Conceptual-to-workflow model transformation guidelines

Olaka, J.D.

Award date:
2012

Technische Universiteit
Eindhoven
University of Technology

Department of Mathematics and Computer Science

Conceptual-to-Workflow Model Transformation Guidelines

Olaka Joseph Derrick

In partial fulfillment of the requirements for
The degree of Master of Science in
Business Information Systems

Supervisor:
dr.ir. R.M. (Remco) Dijkman

Assessment Committee:
dr.ir. R.M. (Remco) Dijkman

dr.ir. B.F. (Boudewijn) van Dongen

MSc H.L. (Heidi) Romero

Eindhoven, September 2012

Table of Contents

1. Introduction

A typical business process life cycle is structured into design, implementation, deployment and evaluation phases. During these phases, different process models are employed. The design phase encompasses development of conceptual process models from a business analyst perspective [32]. The resulting conceptual models in the design phase serve as an impetus for developing implementation models in the form of executable workflow specifications to help technical analysts. Workflow specifications developed during the implementation phase is then deployed on a workflow engine. Finally, the deployed process is examined and evaluated for improvement purposes in subsequent iteration of the business process life cycle [45]. Currently, the development of enterprise applications starts with conceptual process models followed by transformation of these models to workflow process models [32]. Therefore, in the business process management (BPM) domain, both the conceptual process model and workflow process model are required. A conceptual process model is a high level overview (usually graphical description) of the different activities to be executed in a current or envisioned business process. They generally present an abstracted view of a business process with very minimal technical details (if any) relating to the process. Conceptual process models are typically made by domain experts/business analysts and are predominantly used as a basis for communication. Workflow process models are representations of application processes to be used by workflow management systems for controlling the execution of workflows. It presents a concrete view of a business process with very minimal abstractions (if any) as well as the implementation details associated with it. Workflow process models are made by IT-experts/solution architects and serves as input to a WFMS. A workflow management system (WFMS) is a generic software tool which allows for the definition, execution, registration and control of workflows [40]. A workflow process model must be unambiguous and should not contain any uncertainties. This is a necessary requirement in order to analyze and simulate the described processes and to monitor their execution at run-time. Transforming conceptual process models to workflow process models bridges the gap between design of a process model and its actual implementation. Unfortunately, this transformation in most cases is not straight-forward and often results in mismatches.

Numerous approaches to transform conceptual process models to workflow process models have been reported in literature [13,15,22,28,39,83]. However, they majorly present imperative approaches describing *how* to transform a specific type of conceptual process model (for instance EPC) to specific type of workflow process model (for instance BPEL). The transformation algorithms presented therefore have a limited range of applicability. Furthermore, there are no comprehensive and general set guidelines to be followed during the model transformation hence the quality of the outcome of transformation process is not usually guaranteed. In this research, we shall propose a set of conceptual to workflow transformation guidelines. Figure 1 represents a high-level overview of the concept of our research.

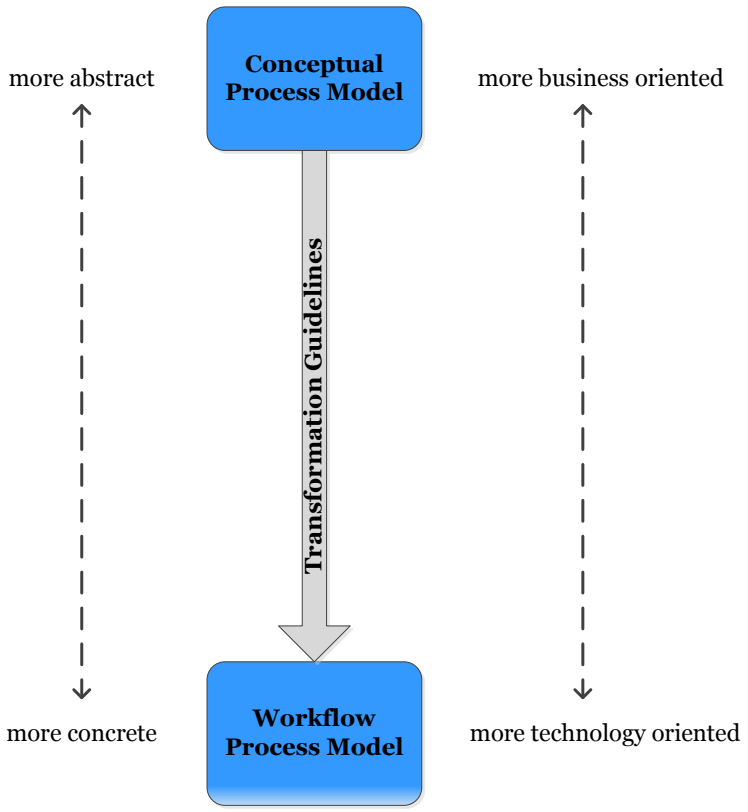


Figure 1. Conceptual Process Model to Workflow Process Model Transformation

1.1 Problem Statement

In order to automate the control, management, and execution of the business process depicted in the conceptual process model, it has to be transformed into a workflow process model. The workflow process model serves as input to the WFMS. Therefore, there is a dire need to bridge the gap between these two concerns: the need for a plain communication basis on the one side and an unambiguous process description, covering details of the implementation, on the other side in any practical BPM project. According to Dehnert et al.[14], existing WFMSs follow a pragmatic approach in trying to bridge this gap. They often use a proprietary modeling language with an intuitive graphical layout. However, the underlying semantics of such a language lacks a formal foundation. As a consequence, analysis issues, such as checking for correctness and reliable execution are not supported at design time [14]. Such potentials flaws can therefore only be realized during process execution. It is clear that flaws detected at run time are generally more costly that those detected during the design phase.

Various specific algorithms have been developed and implemented to transform conceptual process models to workflow process models. According to Dumas et al. [22] ideally, model transformations approaches should achieve four evaluation criteria: completeness, correctness, readability, and reversibility. Completeness in this case refers to the extent to which the transformation involves different classes of process models (both conceptual and workflow process model). The major classes of process model are block-structured process models (such as BPEL model), quasi-structured process models (processes that can be re-written into perfectly structured ones) and graph-structured process model (such as BPMN model) [6,22,44,55,81,83]. Ideally, all

valid input models should be considered by the transformation [39]. With regards to correctness, in addition to ensuring soundness and syntactic correctness of the transformed models, we are also interested in preserving the execution semantics of the original conceptual process model during the transformation. Therefore, the workflow process model obtained after the transformation should be a weak-bisimulation equivalent to the original conceptual process model. However, majority of these researches does not pay attention to these four evaluation criteria. In order to reverse this trend, there is need for a comprehensive, declarative and generic set of transformation guidelines describing what procedures should be followed during the entire model transformation process keeping in mind the four above mentioned evaluation criteria. However, such a comprehensive set of guidelines is still lacking today. There was no single paper that explicitly provided a comprehensive set of transformation guidelines. Rather, very scanty one or two best practices could be abductively derived from certain literature sources.

In this research, we present a set of guidelines for transforming conceptual process models to workflow process models. Following the guidelines proposed in this research should ensure a systematic transformation process and improve on the completeness and correctness of the process of transforming conceptual process models to workflow process models.

1.2 Research Objectives and Scope

The main goal of our study is to develop and validate guidelines for transforming conceptual process models to workflow process models. Specific research questions to be answered under this main research goal are;

- Are there any existing comprehensive set of guidelines for transforming conceptual process models to workflow process models?
- If yes, are they generic set of guidelines, what are their strengths and weaknesses?
- If no, propose new set of generic transformation guidelines
- Which of these proposed transformation guidelines are usable in practice?
- How often is each guideline used in practice?

1.3 Report Structure

This section contains a brief outline of how the rest of the report is structured.

- Chapter 2 presents the research methodology used.
- Chapter 3 discusses in depth conceptual and workflow process models. This serves as background material on which our research is positioned.
- Chapter 4 presents a review of literature about modeling guidelines
- Chapter 5 explains in detail the proposed conceptual to workflow model transformation guidelines
- Chapter 6 presents a validation and evaluation of the proposed guidelines using conceptual process models from Philips healthcare.
- Chapter 7 explains the conclusion of research and proposed future work

2. Research Methodology

In this chapter, section 2.1 presents a general introduction to the two main research paradigms in information systems. Section 2.2 presents our research methodology which is based on the design science research paradigm chosen from section 2.1.

2.1 Research Paradigms in Information Systems

There are basically two paradigms that characterize much of the research in the information systems discipline, namely behavioral science paradigm and design science paradigm.

The behavioral science paradigm seeks to develop and verify theories that explain or predict human or organizational phenomena surrounding the analysis, design, implementation, management, and use of information systems[29]. The theories to be developed or justified using the behavioral science research paradigm describes the interactions among people, technology, and organizations that must be managed if an information system is to achieve its stated objectives[29]. Data collection and empirical analysis is the core technique used when carrying out behavioral science research. Behavioral science research paradigm originated from natural science research methods.

The design science paradigm seeks to extend the boundaries of human and organizational capabilities by creating new and innovative artifacts[29]. It seeks to create innovations that define the ideas, practices, technical capabilities, and products through which the analysis, design, implementation, management, and use of information systems can be effectively and efficiently accomplished[17]. In design science, computational and mathematical methods are primarily used to evaluate the quality and effectiveness of artifacts; however, empirical techniques may also be employed[29]. Design science which is fundamentally a problem solving research paradigm originated from the engineering discipline and the sciences of the artificial.

We used the design science research paradigm during our study. This is because our research seeks to create guidelines to be followed when transforming conceptual process models to workflow process models. This will eventually improve on the efficiency and effectiveness of the transformation process. Based, on this research focus, the design science research paradigm is the most applicable. In the subsequent sections, we shall briefly explain design science research, and how it has been applied in our research work.

2.2 Research Methodology Based on Design Science Research Paradigm

The design science research paradigm is based on an analysis of the reasoning that occurs during a general design cycle as presented by Takeda et al[70] and is shown in

Figure 2 below. The figure also shows what we do in each step of the design cycle [72] in the context of this research. These steps are explained below in more detail.

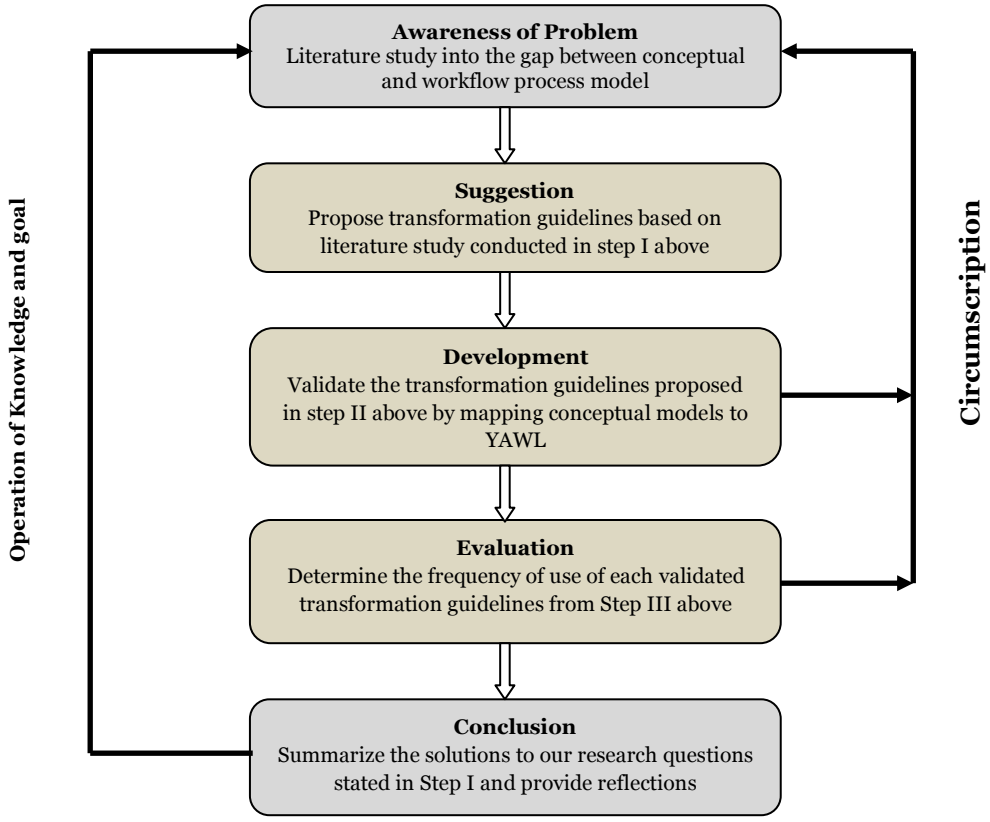


Figure 2. Reasoning on Design Cycle

2.2.1 Awareness of the Problem

This was the starting point of this research. The goal of this step is to create a general understanding of the problem of transforming a conceptual process model to a workflow process model. We perform this step by doing extensive literature study based on the systematic literature review methodology [9,34,35].

Literature Review Protocol Used

A literature review is an evaluation of the current body of knowledge in a specific subject area of interest. The main aim of conducting a literature review is to comprehend what the other researchers have accomplished, identify the strengths and weaknesses of their work and determine how our future work will leverage from their work and contribute to the current body of existing knowledge. An iterative literature search strategy with feedback loops was adopted. Three different steps were followed in conducting this literature review as highlighted in the Figure 3 below. The dotted arrows indicate feedback loops while the thick arrows show the normal flow of control. The overall literature review protocol is made up of three subsections based on the stages of the search methodology used.

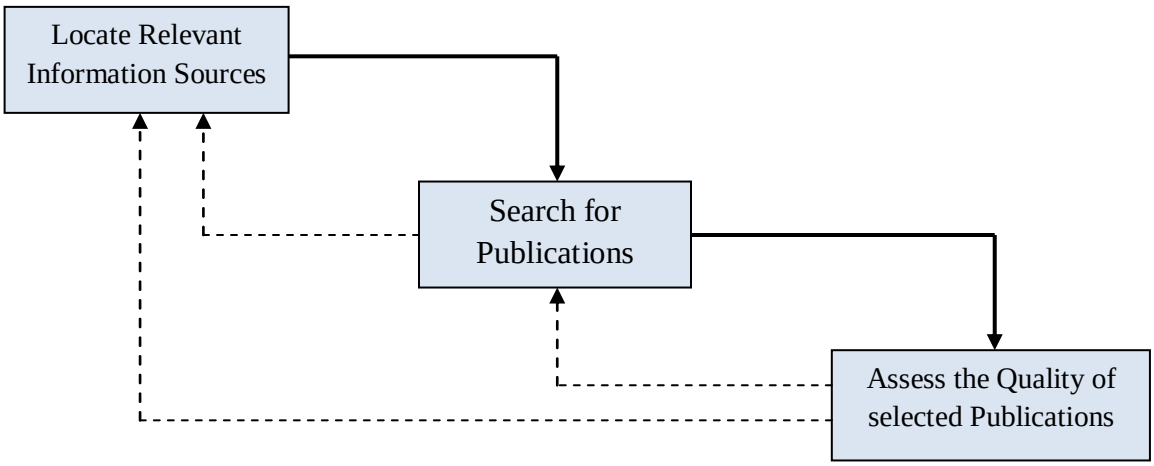


Figure 3. Literature Review Protocol

A detailed explanation of how each of the above process steps was conducted is presented in the subsections below.

Locate Relevant Information Sources

The main goal of this stage was to identify the different information sources where we shall search for the relevant publications. For this study, we used a couple of literature sources namely;

- Google scholar
- ACM digital library
- IEEE xplore digital library
- Springer link digital library, and
- Sciencedirect information database
- Emerald digital library

These sources were chosen because publications can be easily downloaded from them free of charge through the TU/e library account.

Search for Publications

Searching for the relevant publications was done in three different phases as explained below.

- ***Define Search Keywords***: The following search keywords were defined; *conceptual process models, workflow nets, workflow languages, performing*

systematic literature reviews, Workflow Model Representation, *workflow modeling*, *business process modelling, process modeling guidelines, Protos, process modeling languages, modeling methods, viewpoint modeling*, *model transformation, conceptual to workflow model transformation, process model transformation guidelines, conceptual to workflow transformation guidelines*. Synonyms of some of the keywords were also used as a new keyword for the search process.

- ***Search in the Relevant Information Sources***: Using the keywords defined in phase 1 above, a thorough search was performed in all the relevant information sources mentioned above. Inclusion and exclusion criteria's were defined to assess the relevance of the publications retrieved during the search process. These criteria's are explained below.

  A publication is considered potentially relevant for this study if it satisfies the following conditions
  - The study is published in English
  - It presents modeling guidelines, modeling notations, model aspects/perspectives, or workflow patterns supported by various modeling notations, transformation guidelines, and systematic literature review.
  - It explicitly studies modeling from either a conceptual or workflow viewpoint or both.
  - It explicitly studies modeling from the conceptual to workflow viewpoint.

  In addition to the inclusion criteria presented above, one strict exclusion criteria was also defined as explained below.
  - No full text available free of charge through the TU/e library for a publication. Any publication which we were not able to download the entire document free of charge through the library subscriptions were not used. There are cases were only the abstract of a paper can be read online but in order to download the full document, one has to pay. Such papers were immediately left out in this research.

  Only publications that satisfy the first inclusion criterion and at least one of the other inclusion criteria but not the exclusion criterion were kept for further consideration. All the kept publications were later subjected to quality assessment as explained in detail in the next subsection concerning assessment of the quality of selected publications.

- ***Perform Secondary Search***: In this phase, two main activities were performed;
  - For each of the relevant publications found in the previous phase, their respective references were also checked to determine whether we can get other relevant publications cited by it which are of interest for this study. Publications will always cite other publications within the same or related research area under consideration.

- For each of the author(s) of the relevant publications found in the previous phase, Google scholar was also used to check if more relevant publications for our research can found by the same author(s).

The above search strategy was adopted so as to ensure that the search was exhaustive and accurate enough.

Assess Quality of Selected Publication

The main goal of this stage was to further scrutinize all the potentially relevant publications found in the previous steps to ensure that they are of high quality. The publications are judged based on an explicitly stated criterion to determine which ones are of the sufficient quality to be included in the review synthesis [35]. The main quality criterion used to judge the quality of the publications is stated below.

A publication should be cited in at least four other scientific publications. Exceptions to this rule are allowed for publications that are not older than March 2011. In addition, websites are also exempted from this rule.

Google scholar was primarily used to check the number of scientific publications that cite a given publication.

A paper which didn't meet the above quality evaluation criteria but ultimately was used in this study was by L. Wedemeijer & E.G.A.J. de Bruin et al. [78]. This paper according to google scholar was cited by only one other scientific publication despite being published in 2004. However, we felt this paper was quite useful for us specifically since it presented a practical example of how conceptual modeling was used in practice within a Dutch pension fund system. We also feel that the probably reason for it not being widely cited could be because of the small domain of the business case it is reporting on (Dutch pension fund).

Examples of publications retrieved using this search protocol is listed in the reference section of this report.

The output of this step (awareness of the problem) is the realization of the lack of a comprehensive set of guidelines to be followed when transforming conceptual process models to workflow process models. A detailed discussion of the output of this step can be found in section 1.1.

2.2.2 Suggestion

The goal of this step is propose conceptual to workflow model transformation guidelines based on the literature study conducted in previous step (section 2.2.1). The result of this step is a list of guidelines to transform conceptual process model to workflow process model discussed in detail in section 5.

2.2.3 Development

The goal of this step is to validate the guidelines proposed in the previous step (section 2.2.2). Nine major conceptual process models (into total twenty distinct process models) from the Philips healthcare RS Processes were selected and transformed into YAWL workflow. During the transformation process, an analysis was made to determine which guidelines derived from literature during the previous stage was actually usable and/or not usable. Furthermore, new additional guidelines that were discovered during the transformation process are explained. The result of this step is a list of used, unused and newly discovered transformation guidelines as reported in detail in section 6. The implementation YAWL models used for the validation are in the appendices.

2.2.4 Evaluation

The goal of this step is to determine the frequency of use of each of the validated guidelines from the previous step (section 2.2.3). This will bright to light the level of importance of each of the transformation guidelines. The output is statistics indicating how often the guidelines were used during the validation stage. Detailed information about the output of this stage is presented in section 6.5.

2.2.5 Conclusion

The goal of this step is to present a summarized solution to the research questions earlier defined in section 2.2.1. Furthermore, the limitations of our research as well as potential future research areas are highlighted. A detailed discussion of the result of this step is contained in section 7.

3. Conceptual and Workflow Process Models

In this chapter, detailed explanation of the conceptual process models (sometimes referred to a business process models) and workflow process models that corresponds to the two viewpoints of process modeling namely; conceptual and workflow viewpoints are discussed. Section 3.1 discusses conceptual process models and section 3.2 discusses workflow process models. The approach of viewpoint modeling as presented in IEEE standard 1471 [30] is used to analyze both the conceptual and workflow process models. A viewpoint, as defined in IEEE 1471, defines a set of architectural concerns and the resources needed to address those concerns [30]. These resources may include notations, techniques, and guidance. A viewpoint establishes the audience for a view and the methods or techniques employed in constructing the view [30]. In this regards, our architectural concerns are "*what is a conceptual process model?*" and "*what is a workflow process model?*" To address these two architectural concerns in depth, we shall explain *model usage, modeling languages*, and *model aspects* for each concern in the subsequent sections.

3.1 Conceptual Process Model

A conceptual process model is a high level overview (usually graphical description) of the different activities to be executed in a current or envisioned business process [78]. They generally present as an abstracted view of a business process with very minimal technical details (if any) relating to the process. Conceptual process models are intended to be easily comprehensible by different stakeholders such as information systems experts and domain experts and may not need to be executable. Due to the high level of abstraction exhibited in conceptual process models, it makes it independent of the future software implementation packages which will be used to realize it. Therefore from a conceptual process model, various implementation options can be devised to meet the business requirements of a particular company. According to L. Wedemeijer & E.G.A.J. de Bruin et al. [78], the notion of conceptual process model is better understood by its purpose which is to outline all actions indispensable to produce all of the essential results in a customer-centered business process, regardless how, when, by whom or by what means these outputs are produced.

3.1.1 Uses of Conceptual Process Models

There exist numerous uses of conceptual process models, the key notable one's are explained below.

- Conceptual process models are used as checkpoints to verify compliance of the proposed process redesign with the overall business information architecture and requirements[78].
- In addition, conceptual process models forces the core features and business logic to be firmly established, before the complexities and implementation details of the redesign are decided upon[78]. This greatly helps to reduce on the high cost of rework which could have been incurred due to an earlier poorly understood business logic or requirement.

- Conceptual process models are also used to pinpoint the essential aspects of the business process, providing the organization's key players with a shared understanding of the customer event, process deliverables, and indispensable actions/activities[78]. This helps to secure the management commitment towards the project which is one of the key factors for the success of any reengineering project.
- Conceptual process models are also used to establish the scope of the business process being redesigned.

According to Jörg Becker et al. [8], conceptual process modeling is supposed to be an instrument for coping with the complexity of process planning and control. However, existing models may show as well considerable complexity themselves. In summary, conceptual process models are used as a communication tool among different stakeholders in the modeling project in an organization.

3.1.2 Conceptual Process Modeling Languages

Languages used for conceptual process modeling includes UML Activity diagram, EPCs, BPMN, Integrated DEFinition for Process Description Capture Method (IDEF3), Role Activity Diagrams (RAD). A brief explanation of each language is presented below.

Business Process Model and Notation (BPMN): It defines a Business Process Diagram (BPD), based on a flowcharting technique tailored for creating graphical models of business process operations [76]. BPMN provide a standard notation that is intuitive, understandable and usable by all stakeholders in an organization. BPMN [19] is only applicable in the modeling of business processes and excludes the modeling of data models, organizational structures and functional breakdowns. The objective of BPMN is to support business process management by both technical users and business users by providing a notation that is intuitive to business users yet able to represent complex process semantics [69].The BPMN specification also provides a mapping between the graphics of the notation to the underlying constructs of execution languages, particularlyBPEL4WS (or WS-BPEL) [69]. Currently, BPMN is maintained by the Object Management Group (OMG) but it was initially developed by the Business Process Management Initiative (BPMI).

UML Activity Diagram: In general, UML is used in the object-oriented software engineering domain as a standardized general purpose modeling language. UML2 contains 13 diagrams majority of which are used for modeling of object-oriented software, however, the UML activity diagram is the most appropriate amongst the 13 diagrams for modeling business processes (conceptual processes) [76]. The main concepts of the UML Activity diagrams are actions and swimlanes, whereby the latter represent roles [42]. UML was created and is still being maintained by the Object Management Group (OMG).

IDEF3: The IDEF3 Process Description Capture Method provides a mechanism for collecting and documenting processes. IDEF3 captures precedence and causality relations between situations and events in a form natural to domain experts by

providing a structured method for expressing knowledge about how a system, process, or organization works. There are two description modes of IDEF3, namely process flow and object state transition network. A process flow description captures knowledge of "how things work" in an organization, e.g., the description of what happens to a part as it flows through a sequence of manufacturing processes. The object state transition network description summarizes the allowable transitions an object may undergo throughout a particular process. Both the Process Flow Description and Object State Transition Description contain units of information that make up the system description. Further explanation of IDEF3 is also contained in [42].

Event Driven Process Chains (EPC): EPC consists of two main elements namely; functions and events. Functions are active elements and models activities of a business process within a company. Events are passive elements [68] and are created by either an actor outside of the current model or by executing of the functions in the current model. Connectors could possibly be used to link functions and events [52]. Functions have exactly one inbound arc and one outbound arc [52]. However, events have atmost one inbound arc and atmost one outbound arc [52]. For the connectors, they can have multiple inbound arcs and a single outbound arc or vice versa. EPC has been specifically developed for modeling of conceptual business processes with the goal to be easily understood and used by business people [42].

Role Activity Diagrams (RAD): RAD is used in the modeling of the current business processes as well as the future potential business process. They show roles, their activities and interactions, together with external events [42]. RAD uses a notation similar to that in flowcharts to model business processes at a conceptual level. RAD supports a number of abstraction mechanisms, including process composition and process encapsulation/decomposition [52].

Modeling languages that belong to the conceptual modeling viewpoint generally have more intuitive graphical presentation constructs which require very minimal effort to understand even by non modeling professionals. This corresponds with the major use of a conceptual process models which is to facilitate communication among stakeholders of potentially diverse backgrounds. The ease of understandability of conceptual process modeling languages is geared towards facilitating the main usage of conceptual process models.

3.1.3 Process Model Aspects

To adequately describe a business process, many forms of information must be integrated into a process model [42]. These pieces of information typically includes what activities will be executed, who is responsible for its execution, where and when will it be executed, how and why will it be executed, and the sequence of executions of the various activities as well as its dependence with other activities [11]. Components of process models are organized in a modular [73] fashion using the notion of model aspects. This greatly reduced on the complexity of analyzing a process model. In this study, the different categories of the information that are integrated in a process model are called model aspects. The model aspects that are applicable to conceptual process

models differ with those that are strictly applicable to workflow process models. This difference is partly brought about by the extent to which the process modeling languages used to create these two types of models are able to support the integration of the above mentioned pieces of information in the model [42]. This section presents the model aspects that are common to both the conceptual and workflow process model. There are mainly four different model aspects that belong to this category and are widely reported in literature [8,11,42,51,52,71,79]. They include function, information, organizational, and Behavioural aspects. More explanations on each of these aspects are given below.

1. *Function Aspect*: The process elements that are being performed are contained in the function aspect [11]. The basic elements of a business process are activities. They can be either atomic activities or sub-processes, which are recursively refined by activities [42]. The function aspect also presents the functional dependencies between the process elements such as activities and sub-processes [52]. Notations similar to that of a data flow diagram are usually used to encode the function aspect.

2. *Data/Information Aspect*: The information aspects contains the informational entities produced or manipulated by a process; these entities include data, artefacts, products (intermediate and end), and objects [11]. The basic elements of the informational aspect are resources and events [42]. Activities are triggered by events. Entities to be produced or consumed by an atomic activity are referred to as Resources. The basic principle of the information aspect is quite straightforward. However, there exist many technical issues to solve, for instance different data formats of a given data flow, which may require the use of filters to allow seamless integration of different tools using different data formats [79].

3. *Organizational Aspect*: The organization aspect describes where and by whom (which agents) process elements such as functions are performed [11]. This includes the description of the participants in the process as well as a description of the physical (location) and organizational context within which this process is conducted [52].

4. *Control/Behavioural Aspect*: The behavioral aspect represents which elements are performed (e.g., sequencing), as well as details of how they are performed through feedback loops, iteration, complex decision-making conditions, entry and exit criteria, and so forth [11]. This basically represents the relative ordering of the sub-workflows. Examples of elements that belong to this aspect are basic control flow patterns such as XOR Join, XOR Split, AND Join, AND Split and Sequence which constitute part of the control logic [52]. An important point to note is that these basic elements are not workflow specific but is also is used with conceptual business process models [42]. However, there exist more advanced control flow patterns such as critical section and interleaved routing which still fall under the control aspect and are specific to workflow models. The information specified in the behavioral aspect of workflow models is important for a workflow management system to control the execution of workflows [79].

Even though the above four mentioned aspects apply to both conceptual and workflow process modeling, they are more exhaustively and extensively used in the workflow process modeling.

3.2 Workflow Process Model

Workflow process models are representations of application processes to be used by workflow management systems for controlling the execution of workflows[79]. They are specified by workflow modeling languages. A more formal definition of a workflow process model (also called workflow net) is presented by Aalst et al.[5]. A workflow process models presents a concrete view of a business process with very minimal abstractions (if any) as well as the technological details associated with it.

Unlike conceptual process models, workflow models are mainly used by the technical experts actively involved in the project. Workflow models serves as input to the workflow management system which will automatically coordinate the task composing the workflow model [23]. In a WFMS, a task is a generic piece of work defined not for a specific case but for a type of cases [3]. A work item is a task which is enabled for a specific case. An activity is the actual execution of a work item, i.e., a task is executed for a specific case. In terms of petri-net terminology, a task corresponds to a transition, a work item corresponds to an enabled transition (binding) and an activity corresponds to a transition firing. Usually several conceptual process models are later transformed into few workflow models which are used by the technical project team members. Besides the transformation, information which is relevant for the controlled execution of workflows by a workflow management system is added to the model [79]. On the other hand, information which is irrelevant for workflow executions is omitted from the conceptual process model. Therefore, transforming conceptual process models to workflow process models entails abstracting from irrelevant information and adding relevant information mainly of technical nature [79].

3.2.1 Uses of Workflow Process Models

There key notable uses of workflow process models are briefly explained below.

- Workflow models are used as communication platform among the project team members during the planning and development phases of a software system.
- Workflow models are also specifically used for enactment purposes [57]. This implies that based on a workflow model, an enterprise information system like a WFMS or enterprise resource planning (ERP) system can be used to manage and control a workflow.
- Unlike conceptual process models, Workflow models are also uniquely used for analysis and simulation purposes [57]. This is aided by the executable specification of a typical workflow. The behavior of the workflow can then be simulated under varying operating environments/parameters so as to gain more insights on its performance.
- According to H.A. Reijers et al.[57], it is also stated that workflow models can be used for documentation and knowledge management. The workflow model can

be used as a backbone for work instructions on each of its tasks such that instructions can be consulted by the resources responsible for their execution. Therefore, knowledge about for instance regulations or exceptions can be in cooperated into the workflow model making it serve as an operational knowledge base.

Workflow models are also used to verify the syntactic and semantic correctness of application being developed since they can be simulated and tested.

3.2.2 Workflow Process Modeling Languages

There is no standard modeling language that is universal for all workflow models; every WFMS has its own proprietary modeling technique. Modeling languages used for workflow models include WS-BPEL, Workflow Nets, XML Process Definition Language (XPDL), YAWL.

WS-BPEL: WS-BPEL [42,76] is a standard executable language used for the definition of business processes using web services. Information is imported and exported by processes in BPEL via web service interfaces. From a more general perspective, web service interactions are categorized into executable business processes and abstract business processes [56]. Executable processes describe the actual business processes that are internal to an organization and are completely specified (i.e executable) [52]. Abstract processes describe parts of a business process of an enterprise that are exposed to the outside processes in the context of inter-enterprise interactions [52]. WS-BPEL is used to model both the abstract and executable processes specifications namely;

- Correlation of messages as well as process instances
- Sequencing of process activities such as interactions of web services
- Exception handling and performing recovery in case of failure
- Managing Bilateral web service based relationships between process roles

BPEL is a widely accepted process language for web-services orchestration [69]. Its concepts are very low-level, they have no standardized graphic notation and for non-technical users they are unreadable [69]. The major contributors in the development of WS-BPEL are IBM, BEA Systems, Microsoft, SAP AG, Siebel Systems.

Workflow Nets: A Petri net which models a workflow process definition (i.e. the life-cycle of one case in isolation) is called a WorkFlow net (WF-net) [5]. Petri Nets are directed graphs that mainly consist of two different nodes; places and transitions [42,52,69,76]. WF-nets satisfies two minimal requirements namely [5];

- A WF-net has one input place (i) and one output place (o). A token in i corresponds to a case which needs to be handled, a token in o corresponds to a case which has been handled. Any case handled by the procedure represented by the WF-net is created if it enters the WFMS and is deleted once it is completely handled by the WFMS, i.e., the WF-net specifies the lifecycle of a case.
- Secondly, in a WF-net there are no dangling tasks and/or conditions. Every task (transition) and condition (place) should contribute to the processing of cases. Therefore, every transition t (place p) should be located on a path from place i to place o

A more formal definition of Workflow Nets is presented by Aalst et al. [5].

*XML Process Definition Language (XPDL):*An XML-based language from the Workflow Management Coalition (WfMC) for defining business processes [69,80]. The main goal of XPDL is for "the automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rule" [80]. The concepts involved in XPDL are very low-level since it is a configuration language for execution engine [42,52,76]. XPDL has no standardized graphic notation and therefore it is unreadable for non-technical users [69].

Yet Another Workflow Language (YAWL): YAWL is a powerful workflow language based on workflow patterns. It has a strong formal foundation based on the Petri Nets [43]. YAWL modeling language can handle complex data transformations as well as full integration with organization resources and external web services [31].

In addition to the above mentioned modeling languages that are specifically associated to either the conceptual or workflow modeling, Protos modeling language is attributed to both the two modeling viewpoints.

Protos: It's a vendor specific modeling language (since the modeling language cannot be separated from its tool) which is now part of the BPM|one suite [58]. It's allows for workflow models to be executed, enacted and analyzed. However, it should also be noted that protos is also very much used for communication purposes at the conceptual level. It supports freehand specifications of business processes without formal semantics, e.g. to support initial and conceptual modeling [74]. The main use of Protos is to define models of business processes as a step towards either the implementation of quality systems, the redesign of a business process, communication enhancement between process stakeholders, or the implementation of workflow management systems. Protos 7.0 is an attempt to more closely integrate modeling and simulation facilities into one tool [74]. Protos is ultimately classified as a modeling language that belongs to both the conceptual and workflow viewpoint of process modeling. It was developed by Pallas Athena.

The degree of comprehensibility [20] of both conceptual process models and workflow process models greatly vary among user groups. For instance, a workflow process model is usually unreadable to the stakeholders in the management domain of a company while conceptual process model is more user friendly for them. This variation in pragmatic quality [38,41] of models with respect to the user group under consideration is partly influenced by the underlying modeling language which was used to construct it. Process models created by different modeling languages typically have varying expressive powers and levels of detail [20]. Conceptual process models abstracts away from most of the details and it is less expressive. A workflow process model is usually more expressive capturing virtually all the information necessary so that the process modeled in the workflow can be automatically executed by the workflow management

system. This variation in expressive power and level of detail exhibited by process models is influenced by the modeling languages used to construct them, among factors.

3.2.3 Workflow Process Model Aspects

In addition to the above four presented aspects of process models in section 3.1.3, there exist two additional model aspects that are specifically attributed to workflow process modeling. This section of the report is devoted to these two aspects. A brief explanation of each is given below.

1. *Operational Aspect*: The integration of existing tools and application programs into workflow applications is an important feature of workflow management systems. The information required is specified in the operational aspect [79]. These applications cover mainly general applications such as text editor or spreadsheet editor, and include special applications developed for the given task [71]. Technical specifications details like the input and output definition parameters of application programs and how they are mapped to input and output parameters of workflow activities are also presented in the operational aspect. Furthermore, the operational aspect also contains information regarding the invocation environment of application programs. This includes the directory and host of information of the executable program [79]. The operational aspect is strictly unique to a workflow model and it does not need to be present in a conceptual process model. Wegener et al. [51] also briefly explains the operational aspect.

2. *Flexibility Aspect*: Providing flexibility to workflow applications is based on the understanding that during workflow modeling not all aspects of the application process can be specified completely [79]. Unforeseen situations that arise during the execution of the workflow will eventually require a flexible mode of reaction by a user of the system which could also be the administrator. Weske et al. [79] gave various forms of flexibility information covered under this aspect. They include change of role information, change of application program information, adding an activity to a complex workflow while the workflow executes, changing the control structure of sub-workflows of a given workflow (e.g., parallel execution of workflow activities, originally defined to be executed sequentially). Furthermore, facilitating a user to stop, skip or repeat sub-workflow is also a form flexibility contained in this aspect. For all these to materialize, flexibility has to be supported by both the workflow language and the workflow management system [79].

Table 1 below summarizes the distinction between conceptual process models and workflow process models.

Table 1. Conceptual Process Model Versus Workflow Process Model

		Classifications of Process Models	
		Conceptual Process Model	**Workflow Process Model**
Dimensions of Process Modeling	**Model Usage**	▪ Communication platform among all stakeholders in the modeling project ▪ Pinpoint the essential aspects of the business process to provide a shared understanding among stakeholders ▪ Help firmly establish core features and business logic before deciding on implementation details ▪ Establish the scope of the business process being redesigned	▪ Enactment purposes by the WFMS ▪ Analysis and simulation purposes ▪ Communication platform all technical project team members ▪ Operational knowledge base for work instructions
	Modeling Notations/ Languages	BPMN, UML AD, IDEF3, EPC, RAD, Protos	BPEL4WS, XPDL, Workflow Nets, YAWL, BPML, Protos
	Model Aspects	Functional, Organizational, Behavioral, Informational	Functional, Organizational, Behavioral, Informational, Operational, Flexibility

From Table 1 above, it can be observed that modeling languages that belong to the conceptual modeling viewpoint generally have more intuitive graphical presentation constructs which require very minimal effort to understand even by non modeling experts. This is corresponds with the primary use of a conceptual process models which is to facilitate communication among stakeholders of potentially of diverse backgrounds (business and technical backgrounds). The ease of understandability of conceptual process modeling languages is geared towards facilitating the primary usage of conceptual process models. On the other hand, workflow process modeling languages generally have more expressive power but less understandable to stakeholders who are none modeling experts. This is largely attributed to the fact that they are primarily used for enacted purposes hence numerously implementation attributes are in cooperated in the language unlike in conceptual process modeling languages.

4. Review of the State-of-the-Art Modeling Guidelines

In this chapter, a set of modeling guidelines as reported in literature is presented. Process modeling guidelines are a set of rules to be considered when designing process models with the aim of improving the syntactic, semantic or pragmatic quality of models. Modeling guidelines are ideally suppose to help to reduce on the error probability in models and improve on the degree understandability of the model [48]. According to M. Rosemann et al. [59], large projects on process documentation heavily rely on novices and non-expert modelers. Without a well defined set of modeling guidelines, novices and non-expert modelers are bound to make process models which contain more errors. This therefore stresses the need for process modeling guidelines. Section 4.1 presents conceptual process modeling guidelines. Section 4.2 presents workflow process modeling guidelines.

4.1 Conceptual Process Modeling Guidelines

The major goal of the guidelines applicable to conceptual process modeling is to improve the quality of the models so that they become more comprehensible to various stakeholders and contains as few syntactical errors as possible [48]. Mendling et al. [48] presented a very comprehensive set of seven process modeling guidelines. However, a more detailed analysis and experimentation of some of the guidelines presented in [48] was actually works of previous authors in the same domain such as [47,49,50]. Below are the lists of guidelines that are applicable to conceptual process modeling.

- *Use as few elements in the model as possible* [48]. Larger models tend to be more difficult to understand [49] and have a higher error probability than small models [47].
- *Minimize the routing path per element* [48]. The higher the degree of an element in the process model, i.e. the sum of the number of input and output arcs, the harder it becomes to understand the model [49].
- *Use one start and one end event* [48]. There is a strong correlation between the number of start and end events with error probability [47]. Process modelers will easily lose track of several start and end events due to our limited cognitive capability as humans. Models satisfying this requirement are easier to understand [48].
- *Model as structured as possible*. A process model is structured if every split connector matches a respective join connector of the same type [48]. Unstructured models are more likely to include errors and are hard to understand [47,48].
- *Avoid OR routing elements* [48]. Models that have only AND and XOR connectors are less error-prone.
- *Use verb-object activity labels* [48]. The verb-object labeling style, like “Inform complainant”, is considered as significantly less ambiguous and more useful than action-noun labels or labels that follow neither of these styles [50].
- *Decompose the model if it has more than 50 elements* [48]. For models with more than 50 elements the error probability tends to be higher than 50%. Therefore, large models should be split up into smaller models [47,48].

- *The business analyst must decide on a 'best' level of abstraction for conceptual process models*: abstract, yet detailed enough for stakeholders to understand, and sound enough to be used in subsequent redesign [78]. The conceptual process model should neither be too abstract nor too detailed.
- *Use Precedence Analysis*: Most business analysts tend to draw up the Conceptual Process Model by arranging the indispensible actions in a chronological sequence, i.e by thinking "what comes next". However, one ought to determine what must come *before*, instead of what might follow. Precedence analysis must proceed backwards from the concluding actions up to the trigger, thus keeping a strong focus on the key deliverables which is a core feature of the Conceptual Process Model [78].

4.2 Workflow Process Modeling Guidelines

Aalst and Hee [1] proposed a number of general rules of thumbs to be considered when designing workflows. This is further supplemented by workflow modeling guidelines proposed by Becker et al. [8] which is an extension of an earlier work of Schuette and Rotthowe [66]. They [66] presented a general framework of principles that improve the quality of information models through reducing the subjectivism in the information modeling process. Below we present a comprehensive list of all the workflow modeling guidelines.

- *First establish the objective of the process*. We need to make clear the goal of designing the workflow. By reflecting upon this fundamental question, it is possible to define the new workflow without misleading presuppositions [1].
- *Ignore the existence of resources when defining a process*. The process definition is independent of the potential offered by people and machines. If the allocation of work to resources is already being considered when drawing up the process definition, one runs the danger that the resulting process will not be the best one possible [1].
- *As far as possible, make one person responsible for the processing of a case (case manager)*. Processes supported by a workflow system can be quite complicated. For the client, it is therefore often very difficult to gauge the progress of a particular case. This is why it is sensible to appoint a manager for each case. He or she acts as a sort of buffer between the complicated process and the client [1].
- *Check the need for each task*. Critically assess the need for every task in the workflow process. Subsequently eliminate all tasks which add no value [1].
- *Consider the scope of tasks*. A task is a logical unit of work. By combining separate tasks into one composite task, set-up times can be reduced. The involvement of the people performing them is also increased. However, tasks should not be too large. Because a task always has to be performable in one go, without interruptions, 'bite-size chunks' are desirable [1].
- *Strive for the simplest possible process*. Complex process definitions lead to unmanageable processes. It is therefore important that a process not be unnecessarily complex. People involved in carrying out the process should be able to understand it [1].

- *Carefully weigh a generic process versus several versions of the same process.* Always attempt to create a generic process which distinguishes between the various types of cases by using selective routing. Do not, though, attempt to handle two completely different types of cases in a single process [1].
- *Carefully weigh specialization versus generalization.* The division of a generic task into two or more alternative tasks may have either a positive or a negative effect. One advantage can be that the tasks become better suited to the specific qualities of a resource. There can be drawbacks to specialization, though. It often detracts from the flexibility and accessibility of the process. It can also lead to monotonous work, which reduces motivation. Therefore, a careful choice should be made between specialization and generalization [1].
- *As far possible, try to achieve the parallel processing of tasks.* Always consider whether tasks can be performed in parallel. If two tasks can be carried out independently of one another, then it is very important that the process allows their parallel performance. The unnecessary introduction of sequential order relationships results in longer completion times and the inefficient use of resources [1].
- *Investigate the new opportunities opened up by recent developments in networking and (distributed) databases.* The elimination of physical barriers resulting from such developments as the computerization of documents often makes possible entirely new process structures. Tasks which previously had to be performed in sequence can be carried out in parallel following the introduction of, say, a workflow package [1].
- *Treat geographically scattered resources as if they are centralized.* The introduction of a workflow system lowers the physical barriers between the various sections of an organization. It makes it easier for two organizational units to exchange work. If team A is struggling with a flood of work, but team B is operating below capacity, it is logical to transfer work from A to B. It is even better to treat geographically scattered resources as if they are centralized. This enables resources to be allocated to those places where the most work is waiting [1].
- *Allow a resource to do what it is good at:* Making use of resource specific qualities generally improves on the quality of output [1].
- *As far as possible, allow a resource to perform similar tasks in succession.* By performing similar tasks one after the other, set-up times can be reduced and the benefits of routine working can be exploited [1].
- *Try to achieve as much flexibility as possible for the near future.* When allocating work to resources, it is sensible to retain as much flexibility for the near future as possible. This helps to reduce scenarios where in the future a work item has no resource capable of executing it [1].
- *Allow a resource to work as much as possible on the same case.* If an employee performs a number of successive tasks for a specific case, the total processing time is usually shorter than if different employees carry out those tasks. Less time is taken because the member of staff does not have to 'get used' to each new case [1].
- *Avoid the use of manual functions.* This is especially more important if one follows after the other immediately [8]. Very limited (preferably no) use of

manual functions greatly improves the automation of the execution of workflow models.

- *For every function the start and end conditions should be precisely determined*. In particular, it has to be indicated if the function shall be started manually or automatically [8]. As an option, for every function a deadline can be declared. When this deadline is exceeded, a higher authority can be informed, ideally the person in charge for the process (the process owner).
- *Every function within a workflow model must include a link to an organizational construct, if it is not completely automated and shall be executed autonomously* [8]. These organizational constructs includes role, organizational units, position.
- *Information about inclusive OR-join connections evaluated as true at runtime must be provided.* The approach of dead path elimination where, an OR-split forwards information to the corresponding OR-join about all workflow paths which will not be executed can be used. With this input, the OR-join has all the required information for the determination of the continuation of the execution of the workflow [8].

On a closer analysis of the conceptual process modeling guidelines, it can be observed that majority of them focus on the layout and presentation of a process model. This trend was expected since conceptual process models are mainly graphically represented. On the other hand, majority of the workflow modeling guidelines focus on efficient and effective processing of cases. More emphasis has been placed on proposing guidelines that if used will eventually improve on key performance indicators such as average completion time, level of service and utilization of capacity. To bridge the gap between conceptual process models and workflow process models, generic rules of thumb to be followed to translate a conceptual process model to workflow process model must be clearly defined. However, currently, hardly any such set of guidelines does exist. In the next chapter (chapter 5) we propose a set of generic conceptual to workflow process model transformation guidelines.

5. Proposed Model Transformation Guidelines

In this chapter, guidelines describing what procedures should be followed in transforming conceptual process models to workflow process models are discussed. An extensive literature survey was done and a careful analysis of how numerous conceptual process models have been transformed into workflow process models was made[13,14,15,25,28,36,46,53,54]. After this analysis, the guidelines below were derived by identifying best practices that were observed in each of the literature materials. There was no single paper found that explicitly discusses conceptual to workflow transformation guidelines. As a result, we had to analyze each relevant literature material found and identify certain best practices they used in their transformation process. Not all best practices that were identified were used to formulate a guideline, only those that meet the criteria mentioned below were chosen.

A best practice was considered relevant for this research if it is applicable to transformations of conceptual process models represented in any given modeling language to workflow process models in any workflow specification language.

The above criterion was used since from the goal of our research, we are only interested in generic transformation guidelines.

Each of these guidelines is explained below.

1. Decompose all abstract tasks in the conceptual process model

Business process models at the conceptual level are fundamentally very abstract. In order to communicate the business logic depicted in the conceptual process model to process participants on a more operational level, the business logic will have to be concretized unlike at the conceptual level. Typically in conceptual process models, a complex sub-process could be represented as a single abstract task. When transforming a conceptual process model to a workflow process model, such abstract tasks will then have to be identified and decomposed into several atomic/discrete tasks (tasks which cannot be decomposed any further). An atomic task is a single task from the view point of a workflow management system [64]. Usually the decomposition of abstract task is an iterative process done by a business analyst in consultation with the owners of the process or the business owners. The decomposition of a task should immediately stop when any further attempt to decompose it would eventually lead to lose of business relevance. This hierarchical adjustment of tasks facilitates easy modifications of business processes [20,65]. Conceptual modeling languages such as BPMN and EPC support the decomposition procedure [65].

The Figure 4 below shows a fictitious example of part of a company process. In the conceptual process model (depicted under the heading "*Main process path*" in left part of Figure 4), the task "*generate demand forecast*" is an abstract task. At the workflow level, a relatively larger and finer grained sub-process corresponds to this activity only

(right part of Figure 4). When transforming the conceptual process model to workflow process model, this task is decomposed into its corresponding sub-process.

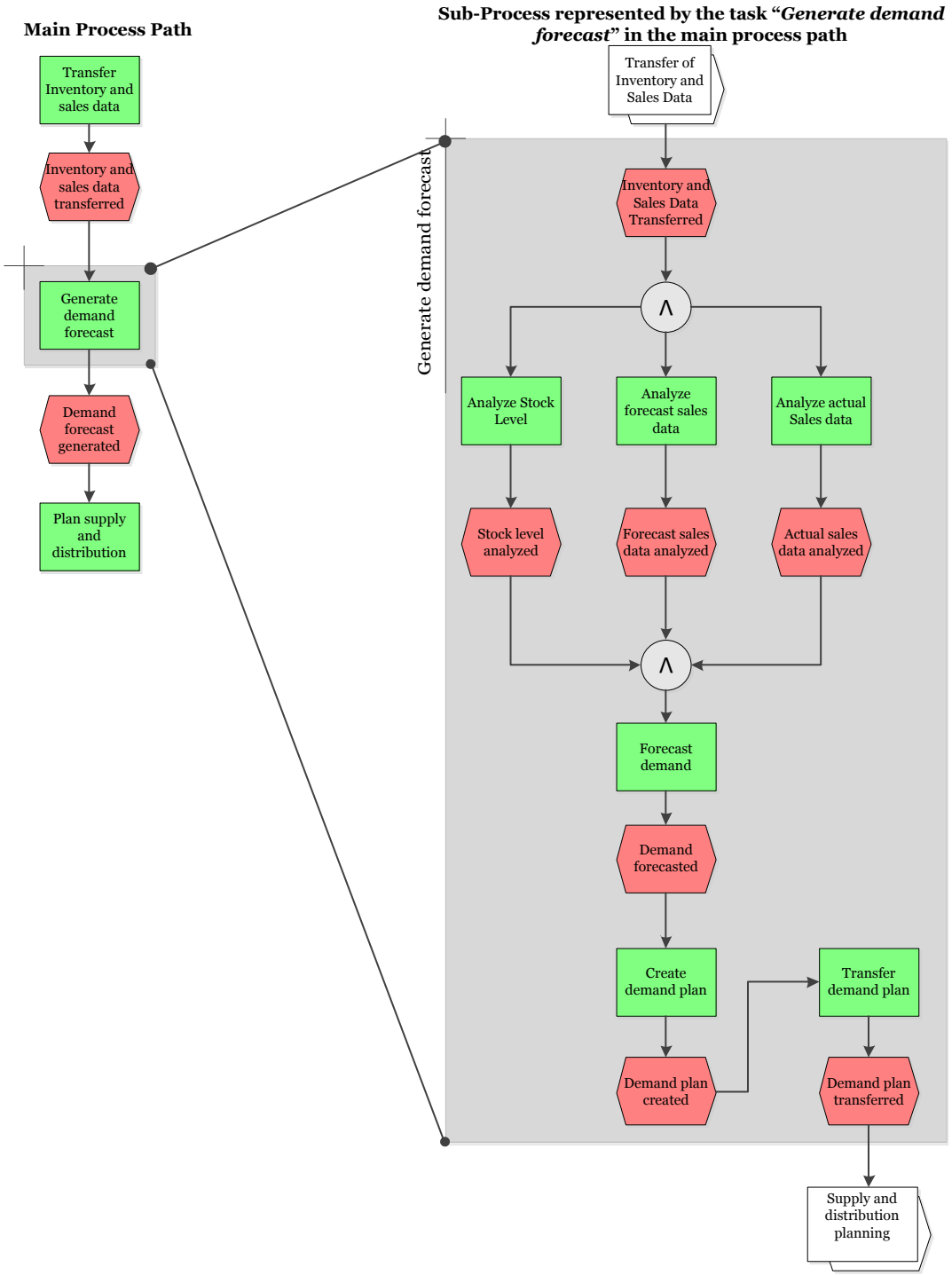


Figure 4. Task Decomposition

2. Complete the process model

Conceptual process models often lack a number of control flow information. The possible causes for this is because process modelers assumed such information is common knowledge to everyone when it might often not be the case, the process modelers themselves didn't know about it, the process modelers got contradicting statements for the same piece of information, or they just forgot to model it. Such missing control flow information should be added when transforming conceptual process models to workflow process models. Examples of the common often missing control flow information include;

- **Missing time-out**

  The time-out construct enables process modelers/designers to set the maximum time that may elapse between the occurrences of two events [7]. It's also serves as a deadline to wait for an event to occur [37]. Fundamentally, one *start timeout* transition and one *end timeout* transition represent these two events. These two transitions respectively model the beginning and ending of the set of transitions that must be completed within the timeout [7]. The *end timeout* transition specifies two actions; successful completion within the timeout and timeout has not been met. However, practically, the timeout construct is mostly modeled in a more compact and elegant way without the need for explicitly a start and an end transition. Consider the example shown in Figure 5 below, after requesting for payment from the client, the process modeler assumed that the client will always respond by paying for the bill. This however might not be the case for all clients, some clients will intentionally or unintentionally not respond at all. In this case, the execution of the process will deadlock since there is no execution path which can be activated from this point on.

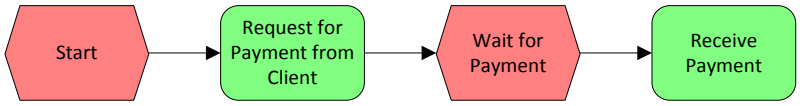


Figure 5. Process Model with Missing Timeout

One option on how to solve this problem is by adding a timeout as depicted in Figure 6 below. Here, the transition "*end the process*" on which a timer has been set. If the client does not respond by paying for the outstanding bill within a specified period of time (say two weeks), then this transition is activated hence ensuring a successful completion of the process.

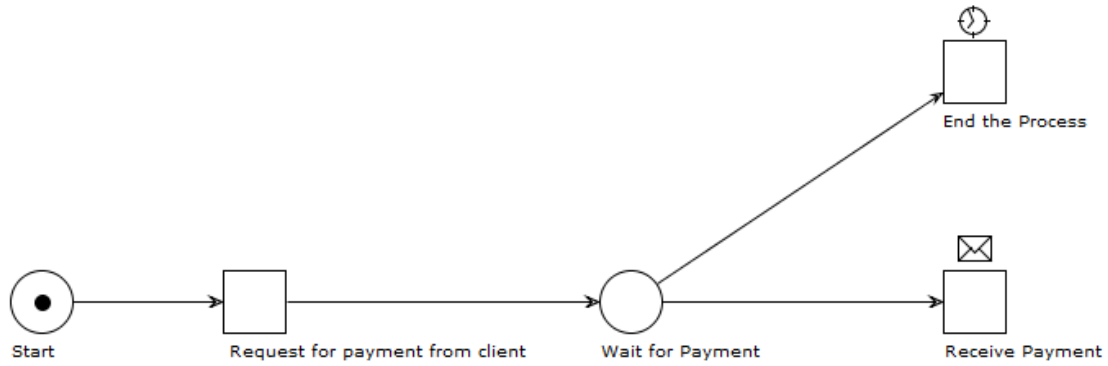


Figure 6. Process Model with timeout added

- **Inappropriate behavior when requesting revision**
  Consider an admission process at university, upon receipt of an application, it is checked for completeness. If the application is complete, then it will be approved, however, for incomplete applications a notification seeking revision will be sent to the applicants. An extract of part of the process containing an inappropriate behavior when asking for revision is shown in Figure 7 below.

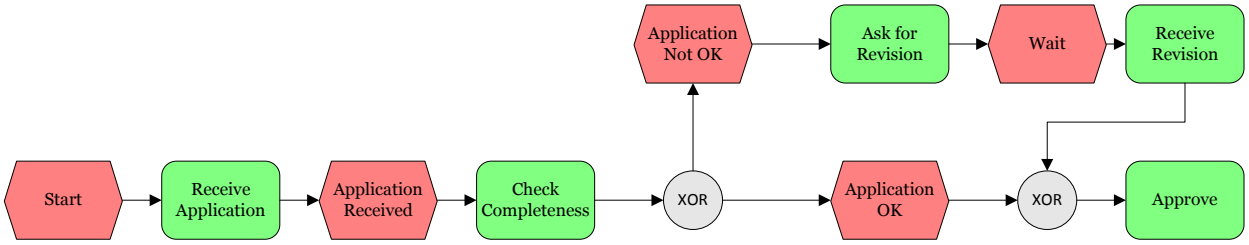


Figure 7. Inappropriate behavior when asking for revision

One preferred option of correcting this anomaly is by adding loopback to the transition "*check completeness*". With this approach, we don't take for granted that all revised application received are complete and should subsequently be approved.

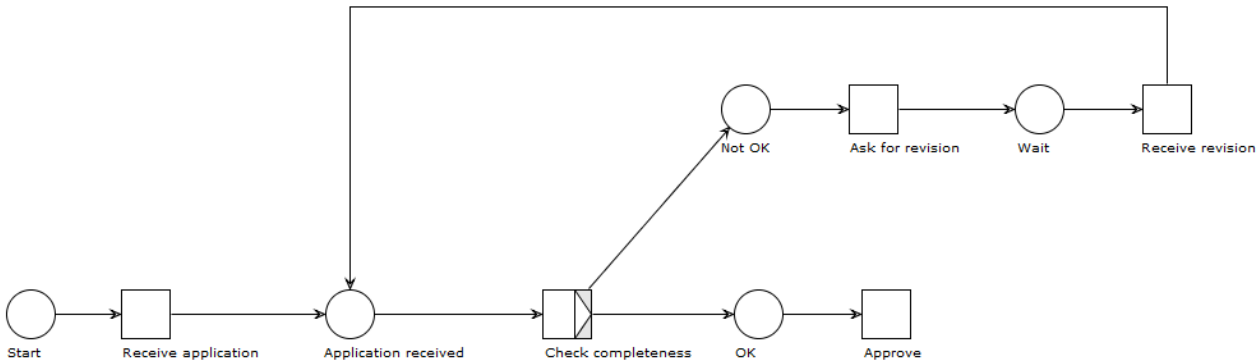


Figure 8. Appropriate behavior when asking for revision

- **Inappropriate end of (sub)process**
  Consider the process shown in Figure 9 below. In this process, for both cases where an application is not OK & a rejection is sent and where an application is OK & an acceptance is sent, subsequently a payout is made. This is not a desirable situation in a real workflow environment.

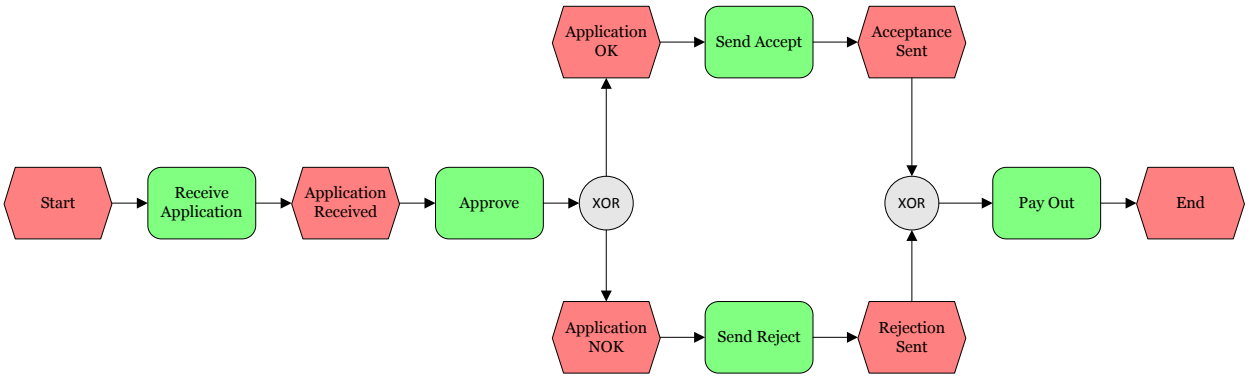


Figure 9. Inappropriate End of Process

The above anomaly can be corrected by adding a different path to cater for a rejection. Figure 10 below show one possible option of above process. Here, when a rejection is sent, the process is immediately ended. It should however be noted that depending on the company policy, the rejection path could be handled differently

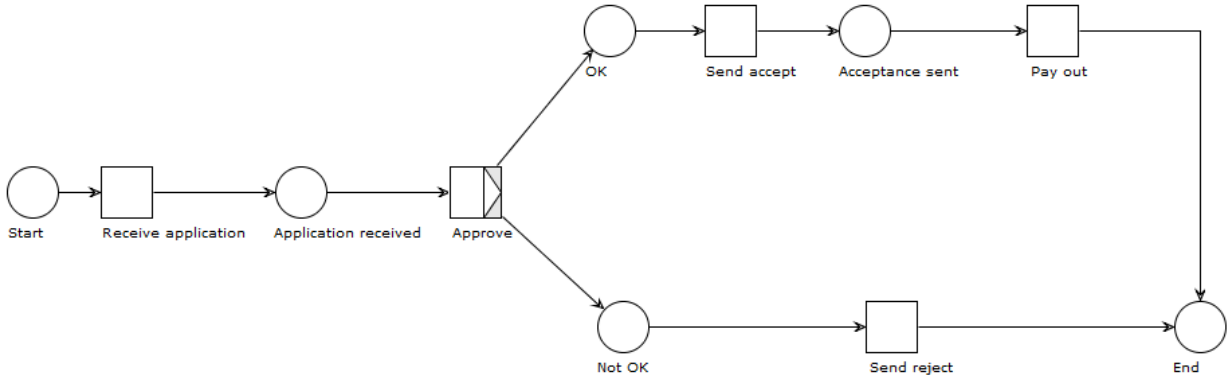


Figure 10. Appropriate End of Process

- **Sunny day scenario**
  The sunny day scenario also called the *happy-path* in the process. Under this situation, only positive process execution options are usually depicted in the process model. In the Figure 11 below, when an application is received, it is always accepted and a payout is made. This is a typical example of a sunny day scenario in process modeling. This however, may not always be the case in practice.

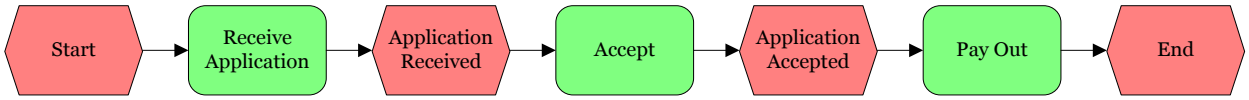


Figure 11. Sunny Day Scenario

Some received applications could be rejected for numerous reasons. This possibility was not catered for in the original process model above. The sunny day scenario can be eliminated by adding alternative process execution path catering for the rejection option as depicted in Figure 12.

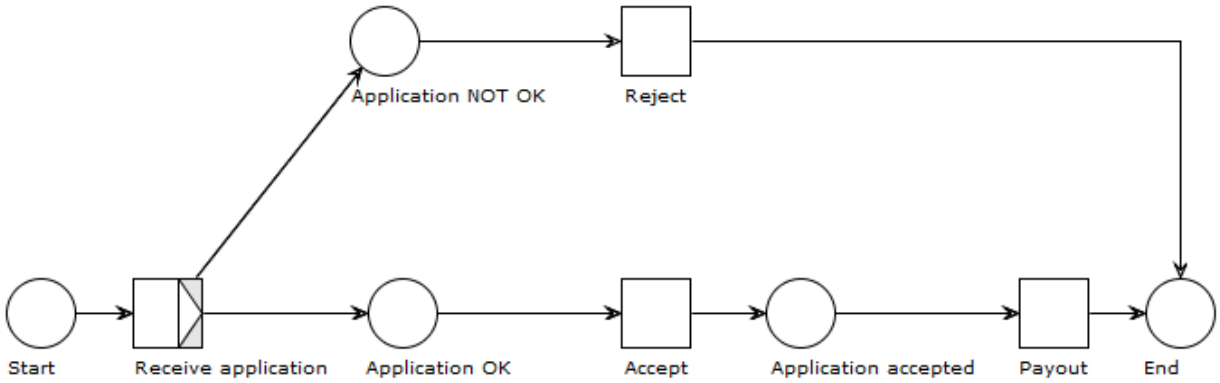


Figure 12. Sunny Day Scenario Corrected

- **Impossible workflow patterns since the chosen workflow engine does not support certain patterns in the process model**
  Consider a case where a company is using staffware as their workflow engine. If a workflow process model contains the *deferred choice* construct, this construct is not be supported by the staffware workflow engine and hence can't be directly implemented. An example of such a construct is shown in Figure 13 below.

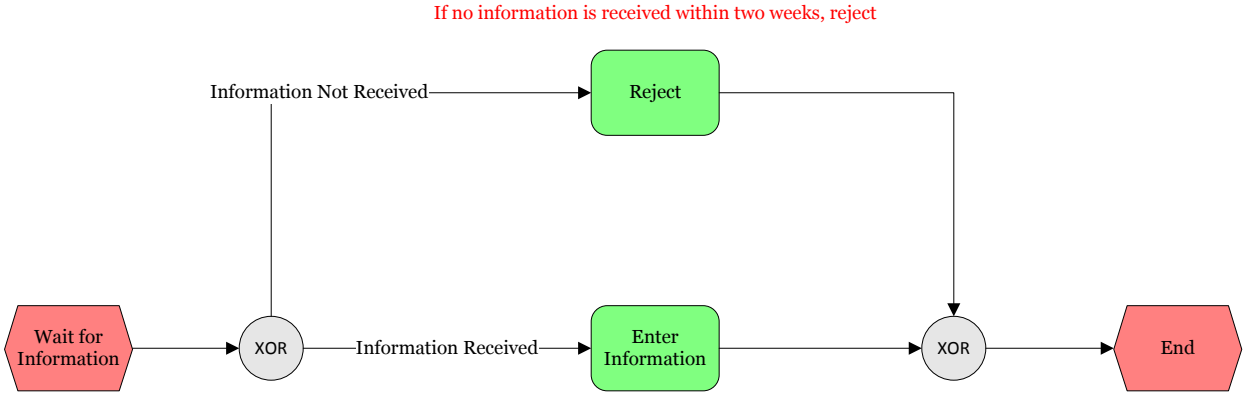


Figure 13. Impossible workflow pattern-deferred choice

One way of modeling the deferred choice construct is to add an explicit (XOR-split) choice that is performed periodically. The example in Figure 13 will subsequently be modified as shown in Figure 14. This construction can subsequently be executed by the staffware workflow engine.

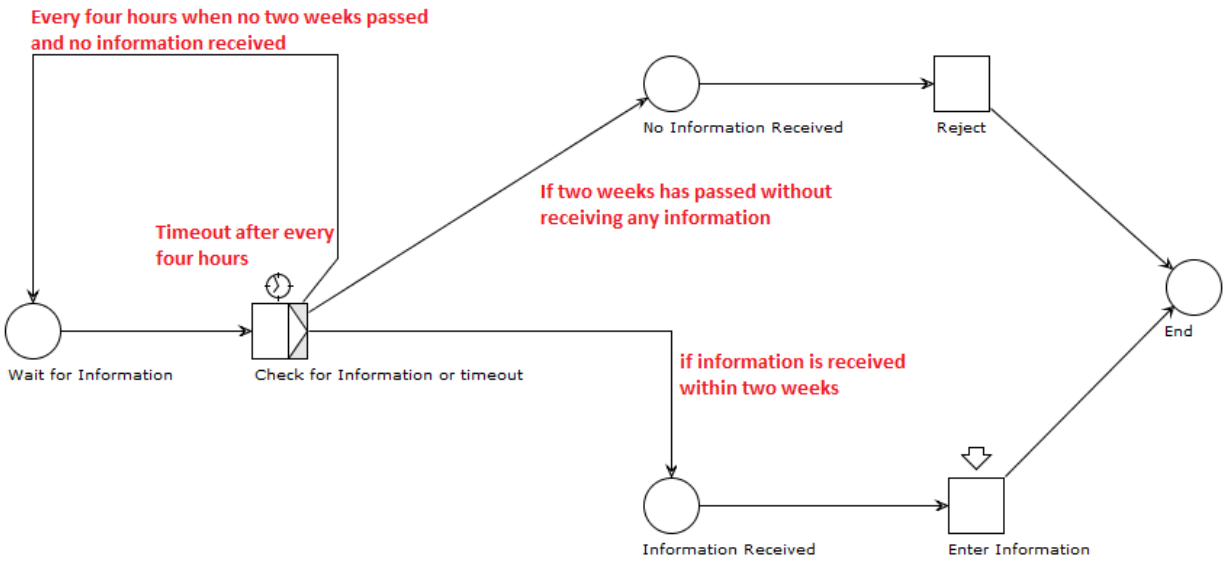


Figure 14. Impossible workflow pattern-deferred choice implemented

- **Alternatives due to different applications**
  A case in point could be when a single task in a process model depending on the situation can be performed on different applications. Consider the task "*compute*" in
  Figure 15, assuming it can be executed either on an SAP or an oracle system depending on the type of the case being handled.

Figure 15. Alternative due to different applications

When transforming such a process model to a workflow net, it is advisable to add an alternative for each application on which it can be executed as illustrated in Figure 16.

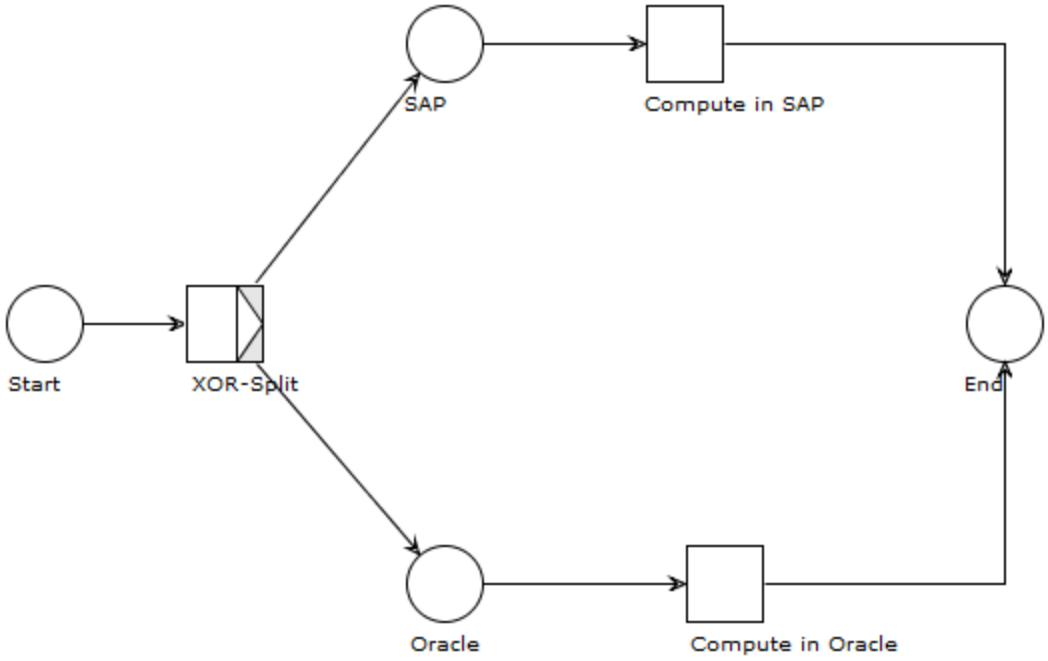


Figure 16. Alternative due to different applications revised

3. Check the process model for semantic correctness

A syntactically correct workflow process model may still have anomalies such as potential deadlocks and inability to terminate. Deadlock refers to a situation where a case is jammed before the *end* condition is reached. These anomalies negatively affect the execution of the workflow process model. Most workflow management systems do not give any support for the verification of workflows [3]. Consequently, workflow process definitions become operational before they are thoroughly checked for correctness. This often results in runtime errors which need to be repaired on-the-fly at high costs. When transforming a conceptual process model into a workflow process model, it is important to assess the correctness of the workflow process model [20]. When checking for correctness of a process model, our main focus is to ensure that it is sound. A process model is sound if and only if it satisfies all the following three conditions [1,3]:

i. For each token put in the *start* place, one (and only one) token eventually appears in the *end* place;
ii. When the token appears in the *end* place, all the other places are empty;
iii. For each transition (task), it is possible to move from initial state to a state in which that transition is enabled i.e. there should be no dead tasks.

The above listed set of minimal requirements for correctness must be satisfied by any workflow process definition [3]. The need for a sound final workflow specification is a necessary requirement in order to guarantee reliable process execution at run-time [3,14]. For instance, when translating a BPMN model to BPEL model, assuming the BPMN model contains deadlocks and infinite loops. The resulting BPEL process model

obtained after the transformation may not be correctly executable. This highlights the need to check for correctness (soundness) of conceptual process models when transforming them into workflow process models. A typical example of an incorrect workflow process model adapted from the paper of Aalst [3] is shown below.

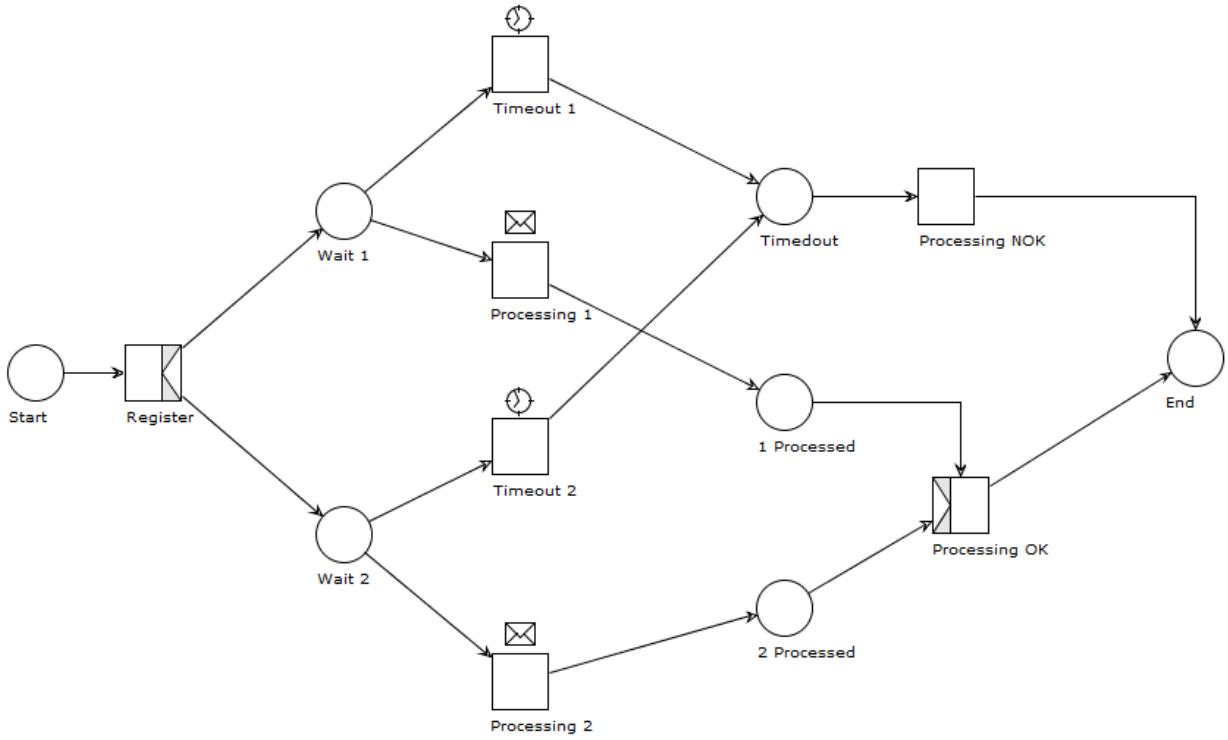


Figure 17. Incorrect Workflow Process Model

It is easy to see that this workflow process definition in Figure 17 contains several deficiencies. Firing *timeout 1* and *processing 2* , or *timeout 2* and *processing 1* will result in the WF-net not terminating properly because a token gets stuck in *2 processed* or *1 processed* respectively. If *timeout 1* and *timeout 2* fire, then the task *processing NOK* will be executed twice resulting in two tokens in *end*. This is an unclear termination scenario violating the first condition (*condition i*) of soundness property which must be exhibited by all workflow process models.

Depending on the language used to develop the conceptual process model, different methods can be adopted to check the correctness of the process model. Brief explanations of two examples of such methods are presented below.

- The conceptual process model is mapped onto Petri Nets. Petri Nets like most industry standards conceptual process modeling languages, such as EPC, BPMN, and UML activity diagrams offers an intuitive graphical notation for the modeling of business processes. However, unlike the other conceptual process modeling languages, Petri nets is based on a strong and well-developed mathematical foundations which define its execution semantics and allow for a variety of

process analysis to be carried out such as soundness checks, liveness, boundedness and many other analysis techniques and tools. Petri nets have long been exploited to formally verify workflow models and it is also state-based rather than just event-based [82]. The Petri net is then checked for correctness for instance by inspecting its reachability graph or coverability graph. A conceptual process model is sound if the corresponding Petri net model is sound. If the correctness check is not satisfied, the deficient elements in the conceptual process model will have to be corrected and the procedure is repeated again. It is necessary that in the end the workflow process model must be sound so as to ensure reliable execution at run-time. Numerous approaches for translating specific conceptual process models to Petri nets have been proposed in literature. For instance, Dijkman et al. [19] propose an algorithm for translating BPMN process models to Petri nets. It should be noted that a Petri net which models the workflow is usually referred to as a Workflow net (WF-net).

- Assuming we are have an EPC process model as our conceptual process model, the ProM framework can be directly used to assess the correctness of the EPC process model [20]. There exist numerous standard tools that can be use to automatically check the soundness of process models. One other common tool used to verify the correctness of petri-net based process model is Workflow analyzer (Woflan). Woflan is WFMS independent hence it's has a greater domain of applicability. In addition, Woflan also provides intuitive guidance on how to detect the sources of errors in workflows and how to correct those errors [4]. Furthermore, in-case we are interested in getting some deeper insights into performance related aspects of our workflow, a tool such ExSpect [2] can be used to for instance measure average throughput time, occupation rate etc [3].

4. Examine the constructs of both the conceptual and workflow modeling language based on the BWW ontology

According to Falkenberg et al. [24], ontology is a discipline concerned with theories of how the "world" may be viewed, conceived or modeled. With respect to the information systems perspective and more specifically business process modeling, ontology studies how business processes can be viewed, conceived and modeled. Ontological examination of both the conceptual and workflow modeling languages is advantageous in that it clearly brings to light the difficulty of mapping constructs of the two languages that typically feature a different level of abstraction. Furthermore, it reveals the incapability of 'high-level' languages to make statements about aspects that can only formulated with more complex languages [20]. The Bunge-Wand-Weber (BWW) ontology was selected and subsequently used for this research because it is widely accepted in the information systems domain and there is extensive scientific literature written about it. The BWW model has five fundamental and core ontological construct namely; thing, property, state, transformation and stable state [27]. Table 2 presents a brief explanation of each of these constructs as adapted with modifications from the article of Peter green and Micheal Rosemann [27].

Table 2. BWW Ontological Constructs

Ontological Construct	Explanation
Thing	A thing is the elementary unit in the BWW ontological model. The real world is made up of things. Two or more things (composite or simple) can be associated into a composite thing.
Property: ▪ In General ▪ In Particular ▪ Hereditary ▪ Emergent ▪ Intrinsic ▪ Non-binding mutual ▪ Binding Mutual ▪ Attributes	Things possess properties, A property is modelled via a function that maps the thing into some value. For example, the attribute "weight" represents a property that all humans possess. In this regard, weight is an attribute standing for a property **in general**. If we focus on the weight of a specific individual, however, we would be concerned with a property **in particular**. A property of a composite thing that belongs to a component thing is called an **hereditary** property. Otherwise it is called an **emergent** property. Some properties are inherent properties of individual things. Such properties are called **intrinsic**. Other properties are properties of pairs or many things. Such properties are called **mutual**. **Non-binding mutual properties** are those properties shared by two or more things that do not "make a difference" to the things involved; for example, order relations or equivalence relations. By contrast, **binding mutual properties** are those properties shared by two or more things that do "make a difference" to the things involved. **Attributes** are the names that we use to represent properties of things.
State	The vector of values for all property functions of a thing is the state of the thing.
Transformation	A transformation is a mapping from one state to another state
Stable State	A stable state is a state in which a thing, subsystem, or system will remain unless forced to change by virtue of the action of a thing in the environment (an external event).

The BWW model presents a useful framework for ontological evaluation [75,77]. Modeling constructs from different process modeling languages that are classified in the same ontological category in the BWW model (e.g *Thing, Transformation, or State*) can be mapped onto each other with ease at meta-level [20]. All modeling constructs of both the conceptual and workflow process modeling languages have to be examined, and the extent to which they fit into the classes of the BWW model determined. If all constructs of both languages have been assigned exactly on ontological construct, pairs of constructs (mappings) of both languages within one class can be constructed [20].

An evaluation of each ontological constructs can be made along the five major views of a process model namely; process view, data view, function view, organizational view and output views. A detailed analysis of the evaluation along these views is presented in [27]. As an example, a simplified ontological analysis of EPCs within the conceptual modeling

language domain and YAWL within the workflow modeling language domain based on the five fundamental and core ontological constructs of BWW model is presented in Table 3 below. The main focus of this analysis was only on the process view. This analysis depicts which ontological constructs from the BWW model are supported by EPC and YAWL. A tick (✔) means that modeling language supports that construct and a dash (-) means that the modeling language does not support that construct.

Table 3. Ontological Analysis of EPC and YAWL

BWW Ontological Constructs	EPC	YAWL
Thing	-	✔
Property	✔	✔
State	✔	✔
Transformation	✔	✔
Stable State	✔	✔

According to Weber [77], a thing as a real world phenomenon is on the instance level, while EPC describes processes on the type level. Therefore, the meta-model of event-driven process chain (EPC) does not capture the concept of a thing. The ontological class property is modeled by a *function* in EPC and also *function* in YAWL since both can be interpreted to represent property in general of an object. Therefore when transforming an EPC model to YAWL, a *function* in EPC could be directly mapped to a *function* in YAWL. The ontological class state is represented in EPC as an *event* and in YAWL as a *place*. Therefore an event in EPC could be transformed into a place in YAWL. The ontological class transformation is represented in EPC as a *function* and in YAWL also as a *function*. Internal events in EPC are usually represented by the triple *event* → *function* → *event*. The stable state ontological construct is represented by only the *end event(s)* and in YAWL it is represented only the *end place*.

5. Assess the degree of workflow pattern support offered by both the conceptual and workflow modeling languages

The main goal is to identify which workflow patterns [18] are directly, indirectly or not supported by a particular modeling language. Workflow patterns are design patterns supported by modeling languages used in the domain of business process engineering and software engineering. These patterns can be used to examine the suitability of a particular process language or workflow system for a particular project, and to implement a certain business requirements in a particular process-aware information system [18]. Process Aware Information System (PAIS) is a software system that manages and executes operational processes involving people, applications, and/or information sources on the basis of process models [21]. There exist a wide variety of process modeling languages used for either conceptual and/or workflow process modeling in the academia and industry. Depending on the process modeling language used, different workflow patterns will be possible to implement in each of the type of models. For both the conceptual and workflow modeling language, the level of support they offer for each of the workflow patterns is compared. In the end, we shall be able to

determine which patterns from the conceptual process model can still be captured in the workflow process model or not. Patterns which are supported by the conceptual process modeling language but are not directly or indirectly supported by the targeted workflow modeling language consequently reduces on the accuracy of the transformation process since such constructs will not be reflected in the transformed model. Mismatches in pattern comparison will help us anticipate future transformation issues which may arise hence we can try to find a workaround through it. Graph-to-block structure transformations will particularly require careful attention when identifying these mismatches.

Consider a scenario where we are interested in transforming an EPC model to YAWL model. We can start by assessing the workflow pattern support offered by both languages for the original set twenty control flow patterns. Table 4 shows a summary of the assessment. The control flow patterns were extracted from [18,31]. If a notation *directly* supports the pattern through one of its constructs, it is rated +. If the pattern is not *directly* supported, it is rated +/-. Any solution which results in spaghetti diagrams or coding, is considered as giving no direct support and is rated -.

Table 4. First Twenty Control Flow Patterns

Patterns	**Modeling Languages**	
	YAWL	EPCs
Sequence	+	+
Parallel Split	+	+
Synchronization	+	+
Exclusive Choice	+	+
Simple Merge	+	+
Multi-Choice	+	+
Structured Synchronizing Merge	+	+
Multi-Merge	+	-
Structured Discriminator	+	-
Arbitrary Cycles	+	+
Implicit Termination	-	+
Multiple Instances without Synchronization	+	-
Multiple Instances with a Priori Design-Time Knowledge	+	-
Multiple Instances with a Priori Run-Time Knowledge	+	-
Multiple Instances without a Priori Run-Time Knowledge	+	-
Deferred Choice	+	-
Interleaved Parallel Routing	+	-
Milestone	+	-
Cancel Activity	+	-
Cancel Case	+	-

An example of an interesting point to look from the analysis presented in the above table is for instance the "*implicit termination pattern*". *Implicit termination* considers a process instance to be complete when there is no remaining work items left to do now or at any future time and the process instance must not be in a deadlock or livelock [31]. This is what is referred to as successful termination of the process execution. Implicit termination captures a scenario of process termination that does not require an explicit end activity [31]. EPC directly supports the implicit termination pattern while YAWL does not support it. Therefore, an EPC model developed based on the notion of implicit termination cannot be directly transformed into YAWL. YAWL does not support *implicit termination* pattern because it has a termination policy of end events such that when a token reaches the output condition, the process instance immediately completes even if there are still active or enabled tasks. However, in an EPC diagram, if the execution of a branch reaches an end event, but there are still other enabled or active tasks, the process instance does not terminate immediately hence obeying the implicit termination policy. It was a deliberate effort by the developers of YAWL not to provide support for implicit termination. This is because this pattern can conceal design errors since it is not possible (very difficult) to distinguish deadlock situations from proper termination. To limit process modelers from making such errors, YAWL developers decided not provide support for implicit termination so as to subsequently force process modelers to carefully consider the termination condition.

For simple EPC process models being transformed to YAWL, the same effect of the notion implicit termination exhibited in the EPC model can be indirectly achieved in YAWL by mapping all end nodes in the EPC model to links to an OR-join which then links to a single final node in the YAWL model. A more detailed explanation and other potential walk-around to this and other complex process models are discussed in depth by Kiepuszewski et al. [33]. The same type of assessment should also be extended to all other classes of workflow patterns namely; *control flow patterns, data patterns, resource patterns, instantiation patterns, abstract syntax presentation patterns, and concrete syntax presentation patterns, and exception handling patterns*.

6. Normalize the conceptual process model

During this stage, additional information associated with the different elements of the process model required for its successful execution is in-cooperated. These include;

- Data necessary for process routing such as routing probabilities of the different alternative paths associated with exclusive gateways and/or workflow attribute. A workflow attribute is a specific piece of information used for the routing of a case [3]. Examples of workflow attributes are age of complainant, claim type, etc. The three major classes of routing are conditional, rule-based, and parallel routing [26]. If tasks are scheduled for execution based on data values contained in the workflow attributes, this is referred to as conditional routing. Rule-based routing is advancement to conditional routing where complex rules are defined to control the execution of a workflow. If multiple tasks are allowed to execute in parallel after the branching of one task, we refer to this as parallel routing. Furthermore, data used to identify a case should also be added so that the people who own the

work-list are availed with more information about the task they are supposed to perform.

- Add priority attributes to tasks [26]; this specifies priorities for activity scheduling during the execution of a workflow process model [67].

It should be possible to change or add elements to the normalized process model [36]. This will greatly improve on the usability of the process model while increasing on its flexibility to accommodate changes to process parameters. The normalization phase is a semi-automatic phase since for instance adding additional resources to an activity is manual process while merging of activities can be done semi-automatically [36].

7. Refine the granularity of tasks/functions

Typically during conceptual process modeling, the granularity of tasks are driven by the semantic considerations that logically distinct tasks should be represented using separate actions. In a workflow implementation, the granularity of tasks is typically driven by the characteristics of the workflow actors, of the assignment criteria, and by the performance and tracking requirements [10]. Choosing tasks of the right size is a key issue in workflow modeling because a workflow engine sends work orders. A task is of the right size if it encompasses the work that a single person can perform uninterrupted. If the tasks are too small, then there is a lot of overhead (setup times) and the workflow becomes too complex to manage. If the tasks are too large, then it may become a problem to execute a task in one go, because it is not possible to commit a partially completed task or put parts of the task out to contract [3]. Furthermore, for task that are too large, the allocation of work items to users may not be fine-grained enough to allow for specialization. Transforming conceptual process models to workflow process models may necessitate the merging or splitting of tasks as explained below.

- A sequence of two tasks is typically merged into one when the actor executing the two is the same [10]. Furthermore, the characteristics of the actor could also necessitate the merging of two tasks. Consider an automated system that offer a function capable of both checking a purchase order and sending it to the delivery service. In designing the conceptual process model for this part, the two functions might be considered as logically separate tasks. However, when designing the workflow process model, the two functions will be merged and implemented as one whole task. An illustration of a scenario where two tasks need to be merged is shown below.

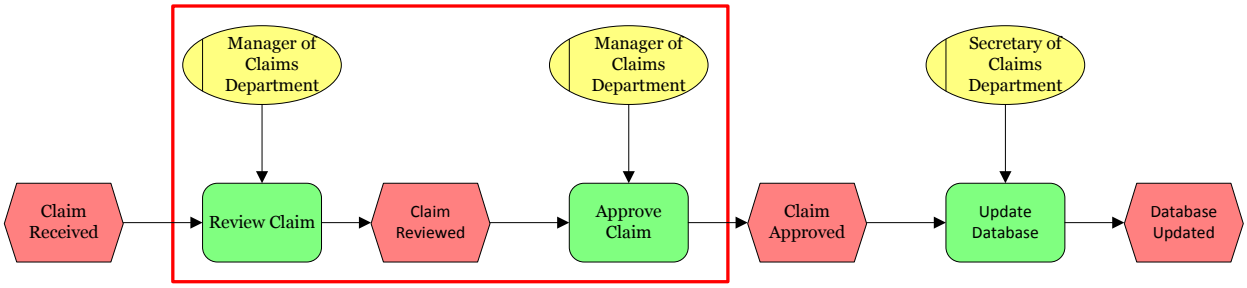


Figure 18. Identifying Tasks to be Merged

From Figure 18 above, suppose the tasks "*review claim*" and "*approve claim*" are both executed by the same manager of claims department, having these two tasks as two separate actions is not desirable. This is because the manager of the claims department will have to log into the workflow system twice in succession to perform them. This is an inconvenience to the manager since (s)he is capable of performing these two tasks uninterrupted in one go. Furthermore, it increases the load on the workflow system because the workflow engine must process two completion messages and also schedule twice instead of once [10]. The resulting process model after merging the two tasks as recommended is shown in below.

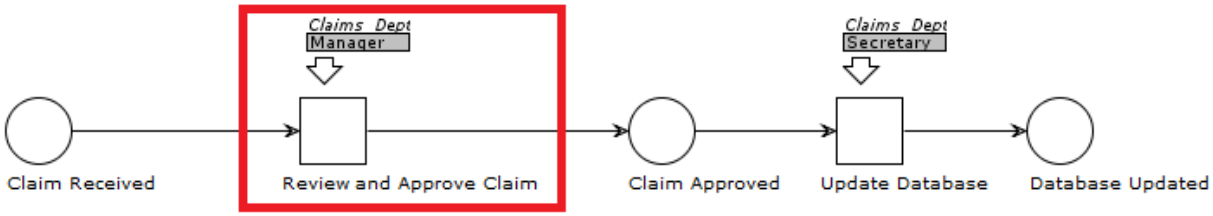


Figure 19. Merge Task

- In other cases it might be necessary to split a task into parallel or sequential tasks [10]. This could be driven by the characteristic of the actors needed to execute that particular task. Consider for example, a task that invokes several other actors for its successful execution. Such a task will clearly need to split up into smaller task that one actor can execute uninterrupted. Typical instances where a task will need to split up are;
  - The execution of a task involves the handing over of work to another role. An example of such a task is "*registering complain by secretary and verifying the complain by the operations managers*" modeled as one single task. In this case there is a hand over of work from secretary to operations manager in the process of performing one task. Such a task should therefore be split-up into two different tasks performed by the two different roles.
  - The execution of a task involves the handing over of work to another person. An example of such a task is "*determining and approving budget allocation by an administrator*" when considering the four eyes principle. In this case, the task should be split up such that an administrator who determines the budget allocation is different from the administrator who approves it. Therefore, there is a handover of work to another person within the same role.
  - The execution of a task involves the handing over of work to the system. Handovers to the system occurs when there is a routing decision to be made where multiple arcs enter or leave a task. The splitting of a task based on routing decision may be postponed in the case of an AND-split or AND-join and the tasks are executed by the same person. Furthermore, a handover to the system is exhibited before or after an automated task.

Furthermore, the need to expose more business logic at the workflow level and to manage/track workflow execution with more detail can be the other motivator to split composite tasks [10].

8. Every task should have temporal aspect/constraints

Different types of temporal constraints are applicable to workflow specifications. These constraints are required in the workflow process model so as to reason about feasibility and effectiveness of the workflow application and enable scheduling of task [64]. Brief descriptions of the most important ones are presented below.

- *Deadline constraint:* At certain points in the workflow, we may require a task to finish by a specific deadline [64]. This should be specified in the workflow process model. The deadline constraint can be specified as a "task specific deadline" which does not depend on the other tasks in the workflow process model [26]. However, we could also set "inter-dependent deadline" which maintains temporal dependencies among other tasks in the workflow process model.
- *Wait constraint:* At particular stages of the workflow execution, we may want to wait for a given time period before proceeding with the execution of a given task [64]. The wait constraint can also be "task specific" or "inter-dependent" just like in the deadline constraint.
- *Obligatory constraint*: This constraint ensures that the workflow process model must be able to handle the execution of any number of cases at any given time. For example, the workflow to process the admission applications for a university has an obligatory constraint to process all the applications received.
- *Iteration constraint*: This is modeled to repeat a task in a workflow for a given time period.
- *Minimum execution time of tasks*
- *Maximum execution time of tasks*
- *Average execution time of tasks*

9. Integrate the process model with existing tools and application programs to be invoked by the WFMS

These applications cover mainly general applications such as text editor or spreadsheet editor, and include special applications developed for the given task [71]. Information regarding the invocation environment of application programs is also specified in the process model in this phase. This includes the directory and host of information of the executable program [79]. The integration of existing tools and application programs with the workflow application is an important feature widely supported by most workflow management systems [79].

10. Configure exception handling and flexibility information in the workflow

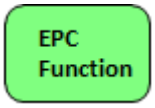
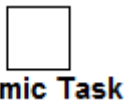
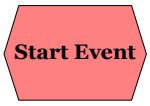
In the dynamic and competitive business environments of recent times, process models are subject to frequent and unavoidable change [12]. Technology supporting business process automation should allow the process models to adapt to changing requirements [62]. In this regards, anticipated behaviors that deviate from the normal process flow are captured and handled accordingly [61]. Unforeseen situations that arise during the

execution of the workflow will eventually require a flexible mode of reaction by a user of the system which could also be the administrator. They include change of role information, change of application program information, adding an activity to a complex workflow while the workflow executes, changing the control structure of sub-workflows of a given workflow (e.g., parallel execution of workflow activities, originally defined to be executed sequentially). Furthermore, facilitating a user to stop, skip or repeat sub-workflow is also a form flexibility which should be supported by a workflow application. A workflow could be pre-maturely terminated during its execution if a system error could result in abnormal workflow termination [64]. An aborted workflow will eventually have to be taken from an undesirable state to a desirable state. This could result in the undoing of certain activities earlier performed. Recovery mechanisms supported by various WFMS greatly vary. For all these to materialize, exception handling and flexibility has to be supported by both the workflow language and the workflow management system [79]. Sadiq and Orlowska [63] classified task failures into two categories, system failures and semantic failures. A system failure is the inability of a task to successfully complete its execution due to problem associated with the processing entity. For example, it could be caused by a break-down in the computing facility on which the workflow is running. A semantic failure is the inability of a task to successfully complete its execution due to underlying task constraints for a specific workflow instance. Earlier defined contingency exception handling plans modeled in the workflow are normally used to mitigate this kind of failures. If we know the expected types of exceptions, then we are able to specify handlers for them. A study has been conducted by Russell et al. [60] to determine a range of exceptions events that are capable of being detected and they also provide a useful basis for recovery handling. This study was based on a comprehensive review of the workflow literature and current commercial workflow systems and business process modeling and execution languages. Example of exception events they discovered includes work item failure, deadline expiry, resource unavailability, external trigger, and constraint violation.

11. Map similar constructs of both Modeling languages

Here, the main focus is to create a translation of pairs of constructs, one from the conceptual modeling language, and another from the workflow modeling language earlier determined from BWW ontology examination. For each pair of construct, a direct mapping of the conceptual modeling construct to the workflow modeling construct is made during the model transformation process. A modeling construct at the conceptual level is considered to be similar (and can be mapped) to another modeling construct at the workflow level if they are both used to represent the same business logic, otherwise they are different. Alternative representations will be considered for conceptual modeling constructs which cannot be directly mapped to any workflow modeling construct. Consider an example where we transforming an EPC model to YAWL, the mapping of their corresponding control-flow constructs will be as depicted in Table 5 below

Table 5. EPC to YAWL Construct Mapping

Constructs	EPC	YAWL
Function		
State		
XOR-Join		
XOR-Split		
AND-Join		
AND-Split		
OR-Join		
OR-Split		
Start Condition		
End Condition		

6. Validation of the Transformation Guidelines

In this chapter, the validation of guidelines was based on the experimentation approach proposed by Hevner et al. [29]. Nine major conceptual process models from the Philips healthcare RS processes were selected and transformed into YAWL. Some of these nine models however also have sub-processes, in total twenty six different process models were converted to YAWL. The YAWL models can be found in appendices **Error! Reference source not found.**, **Error! Reference source not found.**, **Error! Reference source not found.**, **Error! Reference source not found.**, **Error! Reference source not found.**, **Error! Reference source not found.**, **Error! Reference source not found.**, **Error! Reference source not found.**, and **Error! Reference source not found.**. An analysis was thereafter made to determine which guidelines proposed based on the literature study during the previous section was actually usable or not usable during the transformation process. Furthermore, additional guidelines that were newly discovered during the transformation process and were not earlier reported in the previous section are presented. An evaluation was also made to determine the frequency of usage of all the discovered guidelines. Section 6.1 contains the process model selection process, section 6.2 presents the proposed guidelines that were usable during the transformation process, section 6.3 presents the proposed guidelines that were not usable during the transformation process, 6.4 presents additional guidelines discovered during the transformation process, and section 6.5 gives an evaluation of the frequency of use of the guidelines used during transformation.

6.1 Process Selection

6.1.1 Criteria used to Select the Set of Processes

According to Denna et al. [16], in the most generic sense, every organization, regardless of its purpose, goods and services, location, or ownership, has three basic types of business processes namely;

- Acquisition/maintenance/payment process
- Conversion process
- Sales/collections process

The **acquisition/maintenance/payment process** comprise of a series of business activities performed with the aim of acquiring goods and services needed by an organization to perform its functions. This type of process is rather standard across most organizations. This is because only a few different ways of acquiring, maintaining and/or paying for goods and services do exist in practice. The differences can be brought about by the process characteristics such as the means of payment being used, the type

of resource being acquired. Typical examples of goods and services acquired by organizations include [16]

- Human resources (e.g., people's time and skills)
- Financial resources
- Supplies
- Inventories
- Property, plant, and equipment
- New ideas (e.g., research and development)
- Miscellaneous services (e.g., legal, power, telephone, protection, medical, financial, and custodial)

Regardless of the type of good and service being acquired, the basic nature of this type process has the following sequence of events [16] which is fairly standard across most organizations

- Request the good or service
- Select a supplier
- Order the good or service
- Receive the good or service
- Inspect the good or service
- Pay for the good or service

The **conversion process** comprise of a series of business activities performed to transform goods and services serving as raw materials into finished products (goods and services for customers). The nature of the conversion process varies widely across industries ranging from assembling and manufacturing enterprises to distribution and professional service firms[16]. This variation is brought about by the type of good or service being produced, the technology used, and any other restrictions set by regulators, governments, or customers.

The **sales/collection process** comprise of a series of business activities performed to attract customers to buy an organizations goods and services, deliver goods and services to customers and collect payment for delivered goods and services. Just like the acquisition/maintenance/payment type of process, the sales/collection process also has a fairly standard nature across most organizations. Regardless of the types of goods and services being sold, the basic nature of this type process has the following sequence of events [16].

- Receive an order for goods or services
- Select the good or service to be delivered
- Inspect the good or service to be delivered
- Prepare the good or service for delivery
- Deliver the good or service
- Receive payment for the good or service

The above categories of business processes in organizations serve as a framework for classifying any business process. Using this framework we identified a subset of conceptual process models to be to be transformed into workflow process models such that each of the three basic types of business process are fully represented within by our set. With this approach, we can guarantee that our set of selected processes is representative and complete enough of typical organizations processes. Therefore, any analysis or evaluations made using data from this set will be generalizable to any other organization. Below is a table showing mapping of selected sets of processes onto the framework proposed by Denna et al [16]. The "X" in the Table 6 below represents the type business process a particular process model belongs to.

Table 6. Process Classification Framework

	Business Process Types		
Selected Process Models	**Acquisition**	**conversion**	**sales/collection**
Business planning process	X		
Customer management process			X
Acquisition management process	X		
Order management process			X
Business Support (QMS) process	X		
Sales (order acquisition process) process			X
Planning process		X	
Installation process		X	
Packaging process		X	

From the table above, it can be observed that the selected sets of process models equally represent the three basic types of business processes found in any organization. This implies that this set of process models from Philips is representative of typical processes in any other company. Therefore, any analysis carried out based on data drawn from this set of processes will be generalizable to any other organization. Furthermore, the selected processes also had very good documentation compared to the rest. The documentation included detailed description of each process element, included their input and outputs, resources required to execute them, different roles/persons responsible for executing them, etc.

6.1.2 Description of Selected Processes

a) *Business planning process under the PMRS Cleveland Processes:* The objective is to plan business objectives to meet corporate requirements.

b) *Customer management process under the PMRS Cleveland Processes*: The objective is to manage customer expectations and review customer feedback in

order to evaluate and implement improvements to product quality / provided service. The process applies to all customers of PMRS Cleveland.

c) *Acquisition management process under the PMRS Cleveland Processes*: The objective of this process is to procure systems to meet operational requirements as indicated by authorization of purchase (AOP). The process scope is limited to procurement of systems for refurbishment, parts recovery, disposal, and non PMS systems processed by PMRS. The system procurement is handled using external suppliers

d) *Order management process under the PMRS Cleveland Processes*: The objective of this process is to effectively identify and process part returns for PMRS Cleveland "Work In Process (WIP). It only applies to apply to parts being returned during Work in Process from PMRS Cleveland.

e) *Business Support (QMS) under the PMRS Cleveland Processes*: The objective is to identify, correct deviations, and eliminate the cause(s) of potential and actual nonconformities. It applies to all Philips medical refurbished systems Cleveland products and processes, but not limited to:
   - Customer feedback
   - Supplier non-conformities
   - Internal non-conformities
   - Audit deficiencies

f) *Sales (order acquisition process) process under the RS Veldhoven Processes*: This process receives customer orders, screens and accepts/rejects them, then deliver the order.
g) *Planning process under the RS Veldhoven Processes*: This process only focuses on allocation of production slots to the accepted new orders.
h) *Installation process under the RS Veldhoven Processes*: This process models the installation of systems at the client site by Phillips engineers sometimes with the help of third party companies. After the installation, a hand over to the clients is done.
i) *Packaging process*: This process models the packing and processing of system parts due for general operational refurbishment.

6.2 Proposed Guidelines usable during the transformation process

In this section, we discuss the guidelines which were used during the transformation process. Explanations of why they were used are presented as well as our analysis of why we think they were actually useful or not.

The guideline "*decompose all abstract/composite tasks in the conceptual process model*" was used. Due to the need for a high-level of abstraction at the conceptual level, so many composite tasks were used to model sub-processes. These composite tasks later had to be decomposed to reflect more technical oriented details required for implementation in the workflow system. We also earlier think this is a useful guideline since in conceptual process modeling, not much attention is paid to presenting very detail information so as to avoid making the model too complex to be understood by the business-oriented stakeholders.

The guideline "*check the conceptual process model for semantic correctness*" was used. A case in point was when the process "*business planning*" had at infinite loop/recursion in which no work items may be created. This had to rectified for the workflow specification to be valid. This is a very useful guideline from our point of view since a workflow process model is specifically used for enactment purposes. Therefore, a WFMS will eventually have to coordinate the execution of the tasks modeled in the workflow process model. If the workflow process model is not correct, then the execution will not be successful.

The guideline "*Normalize the conceptual process model*" was used. Additional process routing data had to be added on every XOR split and OR split task in the workflow model. This is necessary to define which of the outgoing flows on a given task should be activated at runtime. Therefore, for each outgoing flow from an XOR split and OR split tasks, a boolean expression specifying the condition under which it should be activated at runtime must be specified. For OR-splits, each flow which has an expression evaluating to true will be executed. For XOR-splits, the first flow which has an expression evaluating to true will be executed. Furthermore, all XOR and OR split task must have a default flow set which will be activated only when all the other flow expressions specified for that particular task evaluates to false. This will guarantee that the workflow process does not deadlock during its execution. We also think this is a very useful guideline because various process instances may need to follow different execution paths in the same workflow model. This can be dictated by the business constraints and regulations of a company. The only way to ensure this process instance specific execution is to configure process routing data based on the different parameters of process instances.

The guideline "*every task should have temporal aspect/constraints*" was partly used. We say partly because, unlike as earlier proposed where it was mentioned that every task should have a temporal aspect, during the actual transformation process, we discovered that not all task must have a timer set on it. Depending on the business logic depicted in a particular task, we may be required to strictly have a timer set or not on a particular task. Therefore, we strongly advise that every task should have temporal aspect set if it is dictated by the underlying business logic or if it is a prerequisite for the

sound and complete execution of its underlying workflow model/net. Nevertheless, we think it is a very useful guideline since it enables the scheduling of the execution of tasks which is common feature in workflow modeling.

The guideline "*assess the degree of workflow pattern support offered by both the conceptual and workflow modeling languages*" was used. A case in point is when we needed to implement a "*cancellation region*" depicted in the conceptual process model in the workflow process model. We checked in the list of workflow patterns supported by Yawl and luckily enough the pattern "*cancel region*" is directly supported by Yawl. We were able to implement it in the workflow process model. We strongly believe that this is a very useful guideline because it helps process modelers to determine whether the patterns they intend to implement in the workflow process model is actually achievable given the selected workflow modeling language and workflow management system.

The guideline "*integrate the process model with existing tools and application programs to be invoked by the WFMS*" is also applicable. A case in point is when, at runtime, data from a task instance is selected for viewing and it's a company requirement that such information must be presented using standardized company forms, copies of which are digitally available on the company website. In such a case, a custom (user-defined) form should be defined and integrated with the task by providing its location data e.g the URL of the company forms. At runtime, the standardized company forms will be invoked and the corresponding data selected for viewing will be displayed in the appropriate company form format. We think this is useful guideline because it promotes the interoperability between the newly implemented workflow system and other applications that were already long being used by the company. This will certainly ensure a smooth co-existence between both systems while benefiting from the strength of both of them. However, we think this guideline may be used quite less compared to the other guidelines since not all tasks in the workflow will always lead to invocation of other application programs.

The guideline "*configure exception handling and flexibility information in the workflow*" earlier proposed is usable. Most commercial workflow management systems, have dedicated services which manage exception handling and flexibility of workflows. For instance in Yawl, the worklet service is dedicated for this. Its specifically manages dynamic process selection at runtime and exception handling. The configuration for exception handling and flexibility information can be set so that it is invoked for selected tasks in workflow and/or cases being executed.

The guideline "*complete the process model*" was used. For example, there was a case when a composite task in a "*business planning*" process had no description of the underlying sub-process it represents. In this case, we had to take the decision to model

it as an automatic atomic task just to complete the process. This is a very useful guide from our point of view, process modelers are human beings. They are therefore bound to make mistakes, might not know about certain things, or they could implicitly assume that certain things are common knowledge when it's actually not common to everyone. Such missing pieces of information in models will be brought to light and rectified when following this guideline before the actually workflow is implemented.

The guideline "*map similar constructs of both modeling languages*" was implicitly used. First, we developed an understanding of how the different constructs are modeled in both the conceptual and workflow process model, and then the mapping between them was made. For example, an activity in a conceptual process model was mapped to a corresponding activity in the workflow process model. We think this is a useful guideline because it will lead us to a syntactically correct workflow process model which will be implementable in the WFMS.

The guideline "*refine the granularity of tasks/functions*" is usable. This guideline is of great importance because a workflow engine sends work orders that should ideally be precisely executed by one resource uninterrupted. However, typically task in conceptual process models are not always of the right size for workflow implementation. Therefore, their granularity will eventually have to be refined during the transformation process.

6.3 Proposed Guidelines not used during the transformation process

The earlier proposed guideline below was not used during the transformation process.

- *Examine the constructs of both the conceptual and workflow modeling language based on the BWW ontology*. We didn't use this guideline because we felt it was supplementary in that if one already knows how the different elements of a process are represented in both the conceptual and workflow modeling language, then it's quite faster to immediately map their corresponding constructs without having to strictly follow this guideline. Furthermore, documentations of modeling languages usually provide brief and concise description of their modeling constructs which are much more easier to understand at least from our point of view compared to effort required to understand and use this guideline. For our case, we are well acquitted to the constructs of YAWL so we could directly map the conceptual models directly to YAWL without having to use this guideline.

6.4 Additional Guidelines discovered during the transformation process

During the transformation process we also discovered new guidelines that were not among the earlier reported ones. In this section, we present a detailed discussion of these guidelines.

Specify a trigger for every task in the workflow process model:

A Workflow process model specifies which tasks should be executed and the order in which they should be executed [3]. The fact that a given task can be executed for a specific case does not necessarily imply that it will be executed directly. Consider a case where a task is suppose to be executed by an employee, if that employee is not willing to execute it or is not available, then the task will not be executed directly. This is an indication that the execution of a task may be subject to certain additional conditions being satisfied such availability of resources, or information. Therefore a task in a workflow process model may be enabled but it might not be executed immediately. For an enabled task to be executed, it has to be triggered to do so. A *trigger* is an external condition which leads to the execution of an enabled task [3]. When a task instance of a specific case is triggered, then its execution will start. Furthermore, a task instance can only be triggered if the corresponding case is in a state which enables the execution of the task. There exist four different types of triggers that can be associated with a given task as briefly explained below [1,3];

- *Automatic:* the execution of a task is triggered the moment it is enabled. These types of triggers are used for tasks that are executed by an application without the need for any human intervention.
- *User:* a human participant selects an enabled task instance to be executed. A WFMS maintains an "in-basket" for each user. This in-basket contains tasks instances that are enabled and may be executed by the user [3].
- *Message:* an enabled task instance is triggered by an external event (i.e. a message). Typical examples of such messages are electronic data interchange (EDI) messages, emails, telephone calls and fax messages.
- *Time:* a clock triggers the execution of an enabled task instance at a predefined time.

Depending on the WFMS, and workflow modeling language used, the symbols used to represent these four types of trigger may slightly vary. In this research we shall use the following symbols to represent each type of trigger. The wide downward facing arrow represents a user trigger. An envelope symbol represents a message trigger. A clock symbol represents time trigger.

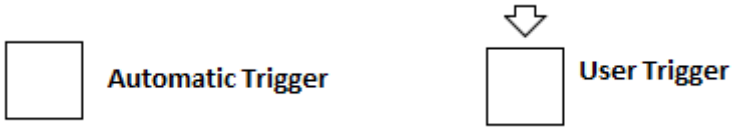


Figure 20 gives an example [3] of a workflow process model with trigger information included.

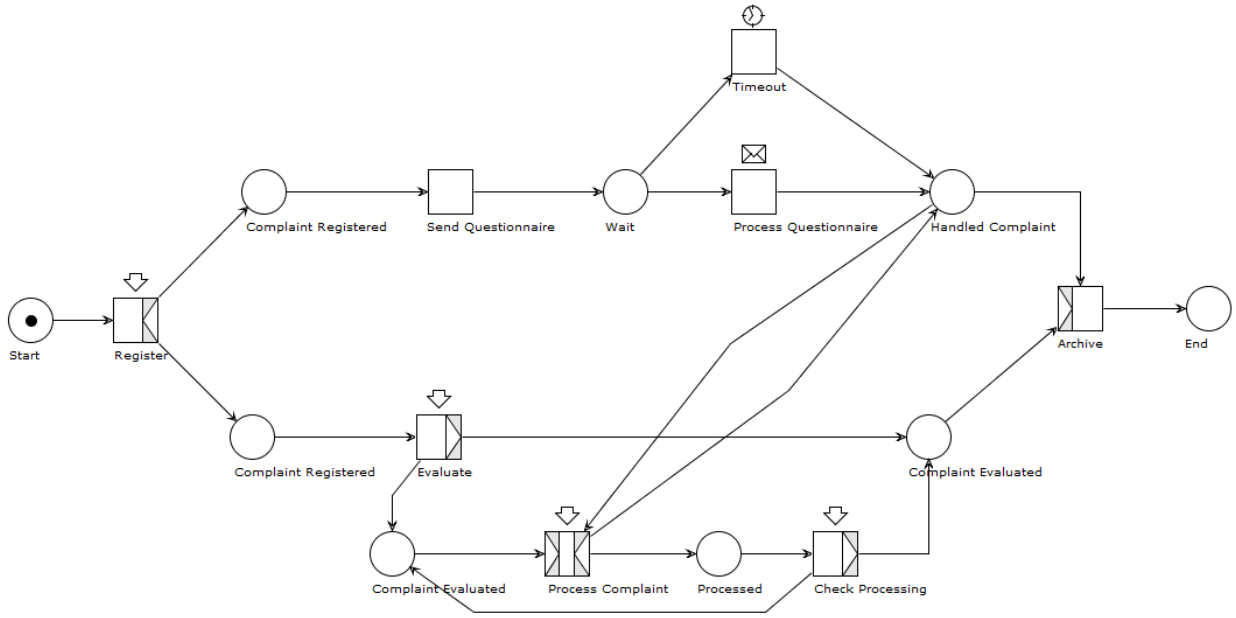


Figure 20. Workflow Process Models with Triggers

A user trigger is necessary for the task *process complaint, evaluate, register, and check processing* since they are performed by human resources. *Send questionnaire*, and *archive* tasks are associated with an automatic trigger since they are executed by an application without the need for human involvement. A message trigger is associated with the task *process questionnaire* since this task can only be executed if the questionnaire sent to the complainant is returned. A time trigger is associated with the task *timeout*. This task is only executed when a certain time period has elapsed and the complainant did not return the filled-in questionnaire.

For every task, specify the workflow engine/service that is responsible for executing the work represented in the task at runtime.
This is particularly useful to make ensure each task is executed by the intended workflow engine/service. Different task may require different services to successfully execute it.

Always make a distinction between explicit and implicit OR-splits.

The distinction between these two types of OR-splits is with respect to the moment choice during the execution of the process. For explicit OR-split, workflow attributes is solely used to determine which process path among the possible process paths will be followed during the execution of a case. For the implicit OR-split, the moment of choice is made at the latest possible time. The importance of making the distinction between the explicit and implicit OR-split is manifested when we consider the idea of triggers where the enabling of a task may not necessary imply its immediate execution. An example of an implicit OR-split is depicted in Figure 21 below, if the place *claim registered* contains a token, the choice between *compute claim in SAP* and *compute claim in Oracle* is non-deterministic and does not depend on the workflow attributes of the case being processed.

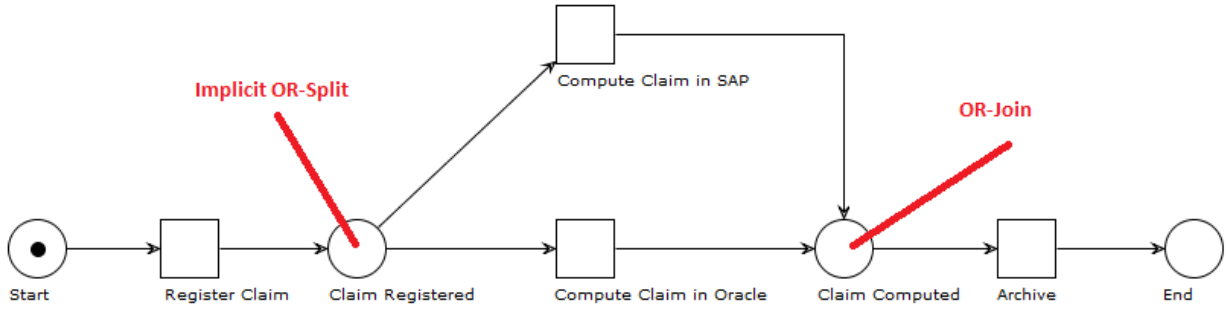


Figure 21. Implicit OR-split

An example of an explicit OR-split is depicted in Figure 22 below, the transition *register claim* produces a token in either the place *claim A registered* or *claim B registered* depending on the workflow attribute claim type. If the claim is of type A, a token is produced in place *claim A registered*. If the claim is of type B, a token is produced in place *claim B registered*. The choice between the two places is deterministic and is made the moment task *register claim* is completed. However, considering the example of implicit OR-split depicted in Figure 21 above, the choice is made the moment *compute claim in SAP* or *compute claim in Oracle* is executed.

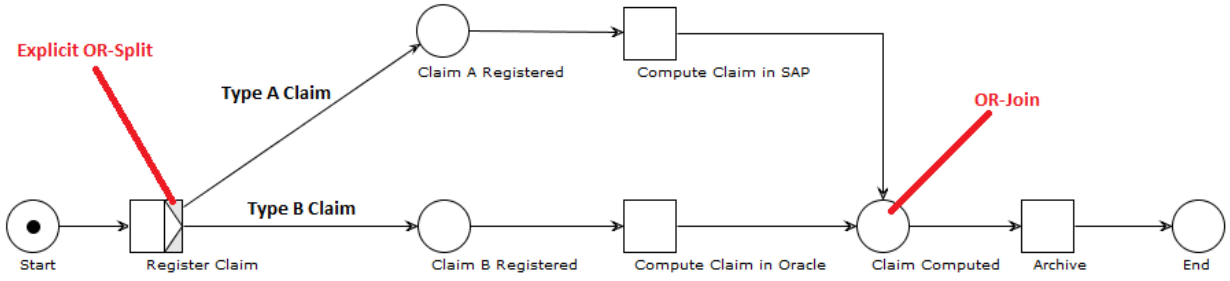


Figure 22. Explicit OR-Split

Making a distinction between these two types of OR-Splits helps to exactly model the intended behavior of the process model.

For every task in the workflow process model, it should be explicitly specified whether it is an automated or manual task. An automatic task is entirely performed by an application/computer program and does not require any intervention by human being. An example of an automatic task is *sending of an auto-reply message acknowledging receipt of an electronic application*. To the contrary, a manual task is entirely performed by human. An example of a manual task is *physically verifying the content of an application*. A task may be executed in part by both a computer program and a person. We refer to such task as semi-automatic task. An example of the semi-automatic task is the physical evaluation of a received application supported by a specially-developed computer application. Making a distinction between a manual and automated task is of great importance because they require different configurations for their successful execution. To execute an automatic task, the workflow management system would need information about the protocol to communicate with the underlying computer application responsible for its execution [64]. For a manual task, information about roles and/or persons who are responsible for the execution of the task must be specified.

All user triggered atomic tasks must be allocated resources authorized to execute it.
Resources include roles and/or participants. Roles can be seen as a group of participant with similar characteristics. Resources therefore refer to an actor or group of actors authorized to execute a particular task. These resources will be offered work items of the task to which they are associated at runtime. An allocation mechanism which determines which role and/or participant should be assigned a particular work item should also be specified. Depending on a company's policy, such allocation mechanism may help optimize resource utilization. However, it is advisable not to explicitly link a task to specific participant but rather to a role. This is because if a task is explicitly linked to a participant, incase that participant is not present then the associated task will subsequently be blocked reducing on the flexibility during the execution of a workflow. Furthermore, in case an existing employee ceases to work for the company, then the workflow process definition will need to be modified increasing on the overhead cost of maintaining the workflow. This situation can be avoided if task are linked to roles since in one role, there could be multiple participants hence if one participant leaves the company, the other remaining participants can still cover up. It may not be necessary to immediately modify the workflow process definition. It is also practically common that a participant is a member of multiple roles.

In workflow modeling, two main types of resource classes are widely reported [1,3] namely; roles and organizational units. Resource classes based on capabilities of its members are called roles. Resource classes based on structure of the organization are called organizational units. Majority of WFMS offer the functionality to assign both a role and an organizational unit to every user triggered task. In this case, a task is only executed by a resource which is a member of a specific role and belongs to specific organizational unit. Furthermore, additional requirements can still be specified to limit the participants authorized to execute a task. Consider the case where a company follows the four-eye principle, if for instance a participant executes a task "*decide on claim amount*" for a specific case, then that same participant should not execute the task "*approve claim amount*" for the same case. Consider Figure 23 below showing workflow process model where resources have been assigned to each of the user triggered tasks.

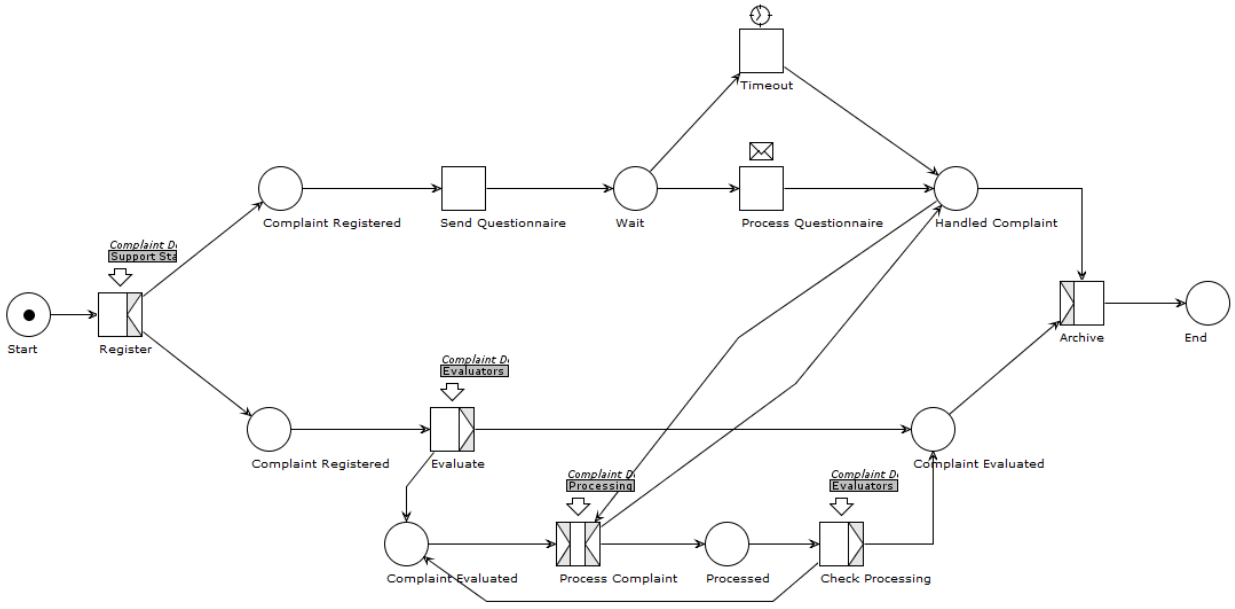


Figure 23. Workflow Process Model with Resources

One organizational unit namely "*complaints department*" and three roles namely *"support staff", "evaluators", and "processing staff"* where defined. The tasks *register* is executed by a member of the role *support staff* within the *complaint department*. The tasks *evaluate* is executed by a member of the role *evaluators* within the *complaint department*. The tasks *process complaint* is executed by a member of the role *processing staff* within the *complaint department*. The tasks *check processing* is executed by a member of the role *evaluators* within the *complaint department*.

For every XOR split task, critically determine the evaluation sequence of the different flow expressions specified for the outgoing flows of the task.
This is of great importance because only the first flow expression that evaluates to true will be activated and subsequently all other flows will be automatically disregarded.

Therefore an unplanned determination of evaluation sequence may lead to some undesirable execution behavior.

Add all the necessary case variables associated with every atomic task in the process model.
For example if you have an atomic task labeled "*Purchase a Book*", the variables "*book name*" and "*ISBN*" will possibly need to defined for this task. Variables should be key attributes of the object(s) being manipulated by a particular atomic task. Defining variables is a necessity because they are used for data transfer and manipulation between a task and a net during the execution of the workflow. It should also be noted that when variables are defined for a given task, the corresponding input and output parameters should also be specified as well as the net variables. The contents of a task variable are copied to net variables and vice versa at runtime with the help of the parameters.

Specify the mechanism to be used to select a resource to execute a work item from a set multiple resources allowed to execute the same work item
Consider a case where multiples resources are authorized to execute a particular work item but only one resource is required to execute it. In this scenario, a choice has to be made for which resource should execute it. Aalst [1,3] identified two mechanisms subsequently explained to resolve this choice.

- *Push-driven*
  The workflow engine makes a decision on which resource should execute a particular work item and subsequently sends the work item to that specific resource. In essence, the workflow engine pushes work items onto resources [1]. A resource does not have the flexibility to choose which work item it wants to execute. The choice of a resource made by the workflow engine could be dependent on parameters such as current work load of each resource, or any other relevant heuristics. Another possible consideration which could be made by the workflow management system is to maintain a balanced workload among all resources. The capturing and analyzing of the task allocation policies to be used workflow management system when adopting a push-driven approach is core aspect of the modeling process [64]. Let take an example of the YAWL workflow system, it has five task allocation policies namely [31];
    - *Round Robin (by time)*: the workflow engine chooses a resource that has not been allocated a task instance for the longest time
    - *Round Robin (by least frequency)*: the workflow engine chooses a resource who has been allocated this task instance the least number of times in the past

- *Round Robin (by experience)*: the workflow engine chooses a resource that has been allocated this task instance the most number of times in the past;
- *Random Choice*: the workflow engine randomly chooses a resource.
- *Shortest Queue*: the workflow engine chooses a participant who has the least number of task instance currently in their work queue

- *Pull-driven*
  The workflow engine sends to each resource that is authorized to execute a particular work item a copy of the work item [1,3]. A resource then has to take initiative and select a work item and start executing it. As soon as one resource selects a work item, copies of the selected work item will disappear from the "*in-basket*" of all other resources to which they were previously sent. In essence, resources pull out work items from a single shared work basket.

Usually the pull-driven approach is preferred by the resources since this gives more flexibility to choose whichever work item they would wish to execute. However, a balance between the push-driven and pull-driven allocation mechanism is commonly used. An example is a situation where the pull-driven mechanism is primarily being used supplemented by the workflow engine ordering work items according any of the well-known queuing principles such as EDD, FIFO, LIFO, etc. The workflow engine then sends to a resource an ordered list of work items it can execute. Ideally, the resource should start by executing the first work item on the list. However, a resource still has the liberty to choose and execute any other work item on the list apart from the first one. The benefit of this balanced approach is that the workflow engine serves the advisory role but resources still have the freedom to choose which work item they will execute. The work assignment rule explaining which person belonging to a particular role should execute a given task is particularly useful if there exist two or more persons that belong to the same role.

Set the queuing principle to be used to order multiple work items

Queuing principles commonly used in workflow management to order pending work items include [1].

- *First In First Out (FIFO)*
  Using this principle, work items or cases are executed in the order in which they are created. It is the most commonly used queuing principle in practice.

- *Last In First Out (LIFO)*

The latest work item to be generated is executed first when using the LIFO principle. It is the direct opposite of the FIFO principle.

- *Shortest Processing Time (SPT)*
  Under the SPT principle, work items which have the least processing time are executed first. Using the SPT principle usually leads to a reduction in the average throughput time of cases.

- *Longest Processing Time (LPT)*
  There could also be rightful justification to have work items with the most processing time executed first. In this case, time-consuming task will have priority over simplest ones. This is a direct opposite of the SPT principle.

- *Shortest Rest-Processing Time (SRPT)*
  Multiple cases can be concurrently executed by the workflow management system. If we are able to determine the remaining total net processing time for every case in the system, we can decide to give priority to cases with the shortest remaining processing time. This will in effect reduce on the quantity of work in progress (WIP).

- *Longest Rest-Processing Time (LRPT)*
  Instead of using the SRPT queuing principle, we could decide to give priority to cases with the longest remaining processing time. Therefore, cases with the longest remaining processing time will be executed first.

- *Earliest Due-Date (EDD)*
  A work item is always carried out in the context of a case [1]. This was initiated at a certain time, and should preferably also be completed by a set time (the 'due date'). The EDD queuing discipline determines the order based upon the case's deadline. So a case which must be finished today takes priority over one which needs to be ready in a week [1].

- *Task with Priority go First (PRIO)*
  Under the PRIO queuing principle, tasks in a workflow process model which have been assigned a high priority are executed first [3].

Table 7 below summarizes the validation of the guidelines

Table 7. Transformation Guidelines Validation

Used Guidelines	Unused Guidelines	Discovered Guidelines
▪ Decompose all abstract/composite tasks in the conceptual process model ▪ Check the conceptual process model for semantic correctness ▪ Normalize the conceptual process model ▪ Every task should have temporal aspect/constraints ▪ Assess the degree of workflow pattern support offered by both the conceptual and workflow modeling languages ▪ Integrate the process model with existing tools and application programs to be invoked by the WFMS ▪ Configure exception handling and flexibility information in the workflow ▪ Complete the process model ▪ Map similar constructs of both modeling languages ▪ Refine the granularity of tasks/functions	▪ Examine the constructs of both the conceptual and workflow modeling language based on the BWW ontology	▪ Specify a trigger for every task in the workflow process model ▪ For every task, specify the workflow engine/service that is responsible for executing the work represented in the task at runtime ▪ Always make a distinction between explicit and implicit OR-splits ▪ For every task in the workflow process model, it should be explicitly specified whether it is an automated or manual task ▪ All user triggered atomic tasks must be allocated resources authorized to execute it ▪ For every XOR split task, critically determine the evaluation sequence of the different flow expressions specified for the outgoing flows of the task ▪ Add all the necessary case variables associated with every atomic task in the process model ▪ Specify the mechanism to be used to select a resource to execute a

		work item from a set multiple resources allowed to execute the same work item ▪ Set the queuing principle to be used to order multiple work items

6.5 Evaluation of the Guidelines

In this section, an evaluation is made to determine how often each of the guidelines was used. This can to a great extend implicitly highlight the level of importance of each guideline. For ease of reference to these guidelines, we have assigned abbreviations to each of them as shown in Table 8.

Table 8. Combined Transformation Guidelines

Guideline	Abbreviations
Decompose all abstract tasks in the conceptual process model	G1
Complete the process model	G2
Check the process model for semantic correctness	G3
Examine the constructs of both the conceptual and workflow modeling language based on the BWW ontology	G4
Assess the degree of workflow pattern support offered by both the conceptual and workflow modeling languages	G5
Normalize the conceptual process model	G6
Refine the granularity of tasks/functions	G7
Every task should have temporal aspect/constraints	G8
Integrate the process model with existing tools and application programs to be invoked by the WFMS	G9
Configure exception handling and flexibility information in the workflow	G10
Map similar constructs of both Modeling languages	G11
Specify a trigger for every task in the workflow process model	G12
For every task, specify the workflow engine/service that is responsible for executing the work represented in the task at runtime	G13
Always make a distinction between explicit and implicit OR-splits	G14
For every task in the workflow process model, it should be specified whether it is an automated or manual task	G15
All user triggered atomic tasks must be allocated resources authorized to execute it	G16

For every XOR split task, critically determine the evaluation sequence of the different flow expressions specified for the outgoing flows of the task	G17
Add all the necessary case variables associated with every atomic task in the process model	G18
Specify the mechanism to be used to select a resource to execute a work item from a set multiple resources allowed to execute the same work item	G19
Set the queuing principle to be used to order multiple work items	G20

Due to time restriction, our evaluation was only limited to the guidelines that apply to the functional and control aspect of the workflow process model. Furthermore, the guideline *G4* which apply to the functional aspect was also not considered in the evaluation because it was not used in the actual transformation process. Therefore the guidelines used in the evaluation are G1, G2, G3, G5, G7, G11, G14. The table below shows how often each of these chosen guidelines was used in the transformation process.

Table 9. Guidelines Frequency of Use

Guideline	Frequency of Use
G1	17
G2	8
G3	26
G5	26
G7	16
G11	9
G14	43

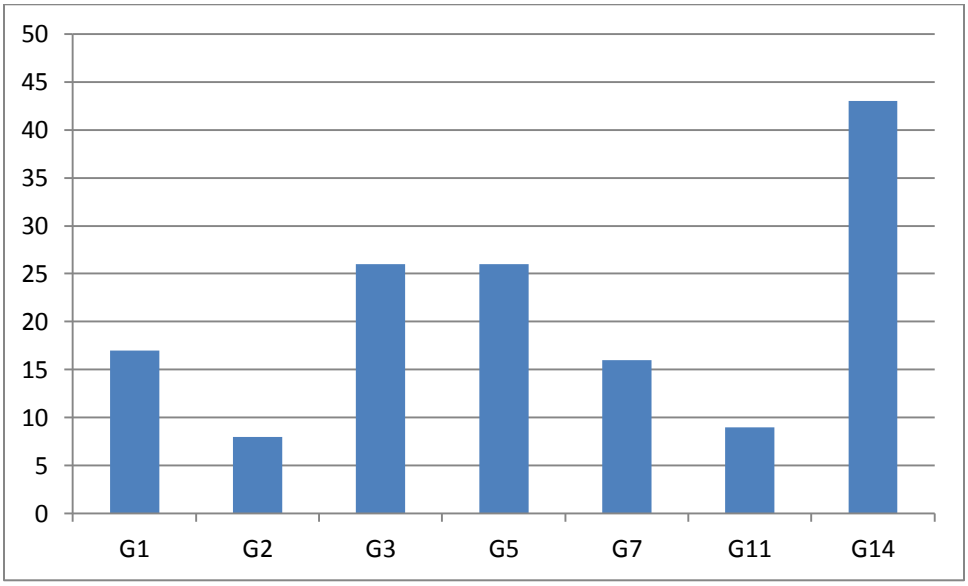


Figure 24. Column Graph Showing Frequency of Use of the Guidelines

From the graph, it can be observed that the guideline "*Always make a distinction between explicit and implicit OR-splits*" was used the most number of times. This can be attributed to the fact that it concerns routing decisions which have to be taken. Typically in any workflow model numerous routing choices have to be taken of which explicit and implicit OR-splits are among. Therefore, we ended up using this guideline most. The least used guidelines are "*Complete the process model*" and "*Map similar constructs of both Modeling languages*". The less usage of the "*Complete the process model*" guideline can be attributed to the fact that the set of process models we selected and used for the transformation process, came along with very details explanation of the process elements and as well as routing process. Therefore, there was less missing control flow information. However, this could be an isolated case, so in the generic sense we might be faced with a situation that this guideline is used quite more often. The guideline "*Map similar constructs of both Modeling languages*" was also used very few times. This is because when transforming any conceptual process model to workflow model, however complex or simple it is, we shall always map their constructs once. Therefore, for each of the nine conceptual models, a mapping was done once for every translation

7. Conclusions

In this chapter, section 7.1 presents a review of our findings by answering the research questions earlier defined in section 1.2. Section 7.2 outlines the major contributions of our research work. Section 7.3 gives some suggestions for future research work and limitations of current research.

7.1 Conclusions

With respects to the earlier defined research questions, the following conclusions were made.

- *Are there any existing comprehensive set of guidelines for transforming conceptual process models to workflow process models?*

Numerous conceptual and workflow modeling guidelines have been proposed in literature. The conceptual modeling guidelines mainly focus on the achieving intuitive graphical layout and presentation of a process model. The workflow modeling guidelines mainly focus on enabling efficient and effective processing of cases so as to improve on key performance indicators such as average completion time, level of service and utilization of capacity.

However, there exist no comprehensive well defined set of guidelines to be used to transform conceptual process models to workflow process models reported in literature.

- *If yes, are they generic set of guidelines, what are their strengths and weaknesses?*

During the literature study, no such set of transformation guidelines was found. Therefore, we skipped this question and didn't answer it.

- *If no, propose new set of generic transformation guidelines*

Since there were no well defined set of generic transformation guidelines reported in literature, we did propose some. The guidelines proposed were primarily derived from analysis of literature pertaining translation of conceptual process models to workflow process models. More details about these proposed guidelines are contained in section 5.

- *Which of these proposed transformation guidelines are usable in practice?*

This was answered during the validation stage of our research. First we selected nine different conceptual process models from Philips healthcare RS such that this set is representative of a typical set of processes in any other company. This will guarantee that conclusions drawn using this set are generalizable to any other company. Secondly,

we implemented these conceptual process models as YAWL workflows. Thereafter, we made an analysis to determine which of the earlier proposed model transformation guidelines we actually used during this transformation process. It was discovered that all guidelines except the guideline "*Examine the constructs of both the conceptual and workflow modeling language based on the BWW ontology*" were used during the transformation process. We didn't use this guideline because we felt it was supplementary in that if one already knows how the different elements of a process are represented in both the conceptual and workflow modeling language, then it's quite faster to immediately map their corresponding constructs without having to strictly follow this guideline. Furthermore, documentations of modeling languages usually provide brief and concise description of their modeling constructs which are much more easier to understand at least from our point of view compared to effort required to understand and use this guideline. For our case, we were well acquitted to the constructs of YAWL so we could directly map the conceptual models directly to YAWL without having to use this guideline. Furthermore, during the validation stage new additional guidelines were also discovered, examples of which include:

- Specify a trigger for every task in the workflow process model
- Always make a distinction between explicit and implicit OR-splits
- For every XOR split task, critically determine the evaluation sequence of the different flow expressions specified for the outgoing flows of the task
- Specify the mechanism to be used to select a resource to execute a work item from a set multiple resources allowed to execute the same work item
- Set the queuing principle to be used to order multiple work items

All this newly discovered guidelines can still be considered usable since they were discovered during the actual transformation process. In summary, all the transformation guidelines except "*Examine the constructs of both the conceptual and workflow modeling language based on the BWW ontology*" are practically usable.

- *How often each guideline is used in practice?*

Only guidelines that apply to the functional and control aspects of the workflow process model were considered during our evaluation to determine how often these guidelines were used. This is motivated by the limited time frame we had to carry out our research. After the assessment, it was noted that the guideline "*Always make a distinction between explicit and implicit OR-splits*" was used the most number of times. This can be attributed to the fact that in all workflow process models, numerous routing choices have to be taken, of which explicit and implicit OR-splits are among. The least used guidelines are "*Complete the process model*" and "*Map similar constructs of both Modeling languages*". The less usage of the "*Complete the process model*" guideline can be attributed to the fact that the set of process models we selected and used for the transformation process, came along with very details explanation of the process

elements and as well as routing process. Therefore, there was less missing control flow information. However, this could be an isolated case, so in the generic sense we might be faced with a situation that this guideline is used quite more often. The guideline "*Map similar constructs of both Modeling languages*" was also used very few times. This is because when transforming any conceptual process model to workflow model, however complex or simple it is, we shall always map their constructs once. Therefore, for each of the nine conceptual models, a mapping was done only once for every translation. A complete list of the usage frequency of all the guidelines is contained in section 6.5.

7.2 Summary of Contributions

The main contribution that our work adds to the existing body of knowledge is it defines a comprehensive set of generic conceptual to workflow transformation guidelines which was until before this research lacking. The definition of this set of guidelines brings forth three key benefits namely;

- It further bridges the gap between the need for a plain communication basis on the one side (conceptual process modeling) and an unambiguous process description, covering details of the implementation (workflow process modeling), on the other side. These two perspectives of process modeling are a necessity in any practical BPM project. Now we have a much clearer set of guidelines to follow while transforming conceptual process models to workflow process models.
- It improves on the completeness and the correctness of transformations of conceptual to workflow process models. Since our guidelines are generic, it is applicable to transformations involving all the three major classes of process models namely; graph-structured, block-structured and quasi-structured process models therefore it has a wider range of applicability. The proposed set of guidelines also puts strong emphasis on ensuring a sound and correct workflow process model. Therefore, two out of four evaluation criteria (completeness, correctness, readability and reversibility) of model transformation approaches stated by Dumas et al. [22] will be positively impacted.
- It reduces on the error probability made when transforming conceptual process models to workflow process as well as process modeling in general. According to M. Rosemann et al. [59], large projects on process documentation heavily rely on novices and non-expert modelers. Without the aid of a well defined set of model transformation guidelines, novices and non-expert modelers are bound to make workflow process models which contain numerous errors. However, this risk is mitigated with the usage of the proposed model transformation guidelines.

7.3 Limitations and Future Research

We think that the validation of the proposed guidelines can be improved further by involving professional process modelers who are using different types of workflow

systems in their company. Our validation process involved only YAWL process models which might not be representative enough. In the future, expert opinions should also be got from different process modelers using a variety of workflow systems. The evaluation of the frequency of use of these guidelines should also be extended to cover all the guidelines. Currently, we only focused on guidelines associated with the function and control aspects of a process model. This will to a great extend highlight the level of importance which should be attached to each guideline. Since transforming conceptual process models to workflow process models is a procedure with numerous steps which could potentially be interdependent, we think it is also wise to determine an ideal sequence in which the transformation guidelines should preferably be applied. This will enable us to better guide process modelers on how to use these transformation guidelines. This is therefore a potential avenue for further research.

References

1. Aalst, W. van der and Hee, K.M. van. *Workflow management : models, methods, and systems*. MIT Press, Cambridge, Mass., 2004.
2. Aalst, W.M.P., Crom, P.J.N., Goverde, R.R.H.M.J., et al. ExSpect 6.4 An Executable Specification Tool for Hierarchical Colored Petri Nets. In, M. Nielsen and D. Simpson, eds., *Application and Theory of Petri Nets 2000*. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 455–464.
3. Aalst, W.M.P. van der. The Application of Petri Nets to Workflow Management. *Journal of Circuits, Systems, and Computers 8*, 1 (1998), 21–66.
4. van der Aalst, W.M.P. Woflan: a Petri-net-based workflow analyzer. *Systems Analysis Modelling Simulation 35*, 3 (1999), 345 – 357.
5. Aalst, W.M.P. Verification of workflow nets. In, P. Azéma and G. Balbo, eds., *Application and Theory of Petri Nets 1997*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1997, pp. 407–426.
6. Asztalos, M., Mészáros, T., and Lengyel, L. Generating Executable BPEL Code from BPMN Models. *5th International Workshop on Graph-Based Tools (GraBaTs)*, (2009).
7. Bandinelli, S., Fuggetta, A., Ghezzi, C., and Grigolli, S. Process enactment in SPADE. In, J.-C. Derniame, ed., *Software Process Technology*. Springer-Verlag, Berlin/Heidelberg, pp. 67–83.
8. Becker, J., Rosemann, M., and Uthmann, C. Guidelines of Business Process Modeling. In, W. Aalst, J. Desel and A. Oberweis, eds., *Business Process Management*. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 30–49.
9. Beecham, S., Baddoo, N., Hall, T., Robinson, H., and Sharp, H. *Protocol for a Systematic Literature Review of Motivation in Software Engineering*. School of Computer Science, Faculty of Engineering and Information Sciences, University of Hertfordshire, 2006.
10. Casati, F., Grazia Fugini, M., Mirbel, I., and Pernici, B. WIRES: A Methodology for Developing Workflow Applications. *Requirements Engineering 7*, 2 (2002), 73–106.
11. Curtis, B., Kellner, M.I., and Over, J. Process modeling. *Communications of the ACM- Special issue on analysis and modeling in software development 35*, 9 (1992), 75–90.
12. Davenport, T.H. *Process innovation : reengineering work through information technology*. Harvard Business School Press, Boston, Mass., 1993.
13. Decker, G., Dijkman, R., Dumas, M., and García-Bañuelos, L. Transforming BPMN Diagrams into YAWL Nets. In, M. Dumas, M. Reichert and M.-C. Shan, eds., *Business Process Management*. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 386–389.
14. Dehnert, J. and Aalst, W.M.P. van der. Bridging The Gap Between Business Models And Workflow Specifications. *Int. J. Cooperative Inf. Syst. 13*, 3 (2004), 289–332.
15. Dehnert, J. *A Methodology for Workflow Modeling - From business process modeling towards sound workflow specification*. Berlin, Heidelberg, 2003.
16. Denna, E.L., Perry, L.., and Jasperson, J. Reengineering and REAL business process modeling. In, V. Grover and W.. Kettinger, eds., *Business Process Change: Reengineering Concepts, Methods, and Technologies*. Idea Group Publishing, London, 1995, pp. 350–375.
17. Denning, P.J. A new social contract for research. *Communications of the ACM 40*, 2 (1997), 132–134.
18. VAN DER AALST, W.M.., ter Hofstede, A.H.., Kiepuszewski, B., and Barros, A.. Workflow Patterns. In, *Distributed and Parallel Databases*. Kluwer Academic Publishers, The Netherlands, 2003, pp. 5–51.
19. Dijkman, R.M., Dumas, M., and Ouyang, C. Semantics and analysis of business process models in BPMN. *Information and Software Technology 50*, 12 (2008), 1281–1294.
20. Dreiling, A., Rosemann, M., van der Aalst, W.M.P., and Sadiq, W. From conceptual process models to running systems: A holistic approach for the configuration of enterprise system processes. *Decision Support Systems 45*, 2 (2008), 189–207.

21. Dumas, M., Aalst, W. van der, and Ter Hofstede, A. *Process-aware information systems : bridging people and software through process technology*. John Wiley & Sons, Hoboken, N.J., 2005.
22. Dumas, M. Case study: BPMN to BPEL model transformation. *5th International Workshop on Graph-Based Tools (GraBaTs)*, (2009).
23. Dumas, M. Workflow Modeling. In, *Encyclopedia of Database Systems*. 2009, pp. 3551–3557.
24. Falkenberg, E.D. *A framework of information system concepts : the FRISCO report (Web edition)*. University of Leiden, Department of Computer Science, Leiden, 1998.
25. Gardner, T. UML modeling of automated business processes with a mapping to BPEL4WS. *First European Workshop on Object Orientation and Web Services*, Springer (2003).
26. Georgakopoulos, D., Hornick, M., and Sheth, A. An overview of workflow management: From process modeling to workflow automation infrastructure. *Distributed and Parallel Databases 3*, 2 (1995), 119–153.
27. Green, P. and Rosemann, M. Integrated process modeling: An ontological evaluation. *Information Systems 25*, 2 (2000), 73–87.
28. Han, Z., Zhang, L., and Ling, J. Transformation of UML Activity Diagram to YAWL. In, K. Popplewell, J. Harding, R. Poler and R. Chalmeta, eds., *Enterprise Interoperability IV*. Springer London, London, 2010, pp. 289–299.
29. Hevner, A.R., March, S.T., Park, J., and Ram, S. Design Science in Information Systems Research. *MIS Quarterly 28*, 1 (2004), 75–105.
30. Hilliard, R. Viewpoint Modeling. *First ICSE Workshop on Describing Software Architecture with UML*, (2001).
31. Hofstede, A.H.M., Adams, M., Russell, N., and Aalst, W.M.P. *Modern Business Process Automation YAWL and its Support Environment*. Springer-Verlag Berlin Heidelberg, Berlin, Heidelberg, 2010.
32. KHATOON, S. and MAHMOOD, A. METHODOLOGY FOR MAPPING BUSINESS PROCESS INTO EXECUTION LANGUAGE. *International Journal of Engineering Science and Technology 3*, 1 (2011).
33. Kiepuszewski, B., ter Hofstede, A.H.M., and van der Aalst, W.M.P. Fundamentals of control flow in workflows. *Acta Informatica 39*, 3 (2003), 143–209.
34. Kitchenham, B., Pearl Brereton, O., Budgen, D., Turner, M., Bailey, J., and Linkman, S. Systematic literature reviews in software engineering – A systematic literature review. *Information and Software Technology 51*, 1 (2009), 7–15.
35. Kitchenham, B. *Procedures for performing systematic reviews*. Keele University and NICTA, 2004.
36. Kloos, O., Nissen, V., and Petsch, M. From Process to Simulation - A Transformational Model Approach. *EMISA*, (2009), 83–96.
37. Koehler, J., Tirenni, G., and Kumaran, S. From business process model to consistent implementation: a case for formal verification methods. IEEE, 96–106.
38. Krogstie, J., Lindland, O.I., and Sindre, G. Defining quality aspects for conceptual models. *IFIP international working conference on Information system concepts: Towards a consolidation of views (ISCO)*, (1995), 216–231.
39. Küster, J.M., Ryndina, K., and Hauser, R. A Systematic Approach to Designing Model Transformations. *IBM Research Technical Report RZ 3621*, (2005).
40. Lawrence, P. and Workflow Management Coalition. *Workflow handbook, 1997*. John Wiley & Sons, Chichester, West Sussex, England; New York, 1997.
41. Lindland, O.I., Sindre, G., and Solvberg, A. Understanding quality in conceptual modeling. *IEEE Software 11*, 2 (1994), 42–49.
42. List, B. and Korherr, B. An evaluation of conceptual business process modelling languages. *2006 ACM symposium on Applied computing*, ACM Press (2006), 1532–1539.

43. Lohmann, N., Verbeek, E., and Dijkman, R. Petri Net Transformations for Business Processes – A Survey. In, K. Jensen and W.M.P. Aalst, eds., *Transactions on Petri Nets and Other Models of Concurrency II*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 46–63.
44. Meertens, L.O., Iacob, M.-E., and Eckartz, S.M. Feasibility of EPC to BPEL Model Transformations Based on Ontology and Patterns. In, S. Rinderle-Ma, S. Sadiq and F. Leymann, eds., *Business Process Management Workshops*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 347–358.
45. Mendling, J., Lassen, K.B., and Zdun, U. Transformation Strategies between Block-Oriented and Graph-Oriented Process Modelling Languages. *Multikonferenz Wirtschaftsinformatik 2006 (MKWI 2006)*, GITO-Verlag Berlin (2006), 297–312.
46. Mendling, J., Moser, M., and Neumann, G. Transformation of yEPC business process models to YAWL. ACM Press (2006), 1262.
47. Mendling, J., Neumann, G., and Aalst, W. Understanding the Occurrence of Errors in Process Models Based on Metrics. In, R. Meersman and Z. Tari, eds., *On the Move to Meaningful Internet Systems 2007: CoopIS, DOA, ODBASE, GADA, and IS*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 113–130.
48. Mendling, J., Reijers, H.A., and van der Aalst, W.M.P. Seven process modeling guidelines (7PMG). *Information and Software Technology 52*, 2 (2010), 127–136.
49. Mendling, J., Reijers, H.A., and Cardoso, J. What Makes Process Models Understandable? In, G. Alonso, P. Dadam and M. Rosemann, eds., *Business Process Management*. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 48–63.
50. Mendling, J., Reijers, H.A., and Recker, J. Activity labeling in process modeling: Empirical insights and recommendations. *Information Systems 35*, 4 (2010), 467–482.
51. Meyer-Wegener, K. and Bohm, M. Conceptual workflow schemas. *Fourth IFCIS International Conference on Cooperative Information Systems*, IEEE (1999), 234–242.
52. Mili, H., Tremblay, G., Jaoude, G.B., Lefebvre, É., Elabed, L., and Boussaidi, G.E. Business process modeling languages: Sorting through the alphabet soup. *ACM Computing Surveys 43*, 1 (2010), 1–56.
53. Ouyang, C., Dumas, M., Aalst, W.M.P. van der, Hofstede, A.H.M. ter, and Mendling, J. From business process models to process-oriented software systems. *ACM Trans. Softw. Eng. Methodol. 19*, 1 (2009).
54. Ouyang, C., Dumas, M., Breutel, S., and Hofstede, A. Translating Standard Process Models to BPEL. In, E. Dubois and K. Pohl, eds., *Advanced Information Systems Engineering*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006, pp. 417–432.
55. Ouyang, C., Dumas, M., H.M., A., and van der Aalst, W.M.P. Pattern-Based Translation of BPMN Process Models to BPEL Web Services. *International Journal of Web Services Research 5*, 1 (2008), 42–62.
56. Ouyang, C., Dumas, M., Ter Hofstede, A., and Van Der Aalst, W. From BPMN Process Models to BPEL Web Services. IEEE (2006), 285–292.
57. Reijers, H.. *Design and control of workflow processes business process management for the service industry*. Springer, Berlin [etc.], 2003.
58. La Rosa, M., Wohed, P., Mendling, J., ter Hofstede, A.H.M., Reijers, H.A., and van der Aalst, W.M.P. Managing Process Model Complexity Via Abstract Syntax Modifications. *IEEE Transactions on Industrial Informatics 7*, 4 (2011), 614–629.
59. Rosemann, M. Potential pitfalls of process modeling: part A. *Business Process Management Journal 12*, 2 (2006), 249–254.
60. Russell, N., Hofstede, A.H.M.T., and van der Aalst, W.M.. Exception Handling Patterns in Process-Aware Information Systems ⋆. *CAiSE'06*, (2006), 288–302.
61. Sadiq, S.W. and Orlowska, M.E. On Capturing Exceptions in Workflow Process Models. In, *BIS 2000*. Springer Verlag, Great Britain, 2000, pp. 3–19.

62. Sadiq, W., Marjanovic, O., and Orlowska, M.E. Managing Change and Time in Dynamic Workflow Processes. *International Journal of Cooperative Information Systems 9*, 1-2 (2000), 93–116.
63. Sadiq, W. and Orlowska, M.E. On Correctness Issues In Conceptual Modeling Of Workflows. *5th European Conference on Information Systems (ECIS '97)*, (1997).
64. Sadiq, W. and Orlowska, M.E. On capturing process requirements of workflow based business information systems. *3rd International Conference on Business Information Systems*, Springer Verlag (1999), 281–294.
65. Scheithauer, G., Wirtz, G., and Toklu, C. Bridging the Semantic Gap Between Process Documentation and Process Execution. *SEKE*, (2008), 549–554.
66. Schuette, R. and Rotthowe, T. The Guidelines of Modeling – An Approach to Enhance the Quality in Information Models. In, T.-W. Ling, S. Ram and M. Lee, eds., *Conceptual Modeling – ER '98*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998, pp. 240–254.
67. Sheth, A., Georgakopoulos, D., Joosten, S.M.M., et al. Report from the NSF workshop on workflow and process automation in information systems. *ACM SIGMOD Record 25*, 4 (1996), 55–67.
68. Söderström, E., Andersson, B., Johannesson, P., Perjons, E., and Wangler, B. Towards a Framework for Comparing Process Modelling Languages. *14th International Conference on Advanced Information Systems Engineering (CAiSE)*, (2002), 600–611.
69. Svatoš, O. Conceptual Process Modeling Language: Regulative Approach. *9th Undergraduate and Graduate Students eConf. and 14th Business & Government Executive Meeting on Innovative Cross-border eRegion*, (2007).
70. Takeda, H., Veerkamp, P., Tomiyama, T., and Yoshikawa, H. Modeling Design Process. *AI Magazine 11*, 4 (1990), 37–48.
71. Tick, J. Workflow Model Representation Concepts. *7th International Symposium of Hungarian Researchers on Computational Intelligence*, (2006), 329–337.
72. Vaishnavi, V. and Kuechler, W. Design Science Research in Information Systems. *design science research in information systems and technology*, (2004).
73. Vanderfeesten, I., Cardoso, J., Mendling, J., Reijers, H.A., and van der Aalst, W. Quality Metrics for Business Process Models. In, L. Fischer, ed., *2007 BPM and workflow handbook*. Lighthouse Point : Future Strategies, 2007, pp. 179–190.
74. Verbeek, E., Hattem, M., Reijers, H., and Munk, W. Protos 7.0: Simulation Made Accessible. In, G. Ciardo and P. Darondeau, eds., *Applications and Theory of Petri Nets 2005*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 465–474.
75. Wand, Y. and Weber, R. On the deep structure of information systems. *Information Systems Journal 5*, 3 (1995), 203–223.
76. Wang, W., Ding, H., Dong, J., and Ren, C. A Comparison of Business Process Modeling Methods. *IEEE International Conference on Service Operations and Logistics and Informatics (2006)*, IEEE (2006), 1136–1141.
77. Weber, R. *Ontological Foundations of Information Systems*. Melbourne, Australia, 1997.
78. Wedemeijer, L. and de Bruin, E.G.A.J. Conceptual process models: using process architecture in practice. IEEE (2004), 1002–1006.
79. Weske, M. and Vossen, G. *Workflow Languages*. Univ., 1997.
80. WFMC. *Workflow Management Coalition Workflow Standard: Process Definition Interface – XML Process Definition Language*. Workflow Management Coalition, 2005.
81. White, S. Using BPMN to Model a BPEL Process. *BPTrends 3*, 3 (2005), 1–18.
82. Yang, W. and Li, F. Workflow modeling: a structured approach. IEEE, 433–438.
83. Ziemann, J. and Mendling, J. EPC-Based Modelling of BPEL Processes: a Pragmatic Transformation Approach. *Seventh International Conference Modern Information Technology in the Innovation Processes of the Industrial Enterprises (MITIP)*, (2005).

Appendices

All the nine Phillips conceptual process models have been deleted for privacy reasons