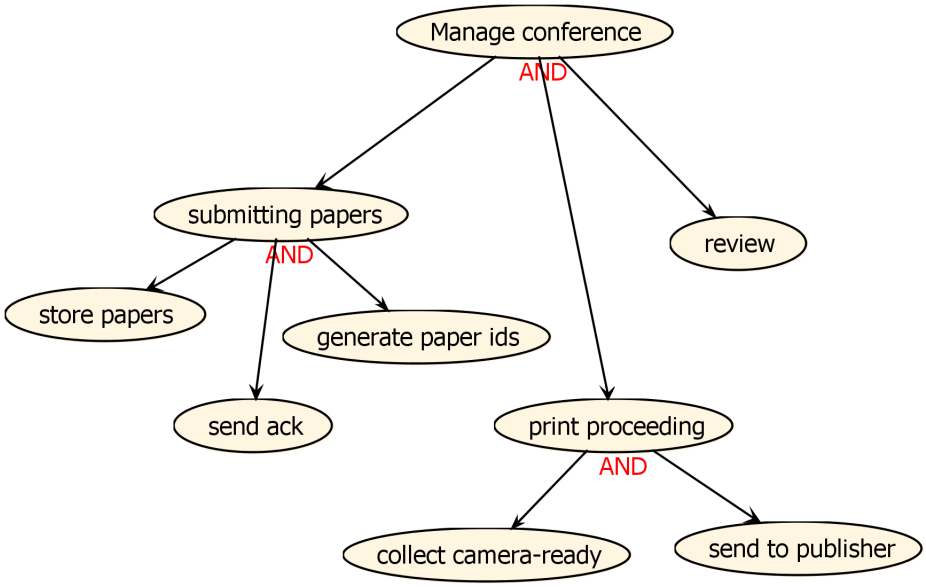

Fig. 2. A goal overview diagram for the CMS

Message Events to Percepts: Message Event which represents the arrival of a message from a participant (i.e. a pool) may trigger the start of the process (Start Event) or cause the process to continue (Intermediate Event). Such message events, if takes place within a system pool, represent information from the environment that the system receives. Hence, those message events can be transformed into percepts in Prometheus. Start Message Events can be translated to percepts that trigger a particular scenario while Intermediate Message Events are mapped to other types of percepts. There are two Message Events in the CMS pool (Figure 1) which can be translated to Prometheus percepts: "paper received" and "camera-ready received"

Message Events to Actions: The sending of a message to a process participant is represented as an Intermediate Message Event in BPMN. If the sender is the system pool and the receiver is a non-system pool, then such a message event represents an output from the system to an actor. Therefore, those events can be transformed to actions. For example, in figure 1 there are two Message Events in the CMS pool which can be translated to Prometheus actions: "ack received", "accept", and "reject".

Processes and Activities to Goals: A complete (sub-)process in a business process model leads to the achievement of a goal. Each activity within a (sub-)process represents a certain things that need to be done to contribute to the achievement of the goal of the process. Therefore, each (sub-)process can be mapped to a goal in Prometheus and each activity within the process can be mapped to a (sub-)goal of the goal corresponding to the process. Figure 2 shows a goal diagram for the CMS. As can be seen, "manage conference" is the top goal which contain a number of sub-goals, including "submitting papers", "print proceeding" and "review". Each of these sub-goals are translated from a corresponding process or sub-process. For instance, the sub-process "review" is mapped to a sub-goal.

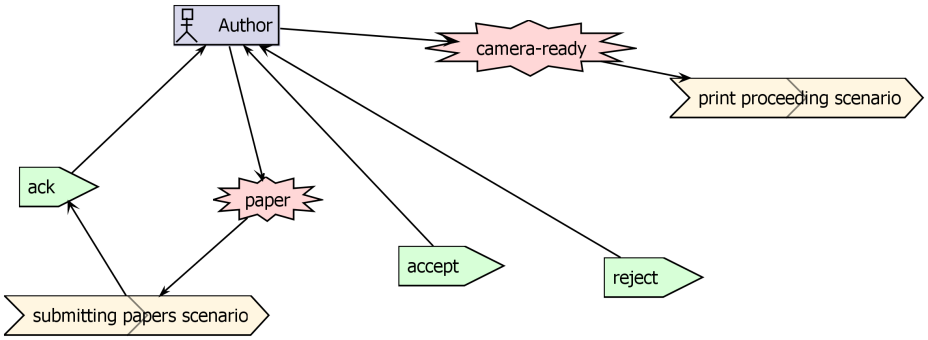

Fig. 3. An analysis overview diagram for the CMS

Processes to Scenarios: Business process models represent and support the dynamic co-ordination of activities by having decision gateways. This allows the model to represent many process instances, each of which contain a unique and supported sequence of activity execution. Each process corresponds to a scenario in Prometheus. An task can be transformed into a goal step whilst a sub-process may be mapped to a sub-scenario. A message event can be mapped into either a percept step or an action step, depending whether it is related to a sending or receiving message. The path resulting from an exclusive (XOR) gateway can be translated to a variation scenario, which represents alternative scenario to a use case. In addition, the transformation from BPMN processes to Prometheus scenarios should also preserve the order of activities and events.

Based on the previous transformation techniques, we can develop an analysis overview diagram in Prometheus as shown in figure 3. The "Author" actor interacts with the system by providing "paper" percepts that contain a paper. The analysis overview diagram also shows which percepts are required by which scenarios, e.g. the "submitting papers" scenario includes the "paper" percept. It is noted that a message flow also implies a link in Prometheus. For example, a message flow from the "submit paper" task in the "Author" pool to the "paper

received" event in the CMS pool indicates a link between the "Author" actor and the "paper" percept.

3 Discussion

In the previous section, we have proposed an approach to translate business requirements in the form of business process models into a system specification using Prometheus notation. We have used a conference management as an example to illustrate our approach. From such a system specification, one can follow the process proposed in the Prometheus methodology to further design and implement an agent-based system that meets the original business requirements. For instance, a similar conference management system has been developed and presented in [5].

Business process models like BPMN tend to be initially designed in a high-level, abstract manner which covers only the normal scenarios. Such business process models do not provide in depth understanding of the processes and their ability to achieve desired goals since they are developed at the analysis level. Further details tend to be added as we move to the later stages of the software development, i.e. design and implementation phases. For instance, at the detailed design level in Prometheus (following a BDI style), we may need to define collection of pre-defined plan recipes (or types) for an agent. Each plan consists of: (a) an invocation condition which defines the event that triggers this plan (i.e. the event that the plan is *relevant* for); (b) a context condition which defines the situation in which the plan is *applicable*, i.e. it is sensible to use the plan in a particular situation; and a plan body containing a sequence of primitive actions and subgoals that are performed for plan execution to be successful. It should be noted that subgoals can trigger further plans.

At run time on a BDI platform [4], the agent achieves responds a particular event by selecting from its plan library a set of plans that are relevant (i.e. match the invocation condition) for handling the event (by looking at the plans' definition). The agent then determines the subset of the relevant plans that is applicable in terms of handling the particular event. The determination of a plan's applicability involves checking whether the plan's context condition holds in the current situation. The agent selects one of the applicable plans and executes it by performing its actions and sub-goals. A plan can be successfully executed, in which case the (sub-)goal is regarded to have been accomplished. Execution of a plan, however, can fail in some situations, e.g. a sub-goal may have no applicable plans, or an action can fail, or a test can be false. In these cases, if the agent is attempting to achieve a goal, a mechanism that handles failure is used. Typically, the agent tries an alternative applicable plan for responding to the triggering event of the failed plan. It is also noted that failures propagate upwards through the event-plan tree: if a plan fails its parent event is re-posted; if this fails then the parent of the event fails and so on.

BDI plans tend to correspond to business processes, e.g. a paper assignment plan corresponding to a process of allocating papers to reviewers. Due to such

a flexibility of BDI plan composition at run-time which discussed above, the plan/process execution at run-time may deviate from the original process specification, i.e. the business process models. More specifically, new behaviours which were not previously specified in the BPMN models can be arisen. Therefore, by capturing those new behaviours we may be able to refine the original business process models. For this reason, agent-based systems can also be used a simulation environment to validate business processes and assist business process redesign. Further investigation on this area is a topic of our future work.

4 Conclusions and future work

On the one hand, the BPM community is facing challenges in modelling and implementing business processes that are able to adapt themselves to a changing environment. On the other hand, although multi-agent systems potentially provide a powerful and flexible platform for process execution, they still fail to attract a wide industry adoption. Therefore, it is very important to bridge the gap between the BPM community and agent technology. This paper has aimed to contribute to such an effort.

We have argued that providing mapping that automatically transforms business languages directly to an agent platform is not feasible due to the significant gap between the two different models and levels of abstraction. Therefore, we have proposed to translate business languages in a form of BPMN models to artefacts of Prometheus, a prominent agent-oriented methodology. Such artefacts are then used to implement an agent system that realizes those business requirements.

There is a number of directions for future work. Firstly, we plan to develop a more complete mapping from BPMN and Prometheus that covers other concepts in the two modelling languages. More specifically, we would like to explore how other types of events (e.g. timer, error, cancel) and decision gateways can be translated to the internal of a plan in Prometheus (e.g. triggering events, context conditions). Secondly, we plan to develop a tool which can be integrated to the Eclipse-based version of the Prometheus Design Tool to support the automation of a mapping between BPMN and Prometheus models. Another major topic for our future work involves further investigation of how new behaviours discovered from the execution of the agent-based system help improve the original business process models that it implements.

References

1. B. Burmeister, M. Arnold, F. Copaciu, and G. Rimassa. BDI-Agents for agile goal-oriented business processes. In Padgham, Parkes, Müller, and Parsons, editors, *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, pages 37–44, Estoril, Portugal, May 2008.
2. L. Padgham and M. Winikoff. *Developing intelligent agent systems: A practical guide*. John Wiley & Sons, Chichester, 2004. ISBN 0-470-86120-7.

3. D. Weyns, H. V. D. Parunak, and O. Shehory, editors. *International Journal of Agent-Oriented Software Engineering (IJAOSE) - Special Issue on the Future of Software Engineering and Multi-Agent Systems*, volume 3, Inderscience Publishers, Geneva, SWITZERLAND, 2009.
4. A. S. Rao and M. P. Georgeff. BDI-agents: from theory to practice. In *Proceedings of the First Intl. Conference on Multiagent Systems*, San Francisco, 1995.
5. L. Padgham, J. Thangarajah, and M. Winikoff. The prometheus design tool: a conference management system case study. In *AOSE'07: Proceedings of the 8th international conference on Agent-oriented software engineering VIII*, pages 197–211, Berlin, Heidelberg, 2008. Springer-Verlag.