SeGA: A Mediator for Artifact-Centric Business Processes[⋆]

Yutian Sun[1], Wei Xu[2,⋆⋆], Jianwen Su[1], and Jian Yang[3]

[1] Department of Computer Science, UC Santa Barbara, USA
[2] School of Computer Science, Fudan University, China
[3] Department of Computing, Maquaire University, Australia

Abstract. Business processes (BPs) can be designed using a variety of modeling languages and executed in different systems. In most BPM applications, the semantics of BPs needed for runtime management is often *scattered* across BP models, execution engines, and auxiliary stores of workflow systems. The inability to capture such semantics in BP models is the root cause for many BPM challenges. In this paper, an automated tool SeGA for wrapping BPs is developed. We demonstrate that SeGA provides a simple yet general framework for runtime querying and monitoring BP executions cross different BP management systems.

1 Introduction

In today's economic market, different Business Processes (BPs) need to engage with each other to achieve competitiveness. Enabling collaboration between different BPs continues to pose a fundamental challenge. A standard BP management system (BPMS) is typically used in internal BP management in a business unit. Such systems are inadequate for business collaboration that involves different independently executing processes, and under different business process models. Therefore, interoperation between BPMSs remains in huge demand and an extremely hard problem.

BP interoperation needs to address two fundamental issues: (1) transformation between different BP model, and (2) runtime BP status and behavior monitoring, The former can smooth the communication, and the latter is critical for execution analysis and management. Web service standards such as WSDL, BPEL, WSCDL provide primitive interoperation support specified in terms of flow of activities, messages to be exchanged, roles and relationships. But they still lack satisfactory support in runtime monitoring, analysis, and process change.

Among emerging data-centric approaches to BPs is artifact-centric BPs initiated in [10]. This approach centers around *business artifacts* (or simply *artifacts*), which are business objects augmented with lifecycle information. Lifecycle information reflects the anticipated status changes of artifacts. However, to support BP mediation and runtime execution monitoring and analysis, we need business data in artifacts as well as process models and execution status information. Unfortunately, the existing artifact

⋆ Supported in part by NSF grant IIS-0812578 and grants from IBM and Bosch.

⋆⋆ Part of work done while visiting UCSB.

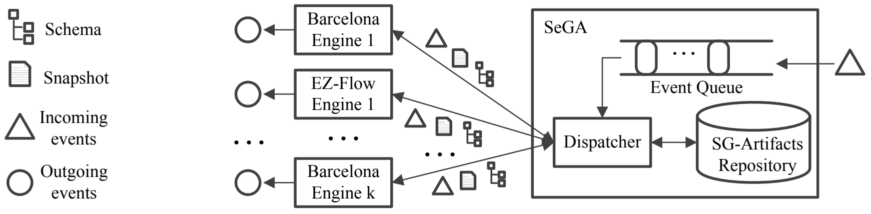

Fig. 1. SeGA Engine

BP modeling formalisms and systems [5,11] are still unable to capture BP semantics in this level of detail, though they provide a good starting point.

In data integration frameworks, wrappers and mediators are frequently used to facilitate interoperation between autonomous data management systems with heterogeneous database schemas. In this short note, we initiate a step towards BP mediation support. We look into two representative artifact BP models: GSM [5] and EZ-Flow [11]. A software tool SeGA is developed to automatically generate BP wrappers for GSM and EZ-Flow BP models. In addition, we demonstrate that SeGA can effectively support runtime management including query, constraints checking, and dynamic modifications.

This paper is organized as follows. §2 reviews GSM and EZ-Flow as well as outlines the SeGA prototype. §3 illustrates support for execution status queries and dynamic model modification. §4 discusses related work.

2 SeGA: Automated Generator for BP Wrappers

This section introduces two existing artifact-centric BP engines. Based on these two engines, a tool "SeGA" is developed to mediate the BPs that run across them.

One artifact-centric BP engine is "Barcelona" [4] that is based on GSM semantics [2]. The communication between the environment and Barcelona is accomplished through events. When an incoming event (sent by a task or a user) arrives, an update to the correlated GSM artifact instance stored in a DB2 database will be performed according to the schema. Some depending GSM artifacts may also change during the same "B-step". Once it is done, the engine will process next event if any arrives.

EZ-Flow [11] is the other artifact-centric BP engine. A step of execution in EZ-Flow moves from one snapshot to another by performing a *transition* on an EZ-Flow artifact instance. A transition is associated with execution of a task. In EZ-Flow, a task (execution) is triggered by an event. During the execution, correlated artifact instances are fetched and modified by the task. When it completes, all the affected artifact instances are stored back to repositories.

Based on the GSM and EZ-Flow artifacts, we develop a tool SeGA (Self-Guided Artifact wrapper) that can serve both as a manager of BP and execution data, and as an automated generator for process wrappers that dispatch execution of individual BPs.

Fig. 1 shows the architecture of a SeGA wrapper (or simply SeGA). Once an engine (GSM or EZ-Flow) is connected to SeGA (through a configure file), all stored artifacts will be automatically transformed to "self-guided artifacts" and stored in a repository. A *self-guided artifact* (or *sg-artifact*) is a GSM/EZ artifact augmented with state and runtime dependency information, and with the artifact schema. When an external event comes, SeGA fetches the relevant sg-artifact from its local repository, separates the schema from the sg-artifact, maps it back to the original form (GSM or EZ-Flow), deposits the artifact schema in the appropriate location where the GSM/EZ-Flow engine can access, and passes the control over to the GSM/EZ-Flow engine via forwarding the event. When the GSM/EZ-Flow engine receives the incoming event, it processes and updates the artifacts according to the schema deposited by SeGA. Once it completes, SeGA is notified and subsequently fetches all updated artifacts, maps them back to sg-artifacts and stores into its own repository. With SeGA the GSM/EZ-Flow engines can focus on execution and have no need for maintaining data, states, dependencies, and even schemas before and after execution.

Based on the design in Fig. 1, a prototype was developed to generate wrappers for GSM and EZ-Flow. The Dispatcher is written in JAVA, implementation of Event Queue and SG-Artifact Repository uses MySQL. A RESTful interface is used for accepting incoming events to SeGA, while SeGA interacts with Barcelona and EZ-Flow through their RESTful interfaces.

3 Runtime Management: Queries, Constraints, and Modifications

In this section, we demonstrate how SeGA can support BP execution querying, constraints checking, and dynamic modifications in a simple manner.

Querying and monitoring
In SeGA, all sg-artifact instances are stored in the form of relations in MySQL system. This can easily facilitate the support for querying (both current and completed) BP execution through the standard query language SQL. However, SQL does not provide constructs/vocabulary for BPs and artifacts. In order to allow BP stakeholders to query and understand BP executions, we develop a query language SGA-Q that incorporates artifact and BP concepts into an OQL-like syntax (Object Query Language [1]).

Checking choreography constraints
In collaborative BPs, choreography constraints are used to restrict how one BP should execute in a collaboration with other BPs to prevent undesirable behaviors by one process. SeGA framework provides a uniformed approach for specifying and maintaining constraints at runtime.

SeGA uses a state machine to model constraints for a BP and monitor running instances. Each BP instance is created with its associated state machine. SeGA maintains a *state table* that records sg-artifact IDs and the current states of the corresponding state machines. The constraints are based on ECA (event-condition-action) rules, i.e., when an event is received (or sent) by a running instance, if the corresponding condition is satisfied, a transition of the associated state machine is made, and this change is recorded in the *state table*. At the end of the lifecycle of the running instance, if the state machine reaches a final state, the constraints are satisfied.

Dynamic modification
BP models often change. In the literature and current practice in BPM and workflow systems, the specification of a BP model is shared by *all* running instances of the model.

Thus modification of the model presents many difficult situations, for example the instance migration problem of deciding a running instance should follow the old model or the new model. By associating each instance with its own schema, SeGA restricts the impact of a schema change to only one running instance, and thus avoids the instance migration problem.

4 Related Work

Process views have been used as an abstraction of BPs to support BP collaboration in [7,3]. Their approach supports design time BP coordination, it does not tackle the hard issue of run time management. [6] proposed a centralized artifact hub in coordinating business processes. It mainly deals with access restrictions that can be placed on stakeholders when they can view the artifacts differently.

Various techniques and formal models are proposed for BP verification [8,9]. They are only performed at the model level. Their applications to runtime analysis therefore are very limited.

Providing process flexibility to support foreseen and unforeseen changes is a very active research area. [11] presented a novel and functional mechanism to handle ad hoc and just-in-time changes at runtime.

References

1. Cattell, R., Barry, D.: The Object Data Standard: ODMG 3.0. Morgan Kaufmann (2000)
2. Damaggio, E., Hull, R., Vaculín, R.: On the Equivalence of Incremental and Fixpoint Semantics for Business Artifacts with Guard-Stage-Milestone Lifecycles. In: Rinderle-Ma, S., Toumani, F., Wolf, K. (eds.) BPM 2011. LNCS, vol. 6896, pp. 396–412. Springer, Heidelberg (2011)
3. Eshuis, R., Grefen, P.: Constructing customized process views. Data Knowl. Eng. 64(2), 419–438 (2008)
4. Heath, T., Vaculin, R., Hull, R.: Barcelona: A design and runtime environment for modeling and execution of artifact-centric business processes (demo paper). In: BPM 2011 (2011)
5. Hull, R., Damaggio, E., et al.: Business artifacts with guard-stage-milestone lifecycles: managing artifact interactions with conditions and events. In: Proc. of the 5th ACM International Conference on Distributed Event-Based System, DEBS 2011, pp. 51–62 (2011)
6. Hull, R., Narendra, N.C., Nigam, A.: Facilitating Workflow Interoperation Using Artifact-Centric Hubs. In: Baresi, L., Chi, C.-H., Suzuki, J. (eds.) ICSOC-ServiceWave 2009. LNCS, vol. 5900, pp. 1–18. Springer, Heidelberg (2009)
7. Liu, D.-R., Shen, M.: Business-to-business workflow interoperation based on process-views. Decision Support Systems 38(3), 399–419 (2004)
8. Lohmann, N., Massuthe, P., Stahl, C., Weinberg, D.: Analyzing interacting ws-bpel processes using flexible model generation. Data Knowl. Eng. 64(1), 38–54 (2008)
9. Nakajima, S.: Model-checking behavioral specification of bpel applications. Electr. Notes Theor. Comput. Sci. 151(2), 89–105 (2006)
10. Nigam, A., Caswell, N.S.: Business artifacts: An approach to operational specification. IBM Syst. J. 42, 428–445 (2003)
11. Xu, W., Su, J., Yan, Z., Yang, J., Zhang, L.: An Artifact-Centric Approach to Dynamic Modification of Workflow Execution. In: Meersman, R., Dillon, T., Herrero, P., Kumar, A., Reichert, M., Qing, L., Ooi, B.-C., Damiani, E., Schmidt, D.C., White, J., Hauswirth, M., Hitzler, P., Mohania, M. (eds.) OTM 2011, Part I. LNCS, vol. 7044, pp. 256–273. Springer, Heidelberg (2011)