THE CONVERGENCE OF WORKFLOWS, BUSINESS RULES AND COMPLEX EVENTS

Defining a Reference Architecture and Approaching Realization Challenges

Markus Döhring, Lars Karg, Eicke Godehardt, Birgit Zimmermann
SAP Research, Bleichstraße 8, Darmstadt, Germany
markus.doehring@sap.com, lars.karg@sap.com, eicke.godehardt@sap.com, birgit.zimmermann@sap.com

Keywords: workflow management, business rules, complex event processing, business process management

Abstract: For years, research has been devoted to the introduction of flexibility to enterprise information systems. There are corresponding concepts for mainly three established paradigms: workflow management, business rule management and complex event processing. It has however been indicated that the integration of the three paradigms with respect to their meta-models and execution principles yields significant potential for more efficient and flexible enterprise applications and that there is still a lack in conceptual and technical guidance for their integration. The contribution of this work is a loosely coupled architecture integrating all three paradigms. This includes a clear definition of its building blocks together with the main realization challenges. In this context, an approach for assisting modelers in solving the question which paradigm should be used in which way for expressing a particular business aspect is presented.

1 INTRODUCTION

The proper integration of workflow management (WfM), business rule management (BRM) and complex event processing (CEP) paradigms is a major concern for future enterprise applications. Those are expected to improve the overall organizational flexibility and performance as well as to provide self-adaptivity capabilities in terms of tailoring themselves to the business needs of an enterprise over time (Gualtieri and Rymer, 2009). One aspect in this regard is the increased semantic expressiveness of the meta-models e.g., by combining workflows and business rules (zur Muehlen et al., 2008). However, since each field has evolved separately and usually targets a definite class of business problems, there is also a particular overlap in modeling constructs by means of the three paradigms. This may lead to confusion regarding the question *what to model where—and how*. Generally, the provision of a properly defined integrated architecture for WfM, BRM and CEP together with specific modeling guidelines is still a long way off (Brett and Gualtieri, 2009). The highlighted gaps are addressed by this work as follows: In Section 2, related work on the combination of WfM, BRM and CEP is discussed. Based on the identified lack of holistic integration concepts, an architecture for the proper interplay of the three paradigms is presented and its building blocks are formalized in section 3. The section also discusses the main architectural realization challenges in terms of meta-model mapping and extension as well as the provision of model recommendations. A basic example illustrates ideas for their solution in section 4. Section 5 concludes this work.

2 RELATED WORK AND MOTIVATION

The work of (Knolmayer et al., 2000) describes the use of reaction rules as an integration layer between different workflow modeling languages. They show how typical workflow con-

structs, for example XOR branching, can be realized by Event-Condition-Action (ECA) rules. The approach is refined in (van Eijndhoven et al., 2008), where a methodology for manually identifying variability points in a workflow (parts which are likely to change in the future) and applying a pattern-based transformation to ECA rules is proposed. The combination of rule and workflow meta-models is targeted by (Milanovic and Gasevic, 2009), who also provide a graphical integration of industry standard modeling languages for the two paradigms. (Decker et al., 2007) present a graphical notation for modeling complex events in workflows. The notation consists in a temporal extension for BPMN[1], e.g., introducing precedence relationship connectors for event nodes. (Paschke and Kozlenkov, 2008) implement whole workflows based on rules, especially leveraging eventing paradigms. Their efforts are of more technical nature on the implementation layer and do not target the special requirements of a business analyst regarding the overall understandability and efficiency of business logic. (Bry et al., 2006) provide a case study realizing a fictive car rental process completely with ECA rules, stating that this bears potential drawbacks for instance regarding the capabilities of monitoring a business process.

In summary, much related work addresses meta-model integration issues. Yet confusion is caused by an improper mixing of the paradigms best suited to express a particular business problem. This multiplies with an insufficient conceptual separation of modeling paradigms and their implementation within an enterprise system. All approaches lack a convenient methodological support for finding the joint balance for employing elements from WfM, BRM and CEP. For this reason, (zur Muehlen et al., 2008) present a framework comprising decision criteria for choosing rule or workflow concepts, however neglecting eventing aspects. Those criteria for instance include how frequently the business aspect changes in its definition. Even though the criteria are principally measurable to a restricted extent, no formal procedure is provided on how to determine thresholds and to offer specific model recommendations. Consequently, an important lack in research is the provision of the conceptual facility for using the paradigms together *and* appropriate guidance to deal with semantic overlaps of the meta-models.

[1]See http://www.omg.org/cgi-bin/doc?dtc/09-08-14 for Business Process Modeling Notation V.2

3 JOINT ARCHITECTURE FOR WFM, BRM AND CEP

Imagine a business analyst who wants to represent a customer solvency rating, whose outcome depends on many interdependent criteria, in an IT system. Would he models those checks as a sequence of workflow steps? Or declaratively as business rules? Or would he rely on events like *customer has outstanding payments > 5000$ for more than 2 months*? Could he even use a combination of all of the paradigms? This section lays the foundation for such a potential combination in terms of architectural building blocks and corresponding realization challenges, followed by a formalization of the architecture components and integration points.

3.1 Building Blocks and Integration

To the best of our knowledge, there is no clear and holistic definition of a loosely coupled architecture supporting all three paradigms. *Loosely coupled* means that it should be possible e.g., to only employ a workflow management system and step-by-step extend it with a rule engine and finally a CEP engine if needed. Consequently the architecture must specify the paradigm integration points for industry standard meta-models like BPMN2 and RIF[2] in a minimal-invasive way, which means there should be preferably no changes to the original meta-models. For an event pattern language (EPL), there is no standard available yet. Therefore the consideration of proprietary languages like Esper EPL[3] will be necessary.

A reference architecture that suffices these requirements is presented in figure 1. Its formal definition is provided in section 3.3. The (meta-)model layers are separated from the implementation layer within the architecture to emphasize their partial independence. This is because in practice, each paradigm is realized with a favored set of implementation variants, as for example petrinet-based workflows, RETE pattern matching for rule conditions or event-driven backward chaining for CEP as in (Anicic et al., 2009). If however the power of the distinct meta-models is correspondingly constrained, all paradigms could be implemented within one monolithic software component. Neglecting im-

[2]See http://www.w3.org/TR/rif-core

[3]See http://esper.codehaus.org

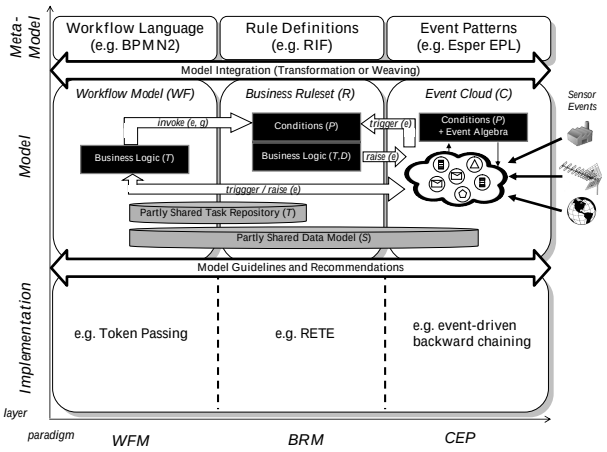


Figure 1: WfM-BRM-CEP Integration Architecture

plementation aspects for now, we describe the features and interplay of the paradigms in figure 1 from right to left.

CEP captures changes in the state of systems (eventually from the real-world via sensors) as events and provides a temporal algebra as well as data-mining and pattern matching algorithms to detect events of higher semantic meaning (Luckham, 2002). These abilities in addition to facilities for handling and filtering masses of incoming events from different sources are the unique features of CEP. For our reference architecture, all detected events which are business relevant according to some conditions are kept within an *event cloud*. This term is coined by (Luckham, 2002) and expresses that events in a business context usually do not stem from one organized stream, but from several sources causing a lot of data noise. One important characteristic of the CEP paradigm is, that it is not responsible for conducting business logic. It is rather a mere *event cruncher*. Events from the cloud can be used as signals to trigger business rules or for catching event nodes within workflows.

BRM aims at enabling the business domain expert to express statements that define or constrain aspects of the business (Hay and Healy, 2000). Being specified in a modular and declarative way as relatively simple and small ECA rules, the realization of complex decision and action logic can be achieved by the collectivity of a ruleset. In summary, the strength of business rules is their easy comprehensibility and direct separate changeability. Within our reference architecture, business rules have at most one event attached as a direct trigger. If constraints on multiple events or temporal constraints need to be specified, this should be achieved in the CEP environment yielding a new single complex event. This has the advantage of disburdening the business analyst (i.e. the rule expert in this case) as well as the BRM engine from having to deal with complex event algebra. The action part of an ECA rule may contain manipulations to the local data context or task executions, which in turn can lead to the detection of new complex events. Business rules in our architecture are of twofold relevance: They can be explicitly invoked by a WfMS as a decision service or they can be triggered by events from the event cloud providing reacting resp. exception handling mechanisms.

WfM aims at the IT-based specification and coordination of *tasks* as units of work within an enterprise and their allowed execution order to jointly realize a business goal (Leymann and Roller, 1999). In practice, a workflow model is implemented by IT experts in cooperation with domain experts and remains relatively stable over a longer period of time. However, in some niche areas, also adaptive WfMS as presented by (Dadam and Reichert, 2009) have evolved. Such WfMS allow for more flexibility in terms of control and data flow changes at design- and runtime. The strength of workflows lies in the overall comprehensibility and traceability of business logic. A workflow as part of our architecture may receive and communicate events with the event cloud. To be able to integrate business rule conditions with workflow task execution, we assume that at least the finishing of a task instance constitutes a business relevant event comprised by the event cloud. As such, it can be reused amongst others for triggering business rules. The branching behavior of gateways can be realized by invoking business rules.

For an efficient integration of the three paradigms, there must be a partly shared data model for all paradigms, e.g., a customer ID must be uniquely defined. Furthermore, there must be a shared task repository for use in rules and workflows, i.e., a predefined task can be started within a WfMS as well as the action part of a rule. The white arrows in figure 1 symbolize the architectural integration points. The two horizontal continuous arrows concern the main scope of this work. The one on the meta-model layer represents the ability to model distributed business logic combining all three paradigms (Section 3.2.1). The one on the model layer relates to the provision of decision and optimization support (Section 3.2.2).

3.2 Realization Challenges

3.2.1 Challenge #1 - Integration of the Meta-Models

Each distinct paradigm, WfM, BRM and CEP has evolved from specific business domains and existing meta-models have been designed accordingly. It is desirable to provide a loose coupling and therefore to allow for a stepwise extension of the system landscape. Consequently the meta-model integration should be achieved preferably leaving their original definition (almost) intact, using either weaving or mapping concepts:

Meta-Model Weaving: One option to realize the integration of meta-models without changing or reducing their original semantics is to define additional *bridge* meta-elements. The resulting overall meta-model is a superset of the original distinct meta-models regarding its semantic expressiveness. For instance, a new integrated meta-element could be defined using multiple inheritance from different meta-model packages and subsequently be enriched with additional properties. An example would be a rule-gateway with joint semantics from BPMN and a rule language.

Meta-Model Mapping: Assuming that not all parts of the meta-models can be reasonably combined by model extension, it is required to be able to map particular constructs from one paradigm to those of another paradigm. For instance, if a company has employed only a rule-based information system in the past, it should be possible to smoothly migrate some of the rules that in fact realize process logic to an explicit workflow representation. The resulting overall meta-model is equal to the original distinct meta-models regarding its semantic expressiveness. An example would be the expression of workflow-patterns via ECA rules.

3.2.2 Challenge #2 - Modeling Guidance

As model mappings will presumably play a chief part within our meta-model integration efforts, we face the problem of a semantic redundancy of some parts of the overall meta-model. That means, when employing more than one paradigm at once within an enterprise information system, it might not be a-priori clear which construct from what paradigm should be used to model a particular business problem. Therefore, we propose to continue and more substantiate work like (zur Muehlen et al., 2008) for providing specific modeling guidelines in terms of grounding them on solid comprehensible metrics. These can be subdivided into design-time metrics relying on static model aspects and log-based execution analysis metrics.

3.3 Formal Definitions

We formalize the architecture components mainly for a definition of the paradigm intersections, e.g., where modeling elements can be commonly reused. The presented notation is also employed in section 4 for sketching the solution approach.

Definition 1 (Event Cloud) *Let C be an event cloud consisting of events $\underrightarrow{e}$. An event e can be represented as a tuple $e = (s, r, \vec{K}, P)$, where:*

- *s is a stamp relating to a fixed point in time when the event occurred or has been detected*
- *r represents a boolean variable whether this event is business relevant in terms of affecting the business logic or not*
- *$\vec{K}$ as a potentially empty causality vector contains all events that caused the detection of e*
- *$P \subseteq M$, $M := A \times V$ is a set which contains attribute-value mappings at time s, which are additional prerequisites constraining the business data context the event may occur in.*

For the definition of a *workflow model* we adopt formalisms from (Yongchareon et al., 2010), especially considering the BPMN2 intermediate event construct.

Definition 2 (Workflow Model) *A workflow model is a set of connected tasks, events and gateways in terms of an extended directed graph. It can be represented as a tuple $WF = (O, T, E, G, F, T^E, X)$, where :*

- *O represents a set of workflow nodes, where:*
  - *$T \subset O$ is a set of tasks. For each $t \in T, \exists e \in E \,|\, r_e = true, \vec{K} = \emptyset$ as an event indicating the proper finishing of a task instance.*
  - *$E \subset O \cap C$ as a set of events $E \cap T = \emptyset$ with a function $type_{event} : E \rightarrow \{start, intermediate_{throw}, intermediate_{catch}, end\}$*
  - *$G \subset O$ is a set of gateways $G \cap E = \emptyset \wedge G \cap T = \emptyset$ with a function $type_{gateway} : E \rightarrow \{or, xor, and\}$.*
- *$F \subseteq O \times O$ represents the directed control flow relations, implying $f(o_i, o_j) \neq f(o_j, o_i), \forall i \neq j$.*
- *$T^E \subseteq E \times T$ is a set of attachments of intermediate events on tasks.*

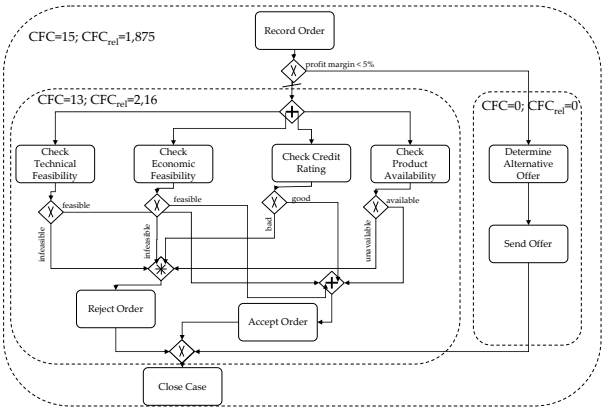
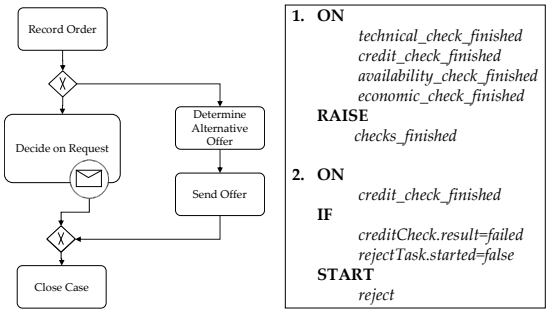


Figure 2: Workflow with Decomposition and Metrics



(a) Abstracted Workflow (b) Workflow Parts Mapped to Events&Rules

Figure 3: Combination of Workflow and Rules

- $X \subseteq M$, $M := A \times V$ *is the valid data context in terms of workflow variables and possible values.*

Definition 3 (Business Rule) *A business rule r from a ruleset R in the ECA format is defined as a triple* $\{e \in E \vee e = \emptyset, P' \subseteq M, \vec{A} := a_i \in \bigcup_{T_i \in O_i \forall O} T_i \cup D\}$, *where* $\vec{A}$ *can be interpreted as an ordered list of either task executions or data context change operations from the set D as a function* $d \in D : m_1 \subseteq M \rightarrow m_2 \subseteq M$, $m_1 \neq m_2$

Implications for architecture bindings :

- $\forall g \,|\, [|(\forall f \in F | f_1 = g)| > 1 \wedge type_{gateway}(g) \neq and]$, $\exists \rho \subseteq R$ specifying the split behavior.
- The shared data model implies that $\exists S \subseteq M$, $M := A \times V$ commonly used in C, R, O.
- The shared task model implies:

$$\bigcup_{T_i \in O_i \forall O} T_i \cap \bigcup_{T_j \in R_j \forall R} T_j \neq \emptyset$$

4 APPROACHING THE CHALLENGES: EXAMPLE

In this section, we concretize some suggestions for approaching the architecture realization challenges. For this purposes, we build upon our aforementioned example of a business analyst who wants to model complex checks for a customer and present an order processing workflow extended from (Seel et al., 2006) in figure 2. After a customer order has been received, the default continuation is to execute four checks, depending on which the order is rejected or accepted. If the profit margin of the ordered product is not high enough, an alternative offer is sent to the customer without accepting or rejecting the actual order. The decision logic encoded in the workflow diagram is quite complex and hard to capture. It therefore makes sense to outsource a part of the flow decision logic to a rule base.

To address challenge #2, design-time model complexity metrics may contribute to the identification of model optimization potentials. For our example, we are interested in complex parts of the workflow which are hard to capture via flow diagrams. Therefore the workflow is first decomposed by applying mechanisms as presented in (Polyvyanyy et al., 2009). The procedure identifies workflow segments as $\phi = (T' \subseteq T, G' \subseteq G, F' \subseteq F)$. Characteristics like intermediate events as permitted by definition 2 are not contained within the example, but are however considered in workflow segmentation work by (Yongchareon et al., 2010). The detected segments are marked with a dashed line in figure 2. To measure the complexity of a workflow segment, (Cardoso, 2006) provides an intuitive and empirically validated metric: the control-flow-complexity (CFC), which is based on the number of mental states a modeler has to consider when modeling the workflow caused by split gateway semantics. The metric can be set in relation to the number of task nodes within a workflow segment to express the "density" of decision complexity for ϕ as $CFC_{rel} = CFC(\phi) / |T'_\phi|$. The corresponding values for CFC and CFC_{rel} are indicated in the upper left corner of the workflow segments. Having identified segments in the workflow most appropriate for being outsourced as rules, they can be transformed into the rule format provided by definition 3 (relating to challenge #1) by applying pattern-based transformations like $\{f_1(t_1, g_1), f_2(g_1, g_2), f_3(g_2, t_2)\}, type_{gateway}(g_1)$

$= XOR$, $type_{gateway}(g_2) = AND \rightarrow r = \{t_1\_finish, P_{f_2}, start(t_2)\}$. As such, rules as presented in extracts in figure 3(b) can be constructed. As shown in figure 3(a), the transformed segment in the workflow is replaced by a placeholder task, with the intermediate event *checks_finished* attached for continuing the control flow as soon as a decision can be drawn from the checks.

As an intermediate conclusion, the benefits of the "outsourced" rules as provided by our approach are manifold. There is a potentially better understandability of single aspects of the decision logic (not necessarily the whole logic!). One can see for example immediately that a failed credit check directly leads to an order rejection. Furthermore, process extensions and exception handling mechanisms can be added more conveniently. A complex event caused by the customer overdrawing his account for more than 2 months could raise a *reject_finished* event for all order processes of the customer and cause the untimely closing of the instances.

5 CONCLUSION

This paper presented an integrated reference architecture for the combined modeling of business logic with workflows, business rules and complex events. The provision of an integration layer for the different meta-models as well as the supply of modeling guidelines were identified as the two main realization challenges for the architecture. To provide a starting point on how the challenges can be tackled, a procedure to identify complex decision logic within a workflow and outsource it to rules was presented. The procedure is based on design-time complexity metrics for workflows and pattern-based transformations. As future work, existing standards for the three paradigms like RIF and BPMN2 are to be examined to determine their meta-model compatibility. Furthermore, it is required to conduct a study on which granularity for each paradigm is appreciated by business analysts and how this can be related to metrics.

ACKNOWLEDGEMENTS

Thanks to Anis Charfi and Todor Stoitsev for helpful comments. The work presented in this paper was developed in the context of the project *Allianz Digitaler Warenfluss* (ADiWa) that is funded by the German Federal Ministry of Education and Research. Support code: 01IA08006.

REFERENCES

Anicic, D., Fodor, P., Stuhmer, R., and Stojanovic, N. (2009). Event-Driven Approach for Logic-Based Complex Event Processing. In *CSE 2009*, Vancouver, Canada. IEEE.

Brett, C. and Gualtieri, M. (2009). Must You Choose Between Business Rules And Complex Event Processing Platforms? Forrester.

Bry, F., Eckert, M., Patranjan, P., and Romanenko, I. (2006). Realizing business processes with ECA rules. In *PPSWR 2006*, volume 4187, pages 48–62, Budva, Montenegro. Springer.

Cardoso, J. (2006). Process control-flow complexity metric: An empirical validation. In *SCC '06*, pages 167–173, Chicago, USA. IEEE.

Dadam, P. and Reichert, M. (2009). The ADEPT Project. *Computer Science - R&D*, 23(2):81–97.

Decker, G., Grosskopf, A., and Barros, A. (2007). A graphical notation for modeling complex events in business processes. In *EDOC '07*, page 2736, New York, USA. ACM.

Gualtieri, M. and Rymer, J. R. (2009). The Forrester Wave: Complex Event Processing (CEP) Platforms, Q3 2009. Forrester.

Hay, D. and Healy, K. (2000). Defining business rules - what are they really. Business Rules Group.

Knolmayer, G., Endl, R., and Pfahrer, M. (2000). Modeling Processes and Workflows by Business Rules. In *BPM*, pages 16–29. Springer.

Leymann, F. and Roller, D. (1999). *Production Workflow - Concepts and Techniques*. Prentice Hall.

Luckham, D. (2002). *The Power of Events*. Addison-Wesley Professional.

Milanovic, M. and Gasevic, D. (2009). Towards a Language for Rule-Enhanced Business Process Modeling. In *EDOC '09*.

Paschke, A. and Kozlenkov, A. (2008). A Rule-based Middleware for Business Process Execution. In *Multikonferenz Wirtschaftsinformatik*.

Polyvyanyy, A., Smirnov, S., and Weske, M. (2009). The triconnected abstraction of process models. In *BPM '09*, number 26, pages 229–244, Ulm, Germany. Springer.

Seel, C., Simoin, B., and Werth, D. (2006). Business Rule-enabled Process Modelling. In *e-Challenges*, Barcelona, Spain.

van Eijndhoven, T., Iacob, M.-E., and Ponisio, M. L. (2008). Achieving Business Process Flexibility with Business Rules. In *EDOC '08*, pages 95–104, Munich, Germany. IEEE.

Yongchareon, S., Liu, C., Zhao, X., and Kowalkiewicz, M. (2010). BPMN Process Views Construction. In *DASFAA '10*, Tsukuba, Jaban.

zur Muehlen, M., Indulska, M., and Kittel, K. (2008). Towards Integrated Modeling of Business Processes and Business Rules. In *19th Australasian Conference on Information Systems*.