Method and System for In-Place Modeling of Business Process Extensions as First-Class Entities

Heiko Witteborg, Anis Charfi, Wei Wei, and Ta'id Holmes

SAP Research Darmstadt
Bleichstr. 8, Darmstadt, Germany
firstname.lastname@sap.com

Abstract. The adaptation of business applications and their underlying business processes is key to meeting the specific needs of a certain customer or domain. The core objective of an extender is to modify the (process) artifacts provided by a base software vendor – yet, existing approaches do not support this directly but force the developer to explicitly use complex extensibility constructs and tools. Based on generic extensibility concepts, this demonstration proposal introduces a novel method and system for business process extensions that supports the intuitive in-place modification of processes. The extensions are extracted automatically and stored as first-class entities, ready to be recomposed or reused in other contexts.

Keywords: process extensions, extensibility, business process modeling, model-driven

1 Introduction

Modern business software systems realize the typical business services of organizations. However, in several cases these services and the underlying business processes have to be adapted to the particular needs of the customer organization or those of its domain. This is typically done via extensions, either built by the customer or by a third party such as an independent software vendor (ISV). In fact, there is an ecosystem of ISVs and consulting organizations around leading business software providers, which specialize in configuring base software and building extensions to it. In some cases reusable extensions are made available to customers through an appropriate marketplace. Yet, extending business applications is a complex task for both the base application provider and the extender. There is a need to raise the abstraction level to other layers such as the process and the service layer in order to ease understanding, developing, and managing extensions and systems that result from base software and extensions.

This demonstration proposal describes a novel method and system for defining business process extensions in an intuitive way. We illustrate our approach by a real-life scenario of extending a sales quotation creation process with a customer rating service (Figure 3). The extension is shown as the blue activity which consults an external credit reporting agency upon receiving a customer request. The obtained customer rating can then be used to better judge customer credibility, such as to lower the risk of fraudulent payments and encourage customers of high credibility with discounts and rewards. Without this extended business process as an abstract level, it would be difficult to realize the extension solely at the code level because (1) it requires a thorough understanding of the base software code; (2) it is hard to estimate the impact of the extension on the original business process; and (3) it is hard to assure its compatibility with the base software as well as with other extensions.

2 Approach for In-Place Modeling of Process Extensions

Based on the process extension meta-model that is introduced in [1], this approach allows the intuitive definition of business process extensions as first-class entities. The generic idea of the method is to give the extender artifacts (i.e., in this instantiation, business processes) of the software applications and to provide a system, in which the extender gets the impression of modifying directly the contained elements (referred to as “in-place extension definition”). Once an extension is saved these tools extract the modification (the delta) in a separate extension module and the original artifacts of the software provided are kept unmodified. Figure 1 depicts a development process for (and the realized features of) the in-place definition of extensions. It comprises four phases:

Definition of extensibility views and base artifacts. In this phase the base software provider decides which business processes he wants to expose to extenders.

Target artifact selection. In the second phase, an extender (e.g., an ISV that develops extensions) selects a business process of the software application as base process for his extension. He may reapply existing extensions to create an extended base process.

In-place extension definition. The extender defines the modifications by apparently directly adapting the base process, although this process is owned by the base software provider and is generally provided as a read-only artifact. This is achieved by annotating the original model in the background, and through appropriate tooling that adapts the visualization accordingly.

Extraction and storage of extensions. In the fourth phase, the modifications defined using the in-place extension technique are automatically extracted into first-class extension modules. They can be later reapplied to the respective base process, restoring the extended process, or they can be reused in a new context.

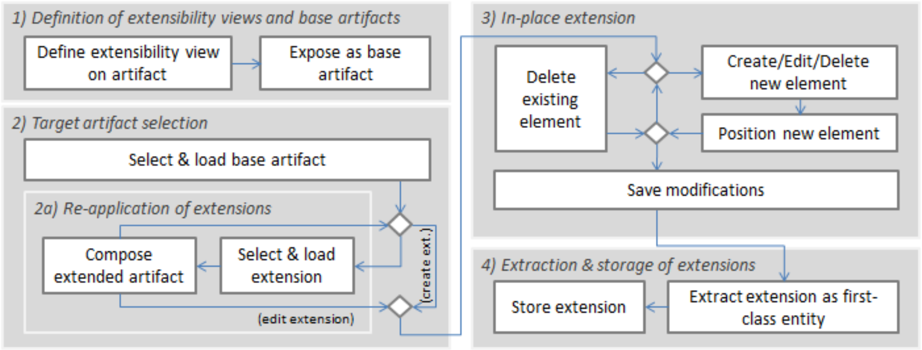
1) Definition of extensibility views and base artifacts
Define extensibility view on artifact
Expose as base artifact
2) Target artifact selection
Select & load base artifact
2a) Re-application of extensions
Compose extended artifact
Select & load extension
(create ext.)
(edit extension)
3) In-place extension
Delete existing element
Create/Edit/Delete new element
Position new element
Save modifications
4) Extraction & storage of extensions
Store extension
Extract extension as first-class entity

Fig. 1: Development Process and Realized Features of In-Place Extensibility

3 System Overview

To support this development process, an instantiation of this approach should provide the repositories and components depicted in Figure 2. With the *Base Artifact Explorer*, an extender can select and load base processes from the *Base Artifact Repository*. He can initiate a new extension for the selected process, or he can retrieve and edit an existing extension from the *Extension Repository* using the *Extension Explorer*. This extension is automatically reapplied to the selected process (*Composer*) by adding and/or annotating the involved elements. In case that the extension of extended base process is supported, a *Conflict Resolution* component may help detecting and resolving potential conflicts.

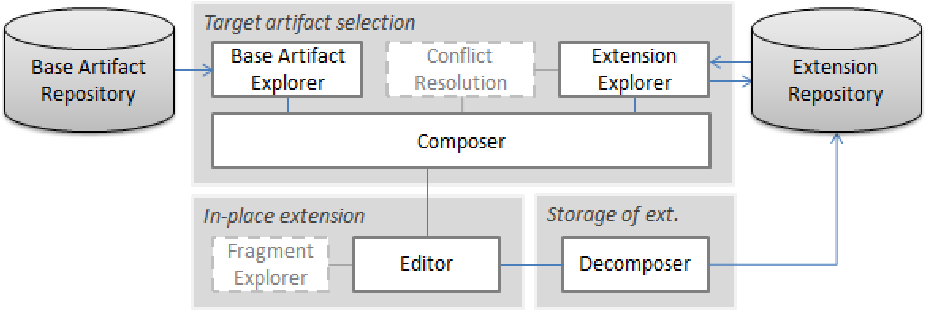

Fig. 2: System Overview of the In-Place Extension Tooling

The *Editor* shows the result of the artifact selection, the target process, and provides functionality to modify this process, similar to other editors of the underlying process language, yet, in accordance with the modification restrictions defined by the base software owner. Optionally, a *Fragment Explorer* component can help to reuse extension parts in different contexts. Finally, the *Decomposer* automatically identifies the modifications, updates the extension accordingly and stores it in the *Extension Repository*.

4 Tooling and Example

A screenshot of the editor implementation is shown in Figure 3. The functionality of the target artifact selection components is bundled in a wizard that can be triggered using the *Process Extension Configuration* button. In this wizard, an extender can (a) choose the base business process from a list of processes retrieved from the model repository of an SAP on-demand solution and (b) select existing or initialize new extensions and apply them to the currently selected target business process. When the wizard is finished the editor is triggered to show the target business process.

The business process editor allows the extender to model modifications, similar to editing the process in a standard editor. Nonetheless, the editor ensures that the deleted or newly added elements are annotated accordingly, in the background. Based on these annotations, the new elements are slightly highlighted or shown as grayed out. In the

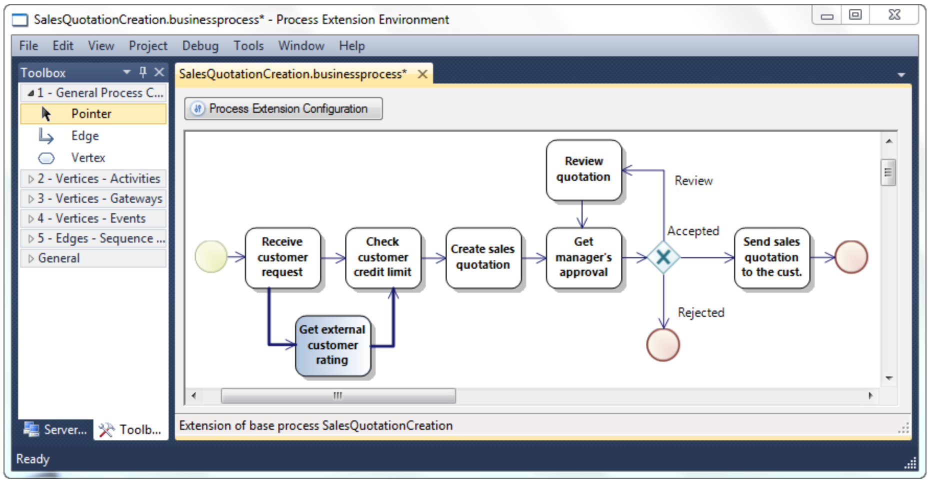
SalesQuotationCreation.businessprocess* - Process Extension Environment
File Edit View Project Debug Tools Window Help
Toolbox
1 - General Process C...
Pointer
Edge
Vertex
2 - Vertices - Activities
3 - Vertices - Gateways
4 - Vertices - Events
5 - Edges - Sequence ...
General
Server...
Toolb...
SalesQuotationCreation.businessprocess*
Process Extension Configuration
Review quotation
Review
Accepted
Receive customer request
Check customer credit limit
Create sales quotation
Get manager's approval
Send sales quotation to the cust.
Rejected
Get external customer rating
Extension of base process SalesQuotationCreation
Ready

Fig. 3: Business Process Extension Editor

example scenario, the extender can add and weave in a new *Get external customer rating* task into the *Sales Quotation Creation* process. The extender can save his work using the established ways, e.g., via menu entry or keyboard shortcut. However, the editor overrides the standard behavior of saving the complete artifact. Instead, it calls the *Business Process Decomposer* component to extract and store the extension.

5 Conclusion

Extensibility is a key requirement of a modern business application and its underlying business processes, to be able to adapt it to the specific use case and to increase the attractiveness of the application, e.g., by establishing an ecosystem of extenders. However, extensions are typically only supported on code layer, or extenders have to cope with complex extensibility concepts. Based on a generic approach, this demonstration proposal presented a method and system for business process extensibility that supports the intuitive in-place definition of extensions, the extraction of the modifications into first-class entities, and the composition of extended processes.

Acknowledgments The authors would like to thank Mohamed Aly for his contributions to the extensibility concept. The work presented was performed in the context of the Software-Cluster project Emergent. It was partially funded by the German Federal Ministry of Education and Research under grant no. `01IC10S01`.

References

1. Witteborg, H., Charfi, A., Aly, M., Holmes, T.: Business process extensions as first-class entities — a model-driven and aspect-oriented approach. In: ICSOC. Lecture Notes in Computer Science (in press), Springer (2012)