SAP WebFlow Made Configurable: Unifying Workflow Templates into a Configurable Model

F. Gottschalk, W.M.P. van der Aalst and M.H. Jansen-Vullers

Eindhoven University of Technology,
P.O. Box 513, 5600 MB Eindhoven, The Netherlands.
{f.gottschalk,w.m.p.v.d.aalst,m.h.jansen-vullers}@tue.nl

Abstract. To facilitate the implementation of workflows, enterprise and workflow system vendors typically provide workflow templates for their software. Each of these templates depicts a variant of how the software supports a certain business process, allowing the user to save the effort of creating models and links to system components from scratch by selecting and activating the appropriate template. A combination of the strengths from different templates is however only achievable by manually adapting the templates which is cumbersome. We therefore suggest in this paper to combine different workflow templates into a single configurable workflow template. Using the workflow modeling language of SAP's WebFlow engine, we show how such a configurable workflow modeling language can be created by identifying the configurable elements in the original language. Requirements imposed on configurations inhibit invalid configurations. Based on a default configuration such configurable templates can be used as easy as the traditional templates. The suggested approach is also applicable to other workflow modeling languages.

Keywords: Process Configuration, Reference Model, Workflow Template

1 Introduction

A workflow engines facilitate the execution of business processes by guiding and monitoring the process while "running through the company". Whenever needed it assigns tasks to the responsible individuals, provides all relevant information, and takes action in case tasks are not performed in time [10].

To execute a workflow in a workflow engine, it must be specified in the engine's workflow modeling language which is quasi an extended business process modeling language (like Event-driven Process Chains (EPCs) [9] or BPMN [17]). Besides depicting the process, it allows for the integration of the process model with other systems like enterprise systems, office software, or intranet portals.

The effort to establish this integration is typically high. When modeling a workflow, it is not only required to ensure the correct control flow, but also the data flow between the different steps and components must be "programmed"

and assignment rules for resources must be set up. Thus, the re-use of workflow models promises huge costs savings when implementing workflows in similar system environments. This holds especially for enterprise systems, which due to the system structure already imply the set-up of processes in quite specific ways.

The biggest enterprise system vendor worldwide is with more than 100,000 installations SAP [14]. The workflow engine that is delivered by SAP together with every installation of its enterprise system since the R/3 Release 3.0 is called WebFlow. Together with the engine, SAP also delivers hundreds of simple, predefined workflow templates for all areas of the system – from logistics to personal time or compensation management [10]. Thus, the template repository can be regarded as a reference model of common workflows in SAP's enterprise system. The templates, which typically fit comfortably on one A4 page, can easily be activated in the SAP system. Without a workflow designer having ever spent a significant amount of time on the workflow definition, they are then triggered automatically whenever their execution is required. Often SAP users are therefore working on the predefined workflows without even knowing it.

For many business processes the repository includes several workflow templates, each suggesting a different implementation of the particular process. For example, a dedicated workflow template exists not only for the approval of a travel request, but also for the automatic approval of a travel request, the approval of a travel plan, and both the approval and the automatic approval of a trip. All these templates are of course similar. To decide on the appropriate template, each template is documented in SAP's online help system, typically also combined with an EPC of the process [13]. However, there is no information available that highlights the differences between the templates. Instead, the workflow designer has to familiarize herself with each workflow template, compare them manually, and find the small differences. If, as, e.g., in the EPCs documenting the templates for supporting the approval of a travel request and the automatic approval of a travel request, a certain degree of inconsistency exists in the documentation of the templates because it is unclear if "Create travel request" and "Enter travel request" actually depict the same task [13], this comparison requires even more efforts. Further on, the workflow designer might even come up with the conclusion that a combination of two templates is the optimal solution as each template has its strength at a different point. As such a template is not available she can then only manually adapt the weak point of one of the templates to match the not selected one here as close as possible.

To help the workflow designers in getting the optimal workflow template we propose in the following to combine the different workflow template variants into a single template using an extension to the workflow modeling language making it a configurable language. Using SAP's workflow modeling language and the travel approval process example, we will show how this configuration extension allows the workflow designer to select or eliminate the relevant or irrelevant template parts in the integrated model. Thus, the designer can focus on the requirements on the workflow instead of searching for the possibilities in the various templates.

An extended description of the approach is available as an internal report [7].

2 Configurable workflow models in SAP

In almost all graphical process modeling languages the routing of cases through the model is determined by the triggering of tasks, functions, steps or any other type of performed action as well as by the release of cases after the actions' completion. We call each possibility to trigger an action an *input port* of the action and each way an action can trigger subsequent paths an *output port* of the action. When integrating several workflow variants into one workflow model, ports are thus the elements of the integrated workflow where we can apply the two general applicable configuration methodologies of blocking and hiding which we identified in our previous research [6].

Actions, ports, and their configuration in SAP WebFlow SAP WebFlow is mainly based on so-called *steps* and *events* which are organized in a block structure (see Figure 1 for the before-mentioned travel approval process which is accessible in SAP as workflow WS20000050). Steps represent either routing constructs or system functionalities. In the simplest case a single step as, e.g., an activity forms a block. However, whenever a step causes the branching of the control-flow (as, e.g., a fork, a condition, or a user-decision) the branching of the control flow is matched by exactly one corresponding join and all elements until (and including) the join belong to the branching step's block. The elements

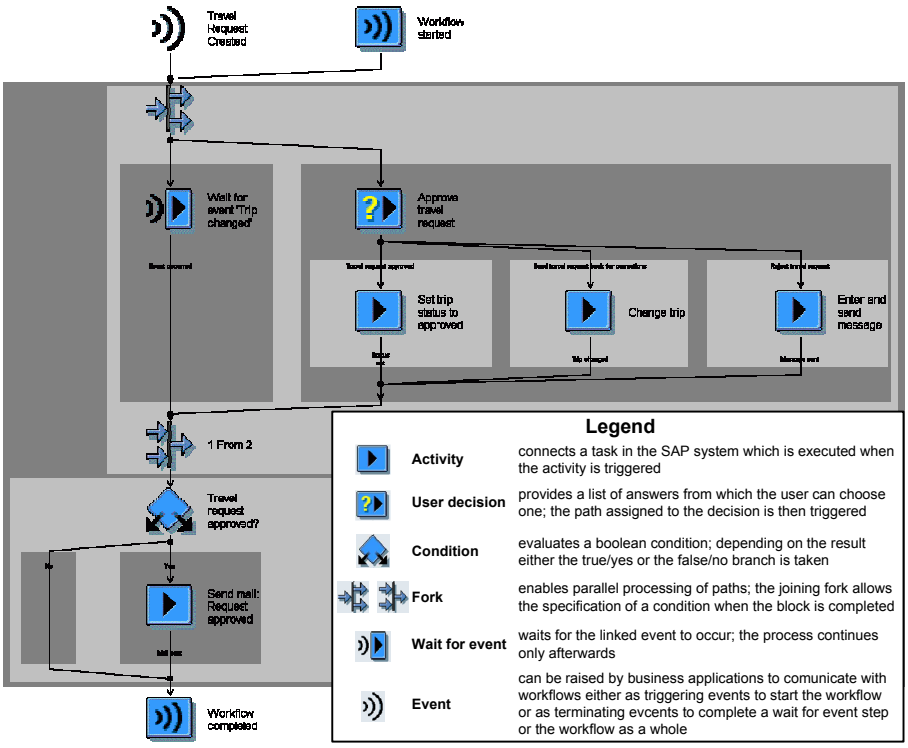

Fig. 1. SAP's workflow template for travel approval processes (WS20000050)

in each of the branches represent then sub-blocks of the branching block. For example, in Figure 1 the block of the fork is highlighted in light grey. It contains two sub-blocks for the two branches. The block of the user-decision *Approve travel request* branches again in three sub-blocks for the particular activities.

Thus, each block can be seen as an action. Each block contains basically just one unique input path and one unique output path which are the ports of the action. The largest block is the complete workflow itself. It is the only block which can be triggered in multiple ways as it cannot only be triggered by a manual start of the workflow but also by (various) events which are linked to the workflow block. In addition, events can be linked to a workflow block or a *wait for event* step to terminate them. Thus, each of these links connecting events to the workflow block can also be seen as a port. As they have some different characteristics from a block's in- and output ports, we call them *event ports*.

The linkage between steps or events and workflows includes also the linkage of the data in the data containers of the step or event and the workflow. This linkage enables starting workflows or steps with the right parameters, e.g. to select responsible resources or correct documents. We will skip such implementation details here, but not without repeating that this modeling and customizing effort is far more time-consuming than the pure creation of a process model. These efforts therefore motivate the development of a configurable SAP WebFlow.

To configure a workflow model, we can use the configuration methodologies of blocking and hiding [6] at the ports of actions. If a port of an action is *blocked*, the action cannot be triggered. Thus, the process will never continue after the action or reach any subsequent action. If an action's port is *hidden*, the action's performance is not observable, i.e. it is skipped and consumes neither time nor resources. But the process flow continues afterwards and subsequent actions will be performed. If an action in a workflow model is neither blocked nor hidden, then we say it is *enabled*, which refers to its normal execution. This concept can be applied to the input ports of blocks in SAP WebFlow in a straightforward manner. If the input port of a block is enabled, cases can normally enter and be executed in the block. If the input port is hidden, a case entering the block is directly forwarded to the unique exit port of the block, quasi bypassing all the content of the block. If the input port is blocked, the case cannot enter the block at all and needs to continue via other alternative branches.

Common soundness criteria for workflow models require that cases must always have a chance to complete a workflow. Thus, a block's input port can only be blocked if an alternatively executable branch exists which leads to the workflow's completion. For example, instead of the *Change trip* step, the *Set trip status to approved* or the *Enter and send message* steps can be executed. It is however impossible to block the input port at the *Travel request approved?* step as no alternative routing exists here. In the case of this particular fork step, it is possible to block one of the two sub-blocks, but only because the join requires just one of the two branches to complete. If the condition at the join would have been "2 From 2" a blocking of one of the sub-blocks would have made it impossible to later satisfy this condition and thus caused a deadlock. Therefore, when

configuring the sub-blocks of a fork, the condition at the joining fork determines the maximal amount of sub-blocks that can be blocked.

Each case entering a block must be able to leave the block via its unique output port. Thus, this port can only be blocked if the block's input port is blocked. However, if the input port is blocked no tokens can arrive at the output port, i.e. the configuration has no influence on the process. Hiding of an output port is not feasible either because the path to the next block does not contain any action that can be skipped. We can therefore consider the output port configuration as practically irrelevant in SAP WebFlow.

In SAP WebFlow an event only triggers a workflow if the link between the event and the workflow is activated. SAP WebFlow already supports the deactivation of such a link, quasi corresponding to the blocking of the particular event port. Although a triggering event port is an inflow port, hiding of such a port is quite useless because it would basically mean skipping the whole workflow block without performing any step. Terminating event ports for wait-for-event steps are output ports. Even though terminating events are externally triggered, they basically enforce the removal of the case from the particular block. Thus, the functionality of SAP to activate or deactivate such linkages already provides exactly the required functionality to configure event ports.

In Figure 2 we combined the workflow template from Figure 1 with SAP's template for the automatic approval of travel requests (WS12500021). By blocking the *change trip* step the corresponding block is quasi removed from the workflow. By hiding of the *Travel request approved?* step, also the sub-block of mailing the request's approval is skipped. All other blocks are enabled. Although the two process templates were integrated, the result of this configuration corresponds exactly to the template for the automatic approval of travel requests. By blocking the sub-block of the *Criteria for Automatic Approval* step's *Automatically Approve Travel Request* outcome and instead enabling the *Change trip* and the *Travel request approved?* blocks, we would get the workflow from Figure 1.

Restricting the configuration opportunities Not all such combinations are feasible in practice. We already mentioned the requirement that a workflow always has to have the opportunity to complete. In addition, there are always a lot of semantic requirements. For example, it is well possible to block or enable the *Wait for event 'Changed'* step's block. However, hiding it prevents the workflow from working correctly as it causes a direct forwarding of cases to the joining fork whose condition would immediately be satisfied. The other branch would get superfluous and cancelled before any decision on the approval can be made.

Using logical expression to denote such requirements, we could for example write `configuration("Enter and send short message")=ENABLED` to depict that the particular block must be enabled or `configuration("Wait for event 'changed'")!=HIDDEN` for the requirement that the block cannot be hidden. Such atomic logical expressions can then be combined, e.g., to formulate a requirement that if the *Change trip* block is blocked then the *Travel request approved?* must be hidden (`configuration("Change trip")=BLOCKED => configuration("Travel request approved?")=HIDDEN`).

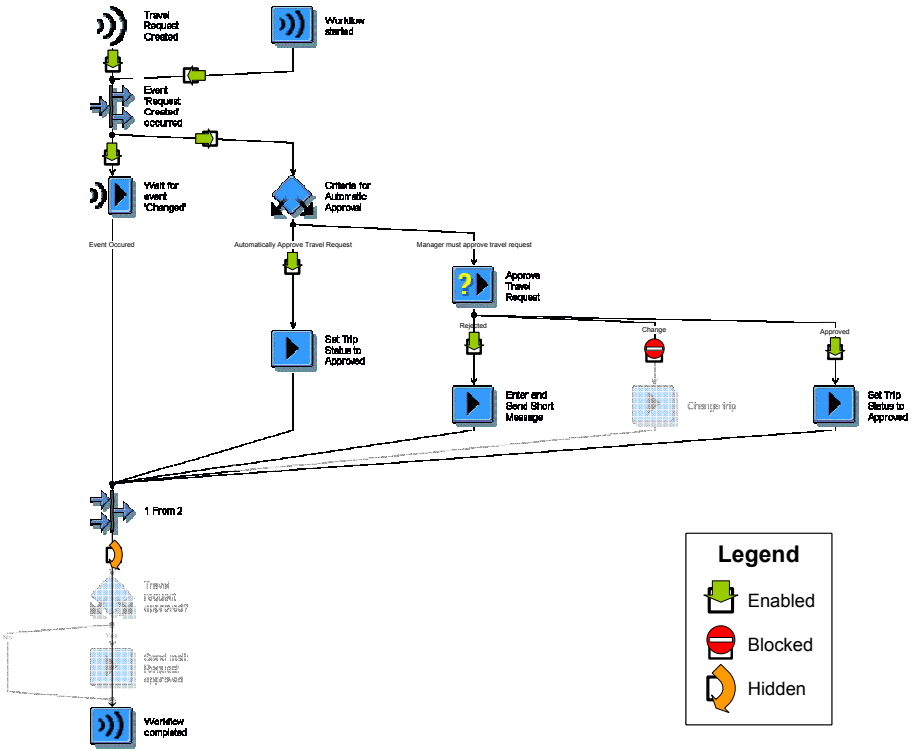

Fig. 2. The combined workflow template of SAP's travel approval and automatic travel approval templates, configured as the automatic approval workflow.

To test if a configuration fulfills all requirements, the requirements can be combined using `AND` operators. By determining blocks which can change their configuration values without breaking these requirements, a tool that regularly re-evaluates the configuration opportunities could even highlight those workflow blocks which are not bound to their current value and thus really configurable.

Plug & Play The current SAP WebFlow templates allow for an easy integration of the predefined workflow templates into a running SAP system by just assigning the relevant resources to the steps and activating the triggering events. To enable such an easy activation also for configurable workflow templates, each workflow template has to have a default configuration that satisfies the specified requirements. For example, the configuration of Figure 2 representing the automatic approval template could be the default configuration for the combined travel approval workflow template. When activating the triggering event, the workflow corresponding to this configuration would automatically be enabled. However, if it is for example desired, that the manager is also able to ask for a change of the travel request, it is sufficient to assign the responsible resource to the *Change* step and activate the currently blocked port. Without any modeling effort the new configuration of the workflow template can be used.

3 Related work

The workflow templates of SAP's WebFlow engine depict suggestions how to execute the particular processes in SAP. Thus, as the conceptual *SAP reference model* [4] they are reference models for processes in SAP, but on an executable level. Motivated by the "Design by Reuse" paradigm, reference models simplify the process model design by providing repositories of generally valid and potentially relevant models to accelerate the modeling process [5].

To be applicable in a particular context (e.g., a specific enterprise), a generally valid reference model must be adjusted to individual needs. To enable the adaptation of reference models by means of configuration, several variants of the process must be integrated. Extensions to conceptual process modeling languages allowing for such integrations are suggested by Becker et al. [3], Rosemann and van der Aalst [12], and Soffer et al. [15]. Although the potential efficiency benefits of using configurable process models during enterprise system implementations are highlighted by all the authors, the suggested usage of the three approaches remains on the conceptual level.

The idea of providing configurable workflow models as suggested here implies to have different variants of the process in different contexts. Of course, the required workflow configuration can change over time which then requires the transfer of running workflow instances to the new configuration. Systems tackling these problems are also called configurable, re-configurable or adaptive workflow systems (e.g., in [8,16]), but typically neglect the preceding aspect of how the change of the workflow model can be supported.

4 Conclusions & Outlook

Based on the block-structured workflow notation of SAP's WebFlow engine, which comes with a huge set of pre-defined workflow templates, we showed the advantages of integrating several workflow templates into a single workflow model from which workflow variants can be derived by means of configuration. To make a workflow modeling language configurable the elements representing *actions* and their *ports* which route the cases through the actions must be identified. Representing runtime alternatives, ports can be *enabled* to allow the action's execution, be *hidden* to skip the particular action, or be *blocked* to prevent any flow of cases via the action. *Requirements* on the configuration ensure the configuration's applicability on the workflow. A *default configuration* enables the usage of a configurable workflow template even without any configuration effort and serves as the starting point for any configuration. All that is needed to use such configurable models in SAP WebFlow is an implementation of the user interface for performing configuration decisions, a tool checking the requirements, and a transformation of the configurable into the configured model.

In future research, we have to show that our ideas are also applicable to non-block-structured workflow modeling languages. For this purpose, we are currently applying these ideas onto YAWL, an open-source workflow system supporting

much more patterns than SAP WebFlow [1,2]. To provide further assistance for the configuration of workflow models, we aim at integrating the idea of configurable workflow modeling languages into a configuration framework enabling the use of advanced decision-making tools for performing the configuration [11], and a synchronized configuration between workflows and other software applications.

References

1. W.M.P. van der Aalst and A.H.M. ter Hofstede. YAWL: Yet Another Workflow Language. *Information Systems*, 30(4):245–275, 2005.
2. W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, and A.P. Barros. Workflow Patterns. *Distributed and Parallel Databases*, 14(1):5–51, 2003.
3. J. Becker, P. Delfmann, A. Dreiling, R. Knackstedt, and D. Kuropka. Configurative Process Modeling – Outlining an Approach to increased Business Process Model Usability. In *Proceedings of the 15th IRMA IC*, New Orleans, 2004.
4. T. Curran, G. Keller, and A. Ladd. *SAP R/3 Business Blueprint: Understanding the Business Process Reference Model*. Prentice Hall, Upper Saddle River, 1998.
5. P. Fettke and P. Loos. Classification of Reference Models – a Methodology and its Application. *Information Systems and e-Business Management*, 1(1):35–53, 2003.
6. F. Gottschalk, W.M.P. van der Aalst, and M.H. Jansen-Vullers. Configurable Process Models – A Foundational Approach. In *Proceedings of the Reference Modelling Conference 2006*, pages 51–66, Passau, Germany, 2006.
7. F. Gottschalk, W.M.P. van der Aalst, and M.H. Jansen-Vullers. SAP WebFlow Made Configurable: Unifying Workflow Templates into a Configurable Model. *Beta Working Paper 221*, Eindhoven University of Technology, The Netherlands, 2007.
8. Y. Han, T. Schaaf, and H. Pang. A Framework for Configurable Workflow Systems. In *Proceedings of the 31st International Conference on Technology of Object-Oriented Language and Systems*, pages 218–224, Los Alamitos, 1999.
9. G. Keller, M. Nüttgens, and A.W. Scheer. Semantische Prozeßmodellierung auf der Grundlage Ereignisgesteuerter Prozeßketten (EPK). Veröffentlichungen des Instituts für Wirtschaftsinformatik, Heft 89 (in German), Saarbrücken, 1992.
10. A. Rickayzen, J. Dart, C. Brennecke, and M. Schneider. *Practical Workflow for SAP – Effective Business Processes using SAP's WebFlow Engine*. Galileo Press, 2002.
11. M. La Rosa, J. Lux, S. Seidel, M. Dumas, and A.H.M. ter Hofstede. Questionnaire-driven Configuration of Reference Process Models. In *Proceedings of the 19th CAiSE*, pages 424–438, Trondheim, Norway, 2007.
12. M. Rosemann and W.M.P. van der Aalst. A Configurable Reference Modelling Language. *Information Systems*, 32(1):1–23, March 2007.
13. SAP AG. *SAP Library – Workflow Scenarios in Travel Management (FI-TV)*, 2006. `http://help.sap.com/saphelp_erp2005vp/helpdata/en/d5/202038541ec006e10000009b38f8cf/frameset.htm`.
14. SAP AG. SAP History: From Start-Up Software Vendor to Global Market Leader, April 2007. `http://www.sap.com/company/history.epx`.
15. P. Soffer, B. Golany, and D. Dori. ERP modeling: a comprehensive approach. *Information Systems*, 28(6):673–690, September 2003.
16. S. Tam, W.B. Lee, W.W.C. Chung, and E.L.Y. Nam. Design of a re-configurable workflow system for rapid product development. *Business Process Management Journal*, 9(1):33–45, February 2003.
17. S.A. White et al. Business Process Modeling Notation (BPMN), Version 1.0, 2004.