Proceedings of the 2011 Winter Simulation Conference
S. Jain, R. R. Creasey, J. Himmelspach, K. P. White, and M. Fu, eds.

WORMS- A FRAMEWORK TO SUPPORT WORKFLOWS IN M&S

Stefan Rybacki
Jan Himmelspach
Fiete Haack
Adelinde M. Uhrmacher

University of Rostock
Albert-Einstein-Str. 22
18059 Rostock, GERMANY

ABSTRACT

Workflows are a promising mean to increase the quality of modeling and simulation (M&S) products such as studies and models. In exploiting workflows for M&S, requirements arise that need to be reflected in the structure and components of a workflow supporting framework, such as WORMS (WORkflows for Modeling and Simulation). In WORMS, we adapt concepts of business process modeling and scientific workflows. Particular attention is given to extensibility and flexibility which is supported by a plug-in based design and by selecting workflow nets as intermediate representation for workflows. The first application of WORMS has been realized for the modeling and simulation framework JAMES II. A small case-study illuminates the role of components and their interplay during evaluating a cell biological model.

1 INTRODUCTION

As discussed by Rybacki et al. (2010) workflows pose a promising mean to increase the credibility and quality of scientific results. Especially features provided by using workflows like provenance, reproducibility and roles support are desired to overcome the "crisis of credibility" of simulation studies (Pawlikowski et al. 2002, Kurkowski 2006). Workflows for M&S can be based on established M&S process models as proposed by Balci (1990), Sargent et al. (2006) or Law and Kelton (2007).

To realize workflow support for M&S one has to consider standards as suggested for general workflow systems (Hollingsworth 1995) and requirements that are specific for scientific workflows and M&S workflows in particular (Rybacki et al. 2010). One possibility to achieve such support is to adapt existing systems supporting workflows, e.g., Kepler (Ludäscher et al. 2006). The resulting benefits of such a reuse are that many functionalities required are already provided. A disadvantage is that a user might be restricted to a particular system and the constraints that come along with it.

The other possibility is to design a system for supporting M&S workflows from scratch. The advantage is a system designed specifically for the requirements of M&S. However, as already argued in (Rybacki et al. 2010), work on supporting workflows in M&S has just started and requirements are likely to be refined in the years to come. Thus, a flexible and extensible software design is mandatory. Therefore, a plug-in-based design appears suitable, as it has already shown its value in conducting M&S experiments, by seamlessly integrating different modeling formalisms, calculation algorithms, and validation methods (Ewald et al. 2010). A central question is how to represent workflows internally, as they are the mean of interaction between the different components. Here a formalism with a clear semantics is required to ensure that each component knows exactly how the workflow is to be interpreted. High level petri nets are such a class of formalism which have been established in the workflow community (Ellis and Nutt 1993, van der Aalst, Wil M. P. 1996a).

With our plug-in based framework WORMS (WORkflows for Modeling and Simulation) we focus on the second possibility, the design of a workflow system supporting M&S from scratch. In the following the central components of WORMS and their interplay will be presented as well as the integration of WORMS into a simulation framework, here JAMES II (Himmelspach and Uhrmacher 2007, Dalle et al. 2009).

2 BACKGROUND

2.1 Workflows & Workflow Systems

Different definitions of a process, a workflow, and a task exist. We use the definitions found in (van der Aalst and van Hee 2004):

Workflow is defined as a network of tasks with rules that determine the (partial) order in which the tasks should be performed. A rule can be either "Sequencing", "Selection of choice", or "Parallelism".

Task is a logical unit of work carried out by one resource.

Resource can be anything that is able to perform specific tasks such as a machine, person, group of machines or persons.

Workflow (Management) System manages resources and the execution of the workflow.

The Workflow Management Coalition (WfMC) has developed a conceptual architecture for workflow management systems (Hollingsworth 1995). This design covers a lot of potential scenarios and so it is no surprise that the framework presented in this paper shares some similarities with the WfMC concept.

2.2 Representing Workflows

One of the crucial questions in any workflow system is how to represent the workflow. Meanwhile there exist a number of description languages and notations originated in the business domain, e.g., XPDL (WfMC 2008), BPEL (Juric 2006), BPMN (White 2004), and UML (Dumas and ter Hofstede 2001, Rodríguez et al. 2011). Further, extensions of the Petri Net formalism are used to represent workflows, such as Colored Petri Nets (Merz, M. and Moldt, D. and Müller, K. and Lamersdorf, W. 1994), Workflow Nets (van der Aalst, Wil M. P. 1996a) and ICNs (Ellis and Nutt 1993). The formal semantics of the Petri Net based representations allow to exploit diverse analysis tools to check diverse properties of the workflows (Schmidt 2000, Verbeek and Aalst 2000). Thereby problems and faults can be detected and thus the quality of the workflows can be improved.

2.3 Requirements for M&S

Although a system supporting workflows in M&S is characterized by the typical requirements of scientific workflows (see Ludäscher et al. (2006), Gil et al. (2007)), a few requirements appear to be rather specific to M&S. Those requirements are: *Scalability,* ***User-interaction****, Detached execution, Reliability & fault-tolerance,* ***Data provenance****, Smart reruns, Workflow monitoring & control,* ***Workflow Roles, Collaborations & Delegation****, System Stability* and *Data Security* (The bold ones are those identified as particularly important for a system supporting workflows in M&S.) (Rybacki et al. 2010).

These requirements form the base for the development of our framework WORMS.

3 THE FRAMEWORK WORMS

3.1 Framework Overview

WORMS is realized as a framework and thus the implementation can be used to ease the integration of workflow support into different M&S systems.

The basic architecture of our framework is plug-in based and follows the strategy pattern (Gamma et al. 1995), by providing extension points where custom "strategies" respectively plug-ins can be plugged

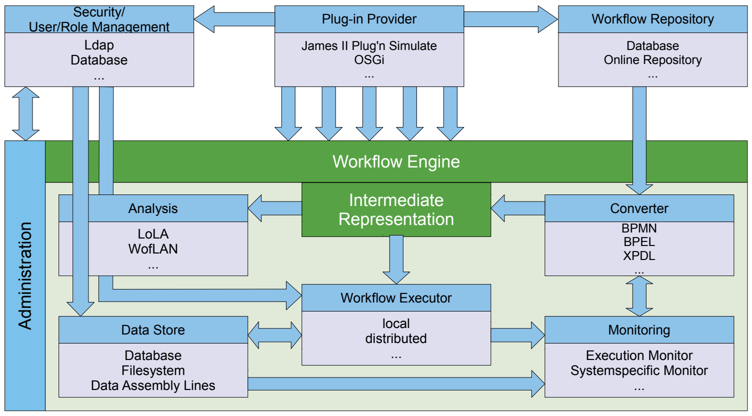


Figure 1: Framework Overview. Fixed parts are painted in green, extensions points in blue.

in as needed. In addition to extensions points there are fixed parts responsible for managing, organizing and orchestrating the plug-ins.

The components of WORMS are sketched in Figure 1. The system is composed of two main parts. The first part consists of the management components which comprise the plug-in management, the workflow repositories, and the user management. The second part is dedicated to the workflow execution (the *Workflow Engine*), which makes intensive use of the first part.

The *Workflow Engine* has a specific *Intermediate Representation*, which is used to represent the workflows internally. The plug-ins that are provided for extension points of the *Workflow Engine*, i.e., *Analysis*, *Converter*, *Workflow Executor*, *Monitoring* and *Data Store*, have to rely on this representation. The components in the first part (*Plug-in Provider*, *Security and User/Role Management* and the *Workflow Repository*) are not *Workflow Engine* specific and can be reused with different engines. A special case is the *Administration* extension point which is *Workflow Engine* specific but also has to be available to the outside.

This implies two major extension strategies when incorporating this framework in M&S software. On the one hand one can extend the provided *Workflow Engine* by adding, removing and/or exchanging components (plug-ins) for workflow specific extension points, such as *Analysis* Tools or different *Converter*. On the other hand it is also possible to reuse another existing workflow engine or system and replace the provided *Workflow Engine* altogether.

3.2 Intermediate Workflow Representation

As already mentioned in the previous section the components of the workflow engine operate on an internal representation of workflows. This is needed for the components to work together seamlessly. In the ideal case, the workflow representation should enable the analysis, and conversion of workflows in addition to the execution of workflows. Aalst motivates in (van der Aalst et al. 1994, van der Aalst, Wil M. P. 1996b) the use of Petri Nets for the description of workflows, an idea which is supported by further authors (Merz, M. and Moldt, D. and Müller, K. and Lamersdorf, W. 1994, Oberweis, A. and Schätzle, R. and Stucky, W. and Weitz, W. and Zimmermann, G. 1996, Russell et al. 2009). Thus we selected "workflow nets" as the underlying workflow model representation as they base on Petri Nets as well. This allows to leverage a well known and researched formalism which provides clear semantics as well as many analysis techniques, that, e.g., help identifying malformed workflows or dead locks (Murata 1989, Schmidt 2000, Jensen et al. 2007). It is important to point out that using workflow nets does not restrict the way we can define workflows. As long as one can transform another workflow representation into a workflow net or petri net (e.g., as shown for BPEL in (Hinz et al. 2005)), the representation can be used and a workflow modeler can benefit from the corresponding tools for workflow modeling.

3.3 Parts of WORMS

WORMS is based on different well separated parts which we will explain herein in more detail. When defining the parts the requirements, presented in Section 2.3, have to be taken into account. Their relation is given in Section 3.4.

Workflow Engine The provided *Workflow Engine* is the central component organizing the execution of a workflow. For instance if the workflow model provided by the *Workflow Repository* is not in the used intermediate format, it will use an appropriate *Converter* (if present) to transform the workflow model into the intermediate format. Additionally, if required, it can also initiate an analysis of the model using one of the provided tools through the *Analysis* extension point. After the conversion (and successful analysis), the *Workflow Engine* instantiates the model and passes it to a *Workflow Executor*, which selects an executor according to the available infrastructure. Further, the engine uses a *Security and User/Role Management* implementation to embed security requirements, e.g., restricting workflow execution to a specific user group.

Converter Since all the plug-ins work on an intermediate workflow representation, workflows need, if not already in the intermediate format, to be converted into this representation. This allows to model and describe workflows in any way, with any tool in any language such as XPDL or BPEL as long as there is a converter present. By allowing different workflow representation existing tooling can be leveraged and existing workflows could be reused or migrated.

This provides means to help to fulfill *Scalability* and *Collaborations & Delegation* requirements.

Workflow Executor By delegating the actual execution of a workflow to the *Workflow Executor* it is possible to exchange the executor executing a given workflow. There might be an executor that can distribute the execution over a network, e.g., to combine processing resources, whereas another executor might execute the workflow locally.

Having different executors is therefore helpful to support different kinds of execution scenarios. For instance the *Collaborations & Delegations* or *Detached execution* requirement for scientific workflows can be achieved by an executor supporting a networking structure, the execution on different platforms, the transfer of a running workflow from one machine to another, and/or the detached execution of a workflow/task. As security is important here as well, the executor should employ the *Security and User/Role Management* implementation to ensure access restrictions while executing the workflow.

Data Store The execution of workflows might imply the need to pass data from one task to the next. In addition data about the execution has to be recorded, e.g., which task did start when, how long did it take, did any errors occur, what has been used to execute the task, and so on. This information is required to document the execution and thus to repeat the execution of workflows.

These tasks are handled by the *Data Store*. Alternative implementations of the data storage can provide solutions dedicated to certain usage scenarios. For instance there can be an implementation based on e.g., Data Assembly Lines (Zinn et al. 2009) or databases.

The *Data Store*, in combination with the *Security and User/Role Management* component has to provide only the information which is actually accessible by a specific task and a specific user/role to ensure consistency and security. As the amount and type of data to be stored might vary from workflow to workflow it has to provide a robust data management. Together with the opportunity to use alternative implementations this allows for the required scalability.

Security and User/Role Management An implementation for this extension point takes care of authentication, authorization and user/role management. It has to be reusable by all those plug-ins provided for the framework needing security and user or role management.

One could implement such a plug-in based on Kerberos (Miller, S. P. and Neuman, B. C. and Schiller, J. I. and Saltzer, J. H. 1987) for authentication and authorization and LDAP (Koutsonikola 2004) for user management and therefore incorporate existing and widely adapted technologies. An implementation for the *Security and User/Role Management* extension point might allow the exchange of any of its subparts by providing further extension points, which can be exploited to integrate, e.g.,, a database-based user management in combination with the existing authentication and authorization elements.

Analysis A benefit of using a formal *Intermediate Representation* as internal workflow representation, is the availability of analysis methods for it. By providing analysis capabilities the quality of workflows and/or their refinement can be enhanced and supported. For example, this can help to verify a given workflow similar to the method used in (Wynn et al. 2007, Mao 2010, Zha et al. 2010, Weiš̆bach and Zimmermann 2010). Examples for such plug-ins are wrapper for LoLA (Schmidt 2000) or WofLAN (Verbeek and Aalst 2000). This allows to reuse existing research on model checking performed on Petri Nets.

Monitoring The *Monitoring* extension point is a key element of the framework because its implementations will track, trace and document what, when, where, and how things are done while executing a workflow. The results are then used for auditing, reengineering and profiling or for documentation, provenance and reproducibility of workflow executions. Important to note is, that although tracking the workflow execution is enough for auditing and profiling, for provenance and reproducibility also the used system, machine, and software components have to be tracked.

So by providing workflow and system specific components provenance and documentation can be achieved at virtually any level of detail. For instance by providing a dedicated monitoring component for a specific M&S system more information of the workflow execution can be recorded, such as used model or simulation parameters, events, or used software components, e.g., random number generators and event queues.

In addition based on the information collected workflows can be checked, reengineered, profiled, and improved (Park and Kim 2010, Accorsi and Wonnemann 2010). Furthermore the information can also be used to profile, verify and improve the used *Workflow Executor*.

However the most important factor is the provenance and reproducibility aspect which helps to ensure quality and credibility of the product generated by the monitored workflow (Wong et al. 2005, Davidson and Freire 2008, Missier et al. 2008, Miles et al. 2008).

Administration This component type supports controlling and administrating different aspects of the Workflow Framework. It can control, e.g., pause, stop, and restart running and scheduled workflows. Another task is the administration of the security, e.g., adding or removing users, map them to roles and change privileges.

Workflow Repository An important part of the framework poses the Workflow Repository. It holds all available workflows as well as additional meta data, such as workflow version, previous versions, descriptions, productivity indicators (valid, broken, analyzed, and so on) By using a repository the history and evolution of a workflow can be captured. In addition it is necessary from the provenance and reproducibility point of view to have access to used workflows even though there might be a newer and improved version of that workflow.

Plug-in Provider Any implementation for the *Plug-in Provider* extension point has to provide the management of the plug-ins installed for WORMS. So the *Plug-in Provider* is a plug-in system abstraction. In our implementation of WORMS the *Plug-in Provider* is implemented for the plug'n simulate mechanism of JAMES II. However, to work with a M&S software using, e.g., OSGi (Alliance 2003) a specific *Plug-in Provider* based on OSGi is needed. Thus instead of using JAMES II, WORMS can work with any other M&S software willing to integrate workflows, documentation, and provenance into their product.

WORMS additionally provides an internal plug-in mechanism which is used to handle the plug-ins provided with WORMS such as basic converters, executors and datastores. The *Plug-in Provider* acts in combination with the internal mechanism to provide plug-ins that are not part of WORMS, e.g., plug-ins that are M&S system specific. The internal mechanisms also works as fallback solution in case a M&S system does not use a plug-in system.

Table 1: Mapping of the parts of WORMS to requirements.

Requirement	**Components**
Scalability	*Data Store*, *Workflow Executor*, *Converter*
User-interaction	*Workflow Executor*, *Security and User/Role Management*, *Administration*
Detached execution	*Workflow Executor*, *Data Store*, *Security and User/Role Management*, *Workflow Repository*
Reliability & fault-tolerance	*Monitoring*, *Data Store*, *Analysis*
Data Provenance	*Monitoring*, *Data Store*
Smart reruns	*Monitoring*, *Workflow Executor*
Workflow Monitoring & Control	*Administration*, *Monitoring*
Workflow Roles	*Security and User/Role Management*
Collaborations & Delegation	*Workflow Executor*, *Data Store*, *Converter*
System Stability	*Workflow Engine*, *Workflow Executor*, *Monitoring*, *Analysis*, *Data Store*
Data Security	*Data Store*, *Security and User/Role Management*

3.4 M&S workflow requirements and WORMS

Table 1 gives a quick overview on which part of WORMS helps to address which of the requirements of a workflow system for M&S.

Our current implementation of WORMS does not yet provide implementations of all the functionalities described above. We currently focus on the execution and documentation of M&S workflows, and the support of different user roles, i.e., modeling expert, simulation expert and validation expert. So right now, e.g., neither specific security measures nor a distributed execution (e.g., by providing different executors) are supported. WORMS comes with a *Workflow Executor* implementation supporting local execution of workflows, LoLA (Schmidt 2000) integration for *Analysis*, a file and database based *Security and User/Role Management* implementation, a memory based *Data Store*, a file based *Workflow Repository*, a basic XPDL and BPMN *Converter* and a *Monitoring* implementation that tracks the execution of a workflow.

Additionally through the integration into JAMES II as described in Section 3.5, a *Plug-in Provider* for JAMES II's plug-in system and a special *Monitoring* implementation, that records JAMES II specific information for each workflow step exist.

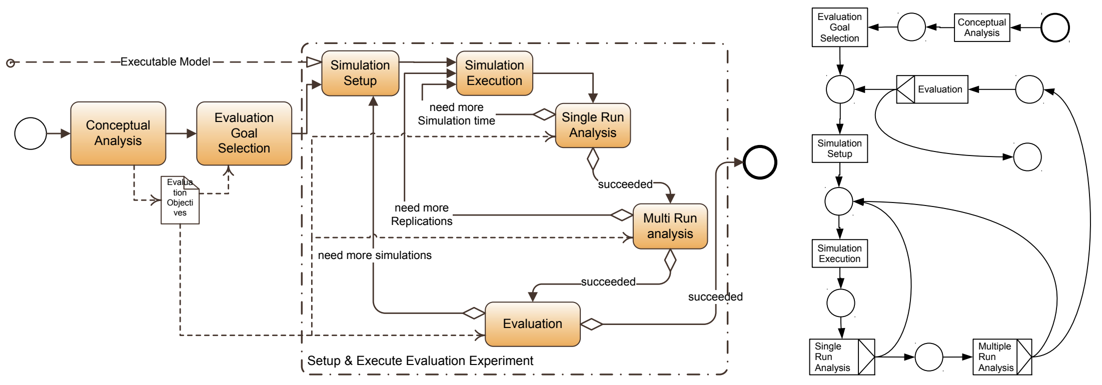
Having these implementations available in WORMS the requirements *Data Provenance, User-interaction, Workflow monitoring & control* and *Workflow Roles* can already be met. *System Stability* and *Reliability & fault-tolerance* are met partially, another *Data Store* not relying entirely on the system memory would be needed here.

3.5 Integration into JAMES II

The experimentation layer of JAMES II has been designed to incorporate many different techniques usable/used for experiments with models (Himmelspach et al. 2008, Ewald et al. 2008). From a software engineering point of view the experimentation layer can be described as an implementation of the skeleton pattern (Gamma et al. 1995).

The experiment skeleton defines the fixed workflow for experiments JAMES II can support (which include parameter scans, optimization, and validation experiments). Although the experimentation layer supports all these (and although you can use any of the alternatives available per option) the current implementation has a number of drawbacks:

- The fixed execution strategy prevents an easy extension for unforeseen experiments.



(a) Workflow described in BPMN.

(b) Workflow converted to Workflow Net.

Figure 2: Workflow for the Evaluation of Simulation Models in JAMES II.

- The experiment is a black box from the point of view of the workflow for the M&S study it is used in.
- Documentation features have to be integrated into the current experimentation layer based on the skeleton and they might not fit too well to the corresponding features of the enclosing workflow environment.
- Detailed control and resilience functions and state views have to be implemented solely for the experimentation layer.
- The experiment workflows cannot be easily stored using one of the emerging repositories for workflows (Roure et al. 2008).
- The initialization of the experiment has to be done via a single interface (containing lots of methods, hard to adapt).
- There is no support for roles - thus anyone having access to an instance of the experiment has full access.

By replacing the existing experimentation layer with WORMS and the definition of the experiment skeleton with a workflow, the above drawbacks are addressed. Little effort had to be made to integrate WORMS into JAMES II, only a few JAMES II specific components had to be created others have been reused without adaption. The following plug-ins were created specifically for JAMES II and basically represent what has to be created for other M&S systems when integrating this framework.

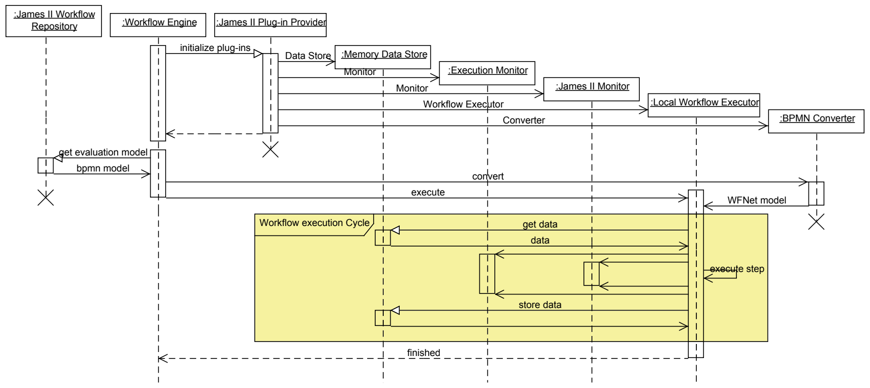
The first plug-in that needed to be realized was the implementation of a *Plug-in Provider* for the plug'n simulate architecture of JAMES II. This way it is possible to add additional components for the framework using JAMES II.

Secondly a *Monitoring* component had to be created to record system specific information mandatory for ensuring reproducibility and provenance. This component provides information which JAMES II plug-ins (with their parameters) were used during each workflow step, e.g., the used random number generator and its seed.

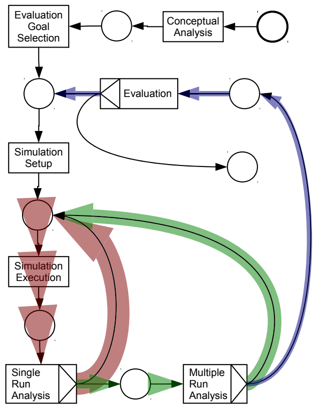
Thirdly the workflows itself had to be created and specified according to JAMES II because each task in the workflow has to map to a task or function in JAMES II.

4 MODEL EVALUATION EXPERIMENT

To give a better understanding of how the framework works in JAMES II, the following model evaluation experiment is shown. As test case we employ a model that describes spatial membrane dynamics. The aim of the biological model is to study the diffusion of membrane integral proteins (proteins attached to the plasma membrane) with respect to lipid rafts dynamics.



(a) Steps and Components within the framework during evaluation.



(b) Report for evaluation experiment.

Lipid rafts are specific membrane structures, that affect the diffusion of membrane integral proteins. Therefor their spatial dynamics is of peculiar interest for biologists. Based on the simulation study by Nicolau et al. (2006), a simplified membrane model was implemented using a cellular automaton. For the given simulation study the main objective is to determine protein collision rates with respect to specific lipid raft characteristics. However, therefore different sets of lipid raft parameters, including raft size, fluidity (i.e., diffusion coefficients of proteins within lipid rafts) or affinity (i.e., rejection probabilities of proteins for entering and leaving lipid rafts) have to be defined as well as global parameters like membrane size or protein/raft concentration. Further the objective includes simulation constraints like confidence and accuracy, that determine the number of time steps (simulation time), such that the steady state is reached and the number of replications, such that certain accuracy is given, respectively.

For the evaluation experiment the evaluation workflow in Figure 2 is used. The execution of the workflow with respect to WoRMS is depicted in Figure 3(a). After initializing the needed components provided by the *Plug-in Provider*, the workflow shown in Figure 2(a) and provided by the *Workflow Repository* has to be converted into the intermediate representation that all the other components understand. This is achieved by using a BPMN converter. Figure 2(b) shows the resulting workflow net which is then passed to the *Workflow Executor*, for execution. It follows a loop of steps within the *Workflow Executor* involving the *Data Store* to store output and load input data for each workflow step, executing the workflow step, as well as the notification of available *Monitoring* components, which record execution data (“Execution Monitor”) and simulation specific data (“JAMES II Monitor”). The “Execution Monitor” collects information such as start and end time of the current workflow step, which *Workflow Executor* and *Data Store* was used, which workflow step was executed and what was the execution status, e.g., successful, with error or aborted. This way a trace of the executed workflow is constructed. Additionally the “JAMES II Monitor” collects M&S specific information. In this case, this would be the defined objectives, executed parameter configurations, which include used random number generators and seeds, used models, used simulation algorithm, and their parameters as well as results of the analysis steps.

After reaching the end of the workflow execution the *Workflow Executor* finishes, notifies the *Workflow Engine*, and the experiment finishes.

With only the evaluation results it is not possible to reproduce them and it would not be possible to compare the generated results with results generated with, e.g., another version of the model. By employing WoRMS and its monitoring capabilities it is easy not only to reproduce the experiment but also to envision it. For instance as shown in Figure 3(b) it is possible to trace a workflow execution and to visualize how often a certain path was taken. The red path shows the simulation step cycle, the green path represents the replication cycle and the blue path is responsible for the different parameter configurations to evaluate. A path indicates the average cycle count by its width, e.g., that there were in average more simulation steps than replications and more replications than parameter configurations. Additionally more information regarding the accuracy and confidence of the generated evaluation results is available provided by the "JAMES II Monitor".

There is even more information directly extractable from the data provided by WORMS, e.g., it is possible to create statistics of used simulation steps, replications for each parameter configuration or whether a specific simulation algorithm or random number generator was used.

5 SUMMARY

Motivations and concepts behind WORMS, a framework that supports the use of workflows in M&S software, have been presented. WORMS pursues a plug-in based design. A strict separation of concerns makes different parts of WORMS responsible for achieving different requirements for workflow support in M&S, e.g., *Monitoring* and *Data Store* to provide *Data Provenance* and the *Workflow Executor* and *Security and User/Role Management* for *User-interaction* whereas the latter is also responsible for supporting *Workflow Roles*. By using workflow nets as intermediate workflow representation and providing converter capabilities for other representations WORMS provides integrated means for verifying and analyzing workflows, which supports the creation and redefinition of workflows which in return will lead to higher quality workflows for M&S.

Thus WORMS on the one hand supports and assists the creation and redefinition of workflows and on the other hand it presents an architecture which is used to execute M&S workflows which guides and assists M&S scientist and produces reproducible and documented results.

The current implementation is only a prototype and needs to be refined and further tested in diverse case studies. Thereby, we will generate the basis of a workflow repository. After that the next step will be to integrate WORMS into other M&S software and thus to prove WORMS reusability and the value of the workflows defined. Given the set of defined workflows for M&S it will be also interesting to use an existing workflow management system and replace the current *Workflow Engine* and evaluate both approaches. Furthermore the support for adaptive workflows is an ongoing research field (Klein et al. 2000, van der Aalst et al. 2000, Rahaman et al. 2009). Future work will evaluate whether adaptive workflows are a valuable way of describing workflows in M&S and whether they can be integrated and supported with the current framework.

The analysis and interpretation of and report generation from information generated by the *Monitoring* components and the evaluation and investigation of their visual presentation is also a challenging part of future work.

ACKNOWLEDGMENTS

This research is supported by the DFG (German Research Foundation).

REFERENCES

Accorsi, R., and C. Wonnemann. 2010. "Auditing Workflow Executions against Dataflow Policies". In *Business Information Systems*, edited by W. Abramowicz, R. Tolksdorf, W. Aalst, J. Mylopoulos, N. M. Sadeh, M. J. Shaw, and C. Szyperski, Volume 47 of *Lecture Notes in Business Information Processing*, 207–217. Springer Berlin Heidelberg.

Alliance, O. 2003. *Osgi Service Platform, Release 3*. IOS Press, Inc.

Balci, O. 1990, December. "Guidelines for successful simulation studies (tutorial session)". In *Proceedings of the 1990 Winter Simulation Conference*, edited by O. Balci, R. P. Sadowski, and R. E. Nance, 25–32. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Dalle, O., J. Ribault, and J. Himmelspach. 2009, December. "Design considerations for m&s software". In *Proceedings of the 2009 Winter Simulation Conference*, edited by M. D. Rossetti, R. R. Hill,

B. Johansson, A. Dunkin, and R. G. Ingalls, 944–955. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Davidson, S. B., and J. Freire. 2008, June. *Provenance and scientific workflows*. New York, New York, USA: ACM Press.

Dumas, M., and A. ter Hofstede. 2001. "UML Activity Diagrams as a Workflow Specification Language". In *«UML» 2001 — The Unified Modeling Language. Modeling Languages, Concepts, and Tools*, edited by M. Gogolla and C. Kobryn, Volume 2185 of *Lecture Notes in Computer Science*, 76–90. Springer Berlin / Heidelberg. 10.1007/3-540-45441-1_7.

Ellis, C. A., and G. J. Nutt. 1993. "Modeling and Enactment of Workflow Systems". In *Proceedings of the 14th International Conference on Application and Theory of Petri Nets*, 1–16. London, UK: Springer-Verlag.

Ewald, R., J. Himmelspach, M. Jeschke, S. Leye, and A. M. Uhrmacher. 2010, January. "Flexible experimentation in the modeling and simulation framework JAMES II–implications for computational systems biology.". *Briefings in bioinformatics* 11 (3): 290–300.

Ewald, R., J. Himmelspach, and A. M. Uhrmacher. 2008, June. "An Algorithm Selection Approach for Simulation Systems". In *2008 22nd Workshop on Principles of Advanced and Distributed Simulation*, 91–98. Rome, Italy: Ieee.

Gamma, E., R. Helm, R. Johnson, and J. Vlissides. 1995. *Design patterns: elements of reusable object-oriented software*. Addison-Wesley Professional.

Gil, Y., E. Deelman, M. Ellisman, T. Fahringer, G. C. Fox, D. Gannon, C. Goble, M. Livny, L. Moreau, and J. Myers. 2007, December. "Examining the Challenges of Scientific Workflows". *Computer* 40 (12): 24–32.

Himmelspach, J., R. Ewald, and A. M. Uhrmacher. 2008, December. "A flexible and scalable experimentation layer". In *Proceedings of the 2008 Winter Simulation Conference*, edited by S. J. Mason, R. R. Hill, L. Moench, O. Rose, T. Jefferson, and J. W. Fowler, WSC '08, 827–835. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Himmelspach, J., and A. M. Uhrmacher. 2007. "Plug'n Simulate". In *40th Annual Simulation Symposium (ANSS'07)*, 137–143: IEEE.

Hinz, S., K. Schmidt, and C. Stahl. 2005, September. "Transforming BPEL to Petri Nets". In *Proceedings of the Third International Conference on Business Process Management (BPM 2005)*, edited by W. M. P. van Der Aalst, B. Benatallah, F. Casati, and F. Curbera, Volume 3649 of *Lecture Notes in Computer Science*, 220–235: Springer-Verlag.

Hollingsworth, D. 1995. "WfMC: Workflow Reference Model". Specification, Workflow Management Coalition.

Jensen, K., L. M. Kristensen, and L. Wells. 2007, March. "Coloured Petri Nets and CPN Tools for modelling and validation of concurrent systems". *International Journal on Software Tools for Technology Transfer* 9 (3-4): 213–254.

Juric, M. B. 2006, January. *Business Process Execution Language for Web Services BPEL and BPEL4WS 2nd Edition*.

Klein, M., C. Dellarocas, and A. Bernstein. 2000. "Introduction to the Special Issue on Adaptive Workflow Systems". *Computer Supported Cooperative Work (CSCW)* 9 (3): 265–267.

Koutsonikola, V. 2004, September. "LDAP: Framework, Practices, and Trends". *IEEE Internet Computing* 8:66–72.

Kurkowski, S. 2006. *Credible Mobile Ad Hoc Network Simulation-Based Studies*. Phd thesis, Colorado School of Mines.

Law, A. M., and D. M. Kelton. 2007. *Simulation modeling and analysis*. 4 ed. McGraw-Hill International.

Ludäscher, B., I. Altintas, C. Berkley, D. Higgins, E. Jaeger, M. Jones, E. A. Lee, J. Tao, and Y. Zhao. 2006. "Scientific workflow management and the Kepler system: Research Articles". *Concurr. Comput. : Pract. Exper.* 18 (10): 1039–1065.

Mao, C. 2010, November. "Control Flow Complexity Metrics for Petri Net-based Web Service Composition". *Journal of Software* 5 (11): 1292–1299.

Merz, M. and Moldt, D. and Müller, K. and Lamersdorf, W. 1994. "Workflow Modeling and Execution with Coloured Petri Nets in COSM".

Miles, S., P. Groth, E. Deelman, K. Vahi, G. Mehta, and L. Moreau. 2008. "Provenance: The Bridge Between Experiments and Data". *Computing in Science and Engineering* 10:38–46.

Miller, S. P. and Neuman, B. C. and Schiller, J. I. and Saltzer, J. H. 1987. "Kerberos authentication and authorization system".

Missier, P., S. Embury, and R. Stapenhurst. 2008. "Provenance and Annotation of Data and Processes". Chapter Exploiting Provenance to Make Sense of Automated Decisions in Scientific Workflows, 174–185. Berlin, Heidelberg: Springer-Verlag.

Murata, T. 1989, April. "Petri nets: Properties, analysis and applications". In *Proceedings of the IEEE*, Volume 77, 541–580.

Nicolau, Dan V., J., K. Burrage, R. G. Parton, and J. F. Hancock. 2006. "Identifying Optimal Lipid Raft Characteristics Required To Promote Nanoscale Protein-Protein Interactions on the Plasma Membrane". *Mol. Cell. Biol.* 26 (1): 313–323.

Oberweis, A. and Schätzle, R. and Stucky, W. and Weitz, W. and Zimmermann, G. 1996. "INCOME/WF - A Petri Net Based Approach to Workflow Management".

Park, M.-J., and K.-H. Kim. 2010. "A Workflow Event Logging Mechanism and Its Implications on Quality of Workflows". *Journal of Information Science and Engineering* 26 (5): 1817–1830.

Pawlikowski, K., H.-D. Jeong, and J.-S. Lee. 2002. "On credibility of simulation studies of telecommunication networks". *IEEE Communications Magazine* 40 (1): 132–139.

Rahaman, M. A., Y. Roudier, and A. Schaad. 2009, September. *Document-Based Dynamic Workflows: Towards Flexible and Stateful Services*. SERVICES-2 '09. Washington: IEEE.

Rodríguez, A., E. Fernández-Medina, J. Trujillo, and M. Piattini. 2011, February. "Secure Business Process Model specification through a UML 2.0 Activity Diagram Profile". *Decision Support Systems* 51 (3): 446 – 465.

Roure, D. D., C. Goble, J. Bhagat, D. Cruickshank, A. Goderis, D. Michaelides, and D. Newman. 2008. "myExperiment: Defining the Social Virtual Research Environment". In *4th IEEE International Conference on e-Science*, 182–189: IEEE Press.

Russell, N. C., W. M. P. van der Aalst, and A. H. M. ter Hofstede. 2009. "Designing a workflow system using coloured petri nets". *Lecture Notes in Computer Science : Transactions on Petri Nets and Other Models of Concurrency III* 5800:1–24.

Rybacki, S., J. Himmelspach, E. Seib, and A. M. Uhrmacher. 2010, December. "Using workflows in M&S software". In *Proceedings of the 2010 Winter Simulation Conference*, edited by B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesan, 535–545. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Sargent, R. G., R. E. Nance, C. M. Overstreet, S. Robinson, and J. Talbot. 2006, December. "The simulation project life-cycle: models and realities". In *Proceedings of the 2006 Winter Simulation Conference*, edited by L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, 863–871. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Schmidt, K. 2000. "LoLA A Low Level Analyser". In *Application and Theory of Petri Nets 2000*, edited by M. Nielsen and D. Simpson, Volume 1825 of *Lecture Notes in Computer Science*, 465–474. Springer Berlin / Heidelberg. 10.1007/3-540-44988-4_27.

van der Aalst, Wil M. P. 1996a. "Structural Characterizations of Sound Workflow Nets".

van der Aalst, Wil M. P. 1996b. "Three Good Reasons for Using a Petri-net-based Workflow Management System".

van der Aalst, W. M. P., T. Basten, H. M. W. Verbeek, P. A. C. Verkoulen, and M. Voorhoeve. 2000. *Adaptive Workflow*, 63–70. Kluwer Academic Publishers.

van der Aalst, W. M. P., and K. M. van Hee. 2004. *Workflow Management: Models, Methods, and Systems*, Volume 1 of *MIT Press Books*. The MIT Press.

van der Aalst, W. M. P., K. M. van Hee, and Houben. 1994. “Modelling workflow management systems with high-level Petri nets”. In *Proceedings of the second Workshop on Computer-Supported Cooperative Work, Petri nets and related formalisms*, 31–50.

Verbeek, E., and W. M. P. V. D. Aalst. 2000. “Woflan 2.0: a Petri-net-based workflow diagnosis tool”. In *Proceedings of the 21st international conference on Application and theory of petri nets*, 475–484.

Weiŝbach, M., and W. Zimmermann. 2010. “Termination analysis of business process workflows”. In *Proceedings of the 5th International Workshop on Enhanced Web Service Technologies*, WEWST ’10, 18–25. New York, NY, USA: ACM.

WfMC 2008. “Workflow Management Coalition Workflow Standard Process Definition Interface – XML Process Definition Language”.

White, S. A. 2004. “Workflow Patterns with BPMN and UML”. *IBM January*.

Wong, S. C., S. Miles, W. Fang, P. Groth, and L. Moreau. 2005. “Validation of E-Science Experiments using a Provenance-based Approach”. In *4th UK e-Science All Hands Meeting (AHM)*.

Wynn, M. T., H. Verbeek, W. M. van der Aalst, A. H. ter Hofstede, and D. Edmond. 2007. “Business Process Verification - Finally a Reality!”. *Business Process Management Journal*.

Zha, H., W. M. P. van der Aalst, J. Wang, L. Wen, and J. Sun. 2010, February. “Verifying workflow processes: a transformation-based approach”. *Software & Systems Modeling* 10:1–12–12.

Zinn, D., S. Bowers, T. McPhillips, and B. Ludäscher. 2009, November. *Scientific workflow design with data assembly lines*. New York, New York, USA: ACM Press.

AUTHOR BIOGRAPHIES

STEFAN RYBACKI holds a diploma in Computer Science from the University of Rostock and pursues a PhD at the modeling and Simulation Group at the University of Rostock. His main research interests are the pros and cons of workflows in the realm of M&S and how to leverage and integrate them in M&S Software. His email address is stefan.rybacki@uni-rostock.de.

JAN HIMMELSPACH is a post doc in the Computer Science Department at the University of Rostock. He received his doctorate in Computer Science from the University of Rostock. His research interest is on software engineering for modeling and simulation, credibility of modeling and simulation, and on efficient modeling and simulation solutions. His e-mail address is jan.himmelspach@uni-rostock.de

FIETE HAACK holds a MSc in Bioinformatics from the Free University of Berlin and pursues a PhD at the modeling and Simulation Group at the University of Rostock. His main research interests are the modeling and simulation of biological processes at the cell membrane. His email address is fiete.haack@uni-rostock.de.

A. M. UHRMACHER is a Professor at the Department of Computer Science at the University of Rostock and head of the modeling and Simulation Group. Her research interests are in modeling and simulation methodologies and their applications. Her e-mail address is lin@informatik.uni-rostock.de.