Unicorn meets Chimera: Integrating External Events into Case Management

Jonas Beyer, Patrick Kuhn, Marcin Hewelt, Sankalita Mandal, Mathias Weske

Hasso Plattner Institute, University of Potsdam, Germany
{Marcin.Hewelt,Sankalita.Mandal,Mathias.Weske}@hpi.de

Abstract. Case management allows knowledge workers to model and enact flexible, knowledge-intensive business processes. Such processes occur in many domains, e.g. logistics or healthcare, and the exact course of a case can not be pre-specified, because it heavily depends on case data, user decisions, and external events, which take place during runtime. This work extends our case management engine Chimera with the capability to incorporate external events. To this end we integrate Chimera with the event processing platform Unicorn, with the result that external events can now trigger new cases, provide case data, or abort activities. This demo is aimed at practitioners and academics in the field of flexible business processes and case management.

Keywords: Case Management, Business Process Management, Complex Event Processing, Case Execution, flexible Business Processes.

1 Overview

Case Management is an approach suitable for the modeling and execution of knowledge-intensive business processes that center around a case. When executing a case, competent knowledge workers try to achieve the specific case goal, by aligning their activities with the emergent requirements of the case. However, the sequence of activities executed towards the goal, depends on the specific circumstances of the case, which only become apparent during case execution, and hence can not be pre-specified.

When dealing with highly flexible processes, integrating external events is necessary to enable case workers to react efficiently and adapt their work to the current situation. Complex event processing is an already proven mean to provide high level events relevant to the course of the process [1], eventually cases. While process modeling languages like BPMN 2 [5] allow to model several kinds of events, process engines like Camunda or Activiti[1] are limited to their engine-internal events.

[1] see http://camunda.de resp. http://activiti.org

This work builds on the case management engine Chimera[2] that was presented at last year's BPM demo under the name JEngine [2]. Here we extend Chimera with capabilities to deal with external events by registering event queries with the event processing platform Unicorn[3], and reacting on received event notifications. Furthermore, we present the case modeling tool Gryphon. It allows knowledge workers to model event types as part of the domain model of a case model, as well as process fragments that are annotated with event queries.

2 Fragment-based Case Management

In our project, we consider the fragment-based case management approach (fCM) by [3, 4], where a business scenario is represented by a *case model* consisting of a set of fragments, a domain model, and a set of life cycles. Each fragment is a process model that contains control flow necessary to describe how to handle a subsection of the case. Upon execution, the fragment instances are dynamically combined based on their data dependencies. The data classes and their associations, as well as their domain specific attributes are defined in the domain model. Each data class has a corresponding object life cycle that specifies possible state transitions a data object of that class can undergo during case execution. The Chimera approach uses BPMN for the fragments, UML for the domain model, and state transition graphs for the life cycles. The semantics of fragment-based case management has been formally defined in [3].

3 Event Integration into fCM

We explain the concept of integrating events with the usecase of asparagus harvest. External factors like temperature and weather prediction influence the harvesting process. One important aspect of the usecase is depicted in Fig. 1 that shows one of several process fragments contained in the case model of our usecase. In order to balance supply and demand for asparagus the harvest date needs to be coordinated with other asparagus farmers in the region.

The fragment in Fig. 1 is enabled once the `Harvest Plan` has been `[created]`. The farmer then starts preparing resources and equipment for the harvest. But as depicted by the event-based-gateway, if the `Market price dropped` event occurs before the harvest date, the `Market Situation` data object is updated and the harvest has to be postponed.

In the remaining part of the section, the work flow for implementing the use case in our execution environment is discussed. Figure 2 provides an overview of all components and their interactions.

[2] see `https://bpt.hpi.uni-potsdam.de/Public/ChimeraDoc`
[3] see `http://bpt.hpi.uni-potsdam.de/UNICORN`

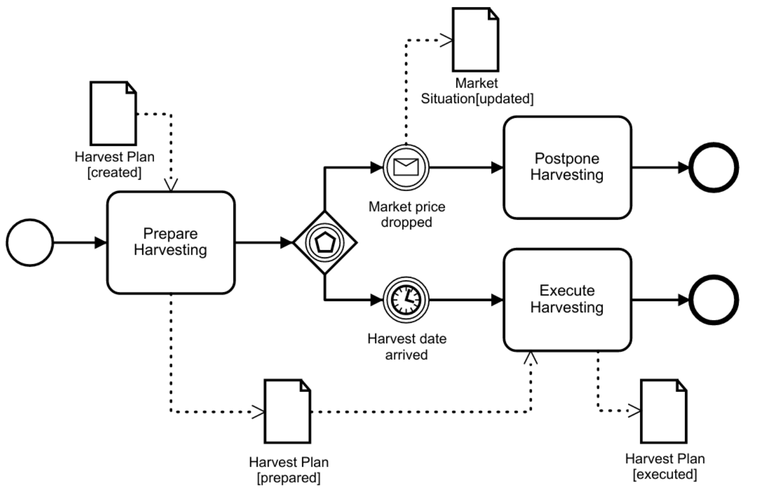


Fig. 1: A process fragment from the case model

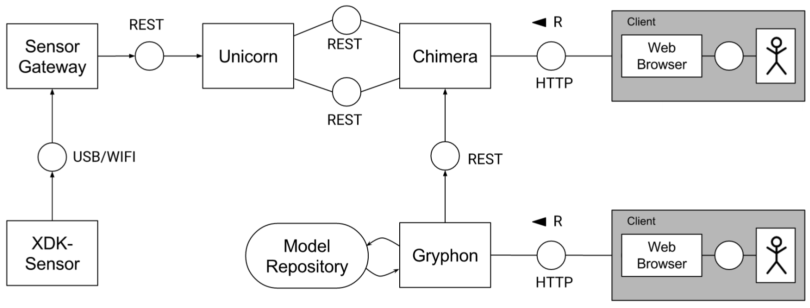
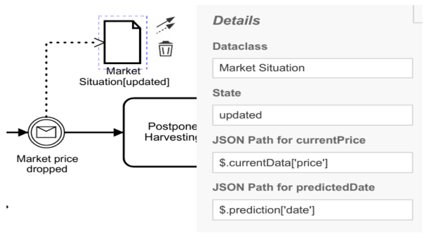


Fig. 2: Architecture

Case model creation. In order to include external events in the execution of a case, they first have to be modelled. It is possible to define event types that describe the domain and the attributes of events. Specific events triggering actions in the case can be modelled using event queries. Event queries are similar to database queries, the difference being that they operate on event streams instead of persistent data. If an event from the stream matches the query, the event is then sent to the execution engine. We chose the Event Processing Language (EPL) provided by Esper[4], as language to express those event queries. We decided to reuse the catching message event to model enabled events as notifications can be considered as receiving messages.

To model the case, we used Gryphon, a web-based tool built around bpmn.io[5] based on a node.js[6]-stack. Gryphon allows to model process fragments, domain model, life cycles, and termination conditions. The completed case model can then be deployed to a running Chimera instance.

[4] see `http://www.espertech.com/products/esper.php`
[5] An open-source BPMN modeler implemented in Javascript `http://bpmn.io`
[6] see `http://node.js`



Details

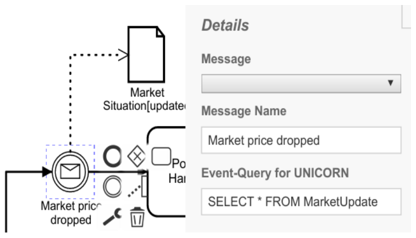
Dataclass
Market Situation

State
updated

JSON Path for currentPrice
$.currentData['price']

JSON Path for predictedDate
$.prediction['date']

(a) Modeling of data attribute bindings



Details

Message

Message Name
Market price dropped

Event-Query for UNICORN
SELECT * FROM MarketUpdate

(b) Modeling of event queries

Fig. 3: Modeling extensions in Gryphon

Model deployment. Chimera is an engine for executing case models, consisting of a web-based frontend and a backend communicating via a RESTful API. Chimera parses the received case model and registers event types and case start queries with Unicorn. The case can then be executed through the web UI, where the user is presented with an overview of all case models.

Chimera also supports incorporating events as case data. Events are received in the form of JSON objects, where each attribute field has a specific value. Thus, we can parse the JSON and set data object attributes according to the event attribute values. This is implemented by evaluating JsonPath[7] expressions. The user can specify one JsonPath expression per attribute of the data object that is used to persist the values, as seen in figure 3a.

Event and query registration. Whenever an event is reached during execution in Chimera, an event query is registered with Unicorn. Unicorn is an event processing platform built around the Esper Event Processing Engine that allows to manage event types, event queries, and notifications both via a web-based UI and a REST API. Events are sent to Unicorn either via a REST API or by means of adapters, that periodically call webservices.

A general overview of the event registration process is shown in Fig. 4. The registered event queries can be divided into two groups. Case start queries have to be registered when a new case model is deployed to Chimera and remain registered until the case is deleted. All other event queries are registered as soon as the respective event control node is reached. The annotation to register event queries has been shown in Fig. 3b. Each query is registered with a specific id, which is used to correlate the event control node to the event query after it was triggered. The queries are unregistered from Unicorn when the event is triggered or skipped.

Event generation. In our use case, events were produced by a sensor unit. Because we did not have access to real “production fields”, the sensor unit used was the Bosch XDK developer kit[8], a package with multiple integrated sensors for prototyping of IoT applications. The unit sends measurement values over

[7] see `http://goessner.net/articles/JsonPath/`
[8] see `http://xdk.bosch-connectivity.com`

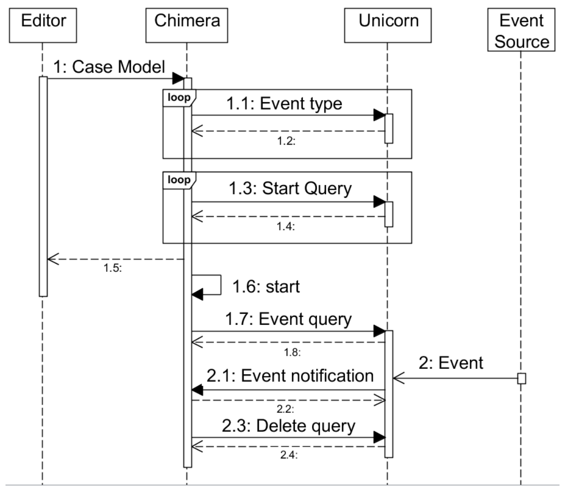


Fig. 4: Event integration sequence

wireless network to a gateway that parses the proprietary format of the received data and forwards it to Unicorn using the REST API.

Reactions to events. If the event is linked to a case start query, the case is initiated. The specified data objects are created using the information given by the start query — this includes the initial data object state and its attribute values. Otherwise, the event control node associated to the query id is retrieved and executed. If the control node has outgoing data objects that define data bindings, the event data is evaluated with the help of the JsonPath expressions, and the attribute values are saved.

4 Conclusion

In this paper, we presented our prototypical adaption of a case management execution engine to handle real life events sent by a sensor. To this end, we adapted the modeling component Gryphon to allow event modeling and integrated the event processing platform Unicorn and a Sensor Gateway into the architecture. The case engine itself had to be adapted to parse and register event types and event queries, as well as to react to received event notifications.

This contribution is part of an ongoing project to develop a highly usable environment for knowledge workers to model and enact fragment-based case models. Source code, documentation, and screencast of the Chimera case engine are available at `https://bpt-lab.org/fcm`.

References

1. O. Etzion and P. Niblett. *Event Processing in Action.* Manning Publications, 2010.

2. S. Haarmann, N. Podlesny, M. Hewelt, A. Meyer, and M. Weske. Production case management: A prototypical process engine to execute flexible business processes. In *Proceedings of the BPM Demo Session*, pages 110–114, 2015.
3. M. Hewelt and M. Weske. A Hybrid Approach for Flexible Case Modeling and Execution. In *BPM Forum*, LNBIP. Springer, 2016. (accepted for publication).
4. A. Meyer, N. Herzberg, F. Puhlmann, and M. Weske. Implementation framework for production case management: Modeling and execution. In *Enterprise Distributed Object Computing (EDOC)*. IEEE, 2014.
5. Object Management Group. Business Process Model and Notation (BPMN), Version 2.0.2, 2013.