Workflows over Workflows: Practical Experiences with the Integration of SAP R/3 Business Workflows in WISE[*]

Christoph Schuler Heiko Schuldt Gustavo Alonso Hans-Jörg Schek

Institute of Information Systems
Swiss Federal Institute of Technology (ETH)
ETH Zentrum
8092 Zürich, Switzerland
Email: {schuler,schuldt,alonso,schek}@inf.ethz.ch

Abstract

Business processes within companies are in general well established and supported by commercially available workflow management systems. However, when processes span multiple companies (such as, for instance, in the case of virtual enterprises or in business to business electronic commerce), software support quite is limited. In such cases, processes need to be implemented on top of already existing systems. The WISE project addresses this problem and provides adequate infrastructure to support the whole life-cycle of virtual enterprise processes, including the integration of legacy systems, especially of workflow management systems. In this paper, we describe our experience with the integration of SAP R/3 Business Workflows into WISE. In addition, and since an important characteristics of the WISE system is the enactment of processes with execution guarantees, we also discuss the provision of execution guarantees when external systems like SAP R/3 Business Workflows are involved.

1 Introduction

In a *virtual enterprise*, different companies temporarily work together in common projects to achieve common goals. In order to support this collaboration, the necessary infrastructure has to be available. In general, business processes within each company can be supported by workflow management systems. In virtual enterprise environments, however, processes go beyond corporation boundaries and encompass services provided by the different participants. A workflow management system supporting virtual business processes in a virtual enterprise must now face the challenge of incorporating existing processes running with a variety of workflow management systems.

The WISE project [AFH$^+$99] addresses this problem and provides an infrastructure to support the execution of virtual business process in virtual enterprises. As part of the WISE project, we have been working on the integration of workflow processes running in different workflow management systems such as, for instance, SAP R/3 Business Workflow and IBM FlowMark, into the WISE system. This

[*]Part of this work has been funded by the Swiss National Science Foundation under the project WISE (Workflow based Internet Services, http://www.inf.ethz.ch/department/IS/iks/research/wise.html) of the Swiss Priority Programme "Information and Communication Systems".

extended abstract focuses on the integration of SAP R/3 Business Workflows, discusses the approach taken, and presents practical experiences made during this endeavor.

The paper is structured as follows: In Section 2, we present the WISE system, a workflow management system developed to support viable business to business Electronic Commerce. Then, in Section 3, we provide a brief overview of the interoperability problems that have to be addressed when embedding legacy applications in workflow processes while at the same time providing execution guarantees for these processes. We then focus in Section 4 on the practical experiences gained by integrating SAP R/3 Business Workflow processes into WISE. Section 5 finally concludes the paper.

2 WISE: Workflow based Internet SErvices

The WISE project [AFH$^+$99] aims to provide a viable infrastructure for business to business electronic commerce in virtual enterprises. To this end, existing services of companies temporarily participating in a virtual enterprise are linked together and embedded within a virtual business process. To do this, WISE implements four different components: *definition*, *enactment*, *monitoring*, and *ad-hoc coordination*.

The *definition* component provides the possibility to create virtual business processes supported by an appropriate graphical interface (IvyFrame, a commercial business modeling tool by IvyTeam [Ivy]). The *enactment* component compiles these process models in a format suitable for execution and controls the execution of virtual business processes. In order to keep track of the state of virtual business processes, the *monitoring* component extracts and visualizes the necessary information. The *coordination* component finally offers the participants of a virtual business process the possibility to establish multimedia conferences based on the information produced during execution.

3 Integration of Applications into WISE

The typical processes implemented in WISE encompass activities which are invocations of different subsystems (e.g. processes running in different workflow management systems). One key feature of the WISE engine is to provide execution guarantees for these processes even in case of failures and of concurrent access to shared resources. To this end, the ideas of transactional process management [SAS99] are applied. Among others, these execution guarantees include guaranteed termination, a more general notion of atomicity than the standard all or nothing semantics. Guaranteed termination is realized by partial compensation and alternative executions. In addition, WISE also controls the parallelization of concurrent processes to guarantee correct interleavings.

3.1 Transactional Coordination Agents (TCAs)

In order to enforce such execution guarantees by the WISE engine, each participating subsystem must meet a series of requirements. These requirements include the following database functionality for single activities: All activities have to be atomic to avoid inconsistencies due to the undefined outcome of non-atomic activities within a subsystem. In case that activities can be semantically compensated once they have been executed correctly, the availability of this compensation has to be guaranteed. When activities can not be compensated, their commit may have to be deferred in certain cases. Finally, for some activities a repeated invocation with “exactly-once-semantics” is necessary in order to guarantee their successful execution. Furthermore, when parallel access to shared resources takes place, orders established between activities by the WISE engine have to be respected in the underlying subsystems.

Since the goal is to integrate arbitrary applications into WISE (and especially arbitrary workflow management systems), these requirements are in general not met. Therefore, each participating subsystem is wrapped by a transactional coordination agent (TCA) [NSSW94, SST98]. The WISE system then acts as a process scheduler on top of several TCAs serving as lower level schedulers. The task of the TCA that has to be provided for each application is therefore not only to exploit the subsystem specific interfaces to allow local services to be invoked from WISE but also to provide the required database functionality on top of the application (a detailed discussion can be found in [SSA99]). Thus, TCAs extend the idea of the application agents defined within the workflow reference model [Hol93] of the Workflow Management Coalition (WfMC) by providing additional database functionality on top of application systems.

3.2 Structure of Generic TCA

With respect to the functionality to be provided by each TCA, there are four different modules to consider [Wun96]: communication, scheduling, monitoring and execution (figure 1).

To support *execution*, activities must be mapped to local operations. For this purpose, subsystem-specific interfaces have to be exploited and thus, the execution module has to be tailored to the application to be integrated.

Furthermore, scheduling of local operations with respect to the activities specified by the WISE system has to be performed by the *scheduling* module. This includes the preservation of the given orders as well as the provision of local atomicity, the guaranteed availability of compensating activities, and eventually the deferment of local commits. For these purposes, the TCA has to persistently log the activities executed within the subsystem and eventually also the activities needed for compensation purposes.

In order to support the interaction between the WISE system and TCAs, a common communication protocol has to be provided by the *communication* module.

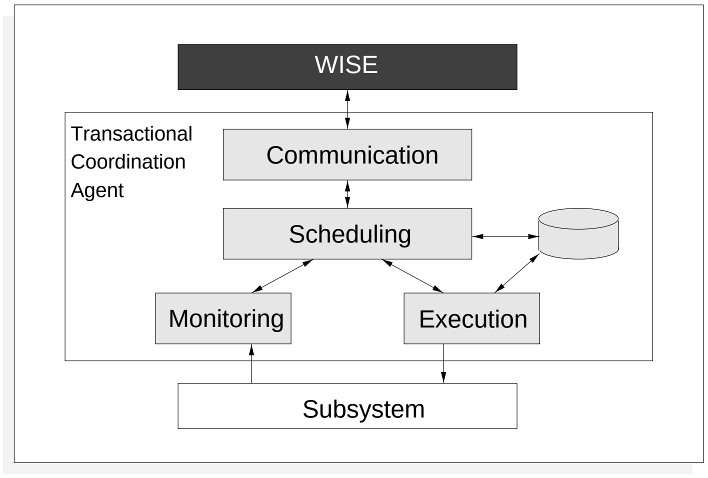

Figure 1: Structure of a generic TCA

Finally, the *monitoring* module covers the extraction of local state information from the underlying application. For this purpose, again existing interfaces have to be exploited such that the monitoring module also has to be tailored to the subsystem.

4 Integration of SAP Business Workflows

SAP R/3 [SAP] is one of the most commonly used application systems for business management purposes. It consists of specialized modules for several application areas (e.g., production planning, logistics, human resource management, or controlling). The system is built in a client/server architecture and is based on a relational DBMS [BEG96].

In order to integrate arbitrary SAP R/3 Business Workflows [WFB+95, SAP96] as part of WISE processes, a transactional coordination agent for SAP R/3 has been implemented [Sch98]. The architecture of this TCA is depicted in figure 2. The agent specific parts are colored in light gray whereas the standard SAP R/3 system is depicted in white color. The agent is tightly integrated into the core system as, for instance, R/3's underlying database is used for the management of the TCAs' metadata and is accessed through the standard DBMS interface of SAP R/3. The kernel of the TCA is itself a SAP workflow process (meta process) in which the function or the workflow to be executed is embedded in a generic way by simply passing its name as parameter to the meta process. In what follows, we describe in detail the architecture of this SAP R/3 agent.

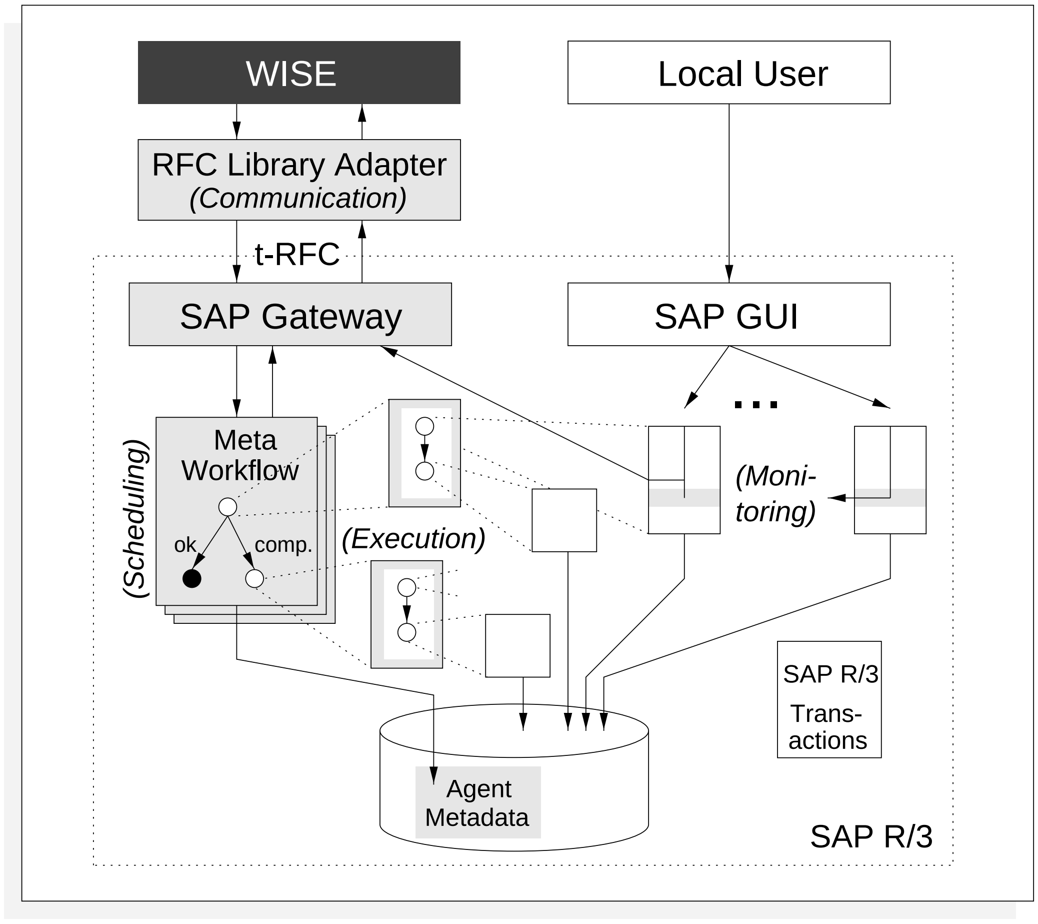

Figure 2: Architecture of the SAP R/3 agent

For execution purposes, the meta workflow of the TCA is called with a specification of the name of the function or process to be executed.

Communication with the WISE system takes place by exploiting the transactional remote function call (t-RFC) of SAP R/3 (t-RFC is the transactional variant of SAP's implementation of the remote procedure call, RPC, which is available via a set of C library functions). It supports both multiple parallel calls of R/3 functions from external applications as well as call-backs of external applications from within the R/3 system via the SAP gateway. The communication module (t-RFC library adapter) is thus a thin software layer transforming requests from WISE into t-RFC calls and vice versa.

The monitoring task is implemented by customization of the necessary R/3 transactions and processes. This is possible since certain hooks (customer exits) are foreseen in R/3 to add user-specific code and since the ABAP/4 [KW97] sources of all R/3 transactions are available (ABAP/4, Advanced Business Application Programming Language, is the 4^{th} generation programming language in which the greatest part of the R/3 system —aside of a small C kernel— is implemented in).

SAP R/3 supports the notion of transactions and guarantees full ACID properties for them. When a single function has to be executed as an activity in WISE, atomicity is provided by the R/3 system. When a complete workflow process has to be executed, however, appropriate failure handling mechanisms need to be implemented within this process to do a rollback in the case of failures in order to guarantee the required all-or-nothing semantics of activities.

In general, the meta workflow encompasses only two activities: the function or process to be executed and its associated compensation. In this, we follow the idea of an explicit registration of the compensation which is advantageous in that all instances of SAP R/3 workflow processes together with the associated parameters are stored persistently in the underlying DBMS. Thus, no additional effort has to be taken to log the parameters associated with an activity executed by WISE. After an activity of WISE (which corresponds to a workflow process of SAP) within the meta workflow has been executed successfully, the meta process performs an idle wait (which does not consume any resources). If the nested SAP process needs to be compensated, an internal event in SAP R/3 is generated via a t-RFC library call and the next activity of the meta workflow, the compensation, can be executed without explicitly specifying the required parameters. Otherwise, when compensation no longer has to be considered, the meta process is terminated. The structure of the meta workflow, which forms the body of the TCA, is depicted in figure 3 using the event-controlled process chain notation of SAP. Although this process conceptually encompasses only two activities, a couple of internal checks and events are relevant. First, the meta workflow writes a log entry to prevent multiple starts of the same task. After this, the desired workflow is started while in the meantime, the meta workflow waits in parallel on its termination signal (a). If an error occurred and the workflow is marked to be repeatable, the flow goes back to start the workflow again (b). After successful termination or abort of the workflow, the result is called back to theWISE system (c). The parallel section (d) communicates with the external WISE system. At this point the meta workflow waits in an idle wait until a decision is sent from the WISE system in order to compensate (e) or not.

The retriability of SAP processes is achieved by exploiting SAP R/3's transactional remote function call mechanisms. After the t-RFC

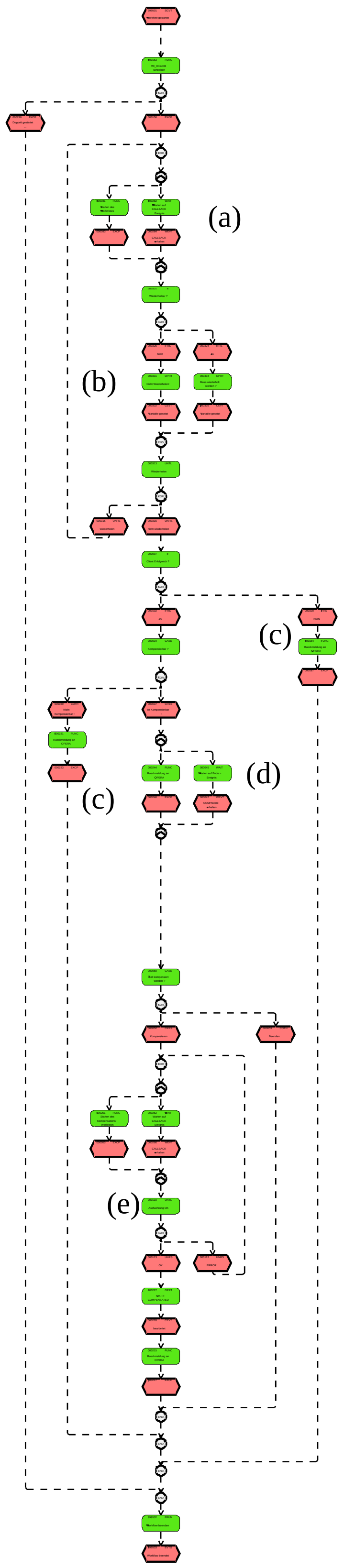

Figure 3: Structure of the agent's meta workflow in EPC notation

library adapter of the TCA has requested a t-RFC identifier from the R/3 system, all invocations of a retriable activity are performed with this ID and the R/3 system guarantees that is is executed only once even when multiple invocations of this activity occur.

SAP R/3's t-RFC was originally designed to start a function on another R/3 system. In order to invoke a workflow, a thin layer of SAP-R/3 functions is needed to propagate requests from outside to the SAP workflow management system. For this purpose, we use ABAP/4 functions in order to propagate calls from the outside world to the workflow component of SAP R/3.

This function layer also implements the mapping from the external ID used in WISE for each activity to SAP's internal workflow identification number. Once a meta workflow has been started it handles all the messages from the WISE system. For every type of command, a function is provided to send this command to the meta workflow by an internal event.

The start command invokes an instance of this meta workflow, passing the name and corresponding parameters to the workflow process to be invoked. The meta workflow marshalls the parameters and starts the desired R/3 workflow as if it was started directly from within the system. All parameters of a SAP workflow process are stored in a so-called container, which is an array of parameter name and value pairs. Since the meta workflow has to store the container of the workflow, this container has to be packed into the container of the meta workflow. That way every built-in SAP workflow can be invoked without changes on its definition. Furthermore, since only standard components of SAP R/3 are exploited, the TCA implementation will not be affected by changes introduced by new releases (it is, for instance, guaranteed that all customer exits of R/3 are also present in future versions of the system).

After the workflow terminates, the meta workflow sends a notification back to the WISE system and stays in idle wait until it can terminate itself or until it has to start the compensation workflow in order to undo the first workflow. The whole container of the workflow in the end state of the workflow is passed through meta workflows container in order to preserve all informations to undo the changes. This way we can store all this information persistently in the meta workflow's container.

The workflow model of SAP R/3 is designed to reach a consistent state after the termination of a process. As in our case we have have an additional workflow manager above SAP, we need a mechanism to propagate some result state back. To provide this feature, we introduced a special variable WF_RESULT which represents the result of the workflow execution.

5 Conclusion

In this short paper, we have discussed the practical experiences gained from the integration of SAP R/3 Business Workflows into the WISE system. This integration not only involves the invocation of SAP processes from WISE but also the provision of execution guarantees for SAP processes which can be exploited by WISE. To this end, a transactional coordination agent (TCA) is required, which is in the case of the SAP R/3 implementation tightly integrated into the system. Together with a TCA that has been implemented for IBM's FlowMark [IBM94], the SAP R/3 TCA complements the effort to allow the WISE system to run processes on top of processes.

Based on the TCAs already implemented, our future work aims in providing further support with respect to the monitoring of nested processes that are, in general, not known to WISE. To this end, an abstract description of nested processes will be made available. The TCA then publishes internal states of nested processes with respect to this abstract description such that it can be displayed by the monitoring facilities of WISE.

References

[AFH+99] G. Alonso, U. Fiedler, C. Hagen, A. Lazcano, H. Schuldt, and N. Weiler. WISE: Business to Business E-Commerce. In *Proceedings of the 9th International Workshop on Research Issues in Data Engineering. Information Technology for Virtual Enterprises (RIDE-VE'99)*, pages 132–139, Sydney, Australia, March 1999.

[BEG96] R. Buck-Emden and J. Galimow. *SAP R/3 System: A Client/Server Technology*. Addison-Wesley, 1996.

[Hol93] D. Hollingsworth. *Workflow Management Coalition: The Workflow Reference Model*. Workflow Management Coalition, December 1993. Document TC00-1003.

[IBM94] IBM. *FlowMark for OS/2: Managing Your Workflow*, first edition, May 1994.

[Ivy] IvyTeam, Zug, Switzerland. `http://www.ivyteam.com`.

[KW97] R. Kretschmer and W. Weiss. *Developing SAP's R/3 Applications with ABAP/4*. Sybex, 1997.

[NSSW94] M. Norrie, W. Schaad, H.-J. Schek, and M. Wunderli. CIM Through Database Coordination. In *Proceedings of the International Conference on Data and Knowledge Systems*, May 1994.

[SAP] SAP AG, Walldorf, Germany. `http://www.sap.com`.

[SAP96] SAP AG, D-69185 Walldorf. *System R/3: SAP Business Workflow*, 1996. White Paper.

[SAS99] H. Schuldt, G. Alonso, and H.-J. Schek. Concurrency Control and Recovery in Transactional Process Management. In *Proceedings of the ACM Symposium on Principles of Database Systems (PODS'99)*, pages 316–326, Philadelphia, Pennsylvania, USA, May 31-June 2 1999.

[Sch98] C. Schuler. Design and Development of a Coordination Agent for the Integration of SAP R/3 in Workflow Processes. Diploma thesis, Database Research Group, Institute of Information Systems, ETH Zürich, July 1998. In German.

[SSA99] H. Schuldt, H.-J. Schek, and G. Alonso. Transactional Coordination Agents for Composite Systems. In *Proceedings of the 3rd International Database Engineering and Applications Symposium (IDEAS'99)*, pages 321–331, Montréal, Canada, August 1999.

[SST98] H. Schuldt, H.-J. Schek, and M. Tresch. Coordination in CIM: Bringing Database Functionality to Application Systems. In *Proceedings of the 5th European Concurrent Engineering Conference (ECEC'98)*, pages 223–230, Erlangen, Germany, April 1998.

[WFB+95] H. Wächter, F. Fritz, A. Berthold, B. Drittler, H. Eckert, R. Gerstner, R. Götzinger, R. Krane, A. Schaeff, C. Schlögel, and R. Weber. Modellierung und Ausführung flexibler Geschäftsprozesse mit SAP Business Workflow 3.0. In F. Huber-Wäschle, H. Schauer, and P. Widmeyer, editors, *GISI 95 – Herausforderungen eines globalen Informationsverbundes für die Informatik*, Informatik Aktuell, pages 197–204. Gesellschaft für Informatik (GI) und Schweizer Informatiker Gesellschaft (SI), Springer-Verlag, 1995. In German.

[Wun96] M. Wunderli. *Database Technology for the Coordination of CIM Subsystems*. PhD thesis, Swiss Federal Institute of Technology Zürich, 1996.