From Requirements via Colored Workflow Nets to an Implementation in Several Workflow Systems

Ronny S. Mans[1,2], Wil M.P. van der Aalst[1], Nick C. Russell[1], Piet J.M. Bakker[2], Arnold J. Moleman[2], Kristian Bisgaard Lassen[3] and Jens Bæk Jørgensen[3]

[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

[3] Department of Computer Science, University of Aarhus, IT-parken, Aabogade 34, DK-8200 Aarhus N, Denmark. {krell,jbj}@daimi.au.dk

Abstract. Hospitals and other healthcare organizations need to support complex and dynamic workflows. Moreover, these processes typically invoke a number of medical disciplines. This makes it important to avoid the typical disconnect between requirements and the actual implementation of the system. In this paper we apply a development approach where an Executable Use Case (EUC) and a Colored Workflow Net (CWN) are used to close the gap between a given requirements specification and the realization of these requirements based on workflow technology. In order to do so, we describe a large case study where the diagnostic process of the gynecological oncology care process of the Academic Medical Center (AMC) hospital is used as a candidate process. The process consists of hundreds of activities. These have been modeled and analyzed using an EUC and a CWN. Moreover, based on the CWN, the process has been implemented using four different workflow systems. In this way, we demonstrate the general application of the approach and its applicability to distinct technology systems.

Keywords: Workflow Management, Executable Use Cases, Colored Petri Nets, healthcare

1 Introduction

For some time, particularly in academic hospitals, there has been the need for better support in controlling and monitoring health care processes for patients [25]. One of the objectives of hospitals is to increase the quality of care for patients [15], while in the future, an increase in the demand for care is expected.

Workflow technology presents an interesting vehicle for the support and monitoring of health care processes as it facilitates process automation by managing

the flow of work such that constituent activities are done at the right time by the proper person [4]. The advantages of successfully applying workflow technology are faster and more efficient process execution [34, 22, 14].

Typically, there is a large gap between an actual hospital process and its implementation in a workflow system. One approach to bridging this gap is to go from a real-world process, via a requirements model and a design model to an implementation in a workflow system as is described in [8, 20]. The different steps in this development approach are shown in Figure 1. First a requirements model is developed, based on a real-life case. The next phase in the process is the construction of a design model, followed by its implementation in a workflow system. The construction of the requirements and the design model is done using *Colored Petri Nets (CPNs)* [17]. CPNs provide a well-established and well-proven language with formal semantics. CPNs are particularly suitable for describing the behavior of systems requiring support for concurrency, resource sharing, and synchronization, and are therefore well suited for modeling business processes.

To be more precise, in the requirements phase, a so-called Executable Use Case (EUC) is created, a CPN model augmented with a graphical animation. The next step in the development process is to build a design model. Here a so-called Colored Workflow Net (CWN), which is also a CPN model, and which is closer to an actual implementation in a workflow system, is used. In an EUC any concepts and entities deemed relevant may be included, whereas for the CWN we are restricted to the workflow domain as only concepts and entities which are common in workflow languages may be used. Ultimately, we have the implementation in several workflow systems. The main advantages of this development process are that concerns, as shown in Figure 1, are dealt with at the right time and that the models constructed in the EUC and CWN phase are based on a formal foundation. In that way, rigor is added to the development

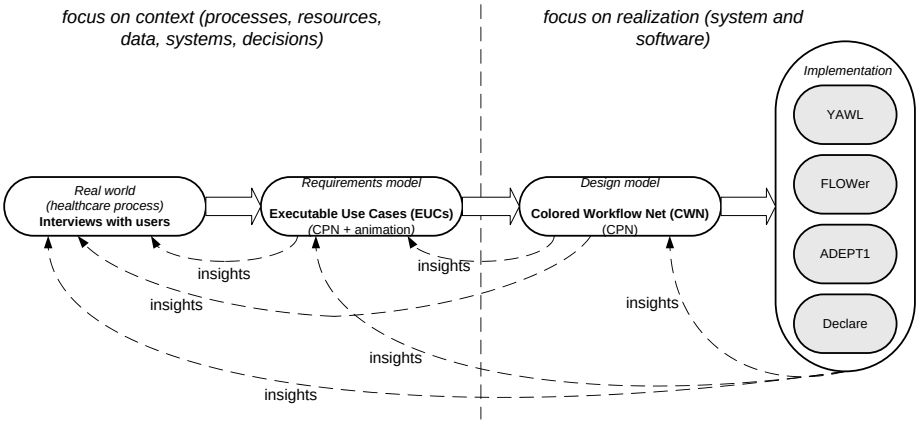

Fig. 1. Overall approach

process but is also allows for a seamless integration between the requirements and design phase.

In [8, 20] rather small cases are used, whereas real world processes typically consist of hundreds of activities and are far from trivial. Therefore, in order to investigate the *general applicability of the approach*, shown in Figure 1, it is applied to a large real world healthcare process which is non-trivial. In this way, we can also investigate whether the approach is *repeatable*.

The healthcare process that will be considered in this paper is the diagnostic process of patients visiting the gynaecological oncology outpatient clinic in the AMC hospital, a large academic hospital in the Netherlands. To give an idea about the size of the healthcare process, it should be emphasized that the EUC consists of *689 transitions, 601 places and 1648 arcs* and that the CWN consists of *324 transitions, 522 places and 1221 arcs*. This shows that the healthcare process is far from trivial.

Typical difficulties that hospitals have to cope with when they want to apply workflow technology stem from the fact that healthcare processes are *diverse*, *flexible* and that *several medical departments* can be involved in the treatment process. For example, as a consequence of the way a patient reacts to the treatment offered, and the condition of the patient themself, it may be necessary to continuously adapt the care process an individual patient [13]. This shows that flexibility is a key requirement in the healthcare domain which consequently needs to be provided by the workflow system.

The following workflow management systems were selected to implement (parts) of this process: YAWL [6], FLOWer [9], ADEPT1 [33], and Declare [29]. These systems were selected because they all provide a certain kind of flexibility, which in this context is deemed relevant[4]. In this way, it allows us to demonstrate the applicability of the development approach to *distinct technologies*. The selected systems cover various flexibility paradigms: adaptive workflow (ADEPT1), case handling (FLOWer), and declarative workflow (Declare). An additional reason for implementing a hospital process in different workflow systems is that we wanted to identify the requirements that need to be fulfilled by workflow systems, in order to be successfully applied in a hospital environment. These requirements have been discussed in [27].

There are some notable differences with the work presented in this paper and that presented in [8, 20] which also went from an informal description of a real world process, via an EUC and CWN, to an implementation in a workflow system. As already indicated, we studied an existing healthcare process of a hospital in detail, whereas in the earlier work rather small cases are used. Moreover, we developed an implementation in four different workflow systems instead of just one and systematically collected feedback from the host care organization (AMC). In [8] there was only user involvement in the EUC phase, and in [20]

[4] Of course other workflow systems could have been selected. An additional reason for choosing these systems is that they were already available to us, or could easily be obtained.

there was no user involvement at all. Note, that an earlier version of this paper has already appeared as an (informal) workshop paper [26].

This paper is structured as follows: Section 2 introduces the approach followed. Section 3 introduces the EUC and the healthcare process we studied. Section 4 discusses the CWN, which is followed by an analysis of the model in Section 5. In Section 6, the implementations in the different workflow systems are discussed. Related work is presented in Section 7. The paper finishes with conclusions in Section 8.

2 Approach

In this section, we first elaborate on the approach that has been followed, as shown in Figure 1, to go from a real-life process to its implementation in several workflow systems via an EUC and CWN. Our approach commences by interviewing users who are involved in the diagnostic part of the gynecological oncology healthcare process. In these interviews we focused on identifying the work processes that constituted the requirements for the system to be built. These requirements were captured during the requirements phase. In this phase we developed a requirements model using the EUC method. EUCs are formal and executable representations of work processes to be supported by a new IT system and can be used in a prototyping fashion to specify, validate, and elicit requirements [19]. EUCs have the ability to "talk back to the user" and can be used in a trial-and-error fashion. Moreover, they have the ability to spur communication between stakeholders [19] about issues relevant to process definition.

In our case, the EUC consists of a CPN model, which describes the real-life process, and an animation layer on top of it, which can be shown to the people involved in the process. Our main reason for using EUCs, was that we did not expect the people to understand the underlying CPN model directly but anticipated that they would be able to derive an understanding from the animations as to how we modeled their work processes. This provided a suitable opportunity for validating our model. Moreover, the use and importance of animations in the very early stages of the development process has been stressed in [30].

After validation of the requirements model, we move on to the next phase, the design phase, in which the CWN method is used. We took the underlying CPN model of the EUC and translated it into a CWN. By developing this workflow model we restricted ourselves to the workflow domain. More specifically, by creating the workflow model, we restricted ourselves to concepts and entities which are common in workflow languages. *In comparison to the EUC model, we now only use a fixed library of concepts and entities, whereas in the EUC any concept or entity deemed relevant may be used.* In addition, the CWN only contains *actions that will be supported by the new system* whereas actions that are not going to be supported by the new system, are left out. Finally, once the CWN model was finished, we used it as a basis to configure each of the four workflow systems.

As is indicated in Figure 1, during the construction of the EUC and the CWN and the associated implementations in the different workflow systems, additional insights were obtained about previous phases. So, there are often iterations involving the repetition of earlier stages of the process as shown in Figure 1. When repeating an earlier phase, the model of that phase and subsequent phases are updated until the current phase is reached again. As we only had feedback from the people involved in the process during the interview, requirements and design phases. This means that we did not receive any user feedback during the implementation phase as we only developed prototypes that were not used as fully operational production systems. Although feedback during the implementation phase could have given us further information about the process, we felt that we had already sufficiently identified the process during the construction of the EUC and CWN.

In Figure 1, there is a dashed line between the first two blocks and the last two blocks indicating a shift of focus. On the left side of the dashed line the focus is on *context*, whereas on the right side the focus is on *realization*. With *context* we mean that the focus is on processes, resources, data, systems and decisions and with *realization* we mean that the focus is on the system and software itself. To support this shift of focus, the CPN modeling language, which has been used for the EUC and the CWN, provides a smooth transition between the two foci. In addition, we believe this addresses the classical "disconnect" which exists between business processes and IT. Moreover, building an EUC and CWN allows for a *separation of concerns*. EUCs are good for capturing the requirements of a process without thinking about how it is realized and this information serves as input for a CWN. CWNs define the control-flow, resource, data and operation perspective while at the same time abstracting from implementation details and language/application specific issues[5]. Using this development process, we are sure that these concerns are dealt with at the right time as we have to deal with them anyway.

Figure 1 should not be read as if we are proposing a waterfall development process. Instead, it should illustrate that the phases are partially ordered. During the development process we can go back to a preceding phase and make changes. Consequently, these changes will result in changes in subsequent phases.

Note that in principle other process languages can also be used with this approach. We used CPN as we are familiar with the language and because they provide a well-established and well-proven formalism for describing and analyzing the behavior of systems with characteristics such as concurrency, resource sharing, and synchronization. In this way, they are well-suited for modeling workflows or work processes [4]. It is worth noting that the process-oriented nature of both the EUC and CWN limits our approach to process-oriented systems,

[5] The control-flow perspective specifies the ordering of activities in a process, the resource perspective deals with the resource allocation required for the execution of the activities within a process, the data perspective deals with data transfer between activities, and the operation perspective describes the elementary operations performed by resources and applications.

like workflow and document handling systems. Nevertheless, it is important to mention that the development approach used in this paper is in no way limited to the healthcare domain. In [8] the work process of bank advisers is considered using similar techniques.

It is important to mention that all the models and translations between models have been done manually. In the end, this leads to a *full* implementation of a *complete* healthcare process in four different workflow systems. Because of the specific nature of the EUC model and the CWN, the CWN cannot be generated automatically from the EUC. This is because certain parts of the EUC need to be refined in order to allow for system support. Depending on what is specified in the EUC, parts can be reused and refined in the CWN model. However, in principle, a (semi-) automatic translation from the CWN to each of the workflow systems is possible and examples of this are shown in [8, 20].

We already indicated that we studied a large healthcare process; for creating the EUC and the CWN more than 100 man hours were needed to develop each model. As we also needed to get acquainted with the workflow systems used, their configuration took around 240 man hours in total. Additionally, around 60 man hours were needed for interviewing and obtaining feedback from the people involved in the process.

3 Executable Use Case for the Gynecological Oncology Healthcare Process

In this section, we first introduce the gynecological oncology healthcare process which we studied. After that, we will consider one part of the healthcare process in more detail and for this part we will elaborate on how the animations have been set-up within the EUC. Moreover, we will elaborate on the experiences associated with this activity. Note that given the size of the CPN model of the EUC (689 transitions, 601 places, 1648 arcs and 2 color sets) it is only possible to show a small fragment of the overall model.

In Figure 2, the topmost page of the CPN model of the EUC is shown, which gives a general overview of the diagnostic process of the gynecologic oncology healthcare process in the AMC hospital. In the remainder of this paper, we will simply refer to the gynecological oncology healthcare process itself, instead of the diagnostic process of the gynecological oncology healthcare process.

As can be seen in Figure 2, the gynecological oncology process consists of two different processes. The *first* process, which is modeled in the upper part of the picture, deals with the diagnostic process that is followed by a patient who is referred to the AMC hospital, up until they are diagnosed. In this process, the patient can have several consultations with a doctor, either via visiting the outpatient clinic or via telephone.

During such a consultation, the status of the patient is discussed and a decision is made about whether diagnostic tests and/or further consultations need to be scheduled, canceled or rescheduled. Moreover, during the course of the

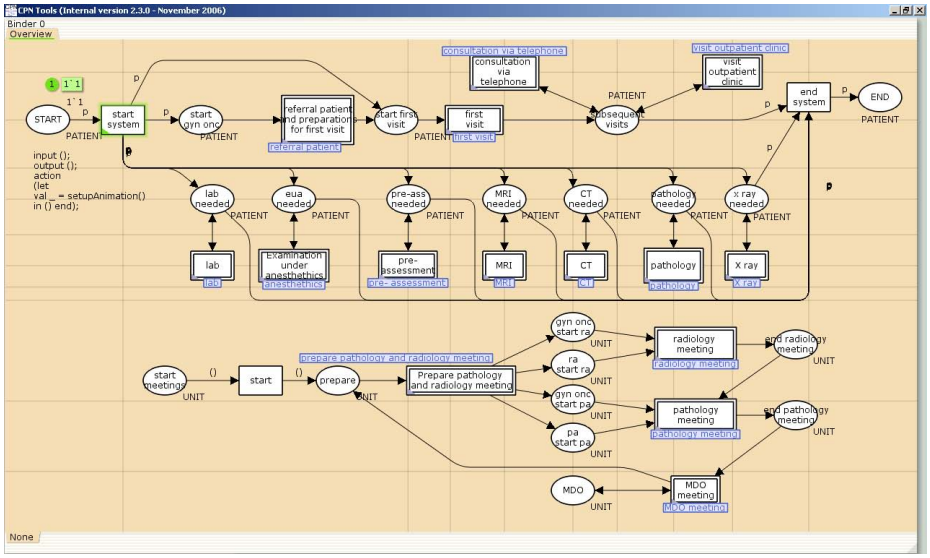

Fig. 2. General overview of the gynecological oncology healthcare process.

process, several administrative activities such as brochure recommendation and patient registration can also occur.

A doctor can request a series of different diagnostic tests, undertaken at different medical departments. The interactions with these medical departments and also the processes within these departments are modeled in the middle of Figure 2. The interactions with these medical departments are considered to be a 'black box' where a request or a cancelation of a diagnostic test is delivered and ends when a result is known or the test is canceled.

The *second* part of the process, which can be found at the lower part of Figure 2, deals with the weekly organized meetings, on Monday afternoon, for discussing the status of patients and what needs to be done in the future treatment of these patients.

Note that some connections exist between the two processes. However, as we only focus on the content and ordering of activities within one process, we did not put any effort in making these connections more explicit.

Figure 3 shows a part of the subnet for the substitution transition "referral patient and preparation for first visit" illustrating the very beginning of the process. At this time, a doctor of a referring hospital calls a nurse or doctor at the AMC followed by the necessary preparations for the first visit of the patient, like planning an MRI (transition "plan MRI").

In Figure 3, we see how the CPN model and the animation layer are related within the EUC. At the top, we see the CPN model that is executed in CPN Tools. At the bottom we see the animation that is provided within the BRITNeY

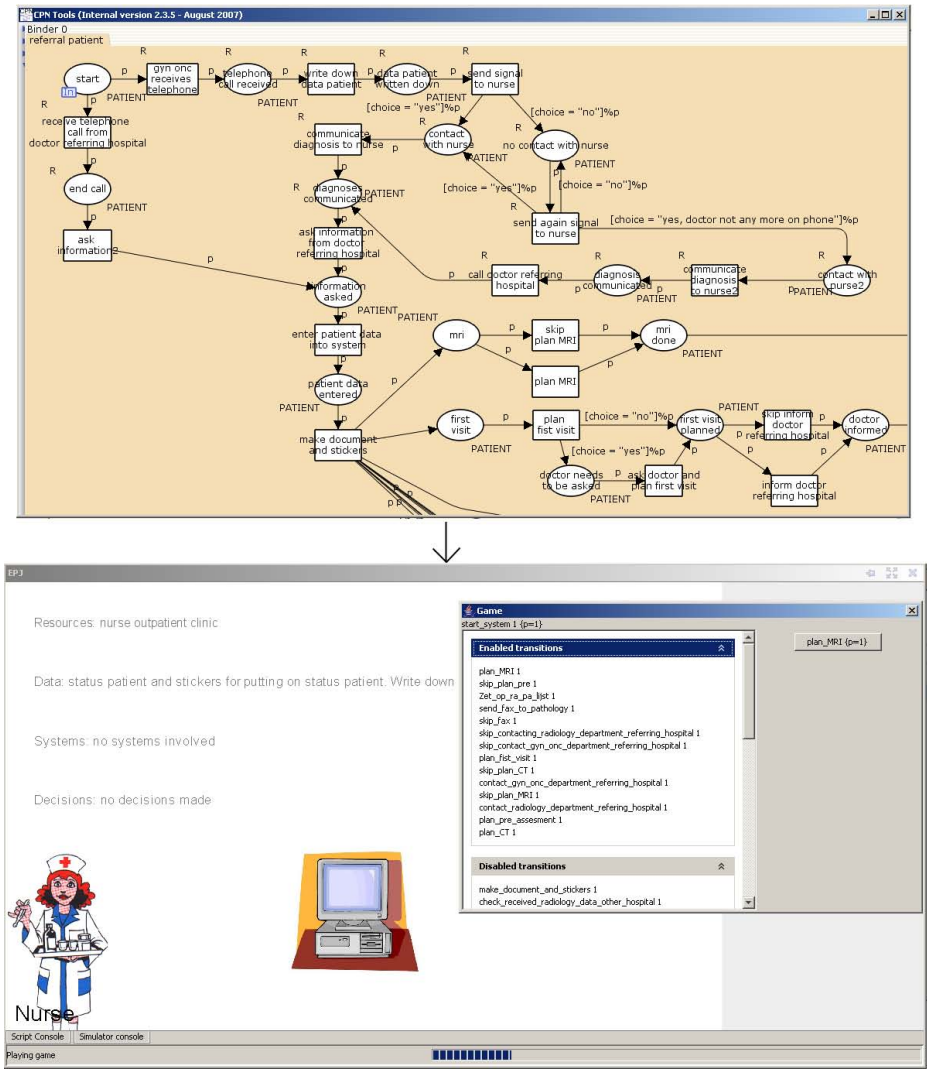

Fig. 3. Animation belonging to the "make document and stickers" activity. In addition, the panel at the top right side shows which activities are enabled now.

tool[6], the animation facility for CPN Tools. The CPN model and the animation layer are connected by adding animation drawing primitives to transitions in the CPN model, which update the animation. The animation layer shows for the last executed activity in the CPN model, which *resources*, *data* and *systems* are involved in executing the activity and it also shows which *decisions* are made during the activity. This ensures the focus is on what happens in the *context* of

[6] http://wiki.daimi.au.dk/britney/britney.wiki

the process which will also be of help when constructing the CWN in the design phase. In addition, a separate panel is shown which indicates which activities are currently enabled and may be executed. One of the enabled activities in the panel can be selected and executed, which changes the state of the process and in this way, we can directly influence the routing within a process. When an activity is executed in the CPN model it is reflected by updates to the animation layer. Consequently, the CPN model and the animation layer remain synchronized.

In Figure 3, the animation visualizes the "make document and stickers" activity. The panel at the top right side of the snapshot in Figure 3, shows which activities can be executed after the "make document and stickers" activity has been executed. It shows that the "plan MRI" activity may be executed but the "make document and stickers" activity may not. Moreover, we see that a nurse of the outpatient clinic is responsible for executing the "make document and stickers" activity and that no decisions need to be made.

We have shown the animations to the people that were involved in the gynecological oncology process. The people were very positive and indicated that through their use, they were able to check whether the process modeled in the EUC corresponded with their work process. Moreover, they provided valuable feedback for improvements, like activities that needed to be reordered. Consequently, we can say that the EUC method was helpful in validating the model and we believe that better results have been obtained than if we had simply shown the plain CPN models or process definitions of a workflow management system.

4 Colored Workflow Net for the Gynecological Oncology Healthcare Process

In this section, we elaborate on CWNs in general. After this, we consider the same part of the CWN as we did for the EUC in more detail and explain the differences. Note that also in this case, given the size of the CWN model (324 transitions, 522 places and 1221 arcs and 53 color sets) it is only possible to show a small fragment of the overall model.

As can be seen in Figure 1, both the EUC and CWN are CPN models. Remember that a CWN is a *workflow* model in which we restricted ourselves to concepts and entities which are common in workflow languages, whereas in the EUC any concept or entity deemed relevant may be used. So, the CWN needs to cover the control-flow, resource, data and operation perspective.

The syntactical and semantic requirements for a CWN have already been defined in [8]. However, when using CWNs in this paper, we have introduced some subtle changes so that they suit our needs. According to [8], a CWN should be a CPN with only places of type Case, Resource or CxR. Tokens in a place of type Case refer only to a case and the corresponding attributes (e.g. patient name, patient id), and tokens in a place of type Resource refer only to resources. Finally, tokens in a place of type CxR refer to both a case and a resource. However, we have decided to separate the case data from the case, so that case data

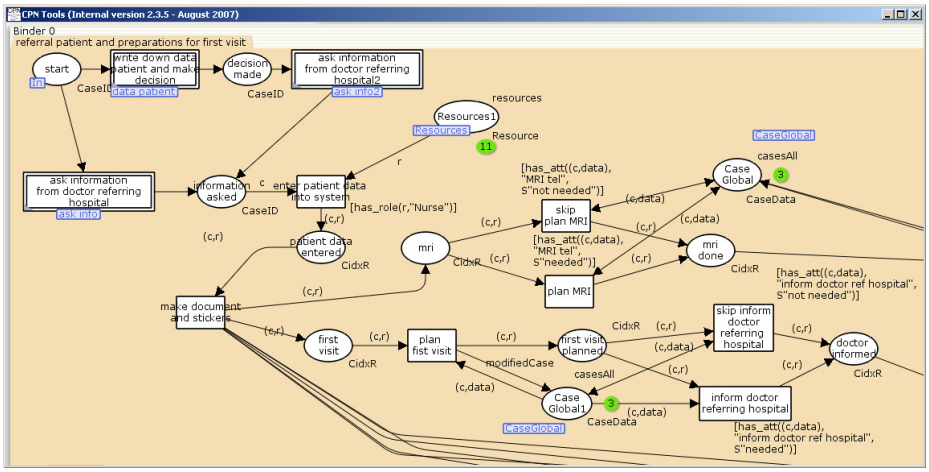

Fig. 4. CWN for the EUC shown in Figure 3.

can be accessed anywhere in the model and is always up-to-date. For example, in Figure 4, the "plan first visit" and "plan MRI" activities might be executed concurrently. In the "plan first visit" activity, some data may be written which is interesting for the "plan MRI" activity, and vice-versa. To this end, instead of places of type Case and CxR, we have places of type CaseID for storing the case identifier, CidxR to refer to both a case identifier and a resource, and CaseData for accessing the data attributes of the case. Furthermore, instead of having data attributes of which the data values may only be integers or strings, we also allow them to have a list data type.

In Figure 4, we see the CWN for the EUC CPN which has been shown in Figure 3. If we compare the CWN of Figure 4 with the EUC CPN of Figure 3, we see that there are some differences. First of all, some activities which are shown in the EUC CPN do not appear in the CWN, as they are not supported by the workflow system. The activities that will not be supported are indicated by the character "R" at the top left of each individual activity that will be removed. On the other hand, the activities of the EUC that will be supported do not have the character "R" at the top left of them. For example, the "plan MRI" activity will be supported by the workflow, whereas the "send signal to nurse" activity is not.

Moreover, the place "Resources1" is only present in the CWN as it contains information about the availability of resources which are needed for the organizational perspective of the CWN. Also, the places "Case global" and "Case global1" are only present in the CWN as they contain the corresponding data attributes for each case instance which are needed for the data perspective of the CWN. Furthermore, the guards belonging to transitions explicitly reference the resource and data perspective.

For example, the guard of the "enter patient data into system" activity indicates that this activity may only be performed by a nurse, whereas the guard of the "plan MRI" activity indicates that this activity may only be performed if data attribute "MRI tel" has value "needed".

When we compare the EUC and CWN with each other it is clear that the CWN provides a complete specification for the control flow, resource, data, and operation perspectives. As a consequence, it is more closely aligned with the way workflow systems are described because similar concepts are used. When we look at the EUC, its goal is to support specification, validation and elicitation of requirements [19]. Its use narrows the gap between informal requirements and the formalization required for supporting the process by a workflow system which on its turn is provided by the CWN. In contrast to the EUC, in the CWN certain parts are refined in order to allow for system support. For example, in the EUC a sequence of activities may be possible which is prohibited in the CWN. Clearly, when defining the control flow, resource, data, and operation perspective of the CWN, parts of the EUC may be reused and refined depending on what is already specified.

A considerable amount of time has been spent in defining both the EUC and CWN (100 man hours for each) in which several iterations, back and forth, have been made. When defining the resource, data, and operation perspective in the CWN, questions arose about the process that had been modeled in the requirements phase. This was due to the fact, that not all process details were obtained when interviewing people involved in the process and showing the animations to them. For example, in the EUC we modeled that after the appointment with the doctor appointments can only be booked with an internist and radiotherapist for treatment of the patient. However, in principle, it should be possible to make an appointment with any specialist if needed. This issue was identified during the design phase. After showing the updated animation to the doctor, it was agreed that this was a correct observation and the CWN was updated accordingly.

Because of the aforementioned reason, it is important that we can jump back to the requirements phase, adapt the model and move on to the design phase again. We believe that this leads to a more complete description of the process itself (EUC) and its implementation in a chosen workflow system (CWN). Consequently, we can say that the CWN provides a useful step towards the implementation of a process in a chosen workflow system.

5 Analysis

In this section, we will focus on the analysis of the CWN model. Within CPN Tools there are two possibilities for the analysis of a CPN model, namely, simulation/animation and state space analysis [17]. Simulation can be used to investigate different scenarios and explore the behavior of the model but does not guarantee that it is free from errors. To this end, we used state space analysis which computes the full state space of a CPN model and makes it possible to *verify*, in the mathematical sense of the word, that the model possesses a

certain formally specified property. The state space analysis of CPN Tools can handle state spaces of up to 200,000 nodes and 2,000,000 arcs [18] and provides, amongst other features, visual inspection and query functions for investigating (behavioral) properties of a model.

The primary task of a workflow management system is to enact case-driven business processes by integrating several perspectives, such as the control-flow, resource, and data perspective. One of the properties deemed to be most relevant in this context, and which can be easily checked for, is the so-called soundness property. We feel this property provides the most insights into the correct operation of a model. Moreover, the soundness property is generic and does not require process or domain specific knowledge. In [1] it is motivated why it is reasonable to abstract from the other perspectives when verifying a workflow. Verification with resources is possible (cf. [16]) but provides few surprising insights as there is little "locking of resources" in this domain. Verification with data is more problematic. As shown in [36] and other papers some verification is possible. However, analysis then only checks for the presence of data rather than the actual values. A full analysis involving data is impossible. If data is infinite, analysis is impossible (cf. halting problem). If data is finite but large, analysis is intractable because of the state-space explosion problem. Moreover, it is often impossible to model human behavior and applications completely. Therefore, we think it is justified to use the classical soundness notion. In [5] various alternative soundness notions are compared and discussed. Moreover, this paper provides a survey of analysis approaches for verifying soundness in the presence of advanced workflow patterns (e.g. cancelation).

Soundness for workflow nets is defined in [2] as: *for any case, the procedure will terminate eventually and the moment the procedure terminates there is a token in the sink place (i.e. a place with no outgoing arcs) and all the other places are empty. Moreover, there should be no dead transitions.* To check for soundness of the CWN, we need to abstract from resources. Moreover, as the CWN is exceptionally large we also need to simplify the color sets and verify things in a hierarchical manner (i.e. in a modular fashion). To be more precise, for each transition on the top page of the CWN which is linked to a subnet, we check the soundness of the subnet, including the net on the top page. Note that such a subnet can also contain subnets.

Checking the soundness of such a subnet has been done according to the following procedure. First, we focused on the individual subnet, removing all nodes and subnets from the total model which did not belong to the subnet being considered. Second, we removed all color sets and all data attributes which were not relevant for the subnet being considered. A data attribute was considered not to be relevant when there was no function in the subnet which actually accessed the data attribute. After finishing the two preceding steps, we were able to check whether the subnet was sound. For the actual check for soundness we added an extra transition $t*$ in the subnet which connected the only output place with the only input place. This is often called a *short-circuited net* [2]. According to [2],

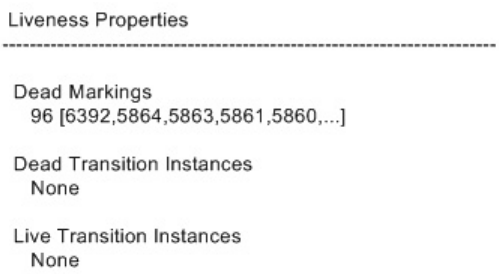

Fig. 5. Liveness Properties section of the state space report generated for the short-circuited net of the erroneous CWN. The report shows that there are multiple dead markings. Moreover, there are no dead transitions and there are no live transitions.

if the short-circuited net is live and bounded, then the original net is sound[7]. Moreover, as we are dealing with a hierarchical structure of subnets, another important requirement that needs to be satisfied for the whole net to be sound, is that all the nets need to be safe (i.e. for each place the maximum number of tokens must not exceed one). If an error was found, the subnet was adapted and again checked for its soundness. This last step was been repeated until the subnet was sound. A limitation of the approach is that a subnet which is checked for its soundness should not be too large and/or have many color sets.

For example, for the CWN, shown in Figure 4, we found that for the "skip plan MRI" activity a double-headed arc had not been used between this transition and place "CaseGlobal". The fact that there was an error could be derived from the liveness properties of the state space report of the short-circuited net which is shown in Figure 5. Informally, liveness means that all transitions in the net have always the opportunity to become enabled no matter which transitions are fired in the net. This means that there may not be any dead markings and all transitions instances in the net should be live. This requirement is not satisfied as the report indicates the existence of multiple dead markings and that no live transition instances exists. After solving the error we obtained a sound CWN consisting of 39808 nodes, 156800 arcs for which all transitions were live and there were no dead transitions.

In general, for each subnet we found one or two errors, but we also had subnets which were error-free. Although there are additional structural properties which could be checked for, we only checked for soundness. Other relevant workflow correctness criteria considering the resource and data perspective, and which can be checked for, are discussed in [36, 16]. Note that the analysis of the above mentioned properties does not remove the possibility of semantic errors.

[7] More details about liveness, boundedness, and short-circuited nets can be found in [2].

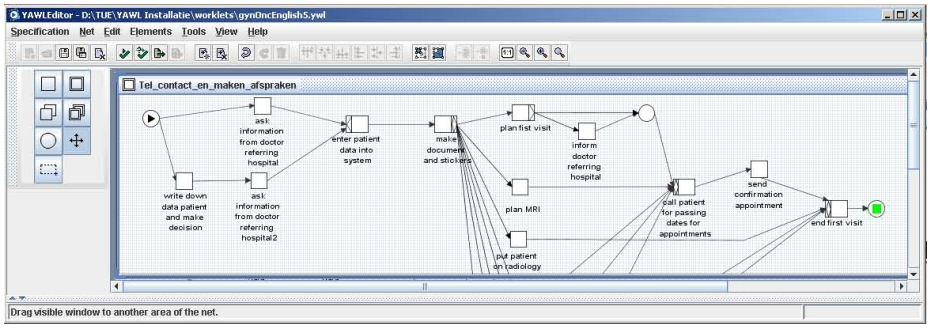

Fig. 6. Screenshot of the YAWL editor.

6 Realization of the System

In this section, we discuss how the four selected workflow systems (YAWL, FLOWer, ADEPT1 and Declare) have been configured in order to support the gynecological oncology healthcare process. For each system, we will first briefly introduce the system. Then the mapping from the CWN to the modeling language used in the workflow system itself, which is done in a manual way, is exemplified. Finally, we elaborate on the applicability of a CWN for implementation of a process in a workflow system.

6.1 YAWL / Worklets

YAWL (Yet Another Workflow Language) [6] is an open source workflow management system[8], based on the well-known workflow patterns[9] which is more expressive than any other workflow languages available today. Moreover, in addition to supporting the control-flow and data perspectives, YAWL also supports the resource perspective.

YAWL supports the modeling, analysis and enactment of flexible processes by, so called, worklets [10] which can be seen as a type of dynamically configurable process fragment. Specific activities in a process are linked to a repertoire of possible actions. Based on the properties of the case at runtime and other associated context information, an appropriate execution option is chosen. The selection process is based on a set of rules. This can be extended at runtime and it is possible to dynamically add new actions to the repertoire.

In Figure 6, we can see how the CWN of Figure 4 is mapped to the YAWL language. Given the fact that YAWL can be seen as a superset of CWNs, it was easy to translate the CWN of Figure 4 to YAWL. It was possible to directly translate the transitions into YAWL tasks. Furthermore, the places of the

[8] YAWL can be freely downloaded from www.yawl-system.com

[9] More information about the workflow patterns can be found at www.workflowpatterns.com

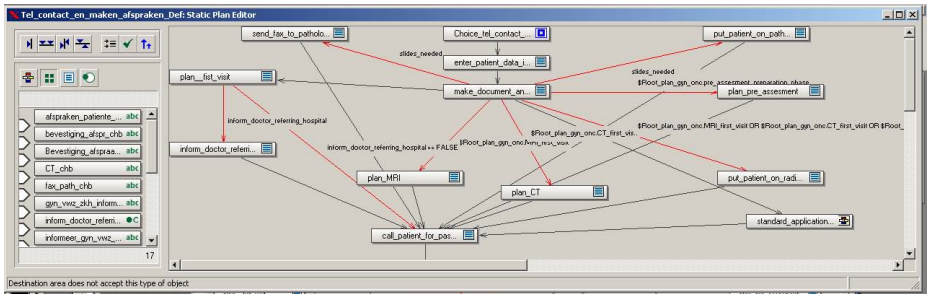

Fig. 7. Screenshot of the FLOWer editor.

CWN model can also be directly translated to YAWL conditions, but due to the syntactical sugaring inherent in YAWL there is no need to add all places as conditions in the YAWL model. For example, the "make document and stickers" activity in YAWL has an associated OR-split, after which it is possible to either plan an MRI or CT. In addition to this transformation, the worklet approach has been used as a selection mechanism for the most appropriate diagnostic test.

The resultant YAWL model consists of 231 nodes and 282 arcs and it took around 120 hours to construct a model that could be executed by the YAWL workflow engine.

6.2 FLOWer

FLOWer is a commercial workflow management system developed by Pallas Athena, the Netherlands[10]. FLOWer is a case-handling product [9]. Case-handling promotes flexibility in process enactment by focusing on the data rather than the control-flow aspects of a process.

In Figure 7, we see how the CWN of Figure 4 is mapped to the FLOWer language. In this case, it was quite easy to translate the CWN in FLOWer. In particular, it was possible to directly translate the transitions into FLOWer activities and the causal relationships could also be taken into account. In Figure 7, all nodes are activities except the "choice tel contact" node. This node represents a deferred choice which needs to be made at the beginning of the process, as can be seen in Figure 4. So, at the commencement of the process, either the "ask information from doctor referring hospital" activity can be chosen or the "write down data patient and make decision" activity can be chosen, but not both.

The resultant FLOWer model consists of 236 nodes and 190 arcs and it took around 100 hours to construct a model that could be executed by the FLOWer workflow engine.

[10] http://www.pallas-athena.com/

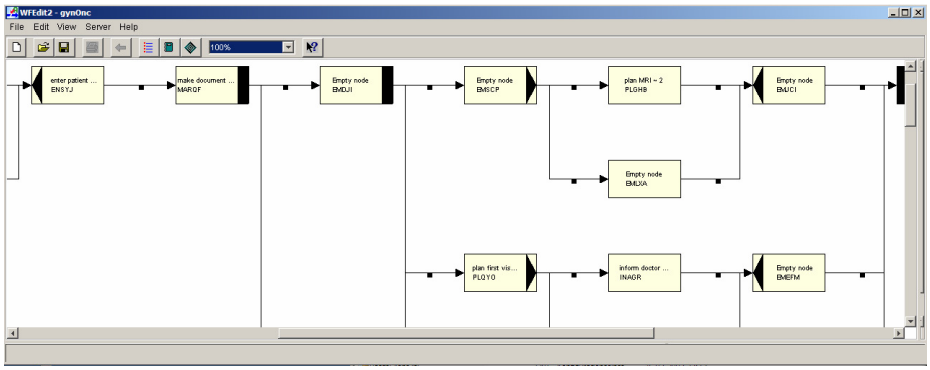

Fig. 8. Screenshot of the ADEPT1 editor. AND-splits/joins are represented by a black rectangle in a node and XOR-splits/joins are represented by a black triangle in a node.

6.3 ADEPT1

ADEPT1 is an academic prototype workflow system [33], developed at the University of Ulm, Germany. ADEPT1 supports *dynamic change* which means that the process of an individual case can be dynamically adapted during execution. So, it is possible to deviate from the static process model (skipping steps, going back to previous steps, inserting new steps, etc.) in a safe and secure way. That is, the system guarantees that all consistency constraints (e.g., no cycles, no missing input data when a task program will be invoked) which have been enforced prior to the dynamic (ad hoc) modification of the process instance are also enforced after the modification.

In Figure 8, we see how the first part of the CWN of Figure 4 is mapped to the ADEPT language. In the ADEPT language, the activities are represented by rectangles. However, as the ADEPT language only has an XOR and an AND split/join, we need to introduce dummy activities, i.e. they are not executed by users. For example, the "make stickers and document" activity in ADEPT is an AND-split which is followed by several dummy XOR-splits such that several activities, like "plan MRI" or "plan CT" can be performed or skipped.

Note that due to the restriction of data elements to simple data types, we only modeled 10% of the whole process[11]. The resultant ADEPT model consists of 40 nodes and 53 arcs and it took around 8 hours to construct a model that could be executed by the ADEPT1 workflow engine.

[11] For ADEPT1 only the part of the CWN associated with the referral of the patient and preparation for the first visit subprocess has been mapped and is shown in Figure 4. The number of activities contained in this subprocess constitutes around 10% of the activities present in the whole CWN. This selection can be considered as representative for the whole CWN.

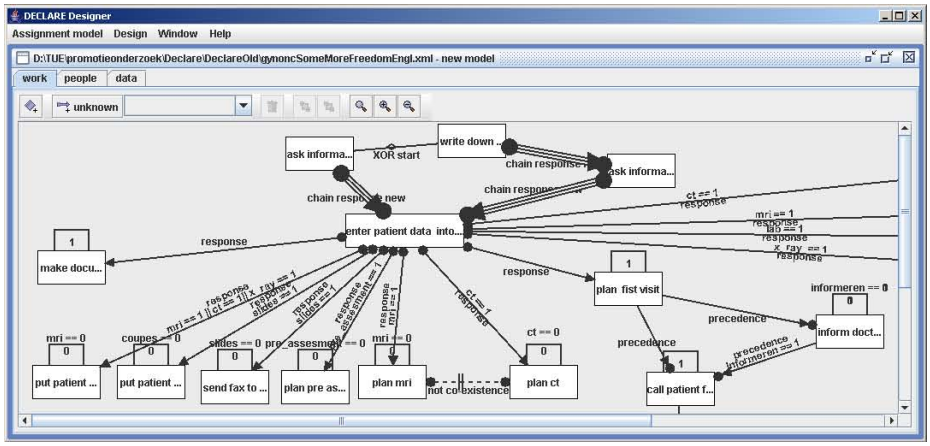

Fig. 9. Screenshot of the Declare editor.

6.4 Declare

Declare is an academic prototype workflow system [29], developed at Eindhoven University of Technology in the Netherlands[12]. In Declare the language used for specifying processes, which is called *ConDec*, is a *declarative* process modeling language, which means that it specifies *what* should be done. *Imperative* process modeling languages, like YAWL and FLOWer, specify *how* it should be done, which often leads to over-specified processes. By using a declarative rather than an imperative / procedural approach, ConDec aims at providing an under-specification of the process where workers have room to "manoeuver". This allows users to execute activities in any order and as often as they want, but they are bound to certain rules. These rules are based on *Linear Temporal Logic (LTL)*. Moreover, Declare also supports *dynamic change*.

In Figure 9, we see how the first part of the CWN of Figure 4 is mapped to the ConDec language. For the same reason as with the ADEPT1 implementation, here we also only modeled 10% of the whole process due to the restriction of data elements to simple data types.

In the ConDec language, the activities are represented by rectangles. Moreover, each different LTL formula, which can be used in the model, is represented by a different *template*, and can be applied to one or more activities, depending on the arity of the LTL formula which is used. Note that the language is extensible, i.e., it is easy to add a new construct by selecting its graphical representation and specifying its semantics in terms of LTL.

In Figure 9, we see that after the "enter patient data into system" activity a lot of subsequent activities need to be done, which is indicated by a *response* arc going from the "enter patient data into system" activity to these activities, e.g. "plan CT" and "plan MRI". However, it is only indicated that these activ-

[12] http://www.win.tue.nl/declare/

ities need to be done afterwards, and no ordering is specified. At runtime, it is indicated to the user which activities need to done. The user can then decide in which order the activities will be executed and how often each one will be done.

The Declare model consists of 23 nodes and 44 LTL formulae and it took around 12 hours to construct a model that could be executed by the Declare workflow engine.

6.5 Effectiveness of CWNs for Moving to an Implementation in a Workflow System

To conclude this section, we will examine on the effectiveness of CWNs for implementing the healthcare process in four different workflow systems. In other words, we try to answer the question of how easy it is to implement the CWN in a workflow system. In order to do this, we demonstrate how CWNs can be used as a starting point for implementation in a workflow system and then evaluate the different systems. To this end, we use five different criteria, which are listed in the top row of Table 1. The different workflow systems are shown in the first column.

As first criterion, we have the number of nodes and arcs in the resultant process model. For the first three workflow systems considered, the nodes in the CWN which referred to activities could be directly translated to activities into the workflow language of the workflow systems. However, for ADEPT1 it was necessary to use dummy nodes as only XOR and AND split/joins are available.

As a CWN covers the *control flow*, *resource*, *data*, and *operation* perspective, we indicate how much effort was required to specify each perspective in each workflow system. This effort corresponded to the time taken by a single person having no experience in configuring processes using these systems. Furthermore, within Declare and ADEPT1 we defined a part of the CWN which constituted 10 percent of the model whereas for YAWL and FLOWer the whole model was implemented. The results are shown in Table 1. The numbers in brackets in the last four columns show the normalized value for the man hours required, so that each of the systems can be compared. However, for the Declare and ADEPT1 system these numbers should only be seen as an indication of the effort required as gradually the level of experience increases when configuring a workflow system. Moreover, some parts of the model might be more difficult to implement than others in different workflow systems.

The main conclusions that can be drawn from the table are discussed below (see [26] for full details). First of all, it can be seen that the control flow perspective can easily be translated for imperative languages (YAWL, FLOWer, ADEPT1). For declarative languages (Declare) more effort is needed. For the resource perspective it can be seen that the resource related parts can be easily translated as in all of the systems roles can be defined which can be linked to activities. Although not immediately visible from the table, the data perspective also could be easily translated for each of the systems. This stems from the fact that for YAWL and FLOWer, although the complex data types that needed to be converted were complex, there is great similarity between data types in these

Table 1. Translation of the CWN into the workflow languages of YAWL, FLOWer, ADEPT1, and Declare. Note that for Declare and ADEPT1 we only defined a part of the CWN which constitutes 10 percent of that model. To this end, to obtain normalized values for these systems, we multiplied the hours spent by 10. The hours enclosed in brackets indicate the normalized time. However, for the Declare and ADEPT1 system these figures should be seen as indicative figures only.

	number of nodes / arcs	effort control flow perspective (hours)	effort organizational perspective (hours)	effort data perspective (hours)	effort operation perspective (hours)
YAWL	231 / 282	20 (20)	5 (5)	30 (30)	65 (65)
FLOWer	236 / 190	20 (20)	5 (5)	20 (30)	55 (65)
ADEPT1	40 / 53	2 (20)	1 (10)	1 (10)	4 (40)
Declare	23 / 44	6 (60)	1 (10)	1 (10)	4 (40)

two systems and the CWN, and therefore the translation is still an easy process. Unfortunately, the operation perspective of the CWN could not be easily translated. This can easily be derived from the table as for each system half or more of the time required for configuring the system is spent on defining this perspective only. This is a consequence of the fact that every system has its own specific way/language for defining the operation perspective and that a big difference exists between these languages and the language used in the CWN. It is worth commenting that in Declare and ADEPT we only had to deal with simple rather than complex data types explaining the difference in effort required. For example, in YAWL and FLOWer, multiple instance tasks needed to be specified which is far from trivial.

During the implementation in the different workflow systems no further insights were gained about the healthcare process itself. So, all the time has been spent on the implementation in the workflow systems. This is due to the fact that we did not receive any user feedback during the implementation phase and the users did not evaluate the resultant systems themselves. If we had received further user feedback during the implementation phase, it is possible that further insights would have been gained about the process captured during the requirements phase (EUC). In essence, the approach focuses on identifying the real world process to be supported during the requirements phase. In the implementation phase the focus is solely on implementation aspects. If new requirements are identified during this phase, in principle, one should go back to the requirements phase and restart the development process from there.

In conclusion, we can say that a CWN is of help when implementing a given process in a workflow system, with regard to the control flow, resource and data perspective. These perspectives have to be defined anyway and the CWN allows for an implementation independent specification of these perspectives. Moreover, it is a useful step towards the implementation of the process in a chosen workflow system.

Furthermore, in principle, a (semi-) automatic translation from the CWN to each of the workflow systems is possible. The control flow and resource perspective are capable of automatic translation. With regard to the data perspective, data attributes which are name-value pairs for a simple data type, e.g. a string or an integer, are capable of automatic translation. However, for the operation perspective and for name-value pairs for which a list type has been used, more work is needed. For both of them, some form of semi-automatic translation would be necessary. However, as in this paper the focus is on the general application of the approach and its applicability to distinct technology systems, we do not focus on developing translation from the CWN to each of the workflow systems. In [8, 20] it has been shown that (semi-) automatic translations from the CWN to a nominated workflow language are possible.

An issue related to the expressiveness of the CWN is the fact that CPNs cannot capture with three categories of workflow patterns, namely advanced synchronization (e.g. the synchronizing merge), multiple instances, and cancelation [6]. When constructing the CWN model already knowledge is gained about how the process exactly needs be supported by a certain workflow system. So, it can already be identified that such pattern might be more appropriate in the final implementation. In that way, if a workflow system offers support for such a pattern it can be selected instead.

As we manually configured each workflow system, there is the risk of making mistakes. Some of the workflow systems offer functionality for checking the soundness property of the configured model. Only in case that such functionality is available we can guarantee the soundness property for the translated model. For example, YAWL offers the opportunity to check for soundness which we subsequently used for checking whether the model was sound. Although such checks can prevent some types of mistakes, they do not prevent mistakes made at the semantic level. For the latter, interactions with users are of vital importance. Note that mistakes made at the semantic level cannot be prevented when making a (semi-) automatic translation. One hopes that these kinds of mistakes have already been detected during the requirements and design phase.

7 Related Work

A workflow application development process is a workflow development methodology aiming at improving the planning and construction of workflow projects [37]. So far, surprisingly little research has been performed on this topic. In [37, 23, 12, 21, 22] different approaches regarding this topic are described. These approaches differ in the number of phases that need to be followed, the steps in these phases, and the scope of the development methodology. Without focusing on specific modeling techniques, [37, 23, 22] specify the steps which needs to be taken in order to end with a workflow implementation of a certain process. These steps are not limited to model construction only, but also address issues like the selection of a workflow system. However, whilst [23, 22] provide a rather high-level description of a workflow development methodology, [37] provides a

detailed overview of the required phases and the steps in these phases, based on experiences obtained during real-world workflow processes.

In contrast, [12, 21] propose specific modeling techniques that have to be used during the development process. In [12], UML descriptions are proposed for identifying business process requirements. Then the WIDE meta-model is used to design an implementation independent workflow schema and associated exception handling which can then subsequently mapped to the required workflow language. In [21], UML use cases and UML interaction diagrams are used to develop a so-called multi-level workflow schema which provides an abstraction of the workflow at different levels.

Compared to these approaches, the development methodology described in Section 2 brings more rigor to the development process as the models constructed in the EUC and CWN phase are based on a formal foundation. This provides various possibilities for verification whereas in [12, 21] different modeling mechanisms are mentioned but nothing is said about the verification of these models. Moreover, as both the EUC and CWN use the same language, this allows for a seamless integration between the requirements and design phase. None of the approaches described in [37, 23, 12, 21, 22] report on obtained experiences when applying the proposed approach.

From the literature, it can be deduced that workflow systems are not applicable to the healthcare domain [11, 24]. The current generation of workflow systems adequately supports administrative and production workflows but they are less suited for healthcare processes which have more complex requirements [11]. In addition, in [31, 32], it has been indicated that so called "careflow systems", i.e. systems for supporting care processes in hospitals, have special demands with regard to workflow technology. One of these requirements is that flexibility needs to be provided by the workflow system [28, 35]. Unfortunately, current workflow systems fall short in this area, a fact which is recognized in the literature [7, 9]. Once a workflow-based application has been configured on the basis of explicit process models, the execution of related process instances tends to be rather inflexible [3, 33]. Consequently, the lack of demonstrated flexibility has significantly limited the application of workflow technology. The workflow systems that we chose in this paper were specifically selected because they allow for more flexibility than classical workflow systems.

This paper uses the approach initially proposed in [8, 20] where an EUC and CWN have been used to progress from an informal inscription of a real world process to an implementation of the same process in a particular workflow system. In [20], an electronic patient record system has been implemented using the YAWL system. However, in both papers, only small examples were used and an implementation was only completed in one workflow system. Furthermore, in [8] direct user involvement was limited to the requirements phase, whereas in [20] there was no user involvement at all. In this paper, we modeled a much larger, representative healthcare process from a hospital using four different workflow systems. Moreover, the approach was evaluated by the people involved in the process.

8 Conclusions

In this paper, we have focused on the implementation of a large hospital process consisting of hundreds of activities, in different workflow systems. To support the implementation process, we first developed an EUC, followed by a CWN. This CWN was used as the input for configuring the different workflow systems. The approach, shown in Figure 1, effectively bridges the gap between the modeling of a real-world process and the implementation of the process in a workflow system. We have successfully shown that the approach works for a large real-world healthcare process and for distinct workflow technologies. However, some important observations can be drawn from this research effort.

The first observation is that the combination of animations and EUCs are of great help when validating the modeled process. Based on the feedback of users we believe that better results have been obtained than when using the plain CPN models.

The second observation is that the CWN helps in elaborating how the candidate process, needs to be made ready for implementation in a workflow system. The CWN covers the control-flow, resource, data and operation perspective, which necessitates the specification of these perspectives during the construction of the CWN. Another advantage is that the CWN abstracts from implementation details and language/application specific issues.

The third observation is that a CWN is helpful during the configuration of the control-flow, resource and data perspective in a workflow system. However, this does not hold for the operation perspective of the CWN.

Furthermore, EUCs and CWNs are useful as they allow for a separation of concerns. The approach adopted enforces that the issues we have to deal during workflow development are tackled at the right time and in the right sequence.

A possible direction for future research is to develop animations specific for CWN. In this way, before a process is supported by a workflow system, the associated staff can already become acquainted with how their process will be facilitated and start experimenting with it. The case study has provided valuable insights into the requirements for workflow technology in care organizations. Besides the need for flexibility, it also revealed the need for a better integration of patient flow with the scheduling of appointments and peripheral systems supporting small and loosely coupled workflows (e.g. lab tests).

Acknowledgements. We would like to thank Lisa Wells and Kurt Jensen for their support with CPN Tools. Also, we would like to thank Michael Westergaard for his support in using the BRITnEY tool. Furthermore, we also want to thank the people of the gynecological oncology, radiology, pathology and anesthetics department, and especially Prof. Mathé Burger of the gynecological oncology department for their comprehensive explanation of the gynecological oncology healthcare process.

Moreover, we would like to thank the reviewers of ToPNoC, as well as the editors, for their fruitful comments to improve this paper.

References

1. van der Aalst, W.M.P.: Workflow Verification: Finding Control-Flow Errors using Petri-net-based Techniques. In: van der Aalst, W.M.P., Desel, J., Oberweis, A. (eds.). LNCS, vol. 1806, pp. 161–183. Springer (2000)
2. van der Aalst, W.M.P.: Business Process Management Demystified: A Tutorial on Models, Systems and Standards for Workflow Management. In: Desel, J., Reisig, W., Rozenberg, G. (eds.). LNCS, vol. 3098, pp. 1–65. Springer (2004)
3. van der Aalst, W.M.P., Barthelmess, P., Ellis, C.A., Wainer, J.: Workflow Modeling using Proclets. In: Etzion, O., Scheuermann, P. (eds.). CoopIS 2000. LNCS, vol. 1901, pp. 198–209. Springer (2000)
4. van der Aalst, W.M.P., van Hee, K.M.: Workflow Management: Models, Methods, and Systems. MIT press, Cambridge, MA (2002)
5. van der Aalst, W.M.P., van Hee, K.M., ter Hofstede, A.H.M., Sidorova, N., Verbeek, H.M.W., Voorhoeve, M., Wynn, M.T.: Soundness of Workflow Nets: Classification, Decidability, and Analysis. Computer Science Report No. 08-13, Technische Universiteit Eindhoven, The Netherlands (2008)
6. van der Aalst, W.M.P., ter Hofstede, A.H.M.: YAWL: Yet Another Workflow Language. Information Systems 30(4), 245–275 (2005)
7. van der Aalst, W.M.P., Jablonski, S.: Dealing with Workflow Change: Identification of Issues and Solutions. International Journal of Computer Systems, Science, and Engineering 15(5), 267–276 (2000)
8. van der Aalst, W.M.P., Jørgensen, J.B., Lassen, K.B.: Let's Go All the Way: From Requirements via Colored Workflow Nets to a BPEL Implementation of a New Bank System Paper. In: Meersman, R., Tari, Z. et al. (eds.). CoopIS 2005. LNCS, vol. 3760, pp. 22–39. Springer (2005)
9. van der Aalst, W.M.P., Weske, M., Grünbauer, D.: Case Handling: A New Paradigm for Business Process Support. Data and Knowledge Engineering. 53(2), 129–162 (2005)
10. Adams, M., ter Hofstede, A.H.M. ter, Edmond, D., van der Aalst, W.M.P.: Facilitating Flexibility and Dynamic Exception Handling in Workflows. In: Belo, O., Eder, J. Pastor, O., Falcao e Cunha, J. (eds.) Proceedings of the CAiSE'05 Forum. pp. 45–50. FEUP, Porto, Portugal (2005)
11. Anyanwu, K., Sheth, A., Cardoso, J., Miller, J., Kochut, K.: Healthcare Enterprise Process Development and Integration. Journal of Research and Practice in Information Technology. 35(2), 83–98 (2003)
12. Baresi, L., Casati, F., Castano, S., Fugini, M.G., Mirbel, I., Pernici, B.: WIDE Workflow Development Metholodogy. In: Proceedings of International Joint Conference on Work Activities Coordination and Collaboration, pp. 19–28 (1999)
13. Dadam, P., Reichert, M., Khuhn, K.: Clinical Workflows - The Killer Application for Process-oriented Information Systems?. In: Abramowicz, W., Orlowska, M.E. (eds.) BIS2000. pp. 36–59. Springer (2000)
14. Graeber, S.: The impact of workflow management systems on the design of hospital information systems. In: AMIA 2001 symposium proceedings.
15. Greiner, U., Ramsch, J., Heller, B., Löffler, M., Müller, R., Rahm, E.: Adaptive Guideline-based Treatment Workflows with AdaptFlow. In: Kaiser, K., Misch, S., Tu, S.W. (eds.) CGP 2004. Computer-based Support for Clinical Guidelines and Protocols, pp. 113–117. IOS Press (2004)
16. van Hee, K.M., Serebrenik, A., Sidorova, N., Voorhoeve, M.: Resource-Constrained Workflow Nets. In: Ciardo, G, Darondeau, P. (eds.) ATPN 2005. LNCS, vol. 3536, pp. 250–267. Springer, Heidelberg (2005)

17. Jensen, K., Kristensen, L.M., Wells, L.: Coloured Petri Nets and CPN Tools for Modelling and Validation of Concurrent Systems. STTT 9(3–4), 213–254 (2007)
18. Jensen, K., Christensen, S., Kristensen, L.M.: CPN Tools State Space Manual. Department of Computer Science, Univerisity of Aarhus (2006)
19. Jørgensen, J.B., Bossen, C.: Executable Use Cases: Requirements for a Pervasive Health Care System. IEEE Software 21, 34–41 (2004)
20. Jørgensen, J.B., Lassen, K.B., van der Aalst, W.M.P.: From task descriptions via colored Petri nets towards an implementation of a new electronic patient record workflow system. STTT 10(1), 15–28 (2006)
21. Kim, J., Robert Karlson, C.: A Design Methodology for Workflow System Development. In: Databases in Networked Information Systems. LNCS, vol. 2544, pp. 15–28. Springer, Heidelberg (2002)
22. Kobielus, J.G.: Workflow Strategies. IDG Books (1997)
23. Kwan, M., Balasubramanian, P.R.: Adding Workflow Analysis Techniques to the IS Development Toolkit. In: HICSS 1998. vol. 4, pp. 312–321. IEEE Computer Society Press (1998)
24. Lenz, R., Elstner, T., Siegele, H., Kuhn, K.: A Practical Approach to Process Support in Health Information Systems. JAMIA 9(6), 571–585 (2002)
25. Lenz, R., Reichert, M.: IT Support for Healthcare Processes - Premises, Challenges, Perspectives. DKE 61, 49–58 (2007)
26. Mans, R.S., van der Aalst, W.M.P., Bakker, P.J.M., Moleman, A.J., Lassen, K.B., Jørgensen, J.B.: From Requirements via Colored Workflow Nets to an Implementation in Several Workflow Systems. In: Jensen, K. (eds.) Proceedings of the Eight Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools. pp. 187–206 (2007)
27. Mans, R.S., van der Aalst, W.M.P., Russell, N.C., Bakker, P.J.M.: Flexibility Schemes for Workflow Management Systems. In: Pre-Proceedings of ProHealth '08. pp. 50–61 (2008)
28. Maruster, L., van der Aalst, W.M.P., Weijters, A.J.M.M., van den Bosch, A., Daelemans, W.: Automated Discovery of Workflow Models from Hospital Data. In: Dousson, C., Höppner, F., Quiniou, R. Proceedings of the ECAI Workshop on Knowledge Discovery and Spatial Data. pp. 32–36 (2002)
29. Pesic, M., van der Aalst, W.M.P.: A Declarative Approach for Flexible Business Processes Management. In: Eder, J., Dustdar, S. BPM 2006 workshops. LNCS, vol. 4103, pp. 169–180. Springer, Heidelberg (2006)
30. Philippi, S., Hill, H.J.: Communication Support for Systems Engineering - Process Modelling and Animation with APRIL. The Journal of Systems and Software 80(8), 1305–1316 (2007)
31. Quaglini, S., Stefanelli, M., Cavallini, A., Micieli, G., Fassino, C., Mossa, C.: Guideline-based Careflow Systems. Artificial Intelligence in Medicine 20(1), 5–22 (2000)
32. Quaglini, S., Stefanelli, M., Lanzola, G., Caporusso, V., Panzarasa, S.: Flexible Guideline-based Patient Careflow Systems. Artificial Intelligence in Medicine 22(1), 65–80 (2001)
33. Reichert, M., Dadam, P.: ADEPTflex: Supporting Dynamic Changes of Workflow without Loosing Control. Journal of Intelligent Information Systems 10(2), 93–129 (1998)
34. Reijers, H.A., van der Aalst, W.M.P.: The Effectiveness of Workflow Management Systems: Predictions and Lessons Learned. International Journal of Information Management 56(5), 457–471 (2005)

35. Stefanelli, M.: Knowledge and Process Management in Health Care Organizations. Methods Inf Med 43, 525–535 (2004)
36. Trcka, N., van der Aalst, W.M.P., Sidorova, N.: Analyzing control-flow and data-flow in workflow processes in a unified way. Computer Science Report No. 08-31, Technische Universiteit Eindhoven, The Netherlands (2008)
37. Weske, M., Goesmann, T., Holten, R., Striemer, R.: Analysing, modelling and improving workflow application development processes. Software Process Improvement and Practice 6, 35–46 (2001)