Business Process Verification - Finally a Reality!

M.T. Wynn[1], H.M.W. Verbeek[2], W.M.P. van der Aalst[1,2], A.H.M. ter Hofstede[1] and D. Edmond[1]

Business Process Management Group, Queensland University of Technology
GPO Box 2434, Brisbane QLD 4001, Australia.
{*m.wynn,d.edmond,a.terhofstede*}*@qut.edu.au*
Department of Mathematics and Computer Science, Eindhoven University of Technology
PO Box 513, NL-5600 MB Eindhoven, The Netherlands.
{*h.m.w.verbeek,w.m.p.v.d.aalst*}*@tue.nl*

Purpose - The goal of this paper is to demonstrate that process verification has matured to a level where it can be used in practice. Earlier techniques assumed simplified process models without the more advanced constructs available in today's modelling languages (e.g., cancellation and OR-joins). This paper reports on new verification techniques that can be used to assess the correctness of real-life models.

Design/Methodology/approach - The proposed approach relies on using formal methods (i.e., mapping a business model to a reset net which is an extension of Petri nets, and performing state space analysis) to determine the correctness of business processes with cancellation and OR-joins. The paper also demonstrates how reduction rules can be used to improve the efficiency. We present these techniques in the context of the workflow language YAWL that provides direct support for 20 most frequently used patterns found today (including cancellation and OR-joins). But the results also apply to other languages with these features (e.g., BPMN, EPCs, UML activity diagrams, etc.). We have developed an editor that provides diagnostic information based on the techniques presented in this paper.

Findings - We propose four properties for business processes with cancellation and OR-joins, namely, *soundness*, *weak soundness*, *irreducible cancellation regions*, and *immutable OR-joins* and develop new techniques to verify these properties. Reduction rules have been used as a means of improving the efficiency of the algorithm. We demonstrate the feasibility of our verification approach using a realistic and complex business process, the *visa application process for general skilled migration to Australia*, modelled as a YAWL workflow with cancellation regions and OR-joins.

Originality/value - Business processes sometimes require complex execution interdependencies to properly complete a process. For instance, it is possible that certain activities need to be cancelled mid-way though the process. Some parallel activities may require complex "wait and see" style synchronisation depending on a given context. These types of business processes can be found in various domains, such as application integration, B2B commerce, web service composition and workflow systems. Even though cancellation and sophisticated join structures are present in many business processes, existing verification techniques are unable to deal with such processes. Hence, this paper plays an important role in making process verification a reality.

Keywords: Verification, Process modelling, Cancellation, OR-joins, YAWL, BPMN, EPC

Paper type - Research paper

The need for verification of process models

Verification is concerned with determining, *in advance*, whether a process model exhibits certain desirable behaviours. By performing this verification at design time, it is possible to identify potential problems, and if so, the model can be modified before it is used for execution. As some systems (e.g., workflow systems) rely on process models for execution of work, careful analysis of process models at design time can greatly improve the reliability of such systems.

Although one would expect verification functionality to be present in any business process modelling tool, workflow management system, or business process management suite, this is not the case. At best these systems do some basic syntactical checks, but allow for the modelling of processes with deadlocks, livelocks, and other anomalies. There are several academic process verification tools. However, until recently, these tools could not verify realistic processes because they assume highly simplified models completely disconnected from real-life languages and systems. Fortunately, as this paper will show, a breakthrough has been realized that makes process verification feasible in a practical setting. As an example, we also refer to the study reported in (Mendling, Moser, Neumann, Verbeek, Dongen & van der Aalst 2006) where we analysed the entire SAP reference model based on similar techniques. In this process, we discovered many errors in the 604 processes contained in the reference model of SAP. This resulted in quite some publicity in the popular press, e.g., IT magazines such as Computable, iX, Automatisering gids, BPTrends, and BPM magazine ran articles on this. This illustrates the interest of practitioners to have correct process models. Moreover, the results illustrate that process verification has become a reality.

This paper, will focus on two features common in any modern process modelling language: (1) *cancellation* and (2) *OR-joins*. The reason is that, until recently, there were no tools and techniques allowing for the analysis of models with these features. *Cancellation* captures the interference of an activity in the execution of others in certain circumstances. Cancellation can be triggered by either a customer request (e.g., a customer wishes to withdraw a credit card application) or by exceptions (e.g., an order cannot be processed due to insufficient stock level). In general, cancellation results in one of two outcomes: disabling some scheduled activities or stopping currently running activities. The complicating factor is that due to concurrency issues, the cancellation action may or may not result in cancelling certain activities, i.e., the process may be in a state before or after the part that is supposed to be cancelled. This can introduce deadlocks (the state where a business process is stuck and cannot proceed). An *OR-join* is used in situations when we need to model "wait and see" behaviour for synchronisation. For example, a purchase process could involve the separate purchase of one or two different items and the customer can decide whether he/she wants to purchase one or the other or both. The subsequent payment task is to be performed only once and this requires synchronisation if the customer has selected both products. If the customer selected only one product, no synchronisation is required before payment. Many systems and languages struggle with the semantics and implementation of the OR-join because its semantics require a synchronisation depending on an analysis of future execution paths. This requires some non-trivial reasoning. The presence of cancellation and OR-joins poses new challenges for deciding the correctness of business processes. Existing

approaches and tools are typically restricted to process models without such features. New techniques are required to enable design time detection of verification problems in business processes with such behaviours.

Proposed approach

The introduction of cancellation and OR-joins leads to *new properties* that need to be checked, the design of *new algorithmic approaches* and the *management of their computational complexity*. We take on these challenges and develop sophisticated verification techniques for process models with cancellation and OR-joins. In this paper, we perform verification in the context of the workflow language YAWL (van der Aalst & ter Hofstede 2005). The YAWL (Yet Another Workflow Language) workflow language supports the most frequent control-flow patterns found in the current workflow and business process modelling practice (van der Aalst, ter Hofstede, Kiepuszewski & Barros 2003). As a result, most workflow and business process modelling languages can be mapped onto YAWL without loss of control-flow details, even languages allowing for advanced constructs such as cancellation regions and OR-joins. Therefore, our results are also applicable to other languages. Some examples:

- The Business Process Modelling Notation (BPMN) is supported by more than 40 tools and has been standardized by the OMG. BPMN provides (in addition to the standard constructs) an "OR-join gateway" and various cancellation constructs.
- The Activity Diagram type of the Unified Modelling Language (UML) has also been standardized by the OMG and is supported by many tools. UML does not provide the OR-join but offers different cancellation features.
- Event-driven Process Chains (EPCs) are used in the reference model of SAP and are used in business process modelling tools such as ARIS. The EPC language provides OR-joins but not cancellation.
- The Business Process Execution Language (BPEL) is supported by the software products of IBM, SAP, Oracle, etc. and is being standardized by OASIS. BPEL supports OR-joins (through the "flow" construct) and cancellation.

These examples show that today's languages support cancellation and/or OR-joins. Hence, it is vital to support the verification of these constructs. We have implemented our verification approach in the context of YAWL. However, as stated before, the results can easily be transferred to other languages supporting cancellation and/or OR-joins.

Introduction to YAWL

The YAWL language has been implemented in an open source workflow system[1] and can be seen as a reference implementation of the workflow patterns (van der Aalst et al. 2003). The YAWL workflow system consists of a number of components including a workflow engine and an editor. Workflow specifications can be designed using the YAWL editor and deployed in the YAWL engine for execution.

[1] http://www.yawl-system.com

A YAWL model is made up of tasks, conditions and a flow relation between tasks and conditions. Each YAWL model has one start condition and one end condition. There are three kinds of split and three corresponding kinds of join in YAWL; they are AND, XOR and OR. The splits, joins, conditions and cancellation symbols for YAWL are shown in Figure 1. A task is enabled when there are enough tokens in its input conditions according to the join behaviour. When a task is executed, it takes tokens out of the input conditions and puts tokens in its output conditions according to the join and split behaviour respectively. The semantics of an OR-join in YAWL waits for synchronisation, wherever possible. Hence, sophisticated analysis is carried out before deciding whether an OR-join will be enabled. YAWL provides direct support for cancellation regions. A task can have a cancellation set associated with it (dotted lines denote the cancellation region of a task). If there is a cancellation set associated with a task, the execution of the task removes all the tokens from the conditions and tasks in the cancellation set. If a task is within the cancellation region of another task, it may be prevented from being started or its execution may be terminated (depending on the timing).

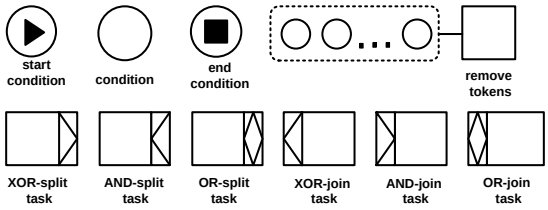

Fig. 1. Splits, joins, conditions and cancellation in YAWL

Properties

Four desirable properties for processes with cancellation regions and OR-joins are proposed. These are *soundness*, *weak soundness*, *irreducible cancellation regions*, and *immutable OR-joins*. While soundness and weak soundness properties concentrate on the correctness of the models, the other two properties; *irreducible cancellation regions* and *immutable OR-joins*, focus on detecting the existence of unnecessary cancellation regions and OR-joins in the process models. Next, we briefly explain these properties informally. Formal definitions of these properties can be found in Wynn (2006).

1. **Soundness:** There are certain desirable characteristics that every business process is expected to exhibit. Firstly, it is important to know that a process, when started, can always complete (*Option to complete*). Secondly, it should not have any other tasks still running for that process when the process ends (*Proper completion*). Thirdly, the process should not contain tasks that will never be executed (*No dead transitions*). The combination of these three desirable properties is known as the *soundness* property.

2. **Weak soundness:** For certain processes with OR-joins and cancellation regions having an infinite state space, it is not possible to detect this soundness property, i.e., although soundness is decidable for most models *the soundness property is undecidable* in the general case.[2] Thus, a weaker notion of soundness, known as the *weak soundness* property is proposed instead. The weak soundness property relaxes the option to complete criterion, to say that, is it possible to complete a process in *some* cases, when started (Weak option to complete). Therefore, if a process model is sound, it will also be weak sound, but not vice versa.
3. **Irreducible cancellation regions:** Reducible elements in a cancellation region represent elements that can never be cancelled while that task is being executed (e.g. conditions may never contain tokens). For example, if a model contains truly alternative branches and one path contains a task that covers some places and conditions from the other path within its cancellation region, then those places and transitions are superfluous as there will never be tokens to remove when the task is executing. A process satisfies the *irreducible cancellation regions* property if there are no reducible elements in any of its cancellation regions.
4. **Immutable OR-joins:** As the runtime analysis of OR-join tasks is time-consuming and (computationally) expensive, it is desirable to determine at design time whether a more appropriate join structure could be found for a given model. A convertible OR-join task is an OR-join task for which it is never possible to mark more than one input condition (the task acts as an XOR-join) or when all the input conditions are always marked (the task acts as an AND-join). Such OR-joins should be replaced by XOR/AND-joins to better reflect their role in the process and to improve the execution speed. A process satisfies the *immutable OR-joins* property if there are no convertible OR-joins in the net.

Algorithmic approaches

There are established results related to the verification of workflows using Petri nets (van der Aalst 1997, Verbeek 2004). We explore how these results can be used for YAWL workflows with cancellation and OR-joins. Reset nets form a natural foundation for workflow languages with explicit support for cancellation as the behaviour of reset arcs closely matches the behaviour of cancellation regions (Dufourd, Finkel & Schnoebelen 1998, Dufourd, Jančar & Schnoebelen 1999). A YAWL workflow is mapped onto a reset net and state space analysis is performed to determine the correctness of the model. The state space analysis generates all possible reachable states of a workflow model to determine whether the model is correct. To determine the weak soundness property and irreducible cancellation regions property, the backwards coverability notion in reset nets has been utilised (Finkel & Schnoebelen 2001).

[2] Note that any language that allows for unrestricted cancellation regions and/or OR-joins is undecidable. Hence, it is not always possible to decide soundness for models constructed using languages like BPMN, UML activity diagrams, YAWL, BPEL, etc. Therefore, it is surprising that existing verification approaches do not acknowledge this and focus on "toy languages". Fortunately, although soundness is not decidable in the general case, it is decidable in most practical cases.

For verification purposes, the processes are divided into those with OR-joins and those without OR-joins. This distinction is necessary as a different verification technique is needed in each case. A process without OR-joins can be mapped to a reset net and it is possible to perform verification on the resulting reset net. However, due to the non-local semantics of OR-joins, it is not possible to map a YAWL workflow with OR-joins to a reset net (without some approximation) and it is not possible to detect the soundness property for a YAWL net with OR-joins using verification techniques available for reset nets. Therefore, an alternative verification technique using the YAWL formal semantics is used. These algorithmic approaches have been derived using the state space analysis and the notions of coverability and reachability.

Managing computational complexity

There is a clear trade-off between the expressive power of a language (i.e., introducing complex constructs such as cancellation and OR-joins) and ease of verification. As the state space analysis results in generation of all possible states of a workflow model, verification is time consuming and can become intractable for large models. There are a number of different approaches to deal with this complexity. Reducing a specification, while preserving its essential properties with respect to a particular analysis problem, is one such approach.

A significant body of research exists that addresses the concept of reduction in the area of Petri nets, see e.g., (Berthelot 1986, Murata 1989) and its various subclasses, see e.g., (Desel & Esparza 1995) and extensions, see e.g., (Sloan & Buy 1996). Even though reduction rules exist for Petri nets, the nature of reset arcs invalidates the transformation rules applicable to Petri nets. When reducing a net it is imperative that certain essential properties are preserved. In the area of workflow verification, soundness is such a property. Therefore, a number of soundness preserving reduction rules for reset nets are proposed (Wynn, Verbeek, van der Aalst, ter Hofstede & Edmond 2006*a*). Furthermore, as verification of YAWL nets without OR-joins is performed without transformation to reset nets, we also propose a number of soundness preserving reduction rules for YAWL elements (Wynn, Verbeek, van der Aalst, ter Hofstede & Edmond 2006*b*).

Visa application example - A YAWL workflow with cancellation regions and OR-joins

We now demonstrate the effectiveness of the proposed verification techniques with reduction rules using a real-life process model: **visa application for general skilled migration to Australia**. This process is modelled "as is" using publicly available information from the Department of Immigration and Multicultural Affairs website.[3] The process starts when a visa application is received by the immigration department and ends when a decision is made to grant or to deny the visa. The model represents the process from the viewpoint of a case officer who handles the visa application. The resulting YAWL workflow contains four nets *Overview*, *Perform main assessment*, *Check basic requirements*, and *Allocate marks*.

[3] http://www.immi.gov.au accessed on 20 April 2006

Figure 2 shows the *Overview* net and the typical process flow is explained first. When an application is received, the case officer opens a file for the applicant, processes visa application fees and performs an initial assessment. If the application is found to be *complete*, the officer continues with the main assessment. If the application is *incomplete*, he/she sends an acknowledgement letter to the applicant requesting further documentation. This is modelled as an XOR-split task after the task *Perform initial assessment*. The *Perform main assessment* task is modelled as a composite task and the internal working of this task is captured in another net. After completing the main assessment, the case officer might request more information, or he/she is ready to make a decision. This is modelled as an XOR-split task. Condition $c9$ represents a state where the officer is waiting for further documentation from the applicant. If he/she receives the requested information, the main assessment task is performed again. On the other hand, the designated time period could have expired, and the officer decides to perform the main assessment again if possible to stop processing the application if it cannot be processed further with existing documentation. Before the officer makes a decision, he/she checks to see if there is any change in circumstances that need to be considered. The *Check circumstances changes* task has a cancellation region containing condition $c2$. Removing a token from $c2$ indicates that there is no need to wait for further circumstances changes. The officer then makes a decision to either grant or deny the visa after taking into account any changed circumstances. The *Make decision* task is an OR-join task with two inputs $c5$ and $c7$. A token in $c5$ indicates that there are changes that need to be considered. If a decision is made to deny the visa, the applicant is then notified. Otherwise, the visa is granted. The process ends when the *Finalise application* task is executed.

While an application is being processed, it is possible for two events to occur. First, an applicant can decide to withdraw his/her application and secondly, an applicant can notify the immigration department of changes in his/her circumstances - such as change of address, correction of errors, etc. Hence, the task *Open applicant file* is modelled as an AND-split to indicate that two tasks (*Wait for possible withdrawal request* and *Monitor circumstances changes*) could occur in addition to the main flow starting with *Process application fees* task. These two tasks represent *external triggers* that can be enabled when a notification is received from the applicant. These triggers affect the main flow of the process and are also captured in the model. Note that YAWL does not explicitly model external triggers and, therefore, the two potential triggers are represented as ordinary tasks subcontracted to a service that handles these triggers. A token in $c6$ indicates that there is some change in circumstances that needs to be taken into account. Similarly, a token in $c4$ indicates that a request has been received for withdrawal. The *Cancel application* task is modelled as an OR-join and when it fires, it removes tokens from conditions and tasks in the process fragment before the *Make decision* task (see the large process fragment enclosed by the dashed lines and connected to *Cancel application* in Figure 2). The application can be withdrawn until a decision is made. The *Make decision* task removes tokens from conditions and tasks associated with the trigger for application withdrawal.

In the *Overview* net, the *Perform main assessment* is represented as a composite task and it is unfolded into the YAWL net with the same name. Similarly, there are two

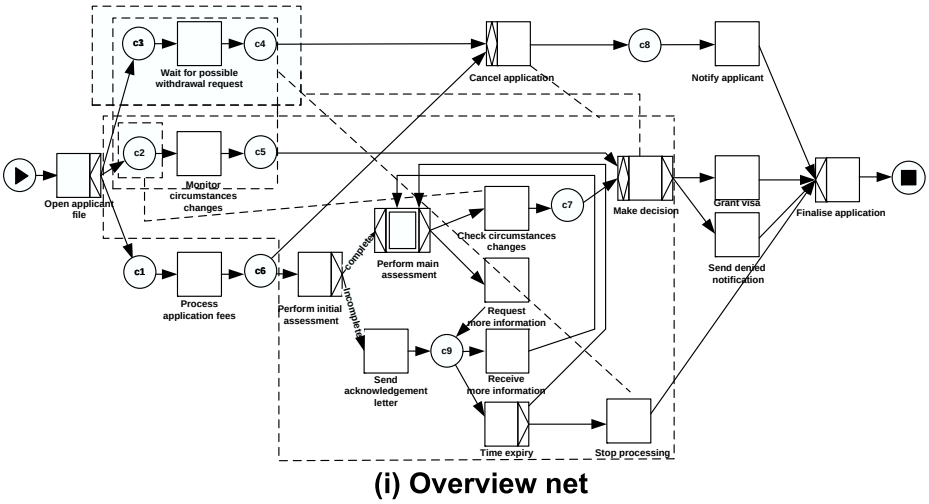

Fig. 2. *Overview*: the main YAWL net in the Australian visa application process

composite tasks: *Check basic requirements* and *Allocate marks* in the *Perform main assessment* net and they are also unfolded into two YAWL nets with the corresponding names. Figure 3 shows the three subnets in the process.

The *Perform main assessment* net contains five tasks: *Check documentation*, *Check basic requirements*, *Allocate marks*, *Compare with pass marks*, and *Perform medical checks*. The *Check basic requirements* task and the *Allocate marks* task are modelled as composite tasks. When the net ends, the process can be at one of the following stages: *insufficient documentation*, *fail*, *pass*. If the process fails due to insufficient documentation, further documentation will be requested from the applicant. Otherwise, the end of perform main assessment net indicates that the officer is ready to make a decision about the visa application.

The *Check basic requirements* net describes how checks for basic requirements are carried out. There are five basic requirements for this class of visa and the *Initialise basic requirements check* task is modelled as an AND-split followed by five tasks, one to check each criterion. A decision is then made about the applicant's ability to satisfy a particular requirement and this is modelled as an XOR-split. A token in condition c_{fail} indicates that at least one of the requirements cannot be satisfied. If an applicant does not meet all of the requirements, he/she will not be granted a visa and the application is not processed any further. This is modelled with a discriminator pattern, where a token in c_{fail} will enable the *Stop checks* task and all the other checks will be cancelled. If an applicant meets all five requirements, the processing continues. This is modelled as an AND-join for the *Finalise basic requirements check* task.

The *Allocate marks* net represents the process for calculating the marks received by each applicant. This visa class uses a points system where marks are given based on the applicant's circumstances assessed on several criteria. The total mark is then compared against the current pass mark for the visa class (110 points) to decide whether the visa will be granted. The net models how these points are allocated for 11 criteria to calcu-

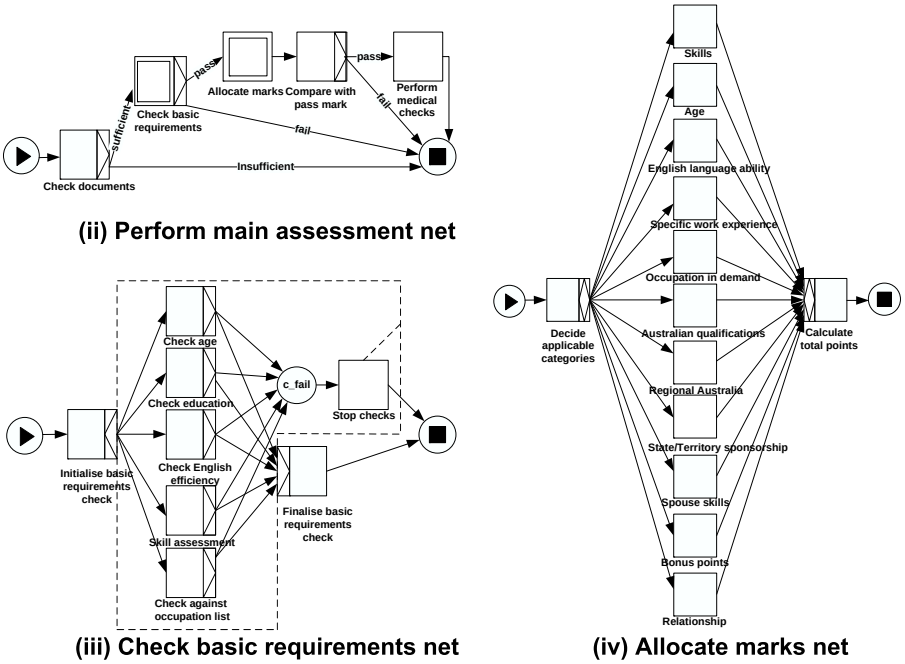

Fig. 3. YAWL nets: *Perform main assessment*, *Check basic requirements*, and *Allocate marks*

late the total number of points. Some criteria such as points for age, skills and English ability are relevant to all applicants, whereas others such as points for Australian qualifications and spouse skills are relevant to some applicants only. The *Decide applicable categories* task is modelled as an OR-split where a decision is made regarding the relevance of a particular criterion. The net completes with an OR-join task that waits for synchronisation of all active paths before calculating the total points allocated to the applicant.

Illustrating properties

We now demonstrate how our verification techniques can be used to diagnose the four properties mentioned before: *soundness*, *weak soundness*, *irreducible cancellation regions*, and *immutable OR-joins*. We will show how the YAWL editor checks these properties and reports the results. Again we would like to stress that, although we demonstrate this in the context of YAWL, the approach is generic and can also be used in the context of other languages supporting cancellation regions and/or OR-joins.

Before showing the diagnostic information provided by the YAWL editor, we discuss the relevant properties for the four YAWL nets in the *Visa application* specification.

1. *Overview*: the net is a large net with two OR-joins and a number of cancellation regions. As it is a net with OR-joins and a finite reachability graph, reachability

results using the YAWL semantics can be obtained. Therefore, three properties: soundness, irreducible cancellation regions and immutable OR-joins are decidable. The weak soundness property check is performed on the corresponding reset net where all OR-joins are first transformed into XOR-joins. Only limited results are available with this approach.

2. *Perform main assessment*: This is a small net with two composite tasks. As it is a net without OR-joins, weak soundness and soundness properties are decidable using coverability and reachability results from reset nets. The immutable OR-joins check is not applicable as there are no OR-joins in the net. The irreducible cancellation regions check is also not needed as there are no cancellation regions in the net.
3. *Check basic requirements*: This is a net with a large cancellation region. As it is a net without OR-joins, weak soundness, soundness and irreducible cancellation regions are decidable using reset net results. The immutable OR-joins property check is not applicable here.
4. *Allocate marks*: This is a structured net with an OR-split task and an OR-join task. As it is a net with OR-joins and a finite reachability graph, soundness and immutable OR-joins are decidable using the YAWL semantics. The irreducible cancellation regions property check is not applicable as there are no cancellation regions in the net. The weak soundness property check is performed on the corresponding reset net where the OR-join is first transformed into an XOR-join. Only limited results are available with this approach.

Verifying soundness

Recall that a net is sound (van der Aalst 1998) if and only if it satisfies three criteria: option to complete, proper completion and no dead transitions. Different verification techniques are proposed to detect the soundness property of nets with and without OR-joins. The two nets representing composite tasks *Main assessment* and *Check basic requirements* are nets without OR-joins. Thus, reset analysis (Wynn 2006) is used to detect the soundness property for these nets (Dufourd et al. 1998, Dufourd et al. 1999). For nets with OR-joins and a finite reachability graph, reachability analysis is carried out using the YAWL semantics. Figure 4 shows a screenshot of the YAWL editor with results of the soundness property check. The three nets (*Overview*, *Check basic requirements* and *Perform main assessment*) are shown to satisfy the soundness property and observation messages are provided to indicate that these nets satisfy all three criteria. For the *Allocate marks* net, the analysis is not completed as it has more than 5000 reachable markings and the editor is configured to use this upper limit to stop the analysis. Note that there may be infinitely many markings, hence the upper bound is set to 5000 to balance responsiveness and precision. Note that even though the *Allocate marks* net satisfies the soundness property, the analysis cannot be completed without using reduction rules for optimisation.

Verifying weak soundness

A net satisfies the weak soundness property if and only if it has the weak option to complete, proper completion and no dead transitions. The weak soundness property check

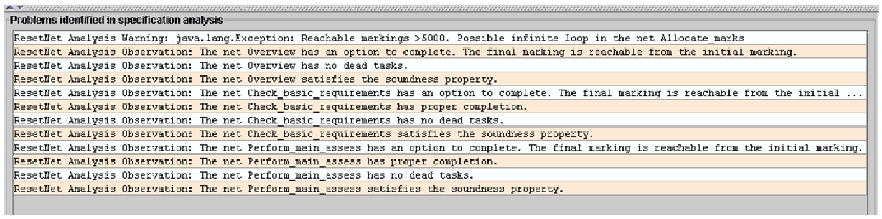

Fig. 4. Screenshot of soundness property check

is performed using reset net coverability analysis for nets with and without OR-joins. For nets with OR-joins, only limited results are available. Figure 5 shows a screenshot from the editor with the results of the weak soundness property check for all nets. We can see that the three nets satisfy the weak soundness property.

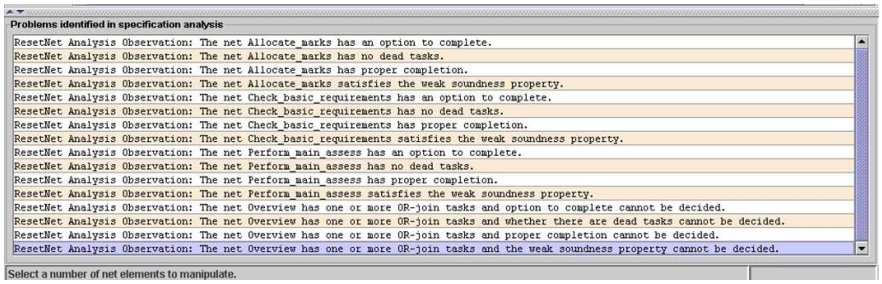

Fig. 5. Screenshot of weak soundness property check

We also found that the *Check basic requirements* net with 21 elements took a long time to complete the check. This is because the corresponding reset net contains 42 elements and as a result the weak soundness property results in 28 calls to the coverability algorithm (one for Weak option to complete, 19 for Proper completion and eight for Dead transitions). It is already known that the backwards coverability algorithm can be time consuming as it needs to calculate a finite basis of the predecessors for the entire net for each coverable method call (Wynn, Edmond, van der Aalst & ter Hofstede 2005). As the weak soundness property check requires 28 calls, the check is quite expensive for nets with a large state space. This experiment also highlights the need for further optimisation techniques to speed up the verification process.

Verifying irreducible cancellation regions

An element within a cancellation region of a task is not necessary if that element can never be marked when the task is being executed. Such elements are called reducible because they can be removed without changing the behaviour. To decide the irreducible cancellation regions property of a net without OR-joins, analysis on the corresponding reset net is performed. The cancellation region for the *Stop checks* task includes the *Finalise basic requirements check* task. As *Finalise basic requirements check* is an AND-join task, it can never be executed while the *Stop checks* task is being executed.

Therefore, it should not be in the cancellation region of the *Stop checks* task. This is reported by the YAWL editor as shown in Figure 6.

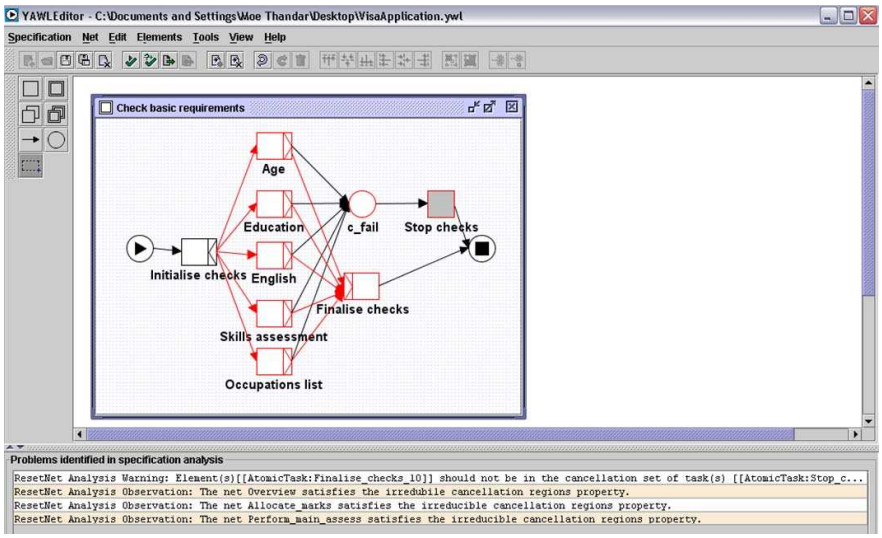

Fig. 6. Screenshot of the irreducible cancellation regions property check for modified *Check basic requirements* net

Verifying immutable OR-joins

An immutable OR-join is one that could not be replaced by either an XOR-join or an AND-join. In Figure 7, the split behaviour of the task *Decide applicable categories* has been changed from OR-split to AND-split for testing purposes. As the net now contains an AND-split followed by an OR-join, the OR-join should be more appropriately modelled as an AND-join.

In this section, we have demonstrated that verification of process models with more advanced constructs such as cancellations regions and OR-joins is indeed possible. Moreover, we demonstrated that the YAWL editor supports the verification process. However, given the complexity of some workflow models, we further improved our approach using so-called reduction rules. These have also been implemented in the YAWL editor and are described in the remainder.

Illustrating reduction rules

In the previous section, we have seen that when a workflow contains a large number of tasks and involves complex control flow dependencies, verification can take an extraordinary period of time or it may even be intractable. Applying reduction rules before carrying out verification could decrease the size of the problem by cutting down the size

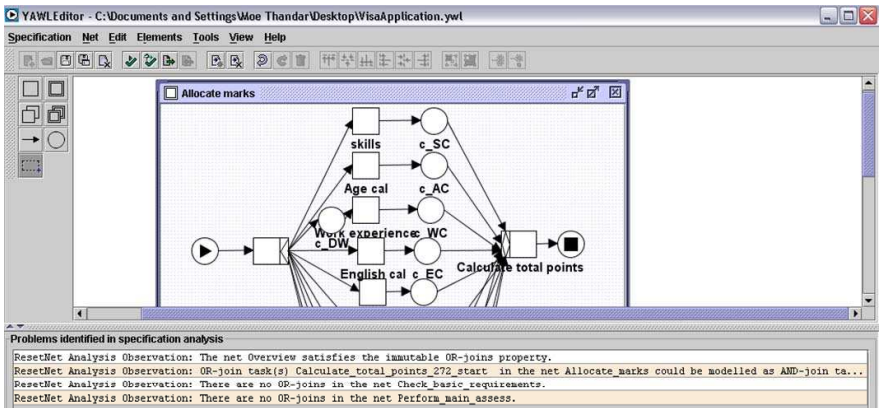

Fig. 7. Screenshot of the immutable OR-joins property check for the modified *Allocate marks* net

of the workflow that needs to be examined while preserving the soundness property. As a result, reduction rules could potentially decrease the average case complexity of performing verification. A number of soundness preserving reset reduction rules as well as YAWL reduction rules are proposed. Due to lack of space, we only provide a brief summary of these rules here in Figures 8 and 9. For further details, see (Wynn 2006). As all of these reduction rules are soundness preserving, it is possible to perform verification on the reduced nets instead of the original net. In general, applying these reduction rules before verification reduces the number of tasks and conditions being considered and hence, assists in speeding up the analysis. Reduction rules together with the new approach for verification using reduced nets are implemented in the YAWL editor.

Table 1 shows the effects of using YAWL and reset reduction rules to detect the soundness property for all nets in the *Visa application* process. The numbers in various columns represent the number of elements in the original net and in the corresponding reduced net. For example, the *Allocate marks* can be reduced significantly from 37 to 3 elements if YAWL reduction rules are applied first followed by the reset reduction rules. The efficiency gain from applying reduction rules is quite significant. The time it takes to verify the soundness property of the *Overview* net decreased from 24.3 sec to 4 sec. Similar gains can be seen for the other two nets: *Check basic requirements* and *Perform main assessment*. As for the *Allocate marks* net, the results are quite spectacular. Even though this net is a structured net - with corresponding OR-split and OR-join tasks, it has a large state space due to the various possible combinations of OR-split and OR-join. Without the use of reduction rules, the net suffers from the state explosion problem when determining the soundness property. After applying the reduction rules, the reduced *Allocate marks* net becomes quite trivial with just one input place, one output place and a task in between. As a result, the soundness check is completed almost instantaneously (less than one second). This is a huge improvement considering the fact that the soundness check for the *Allocate marks* net could not be completed in a reasonable time frame (more than 5 mins) due to state explosion.

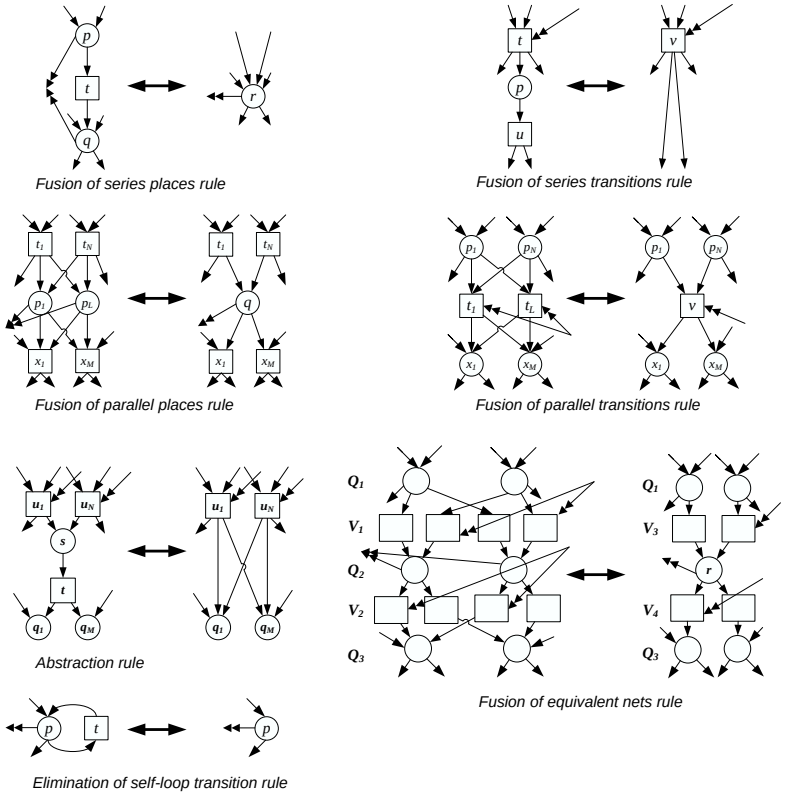

Fig. 8. Reset Reduction rules (Please note that the figure does not capture all the requirements for some rules (i.e., if a transition cannot have a reset arc (double-headed arc) or a place cannot be reset, the figure will not show this.)

Related work

Since the mid nineties, many researchers have been working on workflow verification techniques (van der Aalst 1997, van der Aalst 1998, van der Aalst 2000, Bi & Zhao 2004, Choi & Zhao 2005, Dehnert & Rittgen 2001, van Dongen, van der Aalst & Verbeek 2005, Hee, Sidorova & Voorhoeve 2004, Kindler, Martens & Reisig 2000, Mendling et al. 2006, Sadiq & Orlowska 1997, Sadiq & Orlowska 1999, Verbeek 2004, Verbeek, van der Aalst & ter Hofstede 2006, Verbeek, Basten & van der Aalst 2001, Wynn 2006, Wynn et al. 2005, Wynn et al. 2006*a*, Wynn et al. 2006*b*). It is impossible to give a complete overview here. Moreover, most of the papers on workflow verification focus on rather simple languages, e.g., AND/XOR-graphs which are even less expressive than classical Petri nets . Therefore, we only mention the work directly relevant for this paper.

The use of Petri nets in workflow verification have been studied before (van der Aalst 1997, van der Aalst 1998, Verbeek et al. 2001, Verbeek 2004). van der Aalst (2000) describes how structural properties of a workflow net can be used to detect the

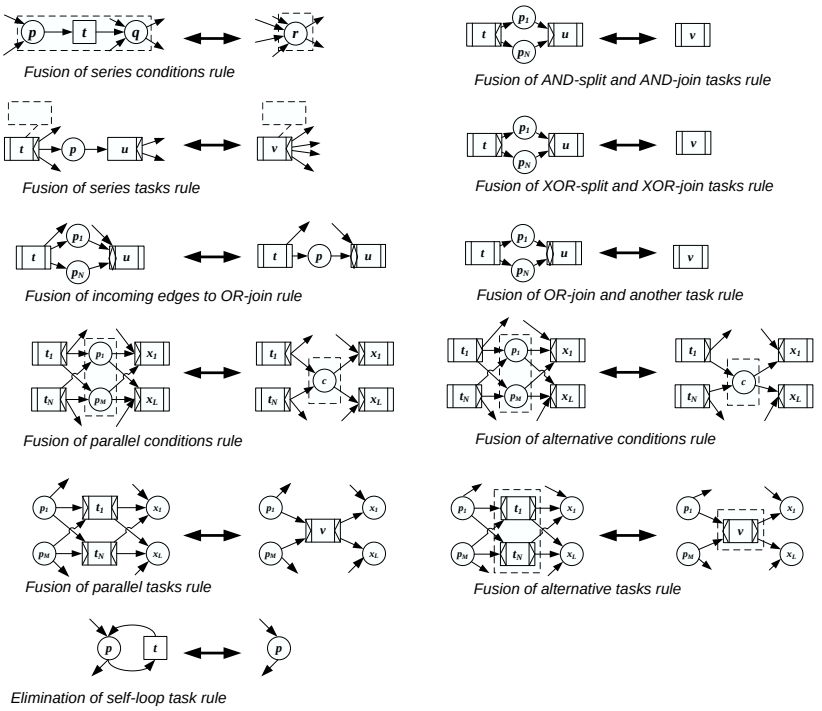

Fig. 9. YAWL Reduction rules

soundness property. Verbeek et al. (2006) present an alternative approach for deciding the relaxed soundness property using invariants. The approach taken results in the approximation of OR-join semantics and transformation of YAWL nets into Petri nets with inhibitor arcs. However, the use of inhibitor arcs instead of reset arcs means that this approach cannot detect problems in certain YAWL specifications with cancellation features. On the other hand, The approximation of OR-join semantics enables the verification of YAWL nets with OR-joins using invariants. It differs from our approach where the reachability analysis is carried out for a net with OR-joins to decide the soundness property. In the general area of reset nets, Dufourd et al.'s work has provided valuable insights into the decidability status of various properties of reset nets including reachability, boundedness and coverability (Dufourd et al. 1998, Dufourd et al. 1999).

A number of authors have investigated reduction rules for Petri nets and for various subclasses of Petri nets. In Murata's paper, six reduction rules are presented for Petri nets (Murata 1989) and this set of rules has been used as a starting point for the rules mentioned in this paper. In the book by Desel & Esparza (1995), a set of reduction rules are proposed for free-choice Petri nets while preserving well-formedness. Berthelot presents a set of reduction rules for general Petri nets (Berthelot 1986). Six reduction rules that preserve correctness for EPCs including reduction rules for trivial constructs, simple splits and joins, similar splits and joins, XOR loop and optional OR-loop have been proposed by van Dongen et al. (2005). However, these reduction rules do not take cancellation into account. Reduction rules have been suggested to be used together with

Number of elements	Overview	Main Assessment	Basic Requirements	Allocate Marks
Original (YAWL)	42	11	21	37
Reduced (YAWL)	28	7	None	3
Original (reset)	89	23	42	2119
Reduced (reset)	35	9	32	2051
Reduced (both)	35	9	32	3
Soundness (original)	24.3 sec	1.9 sec	26.4 sec	>5 mins
Soundness (reduced)	4 sec	0.8 sec	4.1 sec	0.7 sec

Table 1. Demonstrating the effects of reduction rules on soundness property check for *Visa application* process

Petri nets for the verification of workflows (cf. Chapter 4 of the book by van der Aalst & van Hee (2004)). We follow a similar approach with a set of reduction rules for workflow nets with cancellation regions using reset nets.

Recall the analysis of the SAP reference model presented in (Mendling et al. 2006), which is mentioned earlier. Here 604 EPC models were automatically translated to YAWL and analysed using invariants. Note that the translation from Event-driven Process Chains (EPCs) to YAWL is trivial because the EPC language can be seen as a proper subset of the YAWL language. The analysis technique used in (Mendling et al. 2006) (based on transition invariants) is less precise than the analysis described in this paper. Experiments show that using the approach described in this paper indeed reveals more errors but is also more time-consuming.

Conclusion

In this paper, we demonstrated that four desirable properties, i.e., *soundness*, *weak soundness*, *irreducible cancellation regions*, and *immutable OR-joins*, can be verified for *process models with cancellation regions and OR-joins*. The verification approach has been implemented in the context of YAWL and has been illustrated using the Australian *visa application* process. In this paper, we tried to avoid getting in technical details. However, all techniques have been implemented in the open-source tool YAWL[4] and are described in detail in the PhD thesis of the first author (Wynn 2006) which is available for download.[5]

The results presented in this paper show that *verification has become a reality*, i.e., even for languages with advanced constructs such as cancellation regions and OR-joins verification is feasible. Existing approaches presented in the literature tend to focus on very simple languages and are, therefore, not usable in a practical setting.

It is important to note that the verification techniques presented in this paper are *transferable to any other workflow language*. This is particularly interesting for languages that are expressive enough to support cancellation regions and/or OR-joins. Examples of such languages are the Business Process Modelling Notation (BPMN), UML

[4] http://www.yawl-system.com

[5] http://yawlfoundation.org/documents/MoeWynn_Thesis_FinalVersion.pdf

Activity Diagrams (UML-AD), Event-driven Process Chains (EPCs), and Business Process Execution Language (BPEL).

References

van der Aalst, W. (1997), Verification of Workflow Nets, *in* P. Azéma & G.Balbo, eds, 'Proceedings of Application and Theory of Petri Nets', Vol. 1248 of *Lecture Notes in Computer Science*, Springer-Verlag, Toulouse, France, pp. 407–426.

van der Aalst, W. (1998), 'The Application of Petri Nets to Workflow Management', *The Journal of Circuits, Systems and Computers* **8**(1), 21–66.

van der Aalst, W. (2000), Workflow Verification: Finding Control-Flow Errors using Petri Net-Based Techniques, *in* W. van der Aalst, J. Desel & A. Oberweis, eds, 'Proceedings of Business Process Management: Models, Techniques and Empirical Studies', Vol. 1806 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 161–183.

van der Aalst, W. & ter Hofstede, A. (2005), 'YAWL: Yet Another Workflow Language', *Information Systems* **30**(4), 245–275.

van der Aalst, W., ter Hofstede, A., Kiepuszewski, B. & Barros, A. (2003), 'Workflow Patterns', *Distributed and Parallel Databases* **14**, 5–51.

van der Aalst, W. & van Hee, K. (2004), *Workflow Management: Models, Methods and Systems*, MIT press, Cambridge, MA.

Berthelot, G. (1986), Transformations and Decompositions of Nets, *in* W. Brauer, W. Reisig & G. Rozenberg, eds, 'Petri Nets: Central Models and Their Properties, Advances in Petri Nets, Proceedings of an Advanced Course, Part 1', Vol. 254 of *Lecture Notes in Computer Science*, Springer-Verlag, Bad Honnef, pp. 359–376.

Bi, H. & Zhao, J. (2004), 'Applying Propositional Logic to Workflow Verification', *Information Technology and Management* **5**(3-4), 293–318.

Choi, Y. & Zhao, J. (2005), Decomposition-based Verification of Cyclic workflows, *in* D. Peled & Y.-K. Tsay, eds, 'Proceedings of Automated Technology for Verification and Analysis (ATVA 2005)', Vol. 3707 of *Lecture Notes in Computer Science*, Springer-Verlag, Taipei, Taiwan, pp. 84–98.

Dehnert, J. & Rittgen, P. (2001), Relaxed Soundness of Business Processes, *in* K. Dittrich, A. Geppert & M. Norrie, eds, 'Proceedings of the 13th International Conference on Advanced Information Systems Engineering (CAiSE'01)', Vol. 2068 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 157–170.

Desel, J. & Esparza, J. (1995), *Free Choice Petri Nets*, Vol. 40 of *Cambridge Tracts in Theoretical Computer Science*, Cambridge University Press, Cambridge, United Kingdom.

van Dongen, B., van der Aalst, W. & Verbeek, H. (2005), Verification of EPCs: Using Reduction rules and Petri Nets, *in* O.Pastor & J. F. e Cunha, eds, 'Proceedings of the 17th Conference on Advanced Information Systems Engineering (CAiSE 2005)', Vol. 3520 of *Lecture Notes in Computer Science*, Springer-Verlag, Porto, Portugal, pp. 372–386.

Dufourd, C., Finkel, A. & Schnoebelen, P. (1998), Reset Nets Between Decidability and Undecidability, *in* K. Larsen, S. Skyum & G. Winskel, eds, 'Proceedings of the 25th International Colloquium on Automata, Languages and Programming', Vol. 1443 of *Lecture Notes in Computer Science*, Springer-Verlag, Aalborg, Denmark, pp. 103–115.

Dufourd, C., Jančar, P. & Schnoebelen, P. (1999), Boundedness of Reset P/T Nets, *in* J. Wiedermann, P. Boas & M. Nielsen, eds, 'Lectures on Concurrency and Petri Nets', Vol. 1644 of *Lecture Notes in Computer Science*, Springer-Verlag, Prague, Czech Republic, pp. 301–310.

Finkel, A. & Schnoebelen, P. (2001), 'Well-structured Transition Systems everywhere!', *Theoretical Computer Science* **256**(1–2), 63–92.

Hee, K., Sidorova, N. & Voorhoeve, M. (2004), Generalised Soundness of Workflow Nets Is Decidable, *in* J. Cortadella & W. Reisig, eds, 'Application and Theory of Petri Nets 2004', Vol. 3099 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 197–215.

Kindler, E., Martens, A. & Reisig, W. (2000), Inter-Operability of Workflow Applications: Local Criteria for Global Soundness, *in* W. van der Aalst, J. Desel & A. Oberweis, eds, 'Business Process Management: Models, Techniques, and Empirical Studies', Vol. 1806 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 235–253.

Mendling, J., Moser, M., Neumann, G., Verbeek, H., Dongen, B. & van der Aalst, W. (2006), Faulty EPCs in the SAP Reference Model, *in* S. Dustdar, J. Faideiro & A. Sheth, eds, 'International Conference on Business Process Management (BPM 2006)', Vol. 4102 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 451–457.

Murata, T. (1989), 'Petri nets: Properties, Analysis and Applications', *Proceedings of the IEEE* **77**(4), 541–580.

Sadiq, W. & Orlowska, M. (1997), On Correctness Issues in Conceptual Modeling of Workflows, *in* 'Proceedings of the 5th European Conference on Information Systems (ECIS '97)', Cork, Ireland, pp. 19–21.

Sadiq, W. & Orlowska, M. (1999), Applying Graph Reduction Techniques for Identifying Structural Conflicts in Process Models, *in* M. Jarke & A. Oberweis, eds, 'Proceedings of the 11th Conference on Advanced Information Systems Engineering (CAiSE 1999)', Vol. 1626 of *Lecture Notes in Computer Science*, Springer-Verlag, Heidelberg, Germany, pp. 195–209.

Sloan, R. & Buy, U. (1996), 'Reduction Rules for Time Petri Nets', *Acta Informatica* **33**(7), 687–706.

Verbeek, H. (2004), Verification of WF-nets, PhD thesis, Eindhoven University of Technology, Eindhoven, The Netherlands.

Verbeek, H., Basten, T. & van der Aalst, W. (2001), 'Diagnosing workflow processes using Woflan', *The Computer Journal* **44**(4), 246–279.

Verbeek, H., van der Aalst, W. & ter Hofstede, A. (2006), 'Verifying Workflows with Cancellation Regions and OR-joins: An Approach Based on Relaxed Soundness and Invariants', *The Computer Journal* . to appear.

Wynn, M. (2006), Semantics, Verification, and Implementation of Workflows with Cancellation Regions and OR-joins, PhD Thesis, Faculty of Information Technology, Queensland University of Technology.

Wynn, M., Edmond, D., van der Aalst, W. & ter Hofstede, A. (2005), Achieving a General, Formal and Decidable Approach to the OR-join in Workflow using Reset nets, *in* G. Ciardo & P. Darondeau, eds, 'Proceedings of ATPN', Vol. 3536 of *Lecture Notes in Computer Science*, Springer-Verlag, Miami, USA, pp. 423–443.

Wynn, M., Verbeek, H., van der Aalst, W., ter Hofstede, A. & Edmond, D. (2006*a*), Reduction rules for Reset Workflow Nets, Technical report BPM-06-25, BPM Center (bpmcenter.org).

Wynn, M., Verbeek, H., van der Aalst, W., ter Hofstede, A. & Edmond, D. (2006*b*), Reduction rules for Workflows With Cancellation Regions and OR-joins, Technical report BPM-06-24, BPM Center (bpmcenter.org).