Process-aware information system development for the healthcare domain : consistency, reliability and effectiveness

Citation for published version (APA):
Mans, R. S., Aalst, van der, W. M. P., Russell, N. C., Bakker, P. J. M., & Moleman, A. J. (2010). Process-aware information system development for the healthcare domain : consistency, reliability and effectiveness. In S. Rinderle-Ma, S. Sadiq, & F. Leymann (Eds.), *Business Process Management Workshops (BPM 2009 International Workshops, Ulm, Germany, September 7, 2009. Revised Papers)* (pp. 635-646). (Lecture Notes in Business Information Processing; Vol. 43). Springer. https://doi.org/10.1007/978-3-642-12186-9_61

DOI:
10.1007/978-3-642-12186-9_61

Document status and date:
Published: 01/01/2010

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Process-Aware Information System Development for the Healthcare Domain - Consistency, Reliability, and Effectiveness

R.S. Mans[1,2], W.M.P. van der Aalst[1], N.C. Russell[1],
P.J.M. Bakker[2], and A.J. Moleman[2]

[1] Department of Information Systems, Eindhoven University of Technology,
P.O. Box 513, NL-5600 MB, Eindhoven, The Netherlands
{r.s.mans,w.m.p.v.d.aalst,n.c.russell}@tue.nl
[2] Academic Medical Center, University of Amsterdam, Department of Quality
Assurance and Process Innovation, Amsterdam, The Netherlands
{p.j.bakker,a.j.moleman}@amc.uva.nl

Abstract. Optimal support for complex healthcare processes cannot be provided by a single out-of-the-box Process-Aware Information System and necessitates the construction of customized applications based on these systems. In order to allow for the seamless integration of the new technology into the existing operational processes of a healthcare organization, ensuring the correct operation and reliability of the developed system are of the utmost importance. This paper proposes an approach in which the same model is used for specifying, developing, testing and validating the operational performance of a new system. The benefits of using the same model for different purposes are decreased potential for loss of user requirements and increased confidence in *reliability* and *correct operation* of the resultant system before its deployment. This approach has been applied to a schedule-based workflow system developed for the AMC hospital in Amsterdam.

Keywords: Workflow management, scheduling, testing, simulation.

1 Introduction

In healthcare organizations there is increasing pressure to improve medical and organizational efficiency and effectiveness. The overall goal being to provide the highest quality services at the lowest cost. Therefore, more attention is given to the monitoring and control of healthcare processes.

Process-Aware Information Systems (PAIS), which are software systems that operate on the basis of an underlying process model, present an attractive vehicle for the support and monitoring of healthcare processes. However, healthcare processes tend to be both complex and of lengthy duration. Consequently, the application of an out-of-the-box PAIS system does not directly deliver the required benefits. Additional functionality needs to be developed in order to satisfy the specific needs imposed by the healthcare domain. Hence, a combination of

S. Rinderle-Ma et al. (Eds.): BPM 2009 Workshops, LNBIP 43, pp. 635–646, 2010.

different technologies need to be applied or developed in order to be able to deliver this additional functionality.

For the successful application of PAIS technology in the healthcare domain it is vital to precisely identify the required additional functionality and that a system is designed which addresses the existing needs in the healthcare domain. Unfortunately, although a candidate system may have been carefully designed, there still exists a "gap" between its deployment in a healthcare organization. This can easily be understood by the fact that healthcare processes are patient-centric, critical processes for which continuous operation must be guaranteed under all circumstances. Clearly, the introduction of a new technology system requires a seamless integration with the running operational processes of the hospital and no unexpected break-downs may occur. Additionally, it needs to be ensured that user expectations are met.

To address this "gap", we present an approach in which the same model is used for specifying, developing, testing and validating the operational performance of a new system. The different steps in this approach are as follows. First, during the *design* phase, a conceptual model is defined which is a *complete* and *formal* (i.e. executable) specification of the system to be developed. This model serves as a specification for the development of the system during the *implementation* phase. Finally, during the *testing* and *simulation* phase, the conceptual model and operational system are used to both test and validate the operational performance of the system.

The key characteristics of our approach are the *reuse* of the same model throughout the entire development process which *ensures there is minimal potential for loss of user requirements.* In addition, the approach leads to increased confidence in the *reliability* and *correct operation* of the resultant system which can be viewed as minimal requirements that need to be fulfilled before the deployment of such a system in a healthcare organization.

In this paper, we show the applicability of our approach in the healthcare domain. This is illustrated in the context of a schedule-aware workflow management system (WfMS), i.e. a particular type of PAIS, developed in conjunction with the Academic Medical Center (AMC) hospital, a large academic hospital in the Netherlands. In Section 2 we introduce both our conceptual model and a schedule-aware WfMS, and focus on its realization. In sections 3 and 4, we discuss the testing and validation of the operational performance of the resultant system. Finally, Section 5 discusses related work and Section 6 concludes the paper.

2 Schedule-Aware Workflow Management Systems

Deadlines and temporal constraints play an important role in the healthcare domain [7]. For example, many tasks are linked to appointments, e.g., a doctor cannot perform surgery without reserving an operating theater and making sure that the patient is present. Therefore, in this section, we focus on the design and implementation of a *schedule-aware WfMS*. First, some necessary concepts will be introduced. Afterwards, we elaborate on the conceptual model developed and

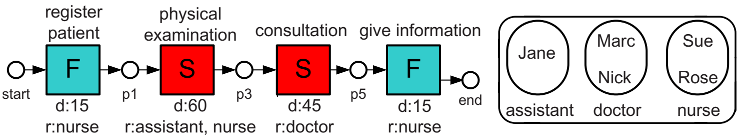

Fig. 1. Running example showing schedule (S) and flow (F) tasks. The prefix "d:" indicates the average time needed for performing the task and prefix "r:" indicates which roles are necessary for performing the task. For both schedule tasks, the patient is also required to be present.

discuss its functionality and how it has been realized. It is assumed that the reader is familiar with basic workflow management concepts, such as case and role [2].

2.1 Concepts

Figure 1 outlines a small example process for patient diagnosis. First the patient is registered by a nurse (task "register patient"). Then, the patient has an appointment with both an assistant and a nurse to check the physical condition of the patient (task "physical examination"), followed by an appointment with a doctor (task "consultation") to discuss the diagnosis. Finally, a nurse provides additional information to the patient (task "give information").

In this figure, we make a distinction between two kinds of tasks. The tasks labeled with an "F" in the figure are called *flow* tasks and can be performed *at an arbitrary point in time* when an available resource can undertake them. In order to do so, these flow tasks are presented in an ordinary *worklist* where a resource can start working on them at a time of their choosing. Only one role is defined for them as only a single resource is required to perform the task.

The other kind of tasks are called *schedule* tasks and are indicated by an "S" in the figure. They are performed by *one or more resources at a particular time*. Because such tasks are planned more than one role may be specified for this kind of task where for each role only *one* resource is involved in the actual performance of the task. Note that for a schedule task, the presence of the patient also might be required. The patient is considered to be a *passive* resource who should be present when the task is completed.

Each resource has its own *calendar* containing the appointments that have been made for schedule tasks. Where an appointment involves multiple resources, it is shown in the calendar for each of these resources. When determining the earliest time that an appointment can be started and the length of the appointment itself, the average duration of each task needs to be known. Despite this, sometimes appointments still need to be rescheduled because of anticipated delays in preceding tasks.

2.2 Conceptual Model

The main innovation of the system developed for the AMC is the incorporation of scheduling functionality. Therefore, the next challenge is to identify before the

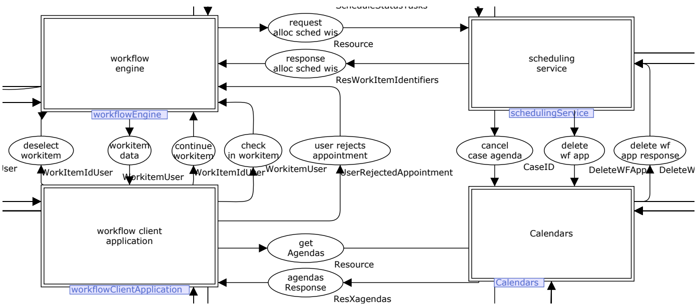

Fig. 2. The topmost level of the conceptual model realized in terms of CP Nets

implementation phase, how the new scheduling functionality being added should be integrated with existing workflow and calendar-based functionality. Therefore, *Colored Petri Nets (CPNs)* [10] have been chosen as the mechanism to identify and formalize the behavior of the system. CPNs provide a well-established and well-proven formal language suitable for describing the behavior of systems exhibiting characteristics such as concurrency, resource sharing, and synchronization.

In Figure 2 we see the topmost net of the conceptual model for the schedule-aware WfMS. Each substitution transition (represented as a rectangle) represents a component in the system. The places between two components (represented as circles) specify the interfaces between them. In total, the developed conceptual model consists of 30 nets, 250 transitions, and 634 places.

One of the main benefits of building a CPN model is that *experimentation* is possible. So, the model (or parts of it) can be executed, simulated and analyzed which leads to insights about the design and implementation of the system. Moreover, as can be seen in Figure 2, the conceptual model consists of several components. This allows for the incremental mapping of the model to an operational system mainly using available, third-party software.

2.3 Architecture

As can be seen in Figure 2, the resultant system consists of four main components. The functionality of each of them will now be explained together with a description of how they have been realized.

- **Workflow Engine:** The workflow engine is the "core" of the system and provides the standard facilities typical of this type of software [2]. For example, the engine takes care of the distribution of workitems and tracking their progress.

  Implementation: The engine is realized using both the open source WfMS YAWL [1] and a service which acts as an adaptor between the workflow client application and the scheduling service. The communication with these two components is based on the interchange of SOAP messages.

- **Scheduling Service:** The scheduling service provides the work scheduling facilities required by the system. Once a scheduling problem is received from the engine, it determines whether schedule tasks need to be (re)scheduled. Each scheduling problem is handled on a case-by-case base. A scheduling problem is represented as a graph which contains all the scheduling constraints for a case which are imposed by the engine (e.g. the ordering of tasks in the corresponding process definition for the case and the current state of the case).

  When scheduling appointments in a case, several issues are important. First, the appointments made for the (re)scheduled tasks need to be in the same order as they occur in the corresponding process definition and there should be sufficient time between them. Second, for the actual scheduling of an appointment multiple roles can be specified for a schedule task. For each role specified, precisely one resource needs to be selected and the corresponding appointment is booked in the calendar of these resources. If the patient also needs to be present, then this must also be taken into account.

  Based on these requirements several scheduling strategies for the booking of an appointment are possible. For example, one approach could be to search for the first opportunity where for each role specified, one resource is available.

  Implementation: The scheduling service is implemented in Java as an AXIS2 service. The communication with the calendar component takes place via a Java interface which exchanges information with this component.
- **Workflow Client Application:** the workflow client application offers a means of showing distributed workitems to a user. In the worktray, workitems corresponding to flow tasks can be seen. If a user is involved in the performance of a schedule task then the corresponding appointment can be found in their calendar. Once a workitem becomes available for the appointment, so that the task is ready to be executed, the corresponding workitem is completed via the calendar. For any appointments made in this way, a user can request their rescheduling.

  Implementation: A Microsoft Outlook 2003 client has been configured which acts as a full workflow client application. It provides both a view on a user's calendar and a view on the user's worktray.
- **Calendar:** The calendar component is responsible for providing display and manipulation facilities for user calendars. Both workflow and non-workflow related appointments can be created or deleted.

  Implementation: For this component, Microsoft Exchange Server 2007 has been selected as the system to support user calendars.

3 Testing

The complete system required the development of 13,692 lines of code (excluding pre-existing software such as YAWL, Outlook and Exchange). For the adaptor service in the workflow engine and the scheduling service, around 4.959 and

7.381 lines of code have been produced respectively. During its realization, the system or parts of it have been tested by executing a multitude of handcrafted scenarios and checking by visual inspection whether the produced output is correct. This approach revealed many programming errors. However, by nature, healthcare information systems should be highly reliable [7]. System failures will directly affect its acceptance by medical specialists. A complicating factor is that PAISs are more complex than traditional function- and data centric information systems. Obviously, a systematic approach for testing (parts of) the system is required to increase confidence in reliability of the resultant system.

3.1 Approach

The conceptual model provides a complete, formal description of the functionality of the system. As this model is executable, it also serves as a prototype implementation of the system. We can easily "replace" one or more components in the conceptual (CPN) model by its concrete implementation by making connections between the conceptual model and components in the actual system. This allows us to test the system on an incremental basis. Our focus is on the *black-box* testing of components where they can only be accessed and observed via their external interfaces. In addition, our focus is on identifying errors in the functionality of the system, also referred to as *functional testing*.

In order to fully automate the process of repeatedly testing one or more components of the realized system, we have built a test environment using Java. This environment consists of an *integration layer* which provides a connection between the CPN model and components in the actual system. Implemented components are tested by sequentially executing multiple tests. One test consists of a single, randomly generated, execution run of the conceptual model in which the implemented component(s) are tested. A test is considered to be complete if (1) an error is discovered in the implemented component or the integration layer, (2) a specified maximal number of steps have been executed in the conceptual model. For the execution of the conceptual model we use a Java interface which allows for loading and simulating CPN models created within CPN Tools.

3.2 Component(s) Testing

Once a testing framework is in place, one or more components of the implemented system can be tested. As for both the scheduling service component and the engine component, more lines of code have been developed in comparison to other components, we decided to (1) test the scheduling service component in isolation and (2) to test the workflow engine and the scheduling service together. In this way, we show that the approach works for both the testing of a single component and also when testing multiple components.

In order, to be able to run tests, the whole system first needs to be initialized. Therefore, a simple process definition has been added to the workflow engine so that cases can be started and workitems can be performed. Several users together with their corresponding calendars have also been added. In order to discover errors in the tested components and the integration layer it is determined whether during execution an exception occurs for one of them, i.e. it is tested whether a Java exception is generated. We will now discuss the different kind of errors that have been found when testing the workflow engine and the scheduling service.

- **Integration layer testing:** The integration layers between the scheduling service and the conceptual model and between the conceptual model and the workflow engine, contained in total 15 errors. All were conversion related.
- **Component testing:** For the workflow engine and the scheduling service component 12 errors have been identified. These ranged from simple coding errors to serious design flaws. For example, it was detected that a scheduling problem was sent to the scheduling service followed by a cancelation request, both for the same case, and that the latter request was handled first. This caused the scheduling service to crash. This issue could only be resolved by changing the corresponding interface.
- **Integration testing:** In total one error was identified which related to the integration of the workflow engine and the scheduling service component.
- **Conceptual model testing:** One challenge we faced was that no meaningful verification of the conceptual model was possible due to its size and complexity. Therefore, in the CPN model we added assertions to check whether certain invariants still hold. In total, 25 errors have been identified ranging from simple modeling errors to serious design flaws. These design flaws were all concurrency related and became visible as many different scenarios have been performed involving the conceptual model. For example, the state of two cases could become corrupted when checking in a workitem for each of them at the same time.

The majority of the identified errors were concurrency related. By using the CPN model for simulation, it is way easy to mimic arbitrary many user. Although our approach to testing revealed many errors which needed immediate attention, it does not guarantee that the tested components are error free as we did not actively check whether the output produced by each of the implemented components was correct. However, we discovered many errors that remained hidden during classical testing, thus increasing the confidence in the system.

4 Simulation

By testing the system more confidence has been gained and reliability increased. However, this is still not enough to know whether the system works properly. For example, in a hospital many critical patient-centric processes are carried out whose operational performance (e.g. waiting time for appointments) must not be negatively impacted by the introduction of the schedule-aware WfMS. If the operational performance is less, service levels offered to patients decrease as well. Obviously, this is not acceptable for a hospital. Therefore, we now elaborate on the use of *simulation* for assessing the impact of the system on the operational performance of the processes that it supports. This is investigated for several different configurations. Note that we use simulation for *operational decision making* and focus on the *transient* behavior of the system, also referred to as *short-term* simulation in [14].

4.1 Approach

In order to investigate the operational performance of our system, we take an existing healthcare process and consider several performance metrics specified for the process. As a candidate healthcare process, we take the diagnostic process for patients visiting the gynecological oncology outpatient clinic at the AMC hospital for which the schedule-aware WfMS was developed. This process deals with the diagnostic process that is followed by a patient, up to the point where the patient is diagnosed (see Figure 3). Given the space limitations, we only elaborate on the most important aspects of the simulation.

The whole process consists of 42 flow tasks and 6 schedule tasks. When a patient is registered, several administrative tasks need to be done before the first visit of the patient ("make conclusion" task). During such a visit, a doctor can request several diagnostic tests that are needed for the patient and for which an appointment needs to be made. These are a CT, MRI, a pre assessment test, and an examination under anesthetic which are all schedule tasks. However, it is also possible that these appointments are made at the beginning of the process, once it is known that they are needed.

In order to investigate the impact of the system on the operation performance of the healthcare process, for the period from 02-07-2007 to 19-03-2008, a group of 143 patients undertook the process. The flow tasks are performed by different actors in the process. For example, we have three nurses that perform all the tasks having prefix "N:". However, for the schedule tasks, corresponding appointments

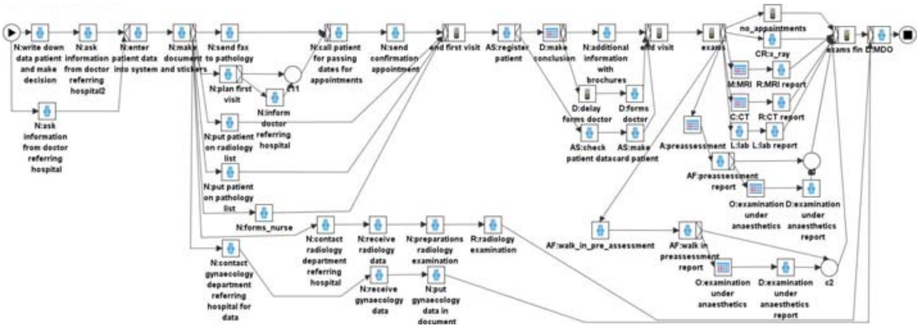
N:write down data patient and make decision
N:ask information from doctor referring hospital2
N:enter patient data into system
N:make document and stickers
N:send fax to pathology
N:plan first visit
c11
N:inform doctor referring hospital
N:call patient for passing dates for appointments
N:send confirmation appointment
end first visit
N:ask information from doctor referring hospital
N:put patient on radiology list
N:put patient on pathology list
N:forms_nurse
N:contact gynaecology department referring hospital for data
N:contact radiology department referring hospital
N:receive radiology data
N:preparations radiology examination
R:radiology examination
N:receive gynaecology data
N:put gynaecology data in document
AS:register patient
D:make conclusion
N:additional information with brochures
end visit
D:delay forms doctor
D:forms doctor
AS:check patient datacard
AS:make patient
exams
no_appointments
CR:x_ray
M:MRI
R:MRI report
C:CT
R:CT report
L:lab
L:lab report
A:preassessment
AF:preassessment report
O:examination under anaesthetics
D:examination under anaesthetics report
exams fin
D:MDO
AF:walk_in_pre_assessment
AF:walk in preassessment report
O:examination under anaesthetics
D:examination under anaesthetics report
c2

Fig. 3. Screenshot of the YAWL model showing the diagnostic process of the gynecological oncology healthcare process. The flow tasks are indicated by a person icon and the schedule tasks are indicated by a calendar icon. For all schedule tasks, the patient is required to be present.

will be made by our system. In order to ensure that the scheduling of these appointments matches reality as closely as possible, the contents of the calendars of the resources allowed to perform these tasks are based on historical data derived from the AMC electronic calendar system. These calendars are filled such that a resource is unavailable outside scheduled hours. In addition, during scheduled hours, a resource is also considered to be unavailable when a patient does not show-up for an appointment and when an appointment exists for a patient which is not in the group of patients we are considering.

One of the main challenges for the simulation is properly estimating the arrival of patients, the length of the appointments that need to be scheduled, and the selection of diagnostic tests that are chosen for each patient. As for all of these characteristics, real data exists in the AMC electronic calendar system, we decided to “replay” these events as they happened in reality. So, in the simulation, a patient arrives on exactly the same day as happened in reality, the same tests are chosen, and the length of the corresponding appointments are exactly the same as happened in reality. The same holds for a request to reschedule an appointment, triggered either by the hospital or the patient. Note that this does not imply that the simulation becomes deterministic. For example, the time spent by a resource on a flow task is determined by a stochastic distribution and the selection of a task is dependent on the availability of resources.

For the actual execution of the simulation experiments, we have used the same system configuration as has been used for testing both the workflow engine and scheduling service component in Section 3 which involves replacing both components with their implemented counterpart. Both the workflow client application and calendar component in the conceptual model have been configured such that the above mentioned aspects are included.

4.2 Results

Several sets of results have been obtained by performing various experiments. One experiment consists of 10 runs of the simulation model. A selection of these results are presented in Figure 4. Here we focus on the waiting time for the first appointment and the time between the first appointment and the CT, MRI, pre-assessment, and the examination under anesthetic respectively. For each of them, the color of the corresponding bars are indicated by the “GO”, “MRI”, “CT”, “ANS”, and “SU” text labels respectively. The average waiting times for these appointments experienced in reality are shown by the bars above the “REAL” text.

Experiment 1: Currently, for the first visit, the average waiting time experienced in reality is 11,333 minutes (7.9 days), which means that only 47% of the patients have an appointment within 7 calendar days. However, as the service level for the group of patients we are studying, it has been defined that for 90% of them, (1) the first visit should take place within seven calendar days of the registration of the patient, and (2) all diagnostic tests should be completed within 14 calendar days of their first visit. Clearly, for the average waiting time

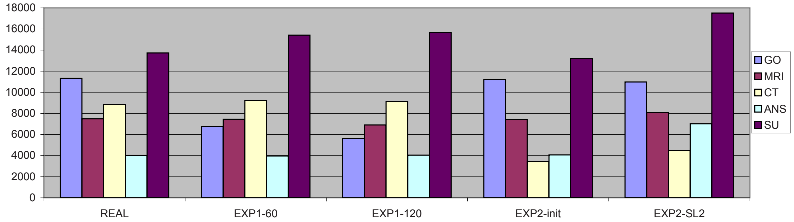

Fig. 4. Results of the experiments

for the first visit the required service level is not met. Note that for the other appointments, the required service level is met.

To examine how this situation might be remedied, the following two variations were examined: for a selected resource, every week, at the same day, an additional 60 and 120 minutes have been added for seeing patients respectively. The results of these experiments can be seen in Figure 4 by the bars above the "EXP1-60" and "EXP1-120" text respectively. Here it becomes clear that the average waiting time for the first visit already significantly drops when adding an additional 60 minutes per week for seeing patients. However, only in the situation where 120 minutes are added, is the service level met.

Experiment 2: Currently, the appointments for a CT, MRI, and pre-assessment may all be scheduled on a different day for a patient requiring multiple visits to the hospital.

In order to increase the service delivered to patients, the AMC likes to offer that the appointments for a CT, MRI, and pre-assessment are scheduled on the same day (although not when rescheduling) with a 1-4 hour gap between them. In order to be able to fully examine the impact of this rule, we simulated the situation in which an appointment for the CT, MRI, and pre-assessment is scheduled for the very first opportunity that all required resources are available. This is shown by the bars above the "EXP2-INIT" text. The bars above the "EXP2-SL" text show the results when applying this service level. Note that for both experiments a small delay of one day is added to the earliest opportunity that these appointments can be booked. In this way, the probability that these appointments need to be rescheduled is minimized which allows us to investigate the true impact of the rule as appointments are only scheduled once. When comparing the results for both experiments, it can be seen that applying this service level has quite some impact on the average waiting time for the pre-assessment and examination under anesthetic appointments. For the pre-assessment this can be explained by the fact that it is now often scheduled together with an MRI or CT test which both have a higher average waiting time. As the examination under anesthetic needs to be scheduled later than the pre-assessment test, this also explains the increased average waiting time for the examination under anesthetic appointment.

5 Related Work

The topic of appointment scheduling has received significant attention in healthcare, particularly the scheduling of appointments for outpatients. Most of this research only focuses on a single unit [9], whereas we take the scheduling of workitems into account for the whole workflow, together with its current state.

On the topic of management of time by WfMS, in [8], the satisfiability of time constraints and the enforcement of these at run-time is investigated. Another field of research is the scheduling of tasks itself [4,6]. Our approach does not present new scheduling algorithms, but instead focusses on augmenting a WfMS with scheduling facilities.

The advantages and disadvantages of using a conceptual model both for the design and testing of the same system have received quite some attention in the model-based testing literature [15]. The general thought is that although a development model typically contains too much detail for the testing phase, it does not describe the dynamic behavior well enough to enable automated test generation [15,12]. Where the dynamic behavior is described in sufficient detail it can be used for code generation. However, the goal of our conceptual model is not code generation. Rather, its level of abstraction is such that it describes precisely the requirements needed for implementing the system such that it can be concretized in many different ways.

Discrete-event simulation is often used in healthcare in order to improve efficiency and costs. However, most of these studies only focus on the analysis of individual units [11]. Our simulation considers the scheduling of appointments across several units. Within the workflow domain, the basic approach is to convert the process definition into a formal model, and then use simulation for optimization using the converted model [5]. A similar approach is described in [5] which focuses on embedding a simulation model within an existing business process management system. With regard to the use of simulation as a preliminary step for the subsequent implementation of a WfMS we are only aware of the work described in [13]. Related to this is [3], in which a business process model is constructed for requirements specification, animation, and validation of a complex software system to be built. Other than that, we are not aware of any approach which uses the same model for specifying, developing, testing, and validating the operational performance of the resultant system.

6 Conclusions

In this paper, we have presented a schedule-aware WfMS which augments a WfMS with scheduling facilities. During development, the same conceptual model is used for specifying, developing, testing, and validating the operational performance of the resultant system. For each of these steps, we have elaborated on the role and usefulness of the conceptual model and demonstrated how the overall development process is expedited through its use. In addition, the development approach pursued in this paper leads to increased confidence about the reliability and correct operation of the resultant PAIS and its ability to satisfy customer requirements. This is of the utmost importance for the deployment of the system in a healthcare organization and we believe that it leads to an increased uptake of the system by the medical specialists and the hospital.

References

1. van der Aalst, W.M.P., Aldred, L., Dumas, M., ter Hofstede, A.H.M.: Design and Implementation of the YAWL System. In: Persson, A., Stirna, J. (eds.) CAiSE 2004. LNCS, vol. 3084, pp. 142–159. Springer, Heidelberg (2004)
2. van der Aalst, W.M.P., van Hee, K.M.: Workflow Management: Models, Methods, and Systems. MIT Press, Cambridge (2002)
3. Barjis, J.: The importance of business process modeling in software systems design. Science of Computer Programming 71(1), 73–87 (2008)
4. Bettini, C., Wang, X.S., Jajodia, S.: Temporal Reasoning in Workflow Systems. Distributed and Parallel Databases 11(3), 269–306 (2002)
5. Choi, B.K., Lee, D., Kang, D.H.: DEVS Modeling of Run-time Workflow Simulation and Its Application. In: 22nd European Conference on Modeling and Simulation (2008)
6. Combi, C., Pozzi, G.: Architectures for a Temporal Workflow Management System. In: Haddad, H., Omicini, A., Wainwright, R.L., Liebrock, L.M. (eds.) Proc. of the 2004 ACM symposium on applied computing, pp. 659–666 (2004)
7. Dadam, P., Reichert, M.: The ADEPT project: a decade of research and development for robust and flexible process support: Challenges and Achievements. Computer Science - Research and Development 23(2), 81–97 (2009)
8. Eder, J., Panagos, E.: Managing Time in Workflow Systems. In: Workflow Handbook 2001. Future Strategies Inc. (2000)
9. Gupta, D., Denton, B.: Appointment scheduling in health care: Challenges and opportunities. IIE Transactions 40, 800–819 (2008)
10. Jensen, K., Kristensen, L.M., Wells, L.: 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 (2007)
11. Jun, J.B., Jacobson, S.H., Swisher, J.R.: Application of discrete-event simulation in healthcare clinics: A survey. Journal of the Operational Research Society 50, 109–123 (1999)
12. Pretschner, A., Philipps, J.: Methodological issues in model-based testing. In: Broy, M., Jonsson, B., Katoen, J.-P., Leucker, M., Pretschner, A. (eds.) Model-Based Testing of Reactive Systems. LNCS, vol. 3472, pp. 281–291. Springer, Heidelberg (2005)
13. Quaglini, S., Caffi, E., Cavallini, A., Micieli, G., Stefanelli, M.: Simulation of a Stroke Unit Careflow. In: Studies in health technology and informatics (MEDINFO), pp. 1190–1194 (2001)
14. Reijers, H.A., van der Aalst, W.M.P.: Short-Term Simulation: Bridging the Gap between Operational Control and Strategic Decision Making. In: Proceedings of the IASTED International Conference on Modelling and Simulation (1999)
15. Utting, M., Legeard, B.: Practical Model-Based Testing. Morgan Kaufman, San Francisco (2006)