

Constructing a Knowledge Base for Business Process Management

DISSERTATION

zur Erlangung des akademischen Grades

Doktor/in der technischen Wissenschaften

eingereicht von

Thanh Tran Thi Kim

Matrikelnummer 0727719

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: Univ.Prof. Dipl.-Ing. Dr.techn. Hannes Werthner

Diese Dissertation haben begutachtet:

(Univ.Prof. Dipl.-Ing. Dr.techn.
Hannes Werthner)

(Univ.Prof. Mag. Dr. Reinhard
Pichler)

Wien, 12.11.2012

(Thanh Tran Thi Kim)

Constructing a Knowledge Base for Business Process Management

DISSERTATION

submitted in partial fulfillment of the requirements for the degree of

Doktor/in der technischen Wissenschaften

by

Thanh Tran Thi Kim

Registration Number 0727719

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Univ.Prof. Dipl.-Ing. Dr.techn. Hannes Werthner

The dissertation has been reviewed by:

(Univ.Prof. Dipl.-Ing. Dr.techn.
Hannes Werthner)

(Univ.Prof. Mag. Dr. Reinhard
Pichler)

Wien, 12.11.2012

(Thanh Tran Thi Kim)

Erklärung zur Verfassung der Arbeit

Thanh Tran Thi Kim
Halmgasse 3, 1020 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum)

(Unterschrift Verfasserin)

Acknowledgements

I would like to express my heartfelt gratitude to my supervisor, Prof. Hannes Werthner. This work would not have been possible without his guidance, support and encouragement.

I am very grateful to Prof. Dieter Merkl, who supported me with valuable technical discussions and mental guidance. His cheerful spirit has made my studying time in office more pleasant.

I am also thankful to Prof. Reinhard Pichler, Prof. Christian Huemer who reviewed and revised my thesis very carefully. Special thanks to Christian Pichler who cheered me up and supported me in writing.

I would like to thank my colleagues for their feedback, advices and sympathy. We have spent great time together in the office as well as outside.

I would also like to extend warm thanks to my Vietnamese and international friends who have always been there when I needed them. Without their support, I would not have overcome the hard time of my study.

Last but not least, I would like to pay high regards to my parents, my brother and my sister for their love, concern and patience.

Abstract

Process operations of a company may be recorded in log data in different ways. For instance, process-aware information systems automatically record business process operations. Several approaches, such as semantic process mining or data warehousing attempt to take advantage of this data for supporting the business process management of companies. While semantic process mining approaches focus on processing log data, data warehousing techniques aim at integrating log data with various related data sources, such as organizational data and resource data. The common objective of these approaches is providing valuable business insight regarding business process operations. However, the approaches mentioned have limitations inhibiting the extraction of valuable business insight. In particular, semantic process mining discovers knowledge from event logs and lifts the extracted information to the conceptual level. However, the support for integrating log data with related data sources is limited. In case of data warehousing approaches, the advantages of semantic technologies are not fully utilized for answering business-relevant questions.

For addressing these problems, the main research question addressed in this thesis is: "How can one combine existing approaches and extend their capabilities with the ultimate goal of providing an alternative solution for answering business-relevant questions?" Thus, the research presented introduces an approach which integrates event log data with related data sources while building upon semantic technologies resulting in a knowledge base containing information on business process operations. Furthermore, the research presented extends the knowledge base through a set of additional axioms used for reasoning, querying, and answering business related questions. Having the resulting ontology at hand allows extracting answers to business-related questions which are beyond the scope of solely applying semantic process mining techniques.

Kurzfassung

Prozessable innerhalb einer Firma können auf verschiedene Weise dokumentiert werden, zum Beispiel durch die automatische Generierung von Log-Daten durch Workflow Systeme. Bekannte Ansätze wie semantisches Process Mining oder Data Warehousing bieten Möglichkeiten, aus diesen Daten wertvolle Informationen für das Management eines Unternehmens zu extrahieren. Dabei basieren die Ansätze im Bereich des semantischen Process Mining auf der Analyse von Log-Daten, wohingegen Ansätze aus dem Bereich des Data Warehousing darauf abzielen verschiedene Datenquellen (wie z.B. Organisationsdaten und Informationen bezüglich Betriebsmitteln) zu integrieren. Jedoch bringen diese Ansätze Einschränkungen mit sich, welche im Rahmen dieser Arbeit behandelt werden. Im Speziellen extrahieren die Ansätze aus dem Bereich des semantischen Process Mining wertvolle Informationen und stellen diese auf einer konzeptuellen höheren Ebene dar. Jedoch ist die Integration unterschiedlicher Datenquellen eingeschränkt. Im Bereich des Data Warehousing werden die Möglichkeiten semantischer Technologien nicht vollständig genutzt.

Daher ist das Ziel dieser Arbeit die Vorteile der oben genannten Methoden zu nutzen um einen erweiterten Ansatz zur Verfügung zu stellen. Dieser Ansatz erlaubt es wertvolle Informationen über das Unternehmen zu extrahieren. Dabei werden Log-Daten mit weiteren unternehmensrelevanten Daten unter der Verwendung von semantischen Technologien integriert. Dies resultiert in einer Ontologie welche Informationen über das Unternehmen enthält. Darüber hinaus wird diese Ontologie durch Axiome ergänzt, welche die erweiterte Analyse von Unternehmensdaten erlauben. Mit Hilfe dieses Ansatzes ist es nun möglich Einsicht in Unternehmensableitungen zu erlangen was durch die Anwendung existierender Ansätze nur in eingeschränkter Möglichkeit war.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Research Questions	3
1.3	Methodology	4
1.4	The Approach in General	5
1.5	Contributions	10
1.6	Thesis Outline	11
2	State of the art	13
2.1	Process-aware Information Systems	13
2.2	Process mining	17
2.3	Semantic Process Mining	20
2.4	Ontology-based data integration	23
3	Background	27
3.1	TOVE ontology	27
3.2	Adonis	34
3.3	ProM	36
4	Data sources and Data Generation	39
4.1	Event log data	39
4.2	Event log data generation	41
4.3	Involved data sources	51
5	Ontology-based Data Integration	53
5.1	Modified TOVE	53
5.2	Ontology-based data integration	56
5.3	Implementation of Ontology-based Data Integration	57
6	Knowledge construction	65
6.1	Question analysis	66
7	Case study and Evaluation	75
7.1	Case Study - An Order Process	75

7.2	Questions and Answers	77
7.3	Evaluation	87
8	Conclusion and open issues	89
8.1	Summary	89
8.2	Contributions	89
8.3	Limitations and Future Works	90
	Bibliography	92

Introduction

1.1 Motivation

The competition among companies in today's markets is intense and it increases steadily. In such a highly competitive environment, improving the quality of products and responding to customer demands are essential factors for success. This requires observing tendencies and changes of the market as well as thoroughly understanding the current situations of other competitive companies. For dealing with these developing trends, technical assistance based on Information Systems (IS) is important for supporting professional staff.

IS-based approaches play an important role and have positively influenced the development strategies of companies in a variety of perspectives, including decision making support, data management, and customer relationship management. Instead of struggling against paper-based systems with tons of documents and accounts, companies can save time and expenses for enterprise management with the advantages gained by using enterprise software.

The history and development of information systems has gone through several trends ranging from (i) building to assembling and from (ii) data orientation to process orientation, resulting in process-aware information systems (PAIS) [15]. Process awareness in information systems has been well researched and applied in industrial projects. Reports on the development of workflow management systems (WfMS) [2], one kind of process aware information system (PAIS), support this observation. As stated in [16], the worldwide revenue derived from workflow technologies grew from \$4.3bn in 2000 to \$8.3bn in 2003, at a compound annual growth rate of 31%. In a report of the *Workflow Management Systems Market 2010-2013* [24], "the adoption of workflow management systems has reduced response time by 20 percent and increased productivity by impressive 50 percent". As a result, the presence of PAISs in companies is progressive and business people are becoming more familiar with the availability of PAISs.

A PAIS is defined as *"a software system that manages and executes operational processes involving people, applications, and/or information sources on the basis of process models"* [15]. As implied in its definition, a PAIS includes sub-systems (i.e., applications), people, and information sources which interact with each other to accomplish particular tasks.

Within a PAIS, process models specify how people interact with resources (such as equipment, information, etc.) to accomplish a particular goal. Process models describe business processes, which are then operated to deliver services or products of a company. Moreover, process models pertain to the way how business processes are operated in an organization. Process models are an effective means to transfer knowledge between humans and are considered as a "visual language" to facilitate the communication among managers and employees in the enterprise sphere. Hence, when building an information system for enterprises, people from the business side can easily communicate with the information system developers via process models.

Having such process models at hand, a PAIS allows assigning users to perform tasks which reside in process models. For achieving these tasks the users may utilize applications or resources. Figure 1.1 depicts an example of a process model in the context of a PAIS [15].

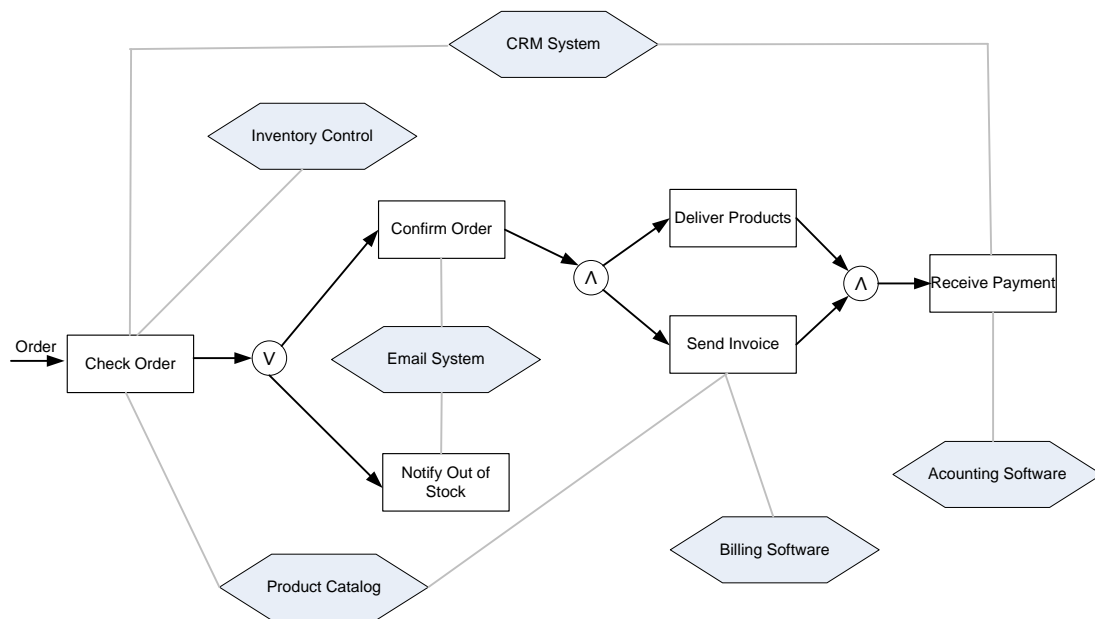


Figure 1.1: A Fulfill Order Process Model Example [15]

In this figure, rectangle and diamond frames represent the activities and the applications, respectively, which are necessary to accomplish an order process. For instance, to confirm an order to a customer, an employee needs to use the *Email System* to compose and send an email to the customer. The use of an *Email System* to perform the activity *Confirm Order* by an employee is accommodated by the PAIS.

In general, a process model can be derived by following different approaches. First, a process model can be designed based on the experience of domain experts. It is the outcome of the cooperation between business people and IT people. In other words, the managers who know exactly what the enterprise needs to improve, exchange their knowledge with IT people. Thereby, the requirements are analyzed and structured in business processes by business analysts and are consequently designed as process models by IT architects. Second, a process model also can be created by utilizing process mining techniques [58] which mine process models based on log data generated and recorded by PAISs. The latter is discussed in the following.

Process models may be automatically constructed by applying process mining techniques on log files recorded and generated by PAISs. This results in raising the semantic level by linking labels contained in the event log data to corresponding concepts in ontologies. Additional areas of application of process mining include utilizing these techniques for supporting the analysis and the monitoring of business processes. However, the results derived from process mining are limited to the labels of extracted from the log data. Consequently, the scope of business-related questions that can be answered based on these results is limited to the scope of the information contained in the mined process model.

The area of data warehousing aims at integrating data from various sources for providing information relevant to business process operations. These approaches take advantage of the data sources from various applications present in an enterprise sphere. In other words, relevant information is gathered from different sources for providing business managers with relevant information for making business-relevant decisions. Consequently, the range of business relevant questions that can be answered by data warehousing can cover many perspectives because of the diversity of data sources. Approaches in data warehousing may also build upon semantic technologies resulting in semantic data warehousing. However, in the context of workflow data and PAISs, data warehousing has only been applied without considering semantic technologies [64].

1.2 Research Questions

Although every approach mentioned earlier has its own techniques, all of them follow similar objectives, which is modeling or providing information to the business process operations of a company. However, the approaches mentioned have limitations inhibiting the extraction of business information for business operation. The process mining approach can discover the knowledge in event logs but does not extend the information contained in the log data with relevant data sources. In case semantic process mining is applied, the information of the log data is lifted to the conceptual level only. In case of the data warehouse approaches, although the integration of log data with other data sources is carried out, the advantages of semantic technologies are not utilized for answering business-relevant questions.

For addressing these problems, the main research question of the work presented in this thesis corresponds to: "How can one combine existing approaches and extend their capabilities with the ultimate goal of providing an alternative solution for answering business-relevant

questions?” Thus, the aim of this research is to create a solution for the problems faced in the aforementioned approaches. The approach proposed in this research tries to take advantage of various data sources to respond to the questions about business process operations of a company. In particular, the approach developed deals with the following issues:

- How can one integrate data sources scattered in the systems of an enterprise?
- What kind of knowledge can be extracted based on such an integration?
- How can the knowledge extracted support business process management within an enterprises?

1.3 Methodology

The methodology of the approach presented follows the following four steps:

1. Data collection: Collecting data from PAIS and related sources is one core aspect of the approach presented. Within the work presented the following three kinds of data are collected: (i) event log data representing information on the execution of business processes, (ii) organizational data, and (iii) resource data. In the context of the work presented the event log data is generated by the simulation of a process model designed using the Adonis tool [23]. The organizational and resource data are assumed to exist and to be ready for further processing by the approach. The result of this step is the preparation of the necessary data for analysis and further processing.
2. Data integration: Following the data collection conducted in Step 1, the different data sources are then integrated using ontology-based data integration techniques. In the work presented in this thesis the TOVE ontology [35] serves as the basis for performing data integration. In particular, the TOVE ontology is modified by adding additional relevant concepts and properties, resulting in the ModTOVE ontology. Actual data instances are transferred into the TOVE ontology. To summarize, the result of this step is the ModTOVE ontology containing extended concepts as well as actual business information from the different data sources.
3. Knowledge Construction: Having the ModTOVE ontology at hand, additional concepts and relations are added to the ModTOVE ontology by defining and inserting axioms. Consequently, the result of this step is the extended ModTOVE ontology serving as the knowledge base of the approach presented.
4. Questions and Answers: Utilizing the ModTOVE ontology allows querying the knowledge base for relevant business information. In particular, querying the knowledge base allows answering questions providing valuable business insight. Such questions are referred to as perspectives in the following. Furthermore, the different perspectives defined

and answered in the context of the work presented are also used for comparing and evaluating the approach presented in regards to existing approaches.

1.4 The Approach in General

Scenario

The work presented can be applied in scenarios which happen in a company using WfMSs to support its enterprise operations, as depicted in Figure 1.2. The business process operations of the company are executed and recorded in event log data. Additionally, there are several invoked applications which have capabilities to accomplish process steps. In the scenario presented in this thesis, the applications support the management of employees and resource consumption. The applications also have their own data containing information related to event log data. With the existence of event log data and its related data sources (i.e., organization and resource management data maintained in the applications), the assumption made in this thesis is the possibility to collect these data sources from a company using WfMSs for business management.

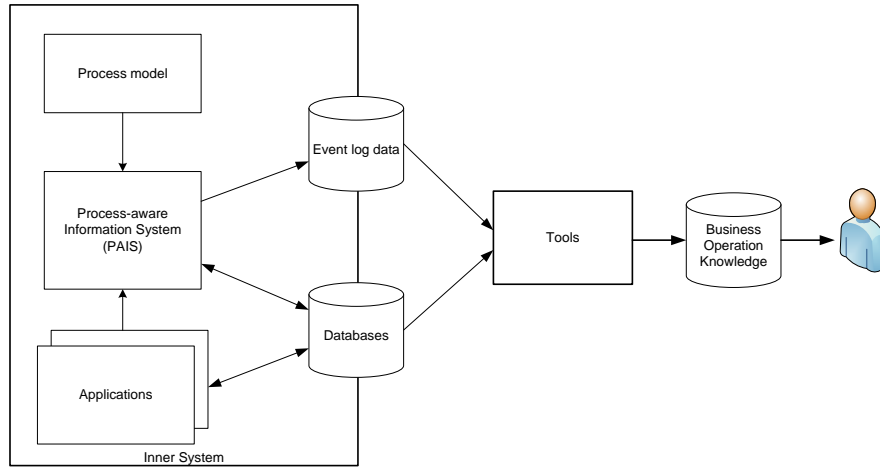


Figure 1.2: Scenario

Design Science Approach

The approach in this thesis is constructed based on the generic approach for Design Science in Information Systems research [30] as shown in Figure 1.3.

As mentioned in [30], information system (IS) research is the combination of the business needs from the environment containing the phenomena of interest and the knowledge base providing the raw materials to accomplish the IS research. In the scenario of the work presented, the environment is a company using a WfMS for its business operation management. Within the environment, the human element is the staff of the company participating in business processes of the company. The *process* element of the environment is represented by the business pro-

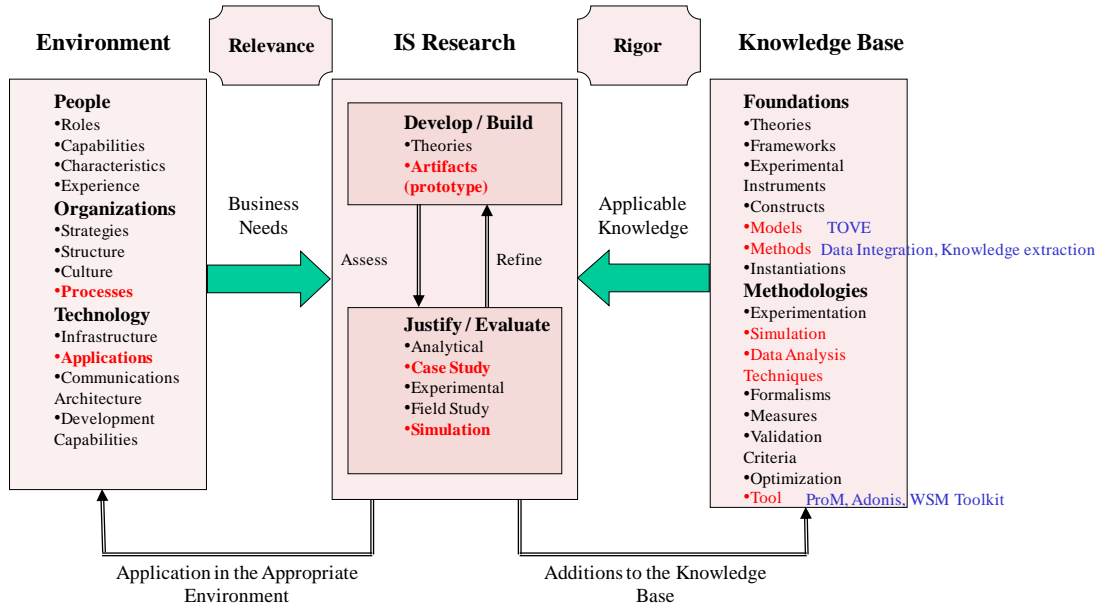


Figure 1.3: Generic Approach [30] (Words in red refer to concepts used in the thesis)

cesses. The *applications* of the environment includes the WfMS and its involved applications. The *business needs* are the research questions that could be answered by the approach, as defined earlier. Specifically, the question tackled is extracting knowledge from data sources maintained in the WfMS and its related data sources for supporting business managers within a company.

The IS in the work presented in this thesis aims at creating *artifacts* to satisfy the business need raised in the environment. In particular, a *simulation* is conducted for the purpose of generating event log data which are extracted from the log data of WfMSs. Furthermore, in the context of this thesis, it is assumed that other data sources on an organization and its resources are available for use. The current research is based on the knowledge foundations including a *model* which is the TOVE ontology. For dealing with the different data sources the *methodologies* simulation and data analysis are applied. Furthermore, the *methods* applied include data integration and knowledge extraction. Finally, the artifacts are evaluated based on a *prototype* implementation and justified based on a simulated case study. Besides, the *tools* used in the current work include ProM, Adonis and the WSM Toolkit. The details of the generic approach are introduced in the following.

Framework

The framework of the approach presented, as illustrated in Figure 1.4, contains three steps including data generation, data integration and knowledge construction. The prototype provides interfaces for users to interact with the systems in the data generation and the data integration processes. The data sources handled by the prototype are Adonis log data, organization data,

resource data and the ModTOVE ontology, resulting in the ModTOVE knowledge base. The prototype is built in the Eclipse [17] environment using the Java programming language. The core building blocks of the framework, namely data generation, data integration, and knowledge construction, are presented in the following.

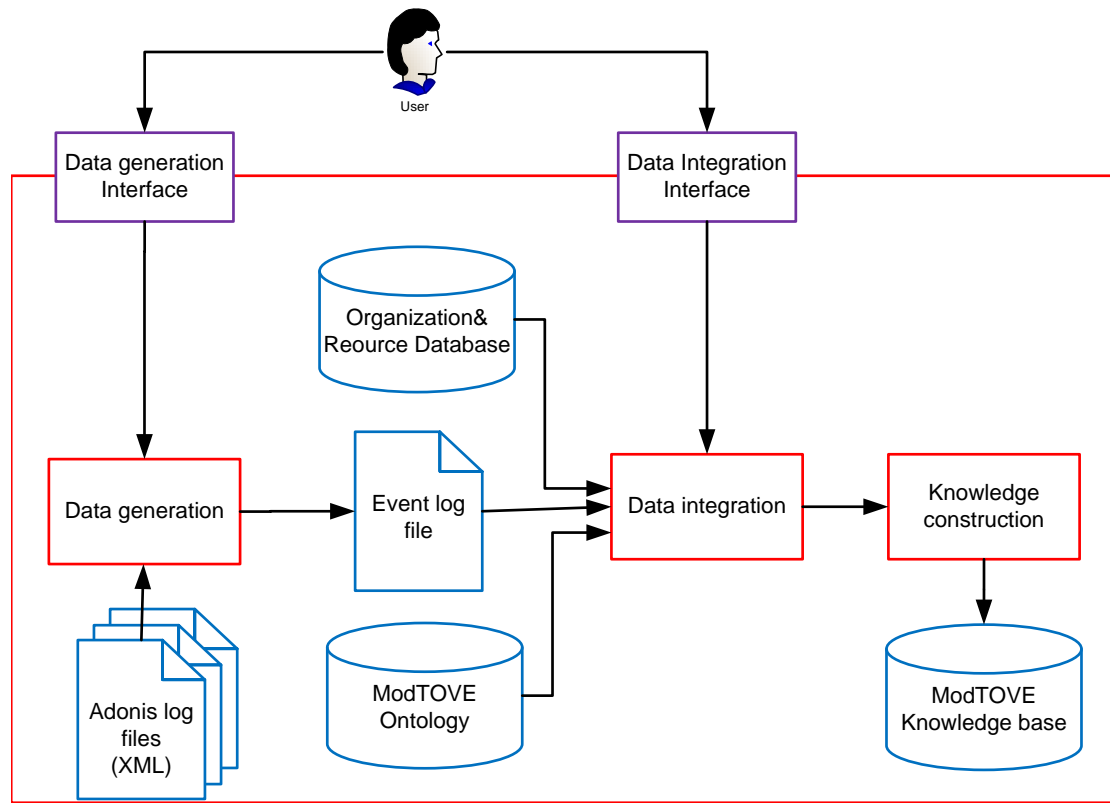


Figure 1.4: Overview of the framework

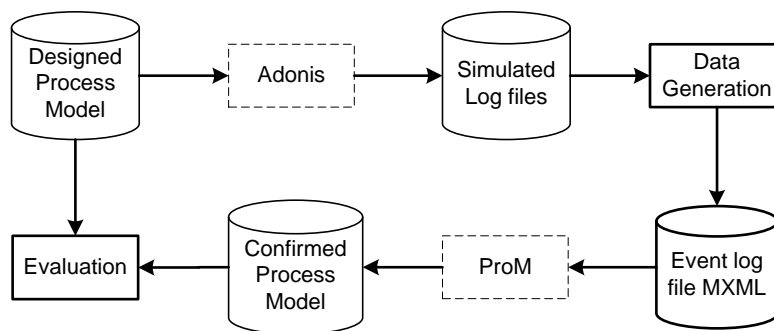


Figure 1.5: Data Generation

Data generation

The business process execution through a WfMS of an organization is recorded in event log data. The objective of the research presented is extracting knowledge hidden in the event log data combined with complementary data sources. Although the event data can be collected from practical systems, the real data always needs preliminary processing which relates to different research areas. Therefore, the log data for conducting the experiment of this research is generated using Adonis instead of collecting real-world data. The generated event log covers essential characteristics of event log data in real systems which are relevant in the course of this thesis. Figure 1.5 provides an overview of the data generation component, which is described in detail in the following.

The generation starts with a process model designed in Adonis. The process model describes the business process of the company described in the scenario earlier. The business process management toolkit of Adonis [29] supports different functionalities to create and simulate business processes. The model created by Adonis contains relevant information describing process operations. By using the process stepper functionality, the business process execution can be simulated and recored in log files in XML format. In particular, all the possible paths of a process model are run step by step with the intervention of users in decision nodes. The log file contains the properties of the model and the process path to which they are simulated. Therefore, the information contained in each log file is semantically similar to the data of a process instance contained in an event log file. After a number of simulations the different log files are collected. The structure and content of the log files are analyzed and transformed to an event log file by the Data Generation tool. The event log data is stored using the MXML format so that it can be used as input data of ProM [1, 60]. In oder to evaluate the generation process, the ProM is used as a tool to create a process model from the simulated event log file. The model created by ProM and the model designed by Adonis are compared visually to evaluate the data generation component.

Data Integration

This component enriches event log data with the related information contained in organizational data and resource data. In particular, this component integrates event log data obtained from the data generation component with organizational data and resource data. As mentioned earlier, the organizational data and resource data are assumed to exist in the in the context of an organization. The component builds upon ontology-based data integration techniques for integrating the different data sources [61, 39, 22]. The data integration step uses the ModTOVE ontology as a conceptual framework [34]. Basically, data integration techniques involve two steps, namely the mapping of concepts and the transformation of instances, as depcited in Figure 1.6.

Concept mapping using the ModTOVE ontology stands for a global ontology as mentioned in [61]. The data tables of the data sources are mapped to the suitable concepts of ModTOVE manually. The prototype provides Concept Mapping and Instance Transferring functionalities. The Concept Mapping supports users with an interface displaying the concepts of ModTOVE

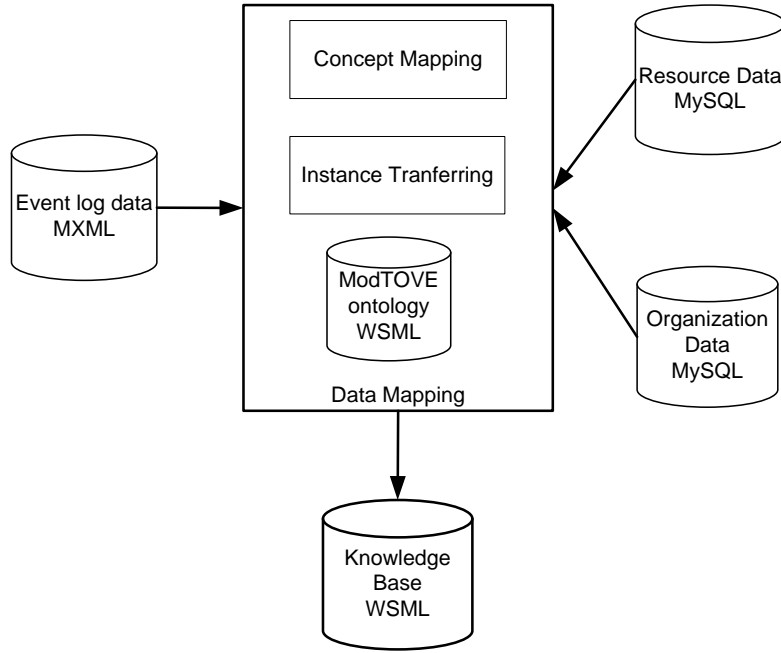


Figure 1.6: Ontology-based Data Mapping

and the data sources in forms of tables and their properties. Users need to decide which concepts of the ontology are suitable to be mapped to the data sources. For transferring instances, the prototype allows users to trigger transferring of instances to the ModTOVE ontology.

There are two types of data transferring called massive dump and query dump [22]. Massive dump is the transferring of instances to ontology without any selection. Query dump is the transferring of selected specified through dedicated queries. The result of the data integration is a knowledge base in WSM (Web service modeling language) format [63]. Although we do not need the support of web services in our approach, we take WSM for storing and querying our knowledge base. In the approach presented in this thesis, both options for transferring data are utilized. However, the knowledge represented in this knowledge base is still simple and limited to simple relations between its concepts. The reasoning of the knowledge base is fostered by adding axioms built in the Knowledge Construction component.

Knowledge Construction

The data integration in the previous section yields an ModTOVE ontology with instances populated from data sources. By using ontology reasoning techniques, new knowledge is constructed based on the primitive relations between concepts in ModTOVE. The result of the knowledge construction is a knowledge base with axioms that can be used for reasoning.

Figure 1.7 illustrates the knowledge construction component. The questions represent in-

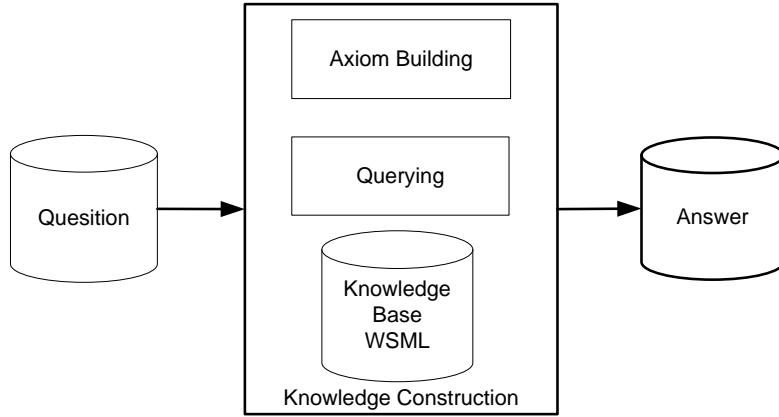


Figure 1.7: Knowledge Construction

sights into business information relevant to an organization and are mainly raised in the context of processes, the organization, and resources. Some of these questions can be answered by querying the ModTOVE ontology obtained from the data integration component. However, other more complex questions are impossible to be answered by the primitive ontology. Therefore, the aim of the knowledge construction phase is to build new concepts and relations based on the primitive ones of ModTOVE. Thus, a set of axioms is created to construct the knowledge. The result of the knowledge construction phase is the extended ModTOVE knowledge base. The knowledge extracted from the knowledge base can answer more complex questions related to business process operations [34]. By using the WSMToolkit [33], the questions are answered by querying and reasoning over the axioms of the knowledge base.

1.5 Contributions

Problem 1

In semantic process mining the data elements in the event log are linked to their corresponding concepts in ontologies. The questions involved in the data elements are more abstract and closer to the human thinking. However, the scope of the business-related questions that can be answered is limited to the scope of the information contained in the ontologies which are not sufficient for providing the desired business insight.

Contribution 1: *The first contribution aims at enriching information contained in log data with external data sources. The enrichment is not only concerned with the semantics of data elements but with additional perspectives necessary for answering business-relevant questions. In other words, the scope of potential questions can be extended by integrating information in addition to the log data. The integration of different data sources in the context of PAISs is achieved by utilizing known techniques from ontology-based data integration. In the course of this, an existing ontology is reused and extended for providing the necessary concepts for performing data*

integration, resulting in the ModTOVE ontology. Moreover, ontology-based data integration has also been performed within the context of data warehousing. The ModTOVE ontology developed in this thesis may be applied for data integration in the context of data warehousing as well for integrating heterogeneous data sources in the context of workflow systems.

Problem 2

For raising questions about process models it is necessary to clearly understand the business process operations. In other words, answering business questions can produce valuable knowledge for business process management. However, the ontology resulting from data integration may be advanced by adding axioms manually.

Contribution 2: *The second contribution is extending the knowledge base of the ontology through a set of axioms which are used for reasoning, querying, and thus, answering business-relevant questions. Having the extended ontology at hand allows extracting answers to business-related questions which are beyond the scope of solely applying process mining techniques.*

1.6 Thesis Outline

The remainder of this dissertation is structured as follows:

- **Chapter 2** introduces the problem of dealing with the integration of event log data and related data sources. Therefore, this chapter discusses existing approaches including process mining, semantic process mining as well as ontology-based data integration.
- **Chapter 3** discusses the tools and materials used for conducting the experiment of the research presented. In particular, Adonis is described which is employed for designing and simulating process models. Furthermore, the ProM tool is used for constructing process models for evaluating the generated data. Finally, the TOVE ontology is reused in the approach presented for performing ontology-based data integration.
- **Chapter 4** elaborates on the different data sources and the data generation approach. In particular, event log data is generated based on process models simulations conducted in Adonis. Thus, this chapter describes the implementation of generating event log data. Furthermore, additional resources including organizational data and resource data are introduced in this chapter as well.
- **Chapter 5** presents the results of the ontology-based data integration based on the ModTOVE ontology. The ModTOVE ontology is an extension of the TOVE ontology and serves as a conceptual framework for integrating event log data, organizational data, and resource data.
- **Chapter 6** describes knowledge construction in the context of the ModTOVE ontology. In particular, a set of axioms is constructed based on the concepts and relations which

exist in the ModTOVE ontology. Having these axioms at hand, this chapter further discusses different business-related questions which can be answered by utilizing the axioms defined.

- **Chapter 7** discusses the results found from conducting a case study. Since real-world data is not available, the case study is used for demonstrating the potential of applying the research work in reality. Moreover, the results of the case study are used for comparing the approach presented to other existing approaches.
- **Chapter 8** concludes this thesis with a summary and open issues. The summary draws the results found in the research work presented. Finally, the the limitations of the models developed are discussed which are subject to future research.

State of the art

This chapter introduces the researches related to the present works. First, the generic architecture of a workflow management system is introduced to have an overview of the environment where the data sources concerned in this approach could come from. Second, the analysis of the process operations recorded in the log data is performed by process mining techniques. The overview of process mining and several techniques are represented. Third, semantic process mining improves process mining to a higher abstract level by linking the labels in log data to the corresponding concepts in ontologies. Last, the ontology-based data integration techniques, which are used to deal with the combination of the data sources, are introduced in this chapter.

2.1 Process-aware Information Systems

Although the information system development has not been around for a long time in the human development technology process, it already has had several significant innovations in its history to produce powerful systems. The first trend is the shift from integrating to building. It has changed the conception of information systems. Information systems have no longer been considered as a processing unit providing specific tasks. Instead, they have been considered a set of independent sub-systems. Each sub-system undertakes one or several tasks of the whole system. Therefore, the system is integrated from several smaller systems which have responsibilities for smaller group of tasks. The system integration promotes the re-usability of the existing sub-systems, and simplifies the system construction and maintenance. Because of these advantages, it is indispensable for information system development to change from building to integrating. The second trend is the shifting of the information system development from data orientation to process orientation. It means the core of development has changed from modeling, storing, and retrieving data to processes while focusing on process orientations, i.e., process orientation. In the process orientation, the system operation is driven by process models. This trend triggers a new era of process-aware information systems.

PAIS is defined as "a software that manages and executes operational processes involving people, applications, and/or information sources on the basis of process models" [15]. There are two options to develop a PAIS: (i) develop a specific process support system, or (ii) configure a generic system. Considering the first option, the organization builds the system from scratch. The system is built to fit with the requirements of the organization. This kind of system is not scalable and expensive. The generic system is generally not developed for a specific organization. Generic system means it does not incorporate information about the structure and processes of any particular organization. The organization using the system needs to configure it by specifying processes, applications, organization entities, and so on. These specifications are then executed by the generic system. An example for the generic system is a workflow management system (WfMS). A typical WfMS is composed of a design tool, an execution engine, a worklist management system, an adapter for invoking various types of applications, and , in a few cases, models for monitoring, auditing, and analyzing existing workflow models [15, 2]. The generic structure of WfMS is depicted in Figure 2.1[62].

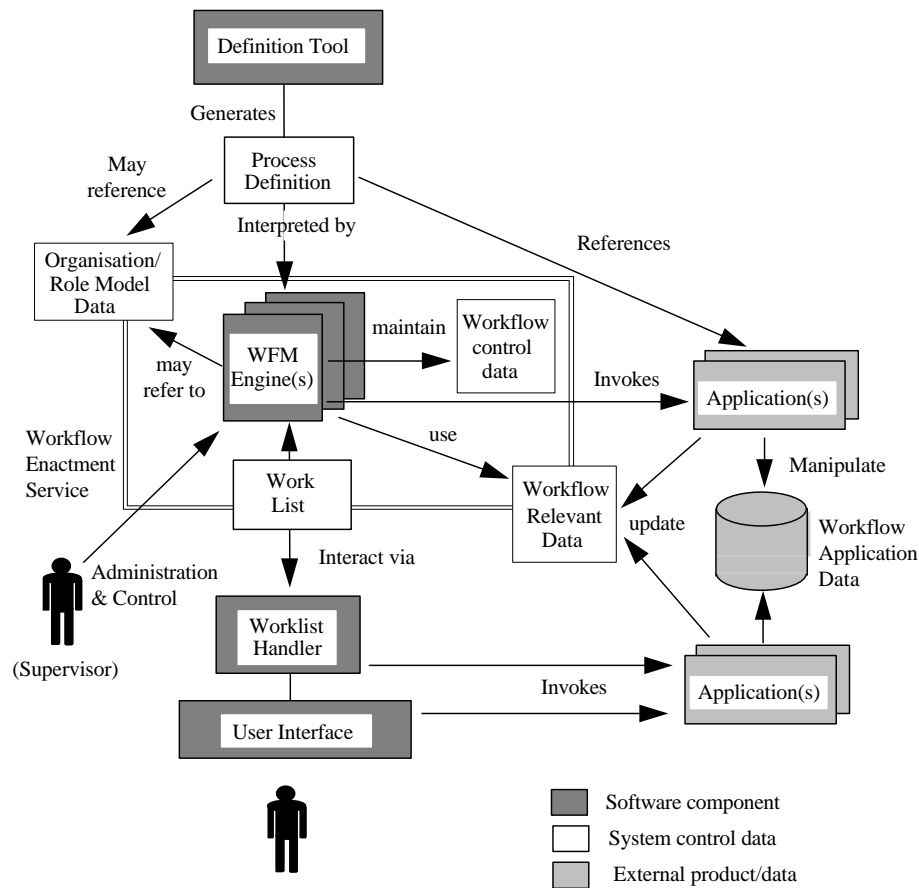


Figure 2.1: Generic Structure of a Workflow System [62]

In this figure, the main functional components of the structure are illustrated in three types of

boxes: the software components are in dark fill, various types of system definitions and control data are shown in unfilled, and applications and application databases in light fill. In general, the center of a PAIS is a *Workflow Management Engine* which interprets the process definition generated by a *Definition Tool* and executes process actions. The process operation is recorded in *Workflow Control Data*. During the execution, the engine needs to invoke *Applications* which can support the task operations. The data produced by the different processes are stored in a *Workflow Relevant Data* database. The applications also have their own data produced for the task operation. However, workflow application data is not accessed by WFM Engines [64]. The human interaction with PAIS can be divided into two types, normal users and supervisors. Normal users of the PAIS, called *Workflow Participant*, access the system via a User Interface and receive task assignments via a *Work List* while *Supervisors* can access the system directly. The following represents the details of the major functional components of the architecture.

Definition Tool The process models driving the PAIS operation are generated by the *Definition Tool* located at the top of the figure. The tool can be a part of a workflow system or a single product for users who are the administrators that have permission to define the process models. In particular, the *Definition Tool* provides a graphic representation to edit, analyze and evaluate workflows. The *Process Definition* generated by the tool contains all necessary information about processes executed by the workflow enactment software. The information may involve starting and completion conditions, constituent activities and rules for navigating between them; user tasks to be undertaken; references to applications which may be invoked; definition of any workflow relevant data which may need to be referenced, and so on [62]. In order to assign the user to actions of processes, the *Process Definition* may refer to an *Organization/ Role Model* which contains information about the organizational structure of a company. Although each workflow editor usually supports one specific workflow modeling language, the process definition is stored in XML format and able to be interpreted by workflow engines [47].

Workflow Enactment Service is defined as a "software service that may consist of one or more workflow engines in order to create, manage and execute workflow instances" [62]. In other words, *Workflow Management Engines* create the run time execution environment to constitute the services for the operation of the system. In particular, the engine creates process instances, and manages their executions such as, start, stop, suspend, etc. It creates work items, matches capabilities (skills, knowledge, and experience) of workflow participants with requirements of tasks, and allocates work items to workflow users. Any change of process instance will execute the corresponding workflow triggers, that in turn call a stored procedure invoking the workflow engine [40]. In fact, it provides facilities to handle the execution of process instances, the identification for work items of users' attention, and the supervisory actions for control, administration and audit purposes. The work engine negotiates the workflow participant and application involved to process models. The control data of these services is maintained in *Workflow control data*. This data contains internal state information about the execution of various processes and activity instances.

Workflow engine invokes an application which has functionality to accomplish specific tasks of the system. Applications existing in the structure are considered as sub-systems integrated in the whole system. The applications have their own data storage, named *Workflow application data*. The applications update the *Workflow relevant data* with the information about the process operation, such as the amount of money of an order within an order process instance. However, the file of the order document created by the application (e.g., document editor) is not accessed by the engine, instead, it is stored in *Workflow application data*.

Worklists are a list of work items that are to be executed [2]. The items on the list are placed by the workflow engine(s) to get attention of the *Worklist Handler*. The *Worklist Handler* manages the interactions between the workflow participants and the items on the worklist. The *Worklist Handler* is a software component responding to progressing work requiring user attention, and interacts with the workflow enactment software via the worklist. The basic functionality of the worklist handler is to present tasks, that may be executed to a workflow participant, and allow them to select tasks, as well as invoke application systems to execute the tasks.

Invoked applications are the external software needed to execute a task. They can be a standardized system for general tasks or a specific software system for a special business function. Invoked applications can be interactive or fully automated applications. The former needs the interaction with users; for example, a user has to fill out a form. The latter can automatically perform a task; for example, the calculation of a bill or the search for data in a database.

Administration and Monitoring is an interface for administrators and persons who manage the execution of workflow instances. With the interface, *Supervisors* of the system can modify the underlying process schema based on certain process parameters.

Involved Data The data produced, exchanged and maintained in the system is the significant object for the current research. As illustrated in Figure 2.1, there are four data sources associated with the system which are:

- *Workflow Control data* contains information about process operating, in run-time, such as the name of the performer who finished the last activity. In the context of this thesis, Workflow Control data implies event log data.
- *Workflow Relevant data* is the information linking between current processes and their involved data in application. For instance, in the fulfill order process model in Figure 1.1, the invoice is managed by the Billing Software system, but the value of the invoice could be Workflow Relevant data.
- *Workflow Application data* is not controlled by the WFM Engine but managed and stored by applications. An example of the data is the file of the invoice, in the above example, that is created and stored in the database of the billing software system.

- *Organization Model* represents the organizational structure of the workflow participants. It contains information about organization entities and their relationships and their properties (e.g., skill, role).

Process-aware information systems, particularly Workflow Management systems, are the environment in which the data for the current research can be derived from. The data produced from the business process operations of the systems contains the interesting information for the experiment of the present PhD work. The assumption made in the present research is the existence of event log data, organization data and resource data. The event log data is the workflow control data. The organization and resource data are the workflow application data maintained in the invoked applications. The functionality of these applications could be the capability to manage the organization and the resource consumption of a company. The data sources are described in detail in Chapter 4.

2.2 Process mining

In section 2.1, the architecture of a PAIS, particularly a generic workflow information system, is described. In general, the operation of the system is executed by a workflow engine and oriented by a predefined process model. However, this kind of information system is not able to control the entire process [15]. In [15], the authors indicate that a WFM system has some degree of freedom, such as, work items are not allocated to a single user but to a group of users, and the routing may be determined by the user or by the arrival of external triggers (e.g., a cancellation by the customer). The flexibility of these systems give the user some form of freedom in the process execution. This raises the interesting question of how people actually work. This question is one of the motivations of process mining which can reveal the knowledge hidden in the log data of the system.

Process mining aims to extract knowledge from event logs generated by PAISs [45]. Based on the information contained in event logs, the knowledge distilled by process mining techniques can be used for supporting process management in terms of process conformance, process monitoring and so on. Thus, process mining is also considered a tool for tracing back the process operations, and bringing out the real process models (i.e., not assumed process models) of the systems. Process mining has become a vivid research area with diverse data sources from the PAISs, which are increasingly used in enterprises. Figure 2.2 shows the overview of process mining.

Process mining techniques can be experimented and implemented in form of plug-ins in ProM [60, 45], a framework for testing the process mining algorithms. The process mining techniques are categorized into three types: discovery, conformance, and extension.

- **Discovery:** The techniques discover the practical process models, the organization context, and the execution properties from event logs [37]. The context for using these tech-

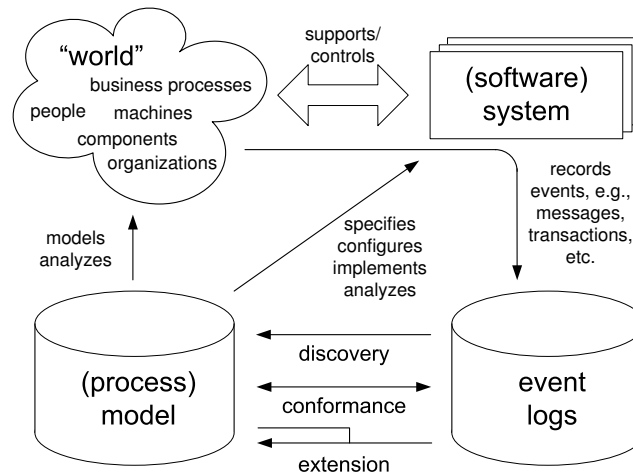


Figure 2.2: Process Mining [37]

niques is that event logs are the only input data for process mining. The constructed process models can show how people and/or procedures really work. The models are the visualization of the real process operations observed in event logs. Therefore, the reality of the results is highly reliable. In this case, process mining can be considered as a tool for process discovery. It could be useful for systems which support processes, but do not enforce a specific way of working, such as the ERP system of SAP [15].

Basically, the principle of the discovery techniques is creating algorithms to automatically produce the models from the event log data. For instance, with event log data, the Alpha algorithm constructs process models in forms of a Petri net [59, 48]. The typical problems of the techniques are the presence of duplicate activities, hidden activities, etc. in the log data [55]. The genetic algorithm described in [8, 10] is an example for the discovery technique which is able to deal with noise and incompleteness in the event log data. Besides process models, organization models are another example of the results obtained from these techniques. In [57], the authors show a social network based on the human information involved in the process executions. Most of case studies used to apply the discovery techniques are in the health care field. In [36, 37, 38], the discovery techniques are applied to figure out the process models of the hospitals in which the process models normally do not visually exist.

The implementation of the discovery techniques is experimented in forms of plug-ins in ProM such as Alpha algorithm, Genetic mining, Multi-phase mining, Social Network miner, the Staff Assignment miner and so on.

- **Conformance:** The techniques are used in case there is an existing process and the log file of its process operation model. The question raised here is "*Do the model and the log conform to each other?*" [51] The techniques take an event log and a process model and compare the observed behavior with the modeled behavior [56, 49, 3]. With the con-

formance capability of process mining, the so-called reference models in the context of SAP can be verified whether the system is operated as shown in the models[15]. The detection of deviations can help enterprises adjust their operation on time. There are two dimensions of conformance: fitness and appropriateness. Fitness is measured to answer the question *"Does the observed process comply with the control flow specified by the process model?"*[51]. Appropriateness is relevant to the question *"Does the model describe the observed process in a suitable way?"*[51].

The implementation of the conformance techniques is experimented and implemented in form of the plug-ins Conformance Checker and LTL Checker in ProM.

- **Extension:** The techniques are used to enrich existing process models with the additional information from event log data. By extracting additional information from an event log and projecting it onto an existing model, the correlations with other process perspectives can be identified. Decision mining is an example for the extension techniques. Based on the additional data, the rules at the decision points of a process model can be discovered [50]. The extension techniques are implemented in ProM with the plug-ins Decision Point Analysis and Conformance Checker with several relevant questions.

The information contained in event log decides the perspectives of process mining can be discovered. For instance, process mining cannot bring out an organization model if event log does not contain any information about the performers. Process mining is classified into three perspectives: process, organization and case [58, 55]:

- If event log supports information about the order of activity performance, a process model which shows all the possible paths can be derived. In this case, process aspect of process mining is discovered
- If the information about the performers involved in the process is contained in event log, the organization aspect can be carried out. The results of this perspective could be an organizational structure in forms of organization model or social network
- If the properties of cases exist in event log, the case perspective can be discovered. For instance, in the fulfill order process, it may be interesting to know the value of the order.

The objective of process mining is providing information of business process operations in a company for managers. Common questions are raised for the information the managers want to know. The questions are answered by process mining by using the plug-ins contained in ProM. The questions are classified by the three perspectives of process mining:

- "How?" questions relate to control-flow perspective
- "Who?" questions relate to the organization perspective

- "What?" questions relate to the case perspective

Related to the present work, process mining is considered a tool to construct process models. The generated event log data is input to the Alpha algorithm plug-in in ProM to construct a process model. This process model is used to evaluate the data generation process. However, the main role of process mining used in the current work is for evaluating the contributions of this research. The criteria for the evaluation is the questions that can be answered by the current work, but cannot be answered by process mining. The answers are analyzed to identify the information sources, from where they come. Through the analysis, the significant contributions of the current work is clarified.

2.3 Semantic Process Mining

Semantic process mining extends process mining. The current process mining techniques are developed to deal with the event log data without any semantics attached in. Because of the characteristics of the input data, the process mining can only handle with the labels, but cannot approach the semantics behind the labels which could cater for more accurate and robust analysis techniques. With this motivation, semantic process mining is proposed with the assumption of the existing linking from labels in event log to ontologies containing their meanings. Semantic process mining techniques are implemented as several plug-ins in ProM.

The semantic process mining approach is built based on three components: ontologies, linkings and a reasoner, as shown in Figure 2.3:

- Ontologies contain a set of shared concepts concerning the meaning of the labels in event logs and process models. Ontologies are defined as an "explicit specification of a conceptualization" [25]. Thus, the ontology in this approach is used to bring out the meaning of the labels in event logs. Questions raised by semantic process mining are more generic, thereby they are understood by users more easily.
- References from elements in logs/models to concepts in ontologies, The references associate concepts defined in the ontologies to labels (i.e., strings) in event logs or models. The references are assumed to be existent in log data and refer to relevant ontologies.
- The reasoner supports reasoning over the ontologies to derive new knowledge, e.g., subsumption, equivalence, etc.

The ontologies and the reasoner used in this approach, thus, raise process mining techniques from the syntactical level to the semantical level.

The data sources for semantic process mining are the event log data, used in process mining, enhanced with semantic elements. For examples, the performer data field in event logs contains

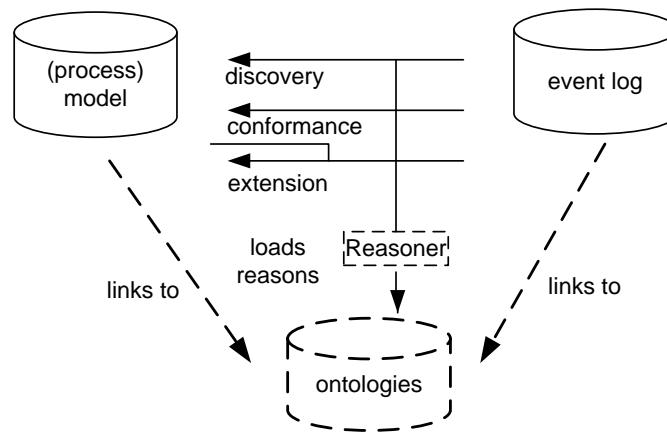


Figure 2.3: Sources of information for semantic process mining. The additional elements (with dashed lines) are necessary to support the mining at the conceptual level

the labels *Richard*, *Paul*, *Laura*, and *Arthur* which are the names of the people who are involved in the process operation recorded in the event logs. In process mining, organization discovery techniques can use these labels to create organization models. In semantic process mining, the labels are linked to the corresponding concepts in ontologies. For instance, *Richard* is linked to the concept *CEO*, and *Laura* and *Arthur* are linked to the concepts *Engineer* and *Network Operational Centre*. With this assumption, the Semantic Organizational Miner plug-in in ProM can create an organization model in an abstract level. Figure 2.4 shows the implementation of the example.

The semantic process mining techniques are developed based on the original process mining techniques. In [13], the following three perspectives of semantic process mining introduced:

- **Discovery:** the techniques basically generate models based on event log. The original techniques mainly discover a model showing all the tasks encountered in the log, thus it could be a large model. With the semantic techniques, the tasks are linked to their corresponding concepts. Therefore, the model can be created in a higher hierarchy level. Besides process models, organization models also apply semantic techniques. Semantic organizational model discovery automatically detects groups and teams in organizations, based on task similarity.
- **Conformance:** the techniques are improved to verify if logs follow prescribed behaviors and/or rules. The problem here is the matching between the labels in event logs and the corresponding elements in the models. Matching in this case is performed at the string level, thus, matching result is not high. The ontologies are applied to define the matching over concepts of event logs and models. An example for such techniques is the semantic LTL conformance checker which allows for the auditing of logs. For example, the performer labels *Laura* and *Arthur* are linked to the concepts *Engineer* and *Network*

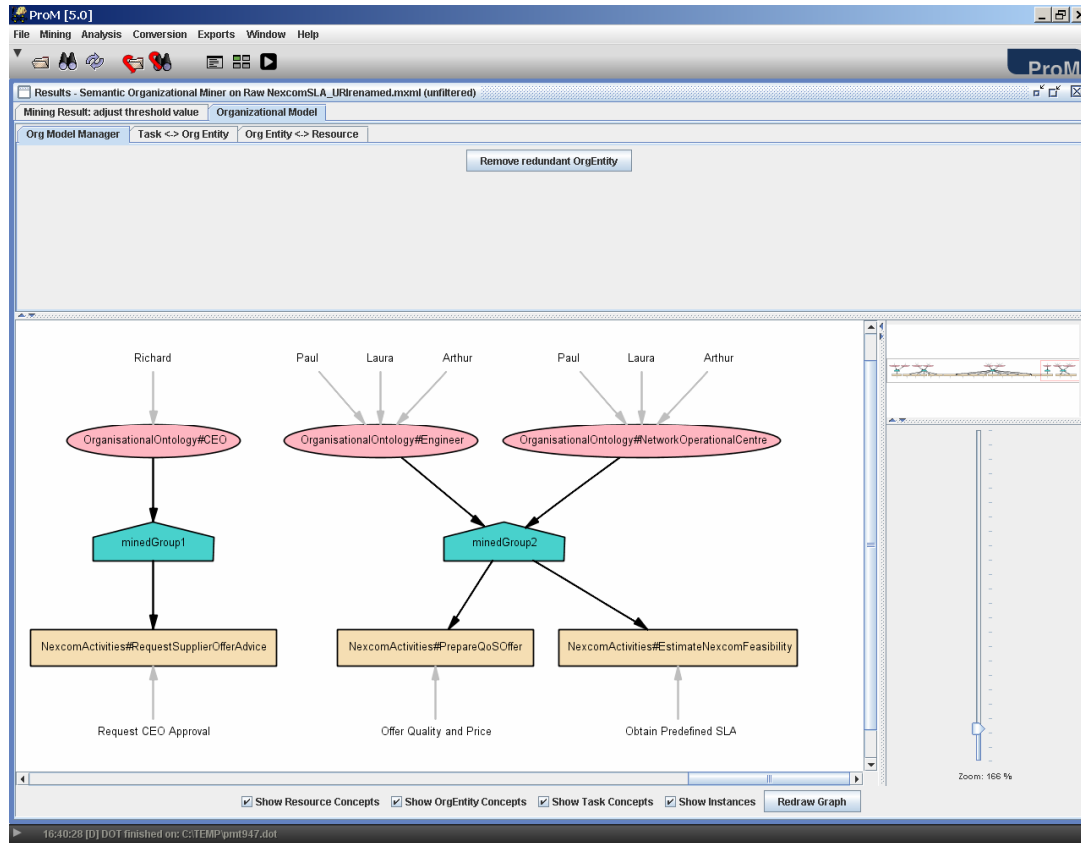


Figure 2.4: Semantic Organizational Miner [12]

Operational Centre. The question, for instance, ” *What activities has Laura from the date X to the date Y done?*”, can be semantically transformed into ” *What activities has a particular Engineer achieved from the date X to the date Y?*”.

- **Extension:** the techniques enhance existing models based on information from event logs. The enhancements also need the matching between elements in logs and models like in conformance techniques. Therefore, the use of ontologies can improve the extension on the semantic perspective.

Semantic process mining is a part of the SUPER (Semantics Utilized for Process management within and between Enterprises) project [46]. This project ”aims at providing a semantic-based and context-aware framework, based on Semantic Web Services technology that acquires, organizes, shares and uses the knowledge embedded in business process within existing IT systems and software, and within employees’ heads, in order to make companies more adaptive” [14]. The objective of SUPER is to raise Business Process Management (BPM) to a business level that is closer to the language of the business experts. The motivation of the project comes

from the urgent issues emerging from BPM when the control of processes is shifting from IT professionals to business natives. It means, business people are becoming more independent of the IT people to operate business process management. In order to bring BPM closer to human thinking, the technical terms need to be tagged with its meaning. In other words, semantic information will be embedded in each of the four phases of the BPM lifecycle, which are design, configuration, execution, and analysis. The implementation of semantic process mining is reported in [12, 11, 4]. Semantic Process Mining techniques are implemented as semantic plug-ins in ProM such as Semantic LTL Checker, Semantic Performance Analysis, Semantic Control-Flow Mining, Semantic Organizational Mining and so on.

Semantic process mining improves process mining techniques from label level to semantic level. The use of ontologies makes the results of process mining more abstract and closer to human thinking. Related to the current work, ontologies are also used to improve the results of process mining. However, the ontologies in the present research are used for the data integration.

2.4 Ontology-based data integration

Ontology-based data integration techniques are created to deal with the information interoperability in [26, 22, 39, 44, 54, 21, 61]. The motivation of the data integration is various applications need to access the heterogeneous and distributed data sources. However, making the different information sources work together with the systems raise the heterogeneity problems in structure and meaning. Structural heterogeneity implies that different information systems store their data in different structures. Semantic heterogeneity refers to the contents of an information item and its intended meaning. Using ontologies for the explication of implicit and hidden knowledge in the data sources is a possible solution for the problem of semantic heterogeneity. Thus, the interoperability is considered a key application of ontologies and many ontology-based approaches to information integration in order to achieve interoperability have been developed [61, 54].

Ontologies, having the role "*explicit specification of a conceptualization*", are used for the data integration in three ways as illustrated in Figure 2.7.

Single Ontology approach

Single Ontology approach is used when all information sources are related to one global ontology. The global ontology provides a shared vocabulary for the specification of the semantics contained in the information sources. SIMS [5] is an prominent example for this kind of integration. By SIMS, the application domain is modeled in forms of objects, actions, and states. The semantics of the objects in information sources are defined by linking to the concepts in the global ontology. The domain of the global ontology needs to cover all the domains of the information sources. The global ontology can be constructed from several single ontologies. The combination is supported by tools, such as ONTOLINGUA [25].

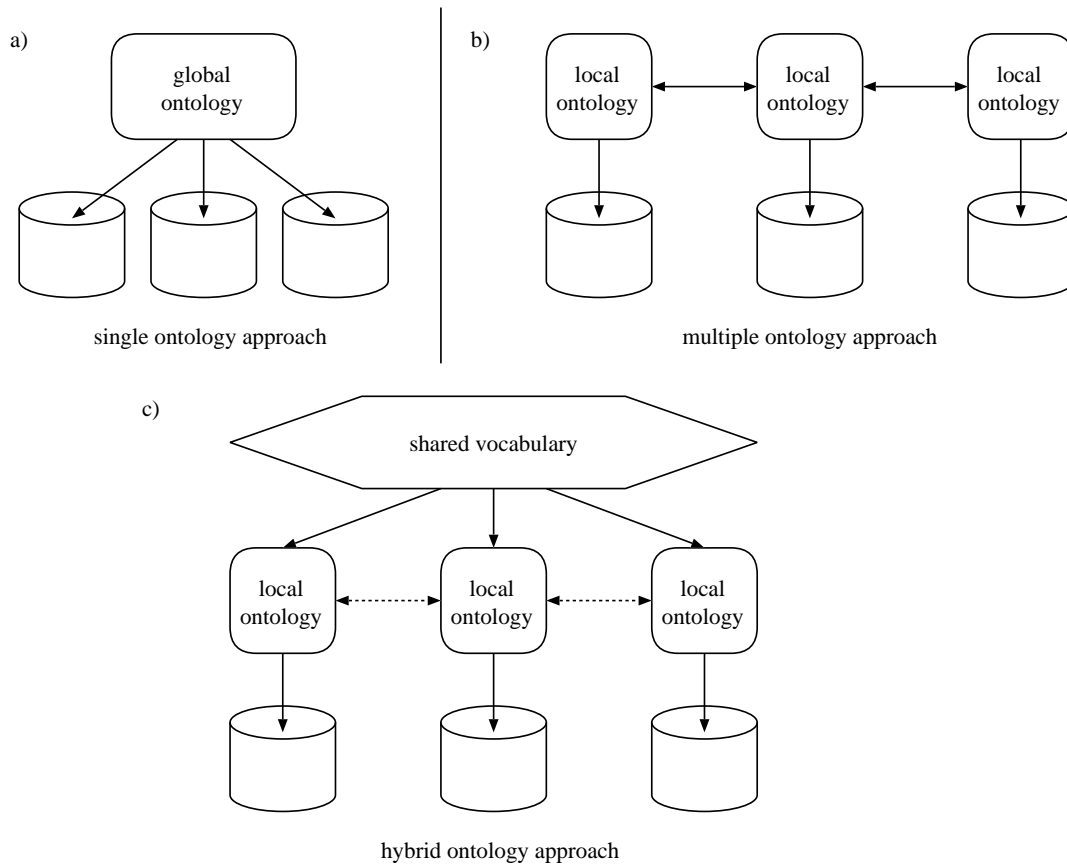


Figure 2.5: Using ontologies for content explication [61]

The single ontology approach can be applied to integrate information sources that have the same view on a domain. However, if one information source has a different domain, the approach cannot be used. The multiple ontology approach can solve this problem.

Multiple Ontology approach

In the multiple ontology approach, the global ontology does not exist. Instead, each information source is described by its own ontology. The local ontologies are properly distinguished and independent of each other, because of the heterogeneity of the information sources. Therefore, changing a local ontology does not effect on the approach. However, in oder to bring out a shared conceptualization, the mapping between local ontologies, known as inter-ontology mapping, is provided and mentioned in [39, 44]. Generally, mapping is performed in several ways, these are defined mapping, lexical relations, top-level grounding and semantic correspondences. Nevertheless, the feasibility of this approach is not high, because of the many semantic heterogeneity problems which may occur in reality [61].

Hybrid Ontology approach

The hybrid ontology approach is the combination of the single and multiple ontology approach to overcome their drawbacks. Each information source is represented in a local ontology by using the shared terms in a global one. The shared vocabulary in the global ontology is built for sharing vocabularies among local ontologies (Figure 2c). This approach is intended to take the advantages of the first two approaches: ease of defining ontologies locally and of querying through a shared vocabulary [21].

The architecture of the ontology based data integration system built by the global ontology approach is introduced in [21] and depicted in Figure 2.6. The goal of the system is to provide a semantic portal for end users at the organization level. The system is composed of data sources, wrappers, a global ontology, a mediator, and other systems.

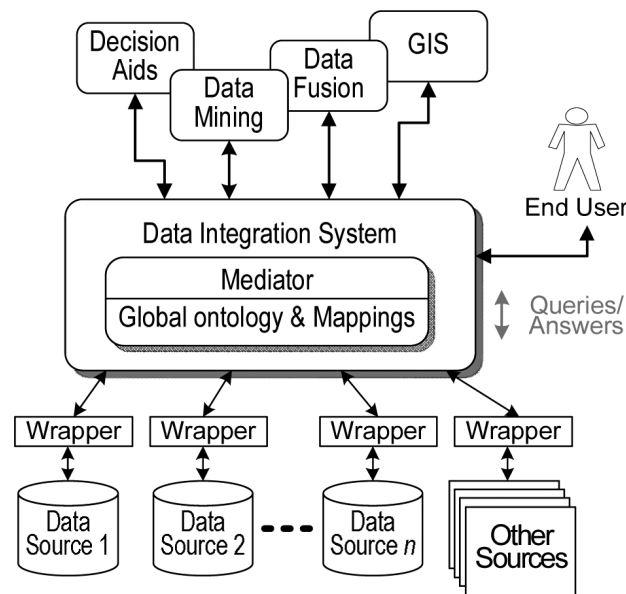


Figure 2.6: An ontology-based data integration architecture [21]

- Data sources contain the different information sources that are integrated in the system
- Wrappers are kinds of mediators which have responsibilities to encapsulate the data sources and create the relations between data sources and the global ontology
- Mediator maps the requests and answers between the global ontology and the local ontologies

A significant issue of the ontology based data integration is mapping. There are two kinds of mapping: ontology mapping and database-to-ontology mapping: Ontology mapping is used

to relate the vocabulary of two ontologies to share the same domain of discourse [32]. There are several methods and tools for mapping ontologies, such as the methods FCA-Merge [52], IF-Map [31], and the tools SMART [42], PROMPT [43] and so on.

Database-to-ontology mapping creates the semantic relations between data sources and ontologies. Several approaches are introduced in [22], such as D2R MAP [7], R2O [6], DB2OWL [22]. The database-to-ontology mapping can be classified into two types:

- Creating an ontology from a relational database model: An ontology is created from a relational database and migrated contents from the the database. Because the data model and the generated ontology are very similar, the mappings here are simply the correspondences between each created ontological component (concept, property, etc.) and its original database component (table, column, etc.).
- Mapping a data base to an existing ontology: the ontology and the database both exist. The goal of the mapping is to create the link between the content of data sources and the concepts of ontologies.

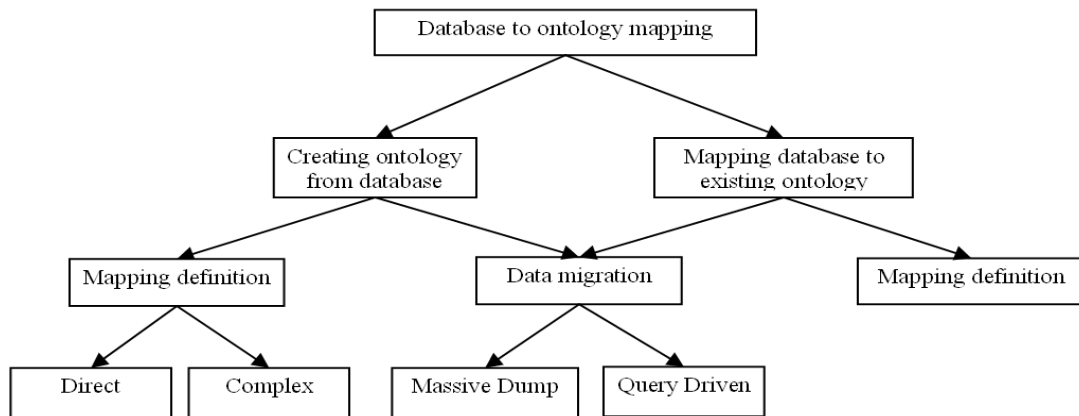


Figure 2.7: Classification of database to ontology mapping approaches[22]

Both mapping approaches contain two processes: mapping definition and data migration as shown in Figure 2.7.

In the current approach, the data-to-ontology mapping is applied to integrate the data sources to ModTOVE ontology. ModTOVE is considered a single ontology covering the domains of the data sources. The mapping of a database to an existing ontology is employed in the current work. Both kinds of data migration are used for transferring instances to ModTOVE depending on the content of the data sources. The detail of the integration is represented in Chapter 4.

Background

The models and tools used in the current approach are introduced in this chapter. First, the TOVE ontology used for ontology-based data integration is represented. Second, the Adonis tool is introduced to have an overview of the data simulation of the present research. Last, the ProM tool used for automatically constructing process models is represented at a glance to show the implementation of process mining.

3.1 TOVE ontology

TOVE project

The TOVE (TOronto Virtual Enterprise) ontology [35] is the main part of the TOVE project executed by a group of researchers of Toronto University. The project is motivated by the increasing demand of enterprises to remain competitive in the rapidly changing market. The desire of every enterprise is to produce products or services with consistently high quality. Thus, enterprises need to be flexible, and quickly respond to the market demands with new techniques, new products, and new business methods. In order to reach expectations, the TOVE ontology is built to formalize the enterprise model by the computational representation. The aims of TOVE is to create an enterprise model which can answer temporal questions of enterprise operations. Specifically, the TOVE ontology represents the organization behaviors which are activity, state, causality and time, and the objects they manipulate: resources, inventory, orders and products [19, 18, 53, 27]. There are four main tasks of the TOVE ontology: *"(1) it provides a shared terminology for the enterprise that every application can jointly understand and use; (2) it defines the meaning (semantics) of each term in a precise and an unambiguous as possible manner using first-order logic; (3) it implements the semantics in a set of PROLOG axioms that enable TOVE to automatically deduce the answer to many commonsense questions about the enterprise; and (4) it defines a symbology for depicting a term, or the concept constructed thereof, in a graphic context"*[20].

The approach

The methodology used to build the TOVE ontology is based on the competency question approach [28, 54]. The approach contains four steps. First, an enterprise's requirements are investigated and transformed into questions that the ontology must be able to answer. These questions are called competency of the ontology. Second, the terminology of the ontology is defined via its objects, attributes, and relations. Third, the definitions and constraints of the terminologies are specified in first-order logic and implemented in Prolog. Last, the competency questions are proved with the Prolog axioms. TOVE is built with the methodology inherited from GEM (Generic Enterprise Model) approach which is considered a solution for the development problems of enterprises. "A GEM is an object library that defines the classes of objects that are generic across a type of enterprise, such as manufacturing or banking, and can be used (that is, instantiated) in defining a specific enterprise [20]." Generally, a GEM contains a set of object classes which relate to each other by a set of relations. However, when a GEM approach is applied in the TOVE project, the meaning of the relations and attributes of the objects are defined by axioms. A GEM is added with the axioms and a deductive engine is called a DEM (Deductive Enterprise Model). In other words, the meaning of the relations between objects in DEM can be inferred by reasoning the axioms.

Competency questions of the TOVE ontology

The domain of the TOVE ontology spans from supply chain management to enterprise engineering. The former is concerned with the enterprise functions in the ordering and receiving of raw materials through the manufacturing of products and the distribution and delivery to the customer [27]. The objects of this area are scheduling, dispatching, resource management, logistics, and transportation. The latter is involved in the design and execution of enterprises, in particular, the formalization of the knowledge required for business process re-engineering. The knowledge found in enterprise engineering perspectives could be related to time-based competition, activity-based costs, quality, agility, and resource management. In order to support the supply chain management and enterprise engineering, the TOVE ontology's competency questions are generated based on the problems raised in these areas. For example, within enterprise engineering, questions such as "What sequence of activities must be completed to achieve a specific goal? At what times must these activities be initiated and terminated?" are represented for planning and scheduling issues.

The TOVE ontology is a set of ontologies which are shown in Figure 3.1. Each sub-ontology represents an object in the domain of discourse, such as an activity, an organization or a resource. Objects are structured into taxonomies and the definitions of objects, attributes and relations are specified in first-order logic. The relations between objects are represented by axioms which constitute a declarative specification for the various tasks of the enterprise model. The following sections represent four sub-ontologies of TOVE (i.e., activity, cost, resource, organization ontology) which are similar to the objects in the domain of the present approach.

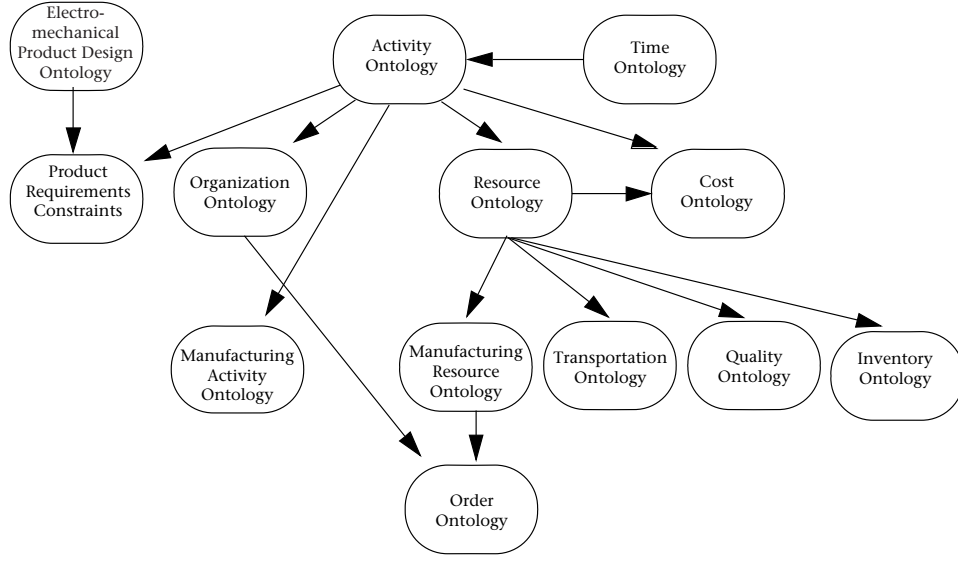


Figure 3.1: The TOVE Ontology [20]

Activity ontology

Activity ontology is described based on the definition of the concepts of time and action. Time in TOVE is considered as a continuous line with time points and time periods. A relation $<$ is defined over time points with the intended interpretation that $t < t'$ iff t is *earlier than* t' . The function $do(a, \sigma)$ indicates the situation that results from performing action a in situation σ . Accordingly, the constant σ_0 is defined as an initial situation of an action. Several predicates are defined to represent the relations between action, situation and time which are denoted by a, s , and t respectively. Situations are assigned to different durations by defining the predicate $start(s, t)$. Each situation begins at time $t = 0$ in situation σ_0 and moves monotonically away from the initial situation.

$$(\forall \sigma)(\exists t) start(\sigma, t) \\ start(\sigma_0, 0)$$

The predicate $holds(f, \sigma)$ is defined to represent the fact that some ground literal f is true in situation σ . Similarly, predicate $holds_T(f, t)$ represents the fact that some ground literal f is true at time t . To represent a notion that actions occur at points in time, the predicate $occurs(a, \sigma)$ and $occurs_T(a, t)$ are defined as follows:

$$occurs(a, \sigma) \equiv \sigma_0 < do(a, \sigma)$$

$$occurs_T(a, t) \equiv occurs(a, \sigma) \wedge start(do(a,), t)$$

Activity ontology is described based on the essential predicates represented above. Activity in the context of TOVE is defined as "an activity cluster containing an activity and its corresponding enabling and caused states" [27]. Enabling states refer to the precondition for the performance of an activity and caused states imply the results obtained when an activity is completed. Figure 3.2 shows an example of a cluster of the activity *fabricate_plug_on_wire* [19]. *es_fabricate_plug_on_wire* and *pro_fabricate_plug_on_wire* are the enabling and caused states of the activity *fabricate_plug_on_wire*, and relates to the activity by the link *enables* and *causes* respectively. Besides, there are conjunct sub-states which are linked to their super-states by the link *conjuncts*. These sub-states show the relationship between the activity and resources which are involved in its performance. In this example, activity *fabricate_plug_on_wire* is performed by using resources, to create a product, a *plug_on_wire*. Therefore, its enabling state concerns the sub-states *consume_wire*, *consume_plug*, and *use_inject_mold*. When the activity is completed, the resources are released and a product is produced which is implied by the sub-states *release_inject_mold* and *produce_plug_on_wire* respectively.

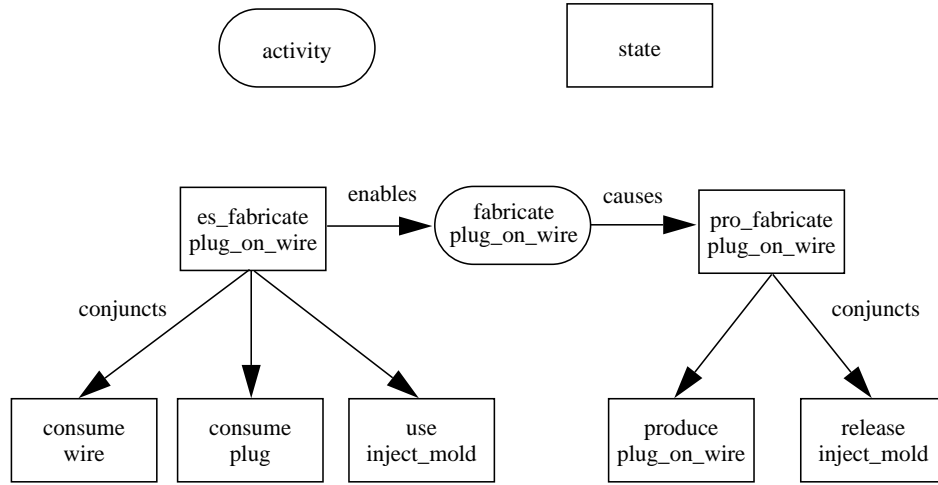


Figure 3.2: An activity cluster [28]

In general, there are four types of relations between activities and resources which are represented by the predicates: *uses(r, a)*, *consumes(r, a)*, *modifies(r, a)*, and *produces(r, a)*. In particular, a resource is used and released by an activity if the properties of the resource are not changed when the activity is completed and the resource is released. A resource is consumed or produced if some properties of the resource are changed after the termination of the activity.

Besides, the predicate $quantity(s, r, q)$ is created to represent the amount q of resource r that is required for a state s to be enabled. In conclusion, an activity in the context of TOVE is defined in the relationships with state and resource represented via the predicates described above.

The status of a state is defined by the following set of constants: *possible*, *committed*, *enabled*, *completed*, *disenabled*, *reenabled*. The status of a state is changed by one of the following actions: $commit(s, a)$, $enable(s, a)$, $complete(s, a)$, $disenable(s, a)$, and $reenable(s, a)$. A set of axioms is built to completely provide the characterization of the state after the performance of any action. These axioms solve the temporal projection problems (determining the value of a status of a state at any point in time). For example, to express the rule "The status of a state is committed in a situation iff either a commit action occurred in the preceding situation, or the state was already committed and an enable action did not occur" [19], its axiom is described as follows:

$$\begin{aligned}
& (\forall s, a, e, \sigma) holds(status(s, a, committed), do(e, \sigma)) \equiv \\
& (e = commit(s, a) \wedge holds(status(s, a, possible), \sigma)) \\
& \vee \neg(e = enable(s, a)) \wedge holds(status(s, a, committed), \sigma)
\end{aligned}$$

More complete specifications about axioms can be found in [19].

Organization Ontology

In the context of TOVE, an organization is considered as a set of constraints on the activities performed by a set of collaborating agents. The organization ontology is structured into taxonomies as shown in Figure 3.3. An organization has several divisions and its goal which is achieved by job functions of the organization. The center of the organization ontology is an *organization-agent* which is an essential human element representing an organization. The relations between organization, its divisions, its goal, and its roles are specified in [19]. In this thesis, only the relations between *organization-agent* and other parts of this ontology (i.e., *division*, *team*, *activity*), which are reused in the present approach, are represented.

An *organization-agent* (denoted by oa) is an individual member, a human being, or a machine agent which performs activities in order to achieve one or more goals of an organization. The properties and relations of organization-agent are implied as follows:

- $member_of(oa, d)$: oa is member of some divisions
- $plays(oa, r)$: oa plays one ore more roles
- $has_communication_link(oa, cl)$: oa communicates with other oas using communication-links

If an *organization-agent* is assigned to a *role* and the *role* has a goal, then the *goal* of the *organization-agent* is defined by the *goal* of the *role* that the *organization-agent* plays. This rule is defined by the axiom:

$$has_goal(oa, g) \equiv (\exists r) plays(oa, r) \wedge has_goal(r, g)$$

The link between the structure of an organization with its behavior is represented by the relations between *organization-agent* and *activity*. In other words, the ontology is constructed to specify "who can do what". The behavior of an organization is represented via the activity and time ontology. This ontology is mentioned in the activity ontology section. Here, we describe the organization ontology in the organization structure perspective. In Figure 3.3, the upper part represents the structure of an organization and the lower implies its behavior. The former represents terms related to the structure of an organization such as role, goal, division, and/or organization-agent. The latter relates to the activity ontology.

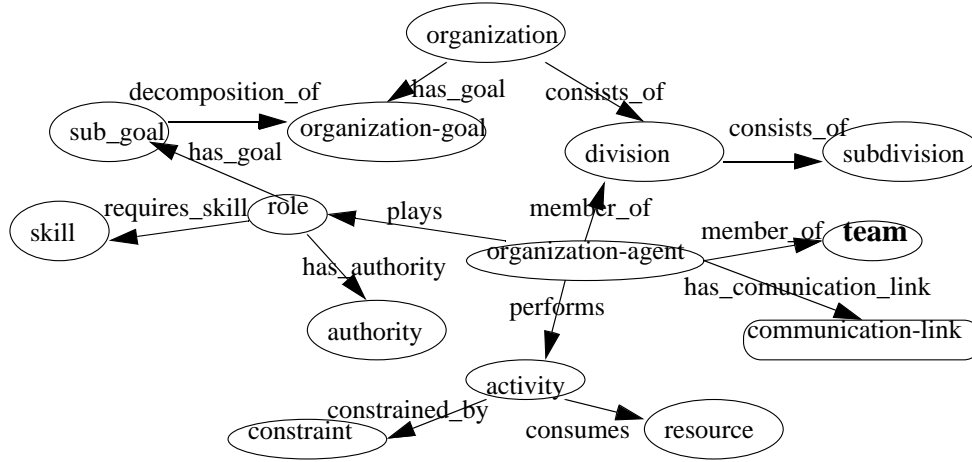


Figure 3.3: Organization Ontology [19]

TOVE supports axioms, which are represented in first-order logic, used to define the terms and answer the competency questions. In [19], the axioms are represented along with the questions they can answer, such as:

Which division does the agent belong to?

member_of(P, ?d)

Which team does the agent belong to?

member_of(P, ?t)

Resource Ontology

Consider the example of an activity cluster shown in Figure 3.2, there are the relations *conjuncts* between enabling states and its sub-states. The *consume_plug* is a state of the resource *plug* which is used for the performance of the activity *plug_on_wire*. A state in TOVE represents what has to be true in the world in order for an activity to be performed, or what is true in the world after an activity is completed. States associate resources with activities through the four types of states which reflect the four ways in which a resource is related to an activity: use, consume, release, and *produce*. The following predicates relate the state with the resource required by the activity:

- *use(s,a)*, *release(s,a)*: a resource is used and released by an activity if none of the properties of a resource are changed when the activity is successfully terminated and the resource is released.
- *consume(s,a)* represents that a resource is to be used up by an activity and will not exist when the activity is completed.
- *produce(s,a)* signifies that a resource, that did not exist prior to the performance of the activity, has been created by the activity.

The resource ontology in TOVE is created for a manufacturing enterprise. The resource in this context is the materials to create products and considering amounts, capacity and availability. Therefore, the competency questions for this ontology is about the ability of consuming resource for the manufacturing enterprise. The questions are classified into several categories, such as divisibility, quality, location, consumption, etc. More specifications of the resource ontology can be found in [18].

Cost Ontology

Cost is defined as "*an entity which represents the temporal fiscal or monetary dimension, attribute, or characteristic of an enterprise activity, and may be referred to as activity cost* [53]. The cost ontology is built based on the Activity Based Costing (ABC) methodology. The ABC methodology identifies the relationship between an activity and the resources needed to conduct it by assigning costs to each of those resources, thus presenting the total expense of the entire activity. The cost ontology uses the terminologies and semantics of the ontologies activity, state, resource and time. Activity is the center object of the ontology, since most of the questions that can be answered by the ontology are related activity.

In TOVE, costs are calculated when the statii of an activity are changed and resources are consumed. In other words, costing happens when an activity is activated with the consumption of resources. The cost ontology is defined by a set of axioms which are described concretely in [53]. In this thesis, the axiom *resource_cost_point*, that is remarkable for the current approach, is represented.

The resource requirement of an activity is identified by the predicate *resource_cost_point*. The *resource_cost_point* predicate, *cpr*, specifies the cost_value, *c*, (monetary units) of a resource, *r*, required by an activity, *a*, up to a certain time point, *t*. The predicate is defined by the axiom:

$$\begin{aligned} & (\forall a, s, r, q, t_s, t_e), (use_spec(r, a, t_s, t_e, q) \wedge enabled(s, a, t)) \\ & \vee (consume_spec(r, a, t_s, t_e, q) \wedge enabled(s, a, t)) \\ & \equiv \exists c, cpr(a, c, t, r) \end{aligned}$$

If a resource of the terminal use or consume states, *s*, for an activity, *a*, is enabled at time point, *t*, there must exist a cost_value, *c*, at time point, *t*, for the activity, *a*, that uses or consumes the resource, *r*. The time interval, *ti* = [*ts*, *te*], during which a resource is used or consumed by an activity is specified in the use or consume specifications as *use_spec(r, a, ts, te, q)* or *consume_spec(r, a, ts, te, q)* where activity, *a*, uses or consumes quantity, *q*, of resource, *r*, during the time interval [*ts*, *te*]. The definition and example are quoted from [53].

An example for the predicate is *cpr(assemble_back, 120, 75, ear)*. It means resource_cost_point for the activity, *assemble_back*, is of cost_value 120 monetary units at time point 75, for the resource, *ear*.

Reuse TOVE in the current approach

The domain of the TOVE ontology is similar with the domain of the current approach. In particular, the supply chain is similar with the business process, when both of them are defined as a set of activities executed to achieve a particular goal of an organization. Moreover, the relation between event log and data source (organization and resource data), is also involved in the enterprise engineering which has knowledge spanning in time-based competition, activity-based costing, and resource management. Thus, the TOVE ontology is reused partly in the present approach.

Several concepts of TOVE are reused to create an derivation of TOVE, called ModTOVE in the current work. ModTOVE contains the concepts *activity*, *organization_agent*, *resource*, and *cost*, which are inspired by the TOVE's original concepts *activity*, *organization*, *resource*, and *cost*. The ModTOVE's concepts are defined to be suitable within the context of the research. The details of ModTOVE are represented in Chapter 5.

3.2 Adonis

The goal of the Adonis tool

ADONIS is a key part of The BOC Management Office [23] which is a family of products for the integrated management of strategy, business processes, people, IT and performance. ADO-

NIS supports the core functionalities for business process management, including information acquisition, modeling and design, analysis, simulation, and evaluation. ADONIS also provides various import/export facilities, Web and standard publishing capabilities, and administration tools. ADONIS supports for a large range of users, spanning from non-technical, such as business analysts, process owners, and process managers, to professional users, such as technically skilled information systems and enterprise architects, who are interested in business processes and business process-related information.

The architecture of Adonis

Figure [29] shows the ADONIS architecture, which consists of three main levels: the repository, the application components, and the user interface. The Modeling and Simulation parts of the ADONIS Modeling Toolkit are used in the present research. In particular, the process models experimented with this thesis are created by the Modeling part, and the simulations of the process operation are executed by the Simulation part.

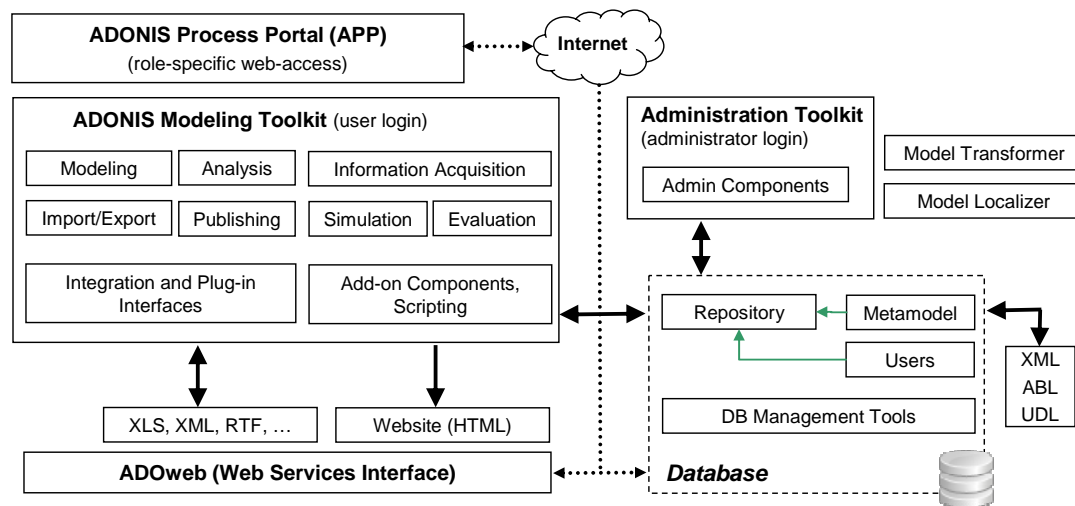


Figure 3.4: Overview of Adonis Architecture [29]

The ADONIS architecture supports XML import/export, scripting, and plug-ins, allowing integration into existing customer specific infrastructures. The model information (XML) can be imported and exported between different ADONIS installations or to other environments. This functionality of ADONIS is applied to collect the log files exported from ADONIS for the data generation of the current research.

Business Process Management toolkit

The Adonis business process management toolkit is the main component of Adonis. Business processes as well as the working environment (i.e., organizational structures) could be designed by using the toolkit. Moreover, the extensive functionality of the toolkit provides users functions

to acquire, analyze, simulate and evaluate the business processes and working environments in a cost effective way. In this thesis, several functions that are used in the current approach are introduced.

- **Notebook:** Each modeling object or connector has a specific set of attributes for capturing information. In the model editor, these attributes are shown via a property dialog box called "Notebook" in Adonis, as shown in Figure 4.5. The fields contained in Notebook are not only simple text fields but also contain many strongly typed attributes such as text, number, date, time, enumeration (simple/multiple selection), record, inter-reference, expression (user-defined formulae), and program calls. The variety of the attribute types provides considerable capabilities for structuring and formatting model information. Thus, Notebook provides users with a powerful and intuitive way to manage processes and process-related information.
- **Graphical Notations:** ADONIS supports a range of pre-defined graphical notations, importable as libraries or model types, including ADONIS (proprietary), BPMN, UML 2.0, LOVEM, EPC, and OSSAD. Figure 3.5 shows the notations that are used for designing process models in the current approach.
- **Simulation:** The ADONIS simulation component is directly integrated within the tool and provides discrete, event-based simulation. A simulation library is included that provides four simulation algorithms as well as animation and playback capabilities. An example of this function is Process Stepper which is applied in the present research. Basically, the Process stepper function of Adonis permits users to operate the process operations by checking each node in the process model. All possible process paths of process models can be examined. The simulation is recorded and exported in XML files.

3.3 ProM

The achitecture of ProM

ProM is an environment for experimenting with process mining techniques. ProM is open source and implemented in Java. Currently, there are more than 170 plug-ins available. The plug-ins are classified into five groups as shown in Figure 3.6.

- **Mining plug-ins:** implement mining algorithms, for instance, control-flow mining techniques, such as the Alpha algorithm or Genetic mining, which construct process models in form of Petri net based on event logs.
- **Export plug-ins:** implement the "save as" functionality for some objects (such as graphs, filtered logs). For example, the plug-in Log Filter is used to save the filtered logs, or the plug-in Petri Net Kernel file for exporting mined models.

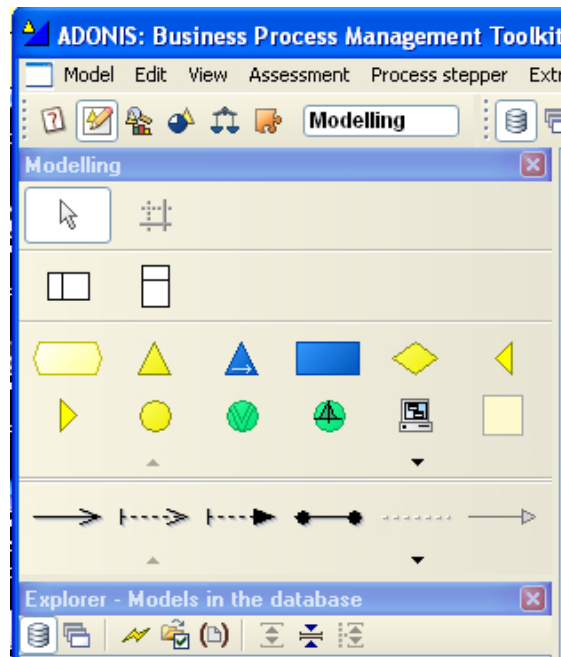


Figure 3.5: The graphical notations in Adonis [29]

- **Import plug-ins:** implement an "open" functionality for exported objects.
- **Analysis plug-ins:** implement some property analysis on some mining result. For example, one wants to know the most frequent path for a process model. The analysis plug-in Performance Sequence Diagram Analysis should be used for this case.
- **Conversion plug-ins** implement conversions between different data formats, e.g., from EPCs to Petri nets.

In the perspective of answering questions, ProM can be divided into:

- **Discovery plug-ins** answer questions like *"How are the cases actually being executed? Are the rules indeed being obeyed?"*
- **Conformance plug-ins** answer questions like *"How compliant are the cases (i.e. process instances) with the deployed process models? Where are the problems? How frequent is the (non-)compliance?"*
- **Extension plug-ins** answer questions like *"What are the business rules in the process model?"*.

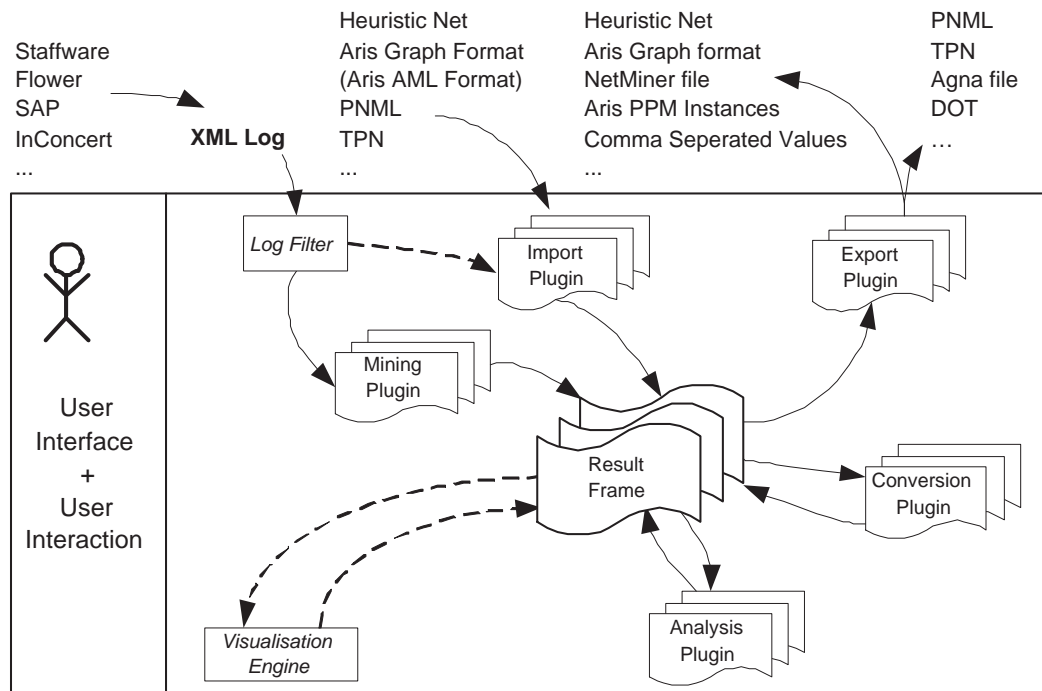


Figure 3.6: The overview of ProM architecture [60]

Regarding to the use of ProM in the current research, the plug-in Alpha algorithm is used to construct the generated event logs. The model produced by a plug-in is compared with the designed one from Adonis to evaluate the data generation.

Data sources and Data Generation

This chapter will introduce how the data sources scattered in business systems can be integrated and what can be obtained from the integration. The data sources mentioned in this chapter include event log data, organization data and resource data. The event log data is obtained by a generating and simulating method. The organization data and resource data are assumed to be existent and suitable for enriching the event log data.

4.1 Event log data

As mentioned in Chapter 1, the operation of a WfMS is driven by a predefined process model. In other words, the process model describes the order of activity operations within the system. To accomplish a particular task, a number of activities are executed following the paths depicted in the process models. The operations are recored and exported into log files. Depending on the system exporting the log files, not only various kinds of information are maintained in log files, the files are stored in different formats. For example, ERP systems from SAP, log all transactions, e.g., users filling out forms, changing documents and so on. Business-to business (B2B) systems log the exchange of messages with other parties. Call center packages as well as general-purpose systems log interactions with customers [58]. No matter how the diversity of the information contained in the log files could be, most event log data typically refers to the activities operated in the systems [58].

Consider an example of an event log shown in Table 4.1 which illustrates the basic information contained in an event log. Each row of the table implies an event happening during the process operation. In other words, when an action is started, the system will record the event as a row in the table. The first column of the table is case id which identifies a case of a process model. The case is the object which is being handled, e.g., a customer order, a job application, an insurance claim, etc. Every case contains several activities which are performed in the order

Table 4.1: Example of an event log

case id	activity id	originator	timestamp	event type
case 1	activity A	Mark	17-05-2008:16:09	start
case 2	activity B	Chris	18-05-2008:09:12	start
case 1	activity C	Tom	18-05-2008:10:06	complete
case 3	activity B	Mary	18-05-2008:15:02	start
...

depicted in the process model. For instance, *case 1* includes a number of activities, such as *activity A*, *activity B*. The second column is the activity identifier, also named task, operation, action, or work-item. An activity is performed by performers whose names are held in the originator column. At a point of time when the activity is performed, it has a status shown in the event type column. For example, in the first row, for a particular case identified by *case 1*, *activity A* is performed by *Mark* at timestamp *17-05-2006, 16:09* with the type of event *start*. In the same vein, the next rows contain information of other events that happened in the process operation. All the events are ordered by their respective timestamps. In short, event log data refers to the information about activity, performer, time, event type of process instances. Moreover, other optional information can be recored in event log, such as information about products, resources, employees which are involved in the case. The other information can be expanded in many perspectives depending on the system recording and producing them.

Event log data is stored in various formats depending on the PAIS producing them. In the current research, event log is considered in MXML which is a standard format for process mining [60]. In [9], the event log data is generated from relational databases which contain related event information. The XES format standard is defined to reduce the restriction of the old format MXML. Although the XES format is an improvement of data input for the process mining, the current research still uses the old MXML format which contains sufficient properties for our approach. The MXML data schema is shown in Figure 4.1. In this figure, a *Process* has from none to many cases or *Process Instances*. A process instance contains from none to many events implied by *AuditTrailEntry*. An event has several properties, such as originator, event type and originator which are contained in *EventType*, *Originator*, *Timestamp*. The *WorkflowModelElement* holds the name of the activity which triggers the event. The *Data* is used to contain the additional information of the event. With the data schema, Figure 4.2 shows an encoded part of the MXML file.

The snippet of the MXML file displays the information of process instance *Case 2*. The Case 2 has two events held in two *AuditTrailEntry*s. The first event is triggered by activity *Check Order*, performed by originator *Tom* at time point *17-05-2008:16:09* in the status *complete*. The second *AuditTrailEntry* contains information of the last event of the process instance *Case 2*. With the specification of the data schema and information containing in MXML file, an event log data is simulated and produced by the data generation represented in the following section.

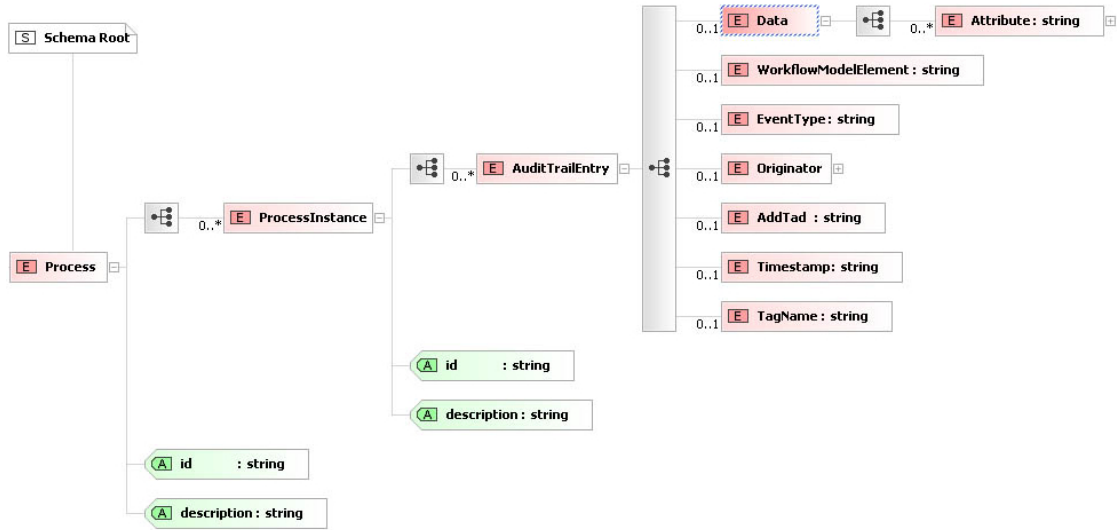


Figure 4.1: Data schema of MXML

4.2 Event log data generation

Process model simulation

The factual event log data exported from the real systems employed in companies always contains noise. Moreover, they are maintained in various formats depending on the exporting systems. Most of them are not ready to use in process mining. Filtering and transforming the data into the standard MXML format is the very beginning step of the whole mining process. Within this thesis, we do not concentrate on converting the real data into the ready-used data structure. To have a well-formed event log data for my experiment, we build the data generation program to create event log data based on the process model simulation function in Adonis.

The data generation is started by a process model designed in Adonis. In particular, the business process modeling functionality of Adonis is used to design and simulate a process model as shown in Figure 4.3. The process model describes an order process, that starts from the yellow triangle node *Process start*, and ends with the yellow round node. The activities of the process model are denoted by blue rectangles with the names of the activities below. In this example, the model has two branches separated by a *Decision* node. In the upper branch, the Assemble product denoted by a blue triangle node with an arrow is a group of activities which are expanded in Figure 4.4. The activity Sent Invoice and Deliver Products can be performed simultaneously and represented by the triangle parallel and merge nodes.

Not only supporting the creation of the skeleton of a process model, Adonis also allows users to manage the properties of each node of the model by the Notebook shown in Figure 4.5. This figure depicts the properties of the activity *Check Order*. The name of the activity is contained

```

    </AuditTrailEntry>
  </ProcessInstance>
  - <ProcessInstance description="" extraInfo="Ex2" id="Case2">
    - <AuditTrailEntry>
      - <Data>
        <Attribute name="Amount">100</Attribute>
        <Attribute name="OriginatorID">or0</Attribute>
      </Data>
      <WorkflowModelElement>Check Order</WorkflowModelElement>
      <EventType>complete</EventType>
      <Originator>Tom </Originator>
      <Timestamp>17-05-2008:16:09</Timestamp>
    </AuditTrailEntry>
    - <AuditTrailEntry>
      - <Data>
        <Attribute name="Amount">100</Attribute>
        <Attribute name="OriginatorID">or1</Attribute>
      </Data>
      <WorkflowModelElement>Notify Out of Stock</WorkflowModelElement>
      <EventType>complete</EventType>
      <Originator>Chris</Originator>
      <Timestamp>17-05-2008:16:15</Timestamp>
    </AuditTrailEntry>
  </ProcessInstance>
  - <ProcessInstance description="" extraInfo="Ex3" id="Case3">
    - <AuditTrailEntry>

```

Figure 4.2: A snippet of MXML file

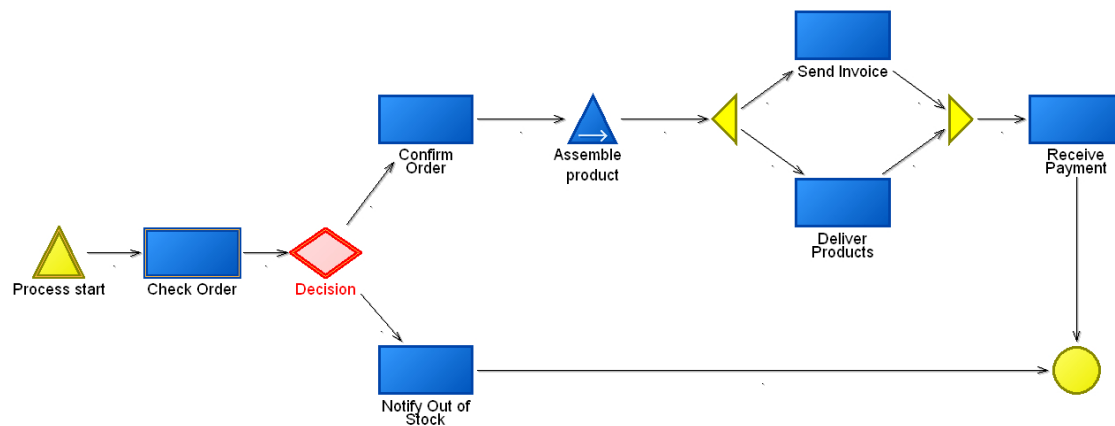


Figure 4.3: A process model designed in Adonis

in the *Name* textbox, people who can execute the activity are listed on the *Performer* box but the person who has done the activity at a particular point time is shown in the *Done by* box. The

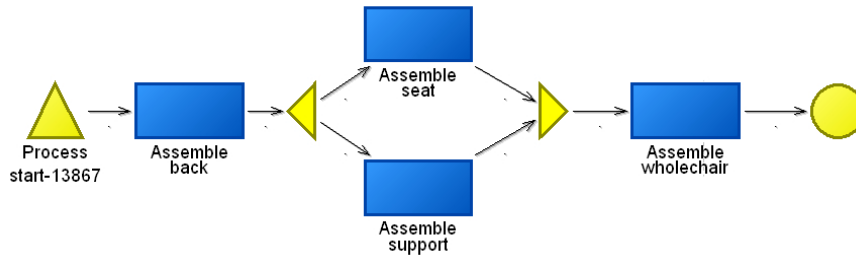


Figure 4.4: The sub process of the activity *Assemble product*

abundance of properties facilitates extending the process model with additional information. Moreover, if a character of the node is not supported, it can be flexibly replaced by the existing one. For example, in Figure 4.5 the point of time when the activity is executed is not supported but can be added in the *Comment* box. Therefore, the additional information about the nodes in the model can be added or edited manually during the simulation process supported by the Process Stepper. With the Notebook (see Chapter 3), the information of the activity node can be added freely by users. For example, in the case of this research, the originator of an activity is stored in the *Done by* field.

Figure 4.5: The *Notebook* for editing properties of a process model in Adonis

To simulate the process operation, the Process Stepper function is used to execute the process model. The execution is understood as the checking of every node of a process path within the process model. A process path is started at the Process start node and ended at the End node. Consider the process model in the checking situation at the decision node in Figure 4.3: there are two process paths which both are started at the activity node *Check Order* and separated at the *Decision* node. The first path contains *Check Order*, *Confirm Order*, *Assemble product*,

Send Invoice, Deliver Products, and Receive Payment. The other path includes *Check Order* and *Notify Out of Stock*. At the decision node, Adonis provides the interface for users to decide manually which branch will be executed. When a process path is executed completely to the end node, a log file is exported to record the simulating process.

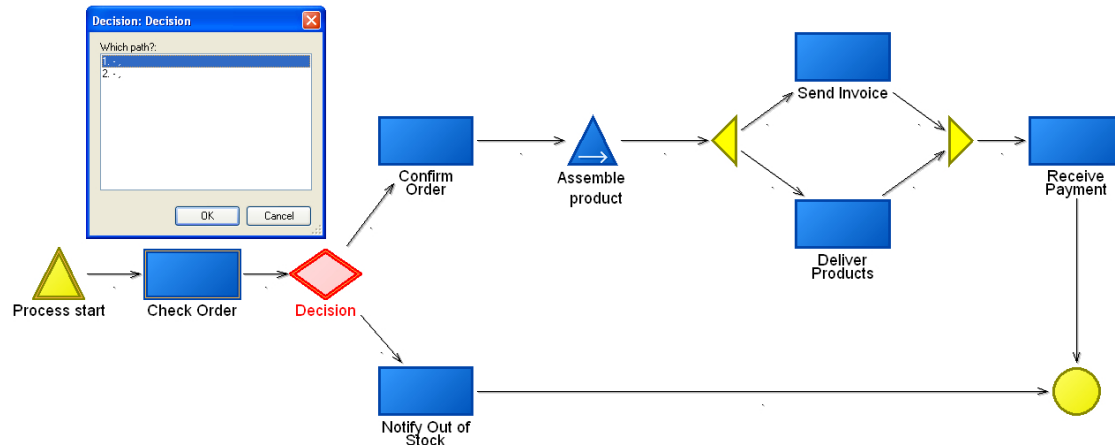


Figure 4.6: Decision Node in a Process simulation

Figure 4.7 shows a snippet of an Adonis log file with a process path. The Adonis log file is actually an encoded version of the process model in XML format and the process path is a list of identifier numbers of activities executed by the Process Stepper. The process model is a set of nodes and connectors between nodes. Therefore, the log files contain the properties of the nodes, connectors, and one process path. In theory, a huge amount of information can be attached to the process model by using the Notebook of its nodes. However, in this thesis, we only consider the significant properties of an activity which are similar to the data elements in the event log data structure, such as name of the activity, performer, timestamp, name of the case etc. A number of log files are collected when simulating the process model manually.

```

    <ATTRIBUTE name="Denomination" type="STRING"/>
    <ATTRIBUTE name="Description" type="LONGSTRING"/>
  </CONNECTOR>
</MODEL>
</MODELS>
</ADOXML>
- <!--
  ADONIS Process stepper path="13510 13706|13512 13620 13614
-->

```

Figure 4.7: A process stepper path in an Adonis log file

In order to generate event log data from the bunch of log files exported from the simulation, the Adonis log files are analyzed to get the data of the process path. Each log file contains the information of the nodes involved in the process path. Figure 4.8 shows the data schema of Adonis log files. Besides the general information contained in the tag *MODELATTRIBUTES*, the significant information is held in the tags *INSTANCE* and *CONNECTOR*. Each node is represented by one tag *INSTANCE*. Each node has one or more properties held in tag *ATTRIBUTE* which can be managed by its Notebook, such as name of performer, timestamp and so on. The tag *CONNECTOR* specifies the links between two nodes in the model. The direction of the link is defined by the tags *FROM* and *TO*. Based on data held in the tag *INSTANCE* and *CONNECTOR*, the process path is described with the information of the involved nodes and their properties.

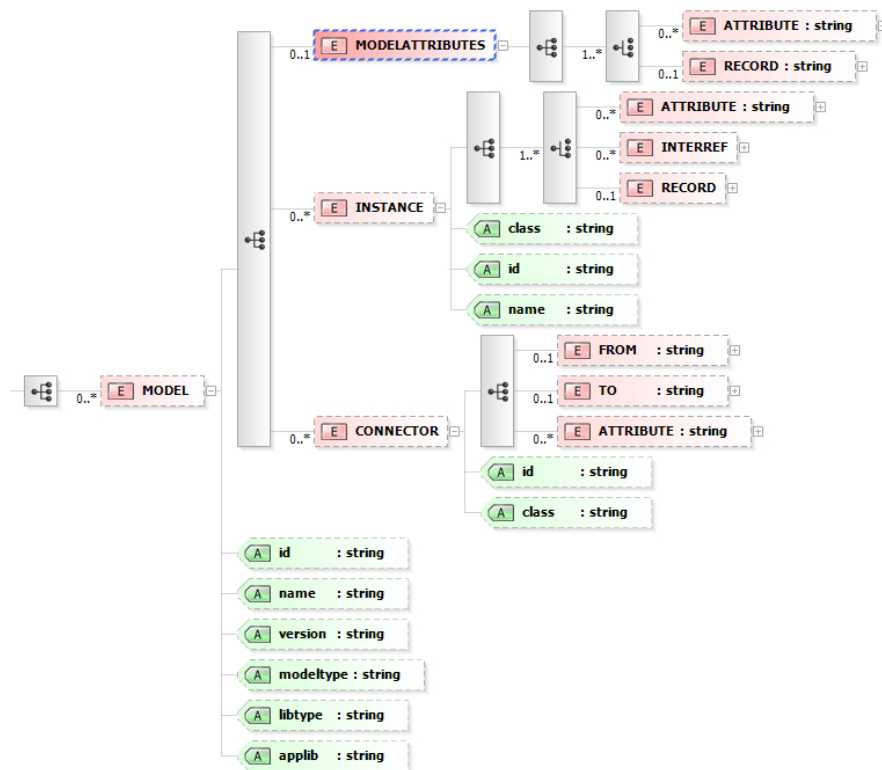


Figure 4.8: The data structure of log files created in Adonis

Consider the process path shown in Figure 4.7. It is a series of identifier numbers of the nodes constructing a process path. Based on the identifier numbers, the information of the node is obtained from the model description part of the log file. The order of the numbers is the order of the execution of the nodes in a process path. Note that a process instance or a case in event log data is a series of events. Therefore, the information contained in one log file is extracted to create a process instance in the event log, and the number of process simulations is the number

of process instances in the event log data.

The analysis above is used to build a prototype for generating event log data. The prototype uses a bunch of Adonis log files as its input data to produce event log data as its output. The techniques for reading and creating XML files are used in the prototype. The function of the prototype is transforming Adonis log files into an event log file. The implementation of the data generation is represented in the next section.

Implementation of data generation

The data generation is constructed to provide the event log data for the experiment of this research. The architecture of the implementation is depicted in Figure 4.9.

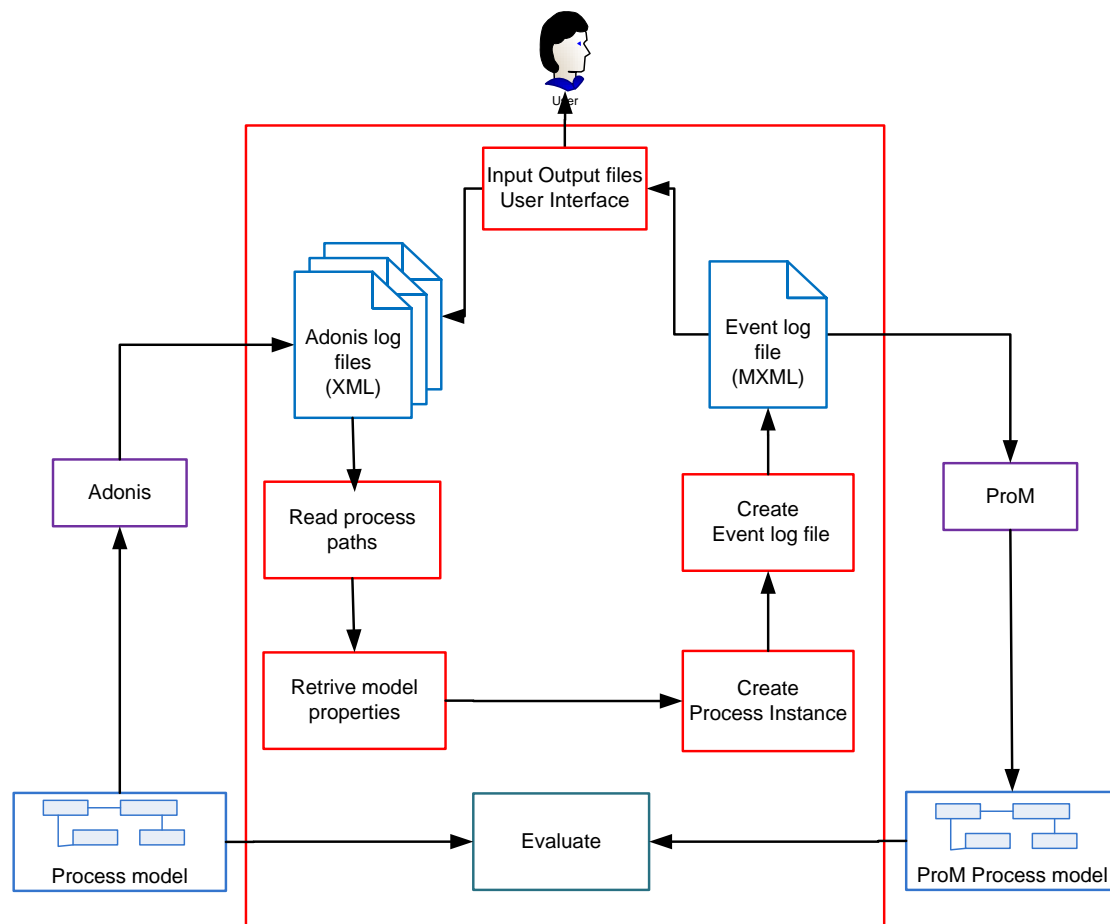


Figure 4.9: Data Generation API

The generation process contains 4 steps:

Step 1 Adonis is used to design a process model. The designed process model is simulated by the Process stepper in Adonis to get the simulated log file.

Step 2 The log files in Step 1 are used as input of the Generate Event log interface. The process path and the properties of the nodes in the model are retrieved to provide data for Step 3

Step 3 Process instances are built to create the event log. An event log file is the result of this step.

Step 4 The event log generated from Step 3 is input to ProM to create process model by the alpha algorithm plug in. The model is evaluated by comparing visually with the model designed in Adonis.

The concrete processing of the data generation is depicted in the algorithm diagram shown in Figure 4.10.

- After the process model is designed and simulated in Adonis, a bunch of log files are generated and used for the event log data generation. The Adonis log files are used as input to an interface called Generate event log data. A user needs to identify the location of the directory containing the set of Adonis log files. He/she also gives the name of the new event log file and the place where the generated file is stored. This interface is built based on the java.awt.* and java.swing.* libraries supporting the construction of graphical interfaces.
- The Adonis log files are accessed by using the packages javax.xml.*, org.w3c.dom.* and org.xml.sax.*. Let n be a number of log files. A loop executed n times, called loop N , is created to access every log file.
- In the log file number i , the process path of the log file is considered. Note that a process path is a series of identifiers of the nodes which are checked in the simulation. Let m be the number of nodes in the process path. A loop executed m times, called loop M , is created to access every node of the process path
- At node k , the properties of the node are retrieved from the process model description part of the log file. In particular, the values of the properties are contained in the tag *ATTRIBUTE* of *INSTANCE* as shown in Figure 4.8. The data of performer, timestamp, event type are retrieved.
- The data retrieved is collected and transformed into an AuditTrailEntry. It means an AuditTrailEntry in XML format is built as shown in Figure 4.2.
- When the loop M is accomplished, we have m AuditTrailEntry created. This is the number of activities performed in a simulation, in other words, a process instance. These AuditTrailEntry paragraphs are arranged in the order identified in the process path to create one process instance.

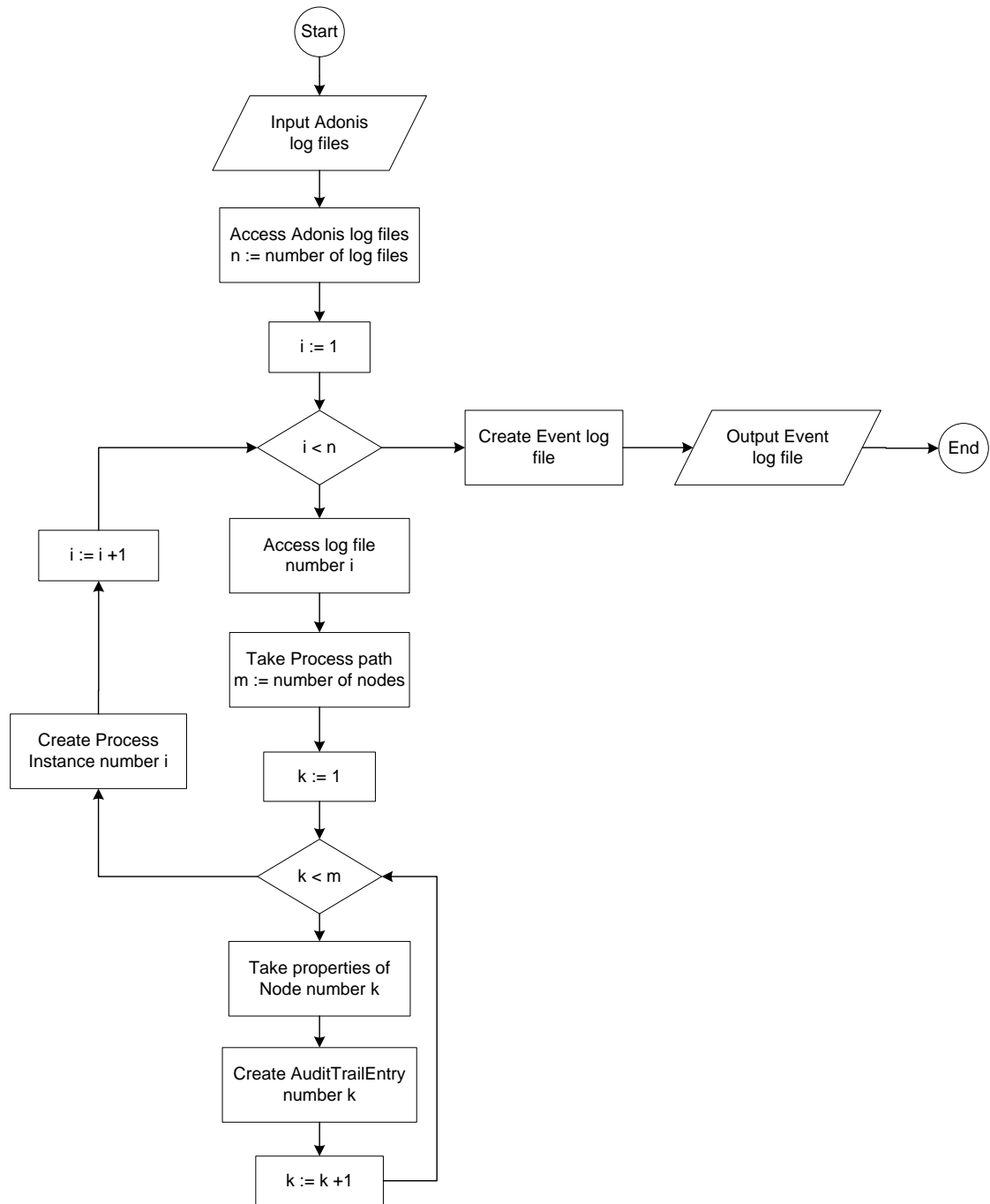


Figure 4.10: Algorithm of the data generation

- The process is continued until all n Adonis log files are accessed. At the end, the process instances are stored in one event log file. The number of log files contained in a directory

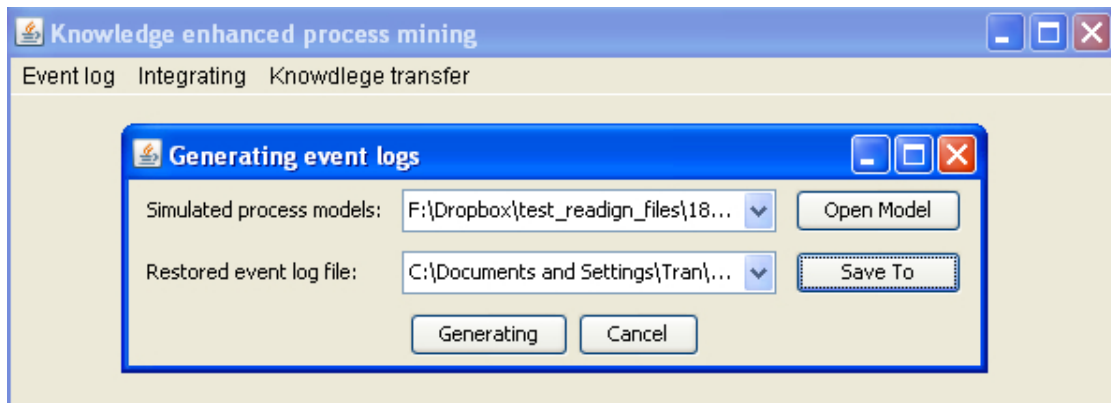


Figure 4.11: Generate Event log data

is the number of process instances contained in the event log file. The location of the event log file is identified by users via the interface in Figure 4.11. The data generation is accomplished.

Evaluation

In order to evaluate the data generation, we use ProM as a tool to create a process model from the generated event log. In particular, the alpha algorithm plug-in in ProM is used to construct a process model. The model created by the plug-in is shown in Figure 4.3. Visually, the model in Figure 4.12 and 4.3 have the same structure and meaning. Consider the model in Figure 4.12, the activities are represented in the rectangles with their names and their *complete* status. The activities *Assemble support* and *Assemble seat*, *Deliver Products* and *Send invoice* can be performed concurrently. In other words, they are parallel operating activities. In the model, there are two paths: one contains *Check Oder*, *Notify Out of Stock*, *Recieve Payment*, and the another contains *Check Order*, *Confirm Order*, *Assemble back*, *Assemble support*, *Assemble seat*, *Assemble wholechair*, *Deliver Products*, *Send Invoice*, *Receive Payment*. The order of the parallel operating activities can be exchanged. Consider Figure 4.3, the *Assemble product* is a group of activities and its extend model is depicted in Figure 4.4. Thus, in the model in Adonis there are also two paths totally similar with the ones in Figure 4.12. Therefore the evaluation of the generation is confirmed. Note that the properties of the process model designed in Adonis can be modified via the Notebook. The data contained in the event log can be changed flexibly every time the simulation is carried out. Therefore, the generation can simulate the event log data covering various perspectives.

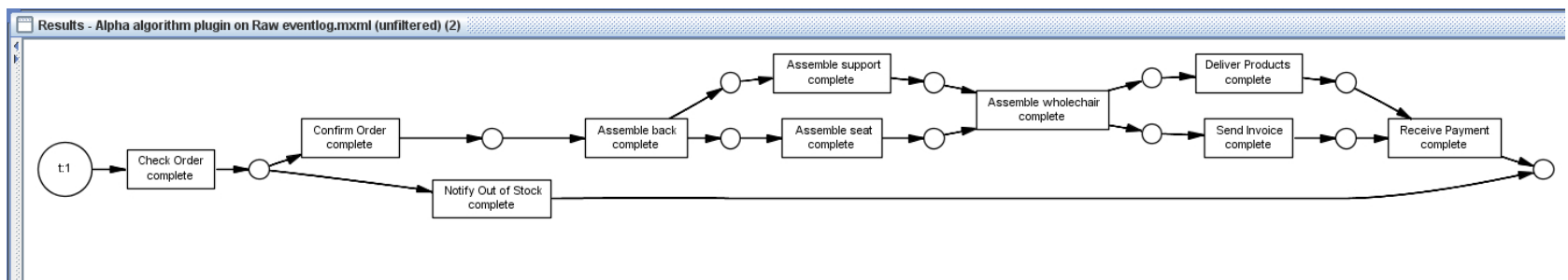


Figure 4.12: Confirmed Process Model in ProM

4.3 Involved data sources

As mentioned in Chapter 2, several applications are associated with the workflow engines to execute business process instances within a PAIS. Based on the existence of the applications, we assume that it is possible to collect the data sources of the applications referring to organization and resource management. The databases of the applications can be complicated in order to satisfy their tasks. However, in my research the data sources are used to extend the information of related elements in event logs, i.e., activity and performer. Thus, the organization data and resource data extracted from the databases of the applications are simplified and condensed as shown in Figure 4.13 and 4.14.

Organization data is used to extend the performer information of event log data. It is about the information of the employees and the organizational structure of a company. The data schema of organization data is shown in Figure 4.13. The organization data within this thesis has three tables: *employee*, *team* and *division*. The employee table has several data fields, these are personal information of an employee (i.e., name, address); his/her working experience which are represented in number of his/her working years in the company (i.e., exp_year), his/her skills (e.g., manager, seller), the cost for his/her labor (i.e., labour_cost), the team he/she belongs to (i.e., team_idteam which is a foreign key linking to team table), the division he belongs to, such as Marketing Department or Delivery Department (i.e., division_iddivision which is a foreign key linking to division table). The employee table connects to two other tables which are the team and division tables via two foreign keys: team_idteam, division_iddivision. The table team includes information about the team constructed during the work-period of an employee and the division contains data of the departments within the company.

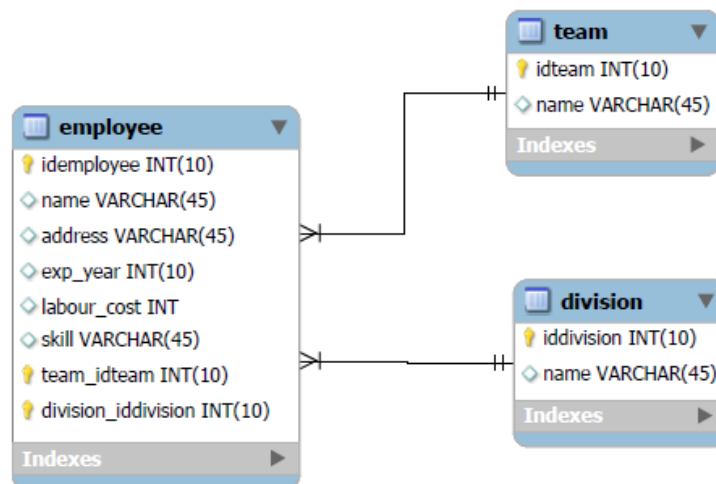


Figure 4.13: Organization Data

Resource data contains information about the consumed resources during the activity per-

formance process. The resource data is simplified into three tables which related to activity operation, which are *activity*, *activityUseResource* and *resource*. These three tables contain main information of resource consumption during the activity operation. The resource table contains information of the cost of resource consumption for each resource. The activity table contains information of the name of the activity. The *activityUseResource* table contains the consumption resource of activity execution by linking the activity table and resource table. Based on this data, the resource consumption of each activity is available.

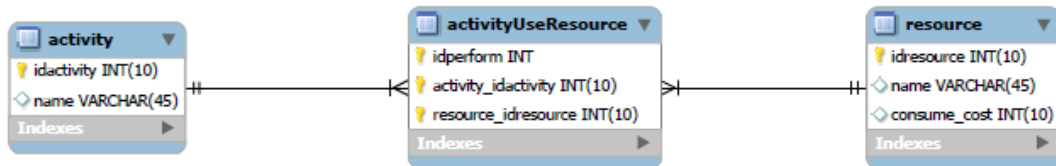


Figure 4.14: Resource Data

The involved data sources are created and stored in the MySQL workbench [41]. We use the SQL development component to create, connect and edit the databases and the Data modeling component to create a data model of the database.

Ontology-based Data Integration

Three types of data represented in the previous chapter will be integrated by using the ontology based data integration technique. The ModTOVE ontology is used as a global ontology in the current approach. The technique contains two phases: mapping and dumping. The mapping implies the schema mapping between the ontology and tables of data sources. It is performed manually with the support of a prototype. The dumping concerns the transferring of instances from data sources to the ontology. In this research, there are two types of dumping: massive dump and query dump. The massive dumping is used to transfer instances to the ontology from event log data when the ontology has no instances. The query dumping is applied after the massive dumping. When the ontology has instances from event log data, the other instances from organization and resource data sources need to be selected to choose the suitable instances. The result obtained from the integration is the ModTOVE ontology with populated instances. The following represents the details of the data sources and the techniques used for the data integration.

5.1 Modified TOVE

The ModTOVE ontology is used in the ontology data integration as a global ontology which covers all domains of the data sources. The ModTOVE ontology is modified from the TOVE ontology. Although the TOVE ontology is quite mature, the data sources assumed in the context of this thesis do not provide the suitable instances for this ontology. The ModTOVE ontology is produced as a simplified version of the TOVE ontology since it inherits the main concepts from TOVE such as organization, activity, resource, cost. The relations and properties of these concepts are simplified and edited to be suitable with my approach. Figure 5.1 shows the concepts and their properties of the ModTOVE ontology.

The center of ModTOVE is an event ontology which is not inherited from the TOVE ontology. This concept describes the process instances recorded in event log file. A process instance

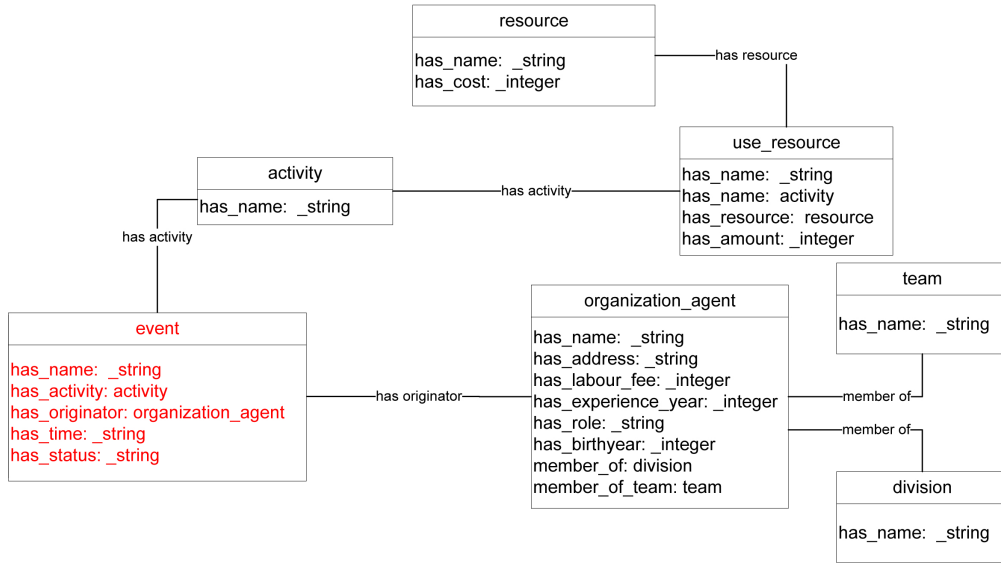


Figure 5.1: ModTOVE ontology

implies an event happened when an action is started by an originator at a particular time point. Therefore, a event contained properties: *has_activity*, *has_originator*, *has_time*, and *has_status*. These predicates are denoted as following with the e, a, t, s denoting for event, activity, time and status respectively.

has_time(e,t) defines the time-point *t* when the event *e* is started

has_status(e,s) defines the status *s* of the event *e*, which are defined by two constants *start* or *end*

has_activity(e,a) defines the activity *a* which is performed in the event *e*

has_originator(e,o) defines the originator *o* who performs the activity contained in the event *e*

The Event concept has relations with the activity concept and the organization-agent concept via the predicate *has_activity(e,a)* and *has_originator(e,o)* respectively. The activity concept does not contain any properties at the integration phase, but new properties and relations will be added in this concept during the knowledge construction phase. However, it is used to define the resource consumption for the performing activities in events. The concept *use_resource* defines the consumption resources for performing an activity. The concept is characterized via these predicates:

has_activity(ur,a) defines the activity *a* consuming the resource *ur*

has_resources(ur,r) defines the resource *r* with the consumed resource *ur*

has_amount(ur,q) defines the quantity *q* of a resource with the consumed resource *ur*

The *resource* concept represents the resources used for the activity performance. A resource has its cost which is defined by the property *has_cost* and is specified by the predicate *has_cost(r,c)* where a resource is denoted by *r* and a cost is denoted by *c*.

The *organization_agent* concept defines a person or an employee who performs activities in processes. This concept contains properties about personal information and working information of an employee. The properties are represented by the following predicates.

has_name(o,n) defines the name *n* of the employee *o*

has_address(o,addr) defines the address *addr* of the employee *o*

has_labor_fee(o,c) defines the money *c* which the employee *o* receives when he/she accomplish an activity he/she was assigned to

has_experience_year(o,y) defines the working years *y* the employee *o* has been working in the organization

has_role(o,r) defines the role *r* of the employee *o* in the organization, such as accountant, delivery man, or warehouse keeper

has_birthyear(o,by) defines the year *by* when the employee *o* was born, this property also defines how old the employee is

has_member_of(o,d) defines the division or department *d* the employee *o* belongs to

has_member_of_team(o,t) defines the team *t* the employee *o* joins

The concept division depicts the organizational structure of an organization. An organization is structured into several divisions. An *organization_agent* belongs to a *division* which manages her/him with an organizational perspective.

An *organization_agent* joins a team if he/she works with others on a particular case. This concept can be populated with instances from data sources or it can be created by specifying an axiom during the knowledge construction phase.

5.2 Ontology-based data integration

The goal of the integration is to enrich the event log data by the organization data and resource data. The integrating process is actually built by the immigrating instances extracted from data sources to the ModTOVE ontology. The techniques used for the integration are based on the literature about the ontology based data integration mentioned in Chapter 2. Basically the integration methods applied in this research contain 2 steps: concept mapping and instances immigration. However, because of the goal of the integration and the characteristic of the data sources, the integration method in this research incorporates several techniques. The overview of the method is shown in Figure 5.2.

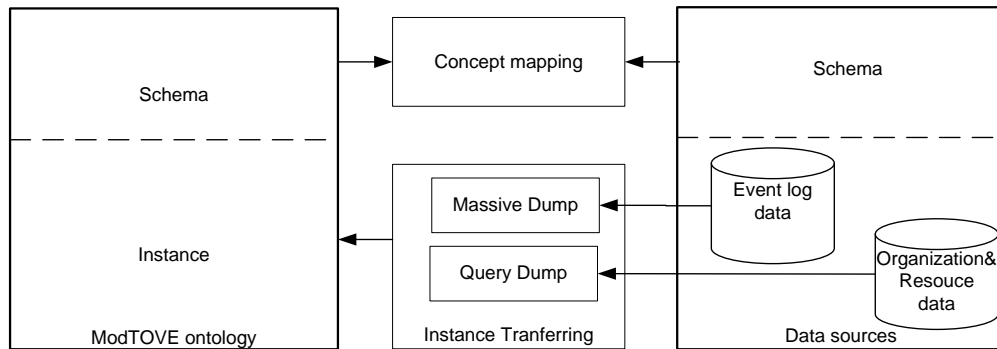


Figure 5.2: Ontology-based Data Integration

The concept mapping corresponds to identifying the correspondence between the tables and their properties in data sources and the concepts and their properties in the ontology. The mapping is executed manually by users with the support of a prototype to show the data schema of the data sources and the ontology. In other words, users determine which tables and properties in the data sources are used to transfer instances to the ontology. The ModTOVE plays the role as a global ontology which covers all domains of the data sources. In particular, ModTOVE contains the concepts of event, activity, originator, and resources which have data stored in the data sources. Based on the concept mapping, the instances are transferred to the ontology in two ways: massive dump and query dump.

To define the mapping for the massive dump, the data fields of the event log data are assigned to the concepts which need to populate instances in ModTOVE. In particular, an event in the event log data is assigned to an event concept. The *originator* field is equivalent to the *organization_agent* concept. The *activity* field is mapped to the *activity* concept. With the determination, the value of the data field identified in the mapping becomes the value of the instances of concepts of ModTOVE. With the mapping, the massive dump technique is applied to transfer the instances from the event log data to the ontology. In particular, the event concept is populated with all records of event log data. The activity concept is populated with the activity value filtered from the activity column of the event log data. The *Origination_agent* concept is populated with the *originator* filtered from the originator column of the event log data.

Consider the schema of ModTOVE in Figure 5.1, the instances of the concepts *event*, *activity* and *organization_agent* are populated from the event log data, and created by the massive dump technique. However, the other properties of the instances are still missing. For example, the instances of the *organization_agent* concept are filled with the *has_name* property, the other ones are empty. Moreover, the relations between the concepts with the other ones such as *resource* concepts are not established in the instance level because of the missing instances. Therefore, the organization data and resource data are used to fill the instances in the ontology. Thereby, the instances from the event log data are enriched by information coming from other data sources. The populated instances from the organization and resource data is executed by the query dumping technique.

The query dumping is executed based on the concept mapping and the query construction. The concept mapping in this phase corresponds to defining correspondences between the tables in the organization data and resource data with the concepts in ModTOVE. With the assumption that the data sources are suitable with the ontology, their tables are quite similar with the concepts in ontology. With the resource data, the tables *resource*, *activity*, and *activityUseResource* are mapped to the concepts *resource*, *activity*, *use_resource* in ModTOVE. The essential point in the query dump technique is the query used to extract the instances in the data sources. With the purpose of the integration mentioned above, the queries are constructed around the event log data. In other words, the value of the event log data exists in the "where clause" of the queries. For example, in the ontology, there is an originator *Tom* in the concept *organization_agent*. The instance transferring from organization data to the concept *organization_agent* is performed as follows. A query is built with the where clause containing the condition "*name = Tom*". If the query returns a positive result, the resulting information about *Tom* in the organization data is added to the ontology as an instance of the *organization_agent* concept. In case the property mapped is a foreign key (e.g., *member_of:division*), the data from the foreign table is obtained by querying created with the foreign key as the condition.

5.3 Implementation of Ontology-based Data Integration

The Ontology-based Data Integration aims to populate instances to the ModTOVE ontology. The goal beyond the integration is to enrich the information of event log data. The API of the data integration contains 2 main phases depicted in Figure 5.3.

The first phase is transferring event log data to the ontology as instances and the second one is populating the ontology with the instances which are selected from the organization data and resource data based on the event log instances existing in the ontology. In particular, the integration contains 4 steps:

First, the event log data and the ontology are input via a user interface. The event log and ontology are accessed to get the data, their fields and concepts.

Second, the data fields and concepts are mapped manually. This step is called concept mapping.

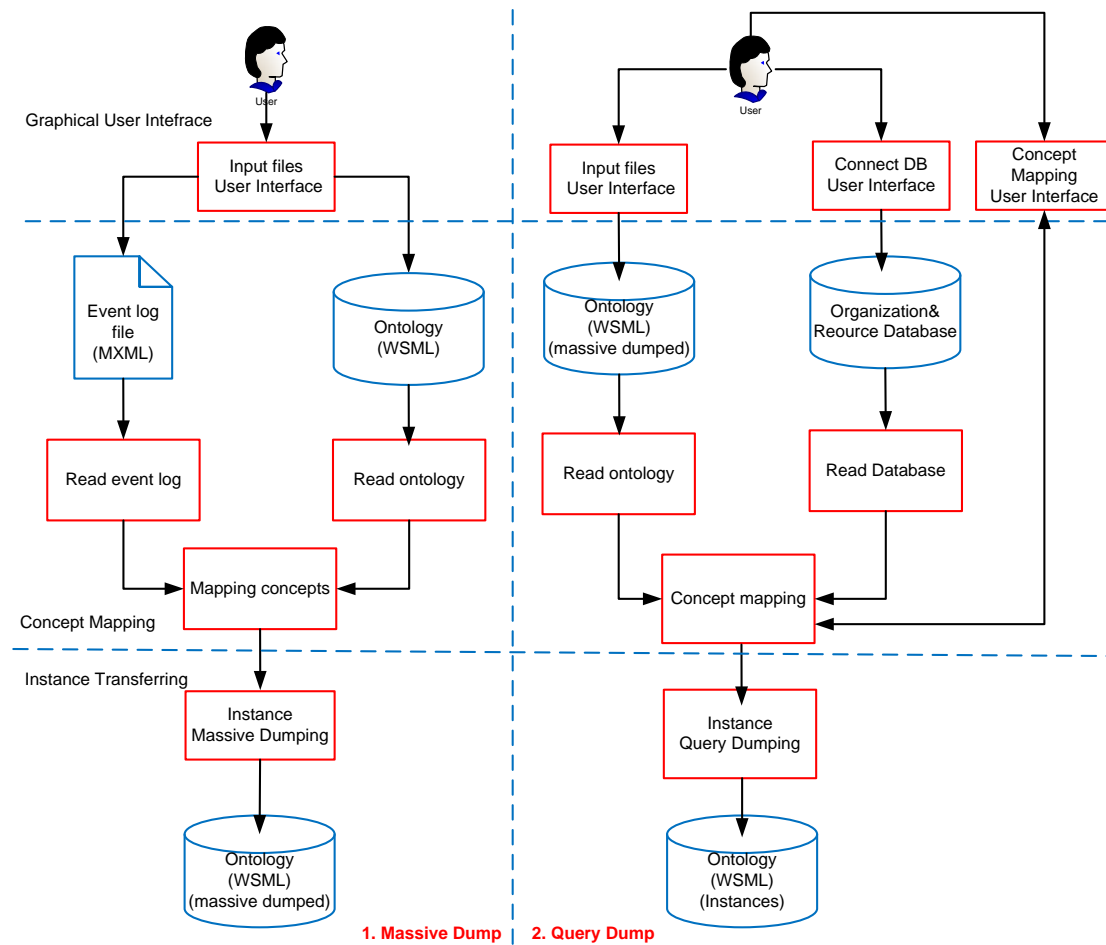


Figure 5.3: Data integration API

Third, data fields from the event log which are mapped with concepts of the ontology are filtered and populated to the ontology as instances. This step, called Massive Dump, fills the ontology with instances from the event log data.

Forth, the ontology from the third step is input again via a user interface to integrate with the organization and resource database. The concepts, tables and their properties from the ontology and the database are shown on a user interface. The mapping is performed manually by users.

Last, based on the concept mapping in Step 4, queries are created to select suitable data from the database for populating instances to the ontology. The ontology is filled with instances from the database and event log data.

The concrete processing of the massive dump is described in the following steps:

- The file paths of the event log data and the ontology are identified by users via the Massive Dump User Interface. In this interface, users also determine the name and the location of the new ontology file where it can be stored after the instance population.
- The Event log file and the ontology are read to get their schema for the concept mapping. In particular, the data fields of the event log are *ProcessInstance*, *WorkflowModelElement*, and *Originator*. The concepts of ontology populated instances from the event log are *process*, *activity* and *organization_agent*. The two elements of the concept mapping between event log and ontology are *ProcessInstance* and *process*, *WorkflowModelElement* and *activity*, *Originator* and *organization_agent*.
- The massive instance dumping is executed in three concepts:
 - The *event* concept refers to populated instances retrieved from *ProcessInstances* in the event log. For each instance created in the *event* concept, the value of the property *has_name* is transferred from the value of the *Case_id* field, *has_activity* from *WorkflowModelElement*, *has_originator* from *Originator*, *has_time* from *Timestamp* and *has_status* from *EventType*.
 - The value of the *WorkflowModelElement* can be repeated in every *AuditTrailEntry*. Thus, the values of the *WorkflowModelElement* field are filtered to exclude the repeat. Then, each value of the *WorkflowModelElement* field is transferred to the ontology and becomes the value of the *has_name* property of each instance created in the *activity* concept.
 - The instance transferring process for the concept *organization_agent* is similar to the one of the concept *activity*. However, only the *has_name* property of the *organization_agent* is filled with the value from the *Originator* field of the event log. The other properties of this concept are empty.
- The ontology is populated instances from the event log data, and stored in a new file which was identified by users at the beginning step.

The algorithm for the query dump execution is depicted in Figure 5.6. The description of the algorithm is represented as follows:

- The path file of the ontology is identified and the database containing the organization data and resource data are connected by users via the user interface shown in Figure 5.4
- The ontology is read to access its concepts. The organization data or resource data is accessed when the connection executed in the previous step was successful. The data tables of the organization data or resource data are obtained.

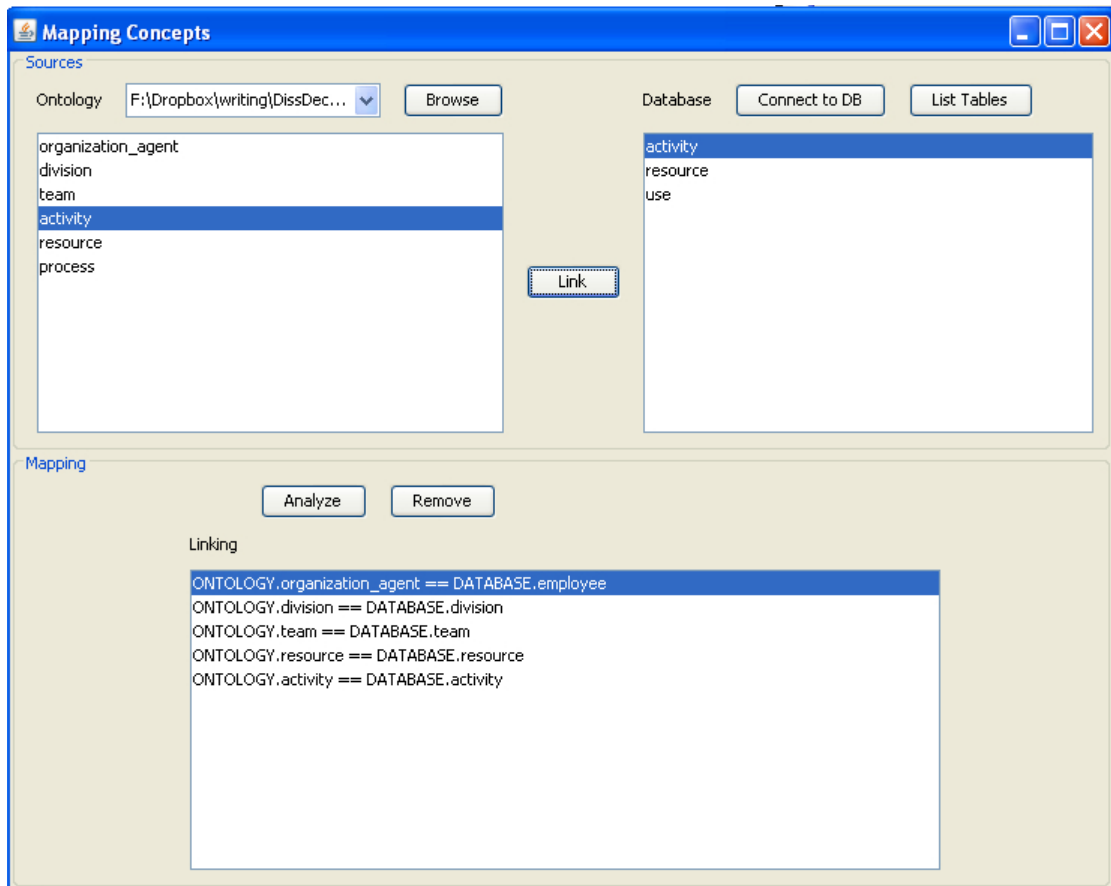


Figure 5.4: Mapping Concepts

- As seen in Figure 5.4, the concepts of the ontology and the data table of data sources are displayed in the interface.
- Users carry out the concept mapping between the ontology and the data sources. For example, Figure 5.4 depicts the mapping between the `organization_agent` concept of ontology and the `employee` table of organization data source, the `division` concept of the ontology and the `division` table.
- The concept mapping table is taken and put into an array. Let n be the number of items in the mapping table. A loop executed n times, called loop N , is created to access each item of the concept mapping.
- At item i , the properties of the concept i and the data fields of the table i are displayed in the user interface shown in Figure 5.5. Users need to identify which property of the concept is suitable with which data field of the data source.

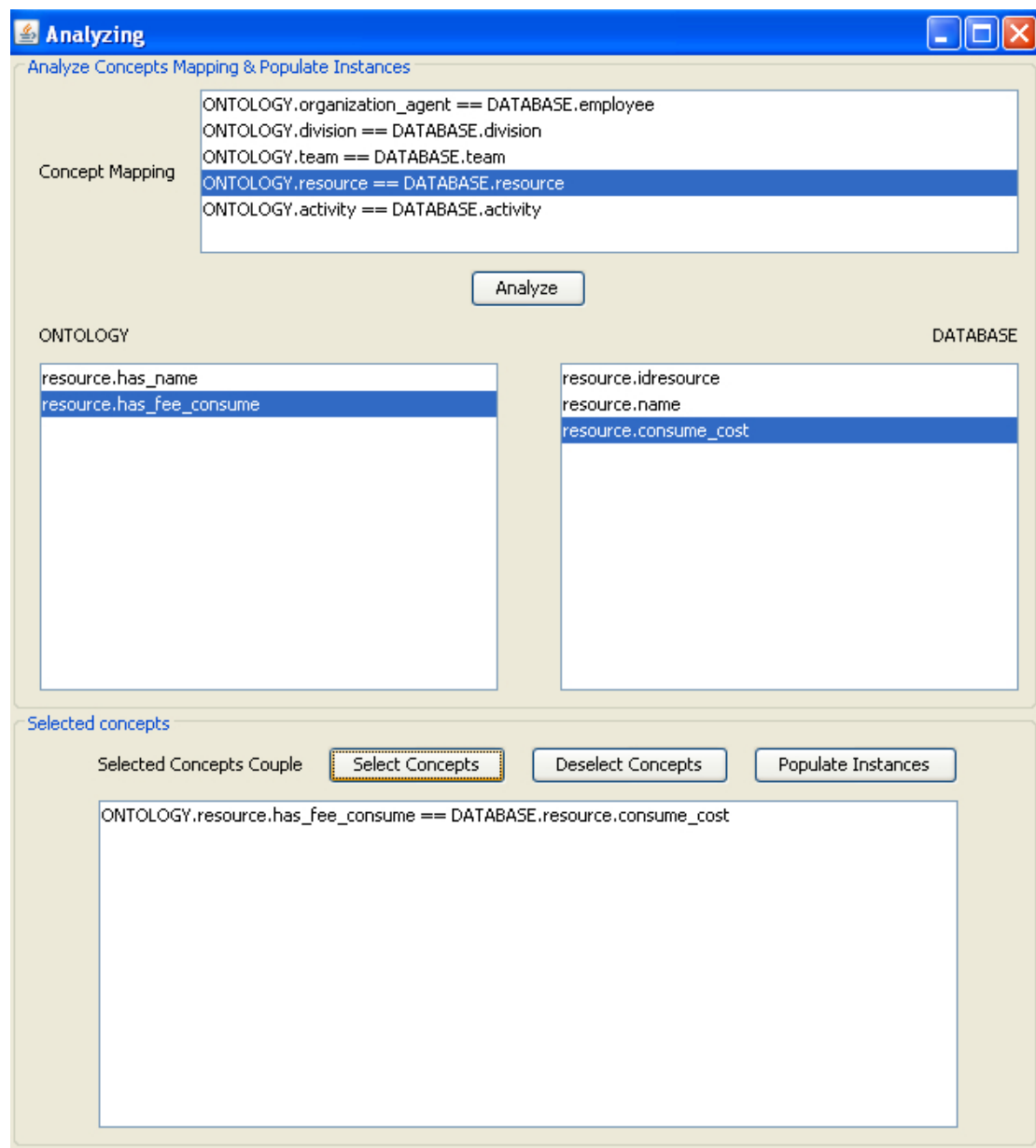


Figure 5.5: Analyze Properties

- The property mapping is taken and put into an array. At the concept i , let m be the number of instances of concept i .
- If $m = 0$, it means the concept i is still empty, the concept is filled by the value from its corresponding table in the database by the massive dump technique. This case happens with the concept, that are not related to the event log data, such as the division concept or

resource concept.

- If $m > 0$, a loop executed m times, called loop M, is created to access each instance of the concept i
- At instance k , a query dump is executed to add values to the empty properties of instance k . The algorithm for query dump is depicted in Figure 5.7
 - A query is constructed to select the properties of the table as identified in the property mapping table. The condition of the query is created with the clause "*instance[k].has_name = table.has_name*". It means the query searches for the data record which contains the missing value of the empty properties of the instance k . This is the essential point which represents the capability for enriching information of the event log data in the data integration.
 - The result obtained from the query is considered. Let l be the number of values. A loop executed l times, called loop L, is created to check every value.
 - * At value j , if the value is a foreign key, the foreign table is accessed to get the real value. For example, if the value is the identifier of a division table, then the name of the division is taken from the division table by querying the value based on the foreign key. The real value is put in a set, named S .
 - * If the value is not a foreign key, the value is put in the set S .
 - * When the loop L is finished, all the value are checked. The set S contains all the values used to edit the instance k
 - Instance k is edited with the set of values filled in the empty properties. The query dump for the instance k is finished.
- When all the instances of concept i are edited with the values filled in the empty properties, the query dumping for the concept is finished.
- When all the concepts in the concept mapping table are populated with instances, the ontology is updated and the process is finished.

The result of the data integration is the ModTOVE ontology which is populated with instances extracted from data sources. The next chapter represents the knowledge construction in the ModTOVE ontology to create a knowledge base which can answer questions related to business process operations.

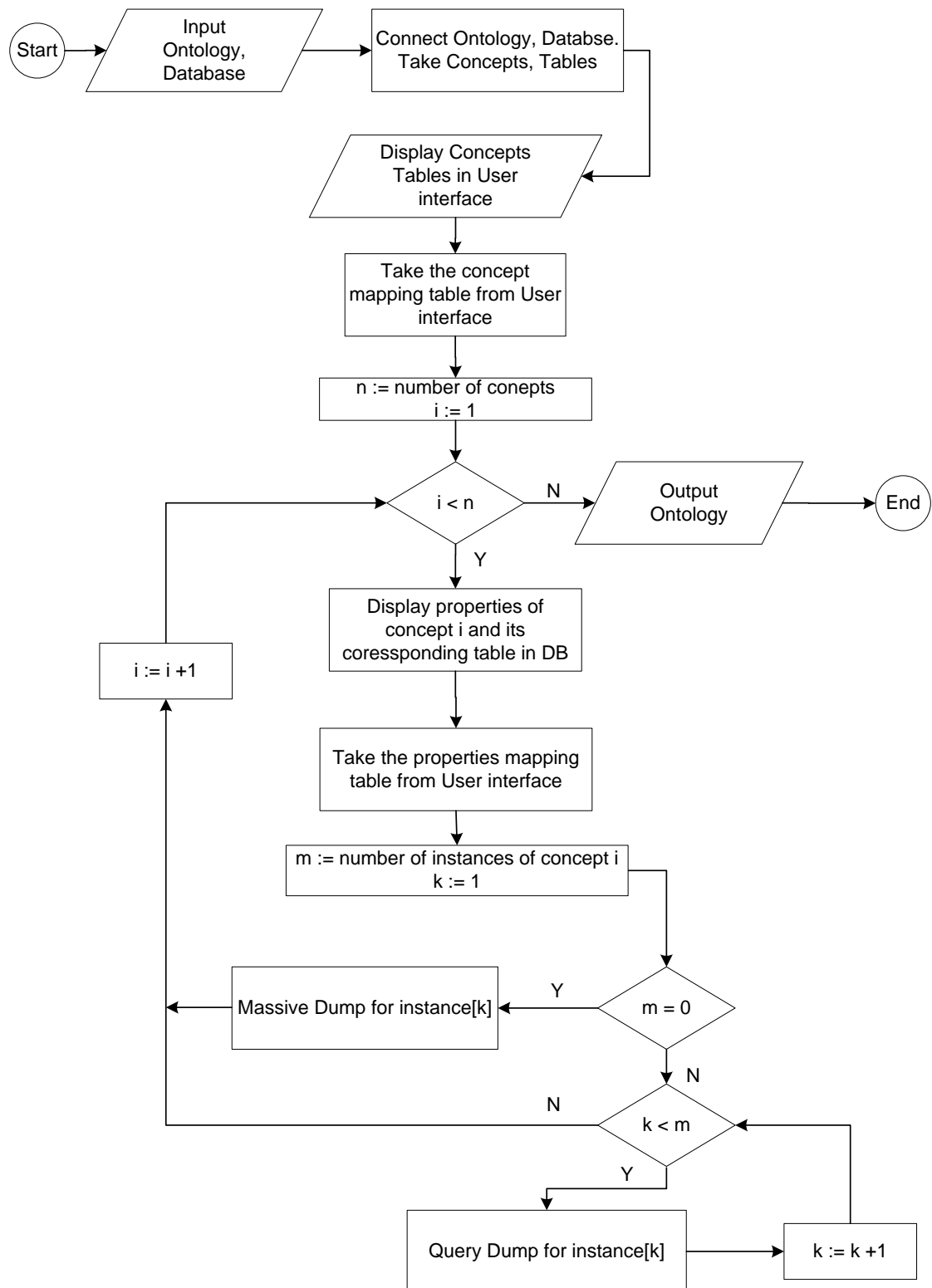


Figure 5.6: Algorithm of data integration

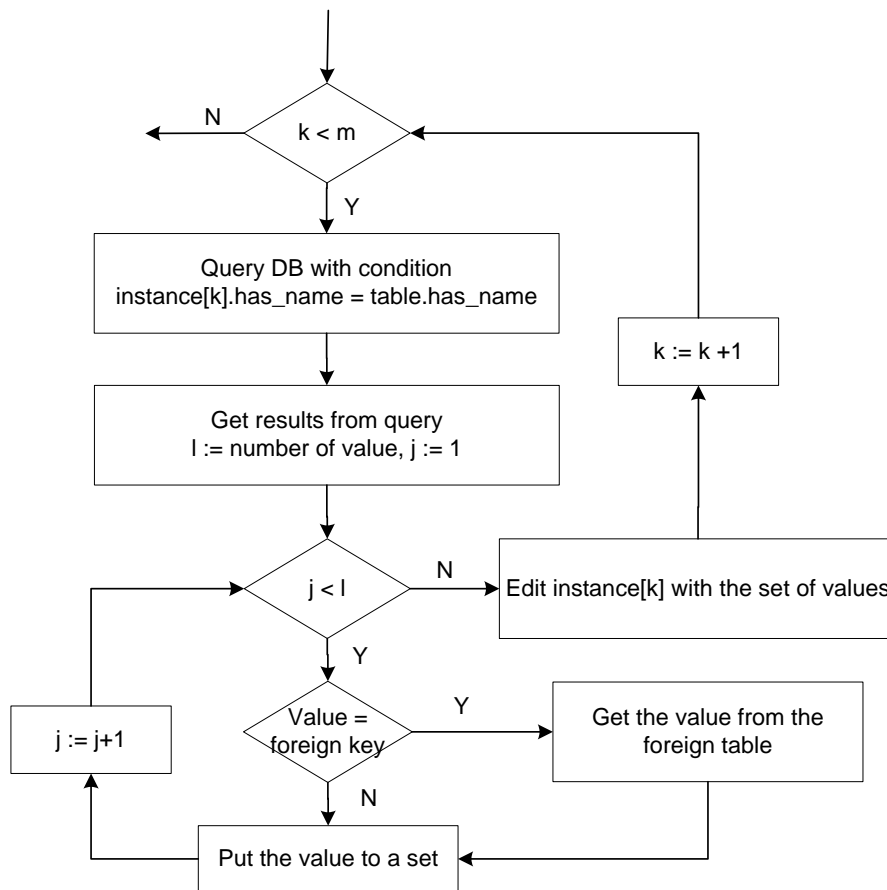


Figure 5.7: Algorithm of query dumping

Knowledge construction

In the previous chapter, the ModTOVE ontology is populated with instances from data sources. The knowledge obtained from the ModTOVE ontology so far is only based on the existing concepts and constraints. In this chapter, the knowledge construction aims to produce other relations and rules based on the current ones existing in the ontology. With the new relations and rules, the querying and reasoning can be performed in the knowledge base to answer more questions in the domain of process operation.

The relations between concepts in the ontology are analyzed to create new concepts and relations in ModTOVE. Then the knowledge base is experimented with a set of questions related to business process operation. The performing process of the knowledge construction is generalized as shown in Figure 6.1. The axioms are tested and operated by the WSM Toolkit.

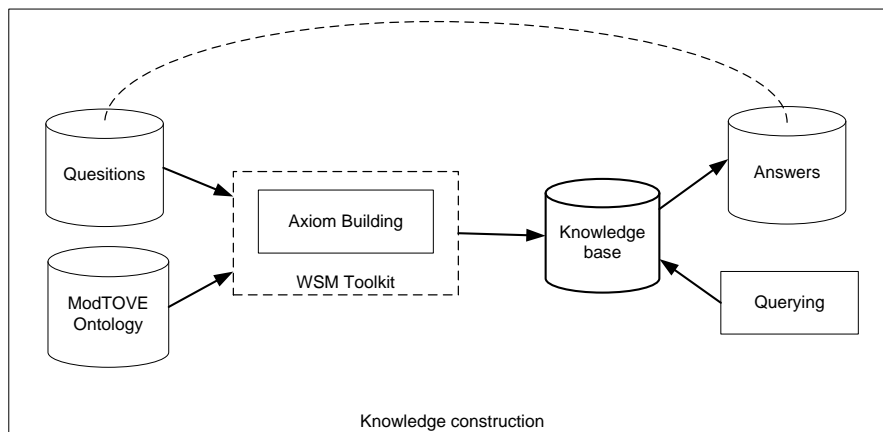


Figure 6.1: Knowledge construction

In the following we analyze questions in order to formulate our axioms needed to complete our knowledge base. The questions are classified into four categories: process operation, organization structure, resource consumption and costs, which are the perspectives of process analysis in this thesis. The questions are raised to provide the information which is not supported by the process visualization in terms of models. A process model combined with the information found in the answers can provide a deeply understanding for people in both business and IT sides.

6.1 Question analysis

1. Process operation

Question 1.1 Which activities are performed in "case 1"?

Analysis 1.1 This kind of question can be answered directly by the properties of the concept *event*.

Query 1.1 *?event[has_activity hasValue ?activity] memberOf event and ?event[has_name hasValue "case 1"] and ?event[has_status hasValue "end"]*.

However, for the questions answered by the knowledge implied beyond the ModTOVE ontology, reasoning is necessary to carry out via the axioms. The analysis of the following questions represent the knowledge as well as the axioms needed to build to answer the questions.

Question 1.2 Which cases are successful? How many cases of the order process are successful?

Analysis 1.2 A case is defined as successful if it contains an activity contained in the assembling process. The axiom in Figure 6.2 shows the definition of a success case. This axiom creates a new class called *success_case_name* containing the names of successful cases.

```

axiom success_case_name
  definedBy
    ?name memberOf success_case_name
  impliedBy
    ?event[has_activity hasValue assemble_back] memberOf
    event
    and ?event[has_name hasValue ?name] memberOf event.

```

Figure 6.2: A new concept "success cases"

Query 1.2 *?name memberOf success_case_name*.

Question 1.3 Which activities do belong to successful cases?

Analysis 1.3 An activity participates in a successful case if the event triggered by the activity belongs to the successful case. Figure 6.3 shows the axiom defining the new concept *activity_belong_successcase* which contains the activities.

```

axiom activity_belong_successcase
  definedBy
    ?activity memberOf activity_belong_successcase
  impliedBy
    ?event[has_activity hasValue ?activity] memberOf event
    and ?event[has_name hasValue ?name] memberOf event
    and ?name memberOf success_case_name.

```

Figure 6.3: Activities belong to success cases

Query 1.3 *?activity memberOf activity_belong_successcase.*

Question 1.4 Which cases have failed? How many cases of the order process have failed?

Analysis 1.4 Similarly, a case has failed if it contains an activity which notifies products out of stock. The new concept *fail_case_name* is defined by the axiom shown in Figure 6.4.

```

axiom fail_case_name
  definedBy
    ?name memberOf fail_case_name
  impliedBy
    ?event[has_activity hasValue notify_outofstock] memberOf event
    and ?event[has_name hasValue ?name] memberOf event.

```

Figure 6.4: A new concept "fail cases"

Query 1.4 *?name memberOf fail_case_name.*

Question 1.5 Which activities do belong to failed cases?

Analysis 1.5 An activity participates in a failed case if the event triggered by the activity belongs to the failed case. Figure 6.5 shows the definition of a new concept *activity_belong_failcase*.

```

axiom activity_belong_failcase
  definedBy
    ?activity memberOf activity_belong_failcase
  impliedBy
    ?event[has_activity hasValue ?activity] memberOf event
    and ?event[has_name hasValue ?name] memberOf event
    and ?name memberOf fail_case_name.

```

Figure 6.5: Activities belong to fail cases

Query 1.5 *?activity memberOf activity_belong_failcase.*

Question 1.6 Which activities need high experienced employees?

Analysis 1.6 To answer this question, first, the concept "*high experienced employee*" need to be defined. An employees is considered "*high experienced*" if he/she has more than 5 years of experience. The new concept *teamHighExperience* represents for hight experienced employees is defined by the axiom shown in Figure 6.6.

```
axiom teamHighExperience
  definedBy
    ?o memberOf teamHighExperience
  impliedBy
    ?o[has_name hasValue ?name] memberOf organization_agent
    and ?o[has_experience_year hasValue ?t]
    and ?t > 5.
```

Figure 6.6: Definition of a "high experience team"

From the concept *teamHighExperience*, a new concept is created to describe the activities performed by the *organization_agents* who belong to the high experience team. The concept is named *activity_need_high_experience* and defined by the axiom shown in Figure 6.7.

```
axiom activity_need_high_experience
  definedBy
    ?a memberOf activity_need_high_experience
  impliedBy
    ?a[has_originator hasValue ?o] memberOf activity
    and ?o memberOf teamHighExperience.
```

Figure 6.7: Definition of activity performed by high experience team

Query 1.6 *?activity memberOf activity_need_high_experience.*

2. Organization structure

Question 2.1 Which activities are performed by "Tom"?

Analysis 2.1 This question can be answered directly by the property *has_originator* of the concept *activity* and the property *has_name* of the concept *organization_agent*.

Query 2.1 *?activity[has_originator hasValue ?organizationagent] memberOf activity and ?organizationagent[has_name hasValue "Tom"].*

Question 2.2 Who performs activity [assemble support]?

Analysis 2.2 This question can be answered directly by the properties *has_originator* and *has_name* of the concept *activity*.

Query 2.2 *?activity[has_originator hasValue ?organizationagent] memberOf activity and ?activity[has_name hasValue "assemble support"]*.

Question 2.3 Who works in which division?

Analysis 2.3 A division contains a number of members. Thus, the property *has_member* is created for the concept division and defined by the axiom shown in Figure 6.8.

```
axiom depart_agent
  definedBy
    ?t[has_member hasValue ?o] memberOf division
impliedBy
  ?o[has_name hasValue ?n] memberOf organization_agent
  and ?o[member_of_division hasValue ?t] memberOf organization_agent.
```

Figure 6.8: Definition of the property *has_member* of concept division

Query 2.3 *?division[has_member hasValue ?organizationagent] memberOf division*.

Question 2.4 Who works in the [Accounting Division]?

Analysis 2.4 A new concept *Accounting_division* is a subconcept of division. It contains the *organization_agent* who belong to division *Accounting*. The definition is shown in Figure 6.9

```
axiom Accounting_division
  definedBy
    ?o memberOf Accounting_division
impliedBy
  ?o[member_of_division hasValue ?t] memberOf organization_agent
  and ?t[has_name hasValue "Accounting"] memberOf division.
```

Figure 6.9: Definition of concept "Accounting_division"

Query 2.4 *?organizationagent memberOf Accounting_division*.

Question 2.5 Who performs which activity?

Analysis 2.5 The property *has_originator* of the concept *activity* is identified by the originator who performs the activity when it participates in a particular event. It is defined by the axiom shown in Figure 6.10.

Query 2.5 *?activity[has_originator hasValue ?organizationagent] memberOf activity*.

Question 2.6 Who works together in a successful team?

```

axiom activity_performer
  definedBy
    ?a[has_organator hasValue ?o] memberOf activity
impliedBy
  ?e[has_activity hasValue ?a] memberOf event
  and ?e[has_organator hasValue ?o] memberOf event.

```

Figure 6.10: Definition of the property *has_organator* of concept *activity*

Analysis 2.6 A team is defined as successful if its members have worked in successful cases. The definition is described by the axiom in Figure 6.11.

```

axiom teamSuccess
  definedBy
    ?o memberOf teamSuccess
impliedBy
  ?e[has_organator hasValue ?o] memberOf event
  and ?e[has_name hasValue ?name] and ?name memberOf success_case_name.

```

Figure 6.11: Definition of a "Successful Team"

Query 2.6 *?organizationagent memberOf teamSuccess.*

Question 2.7 Does "Tim" work in a successful team?

Analysis 2.7 This question can be answered directly by the new concept *teamSuccess* and the property *has_name* of the concept *organization_agent*.

Query 2.7 *?organizationagent memberOf teamSuccess and ?organizationagent[has_name hasValue "Tim"] memberOf organization_agent.*

Question 2.8 Who works together in "team 1"?

Analysis 2.8 A team is defined by a group of *organization_agents* who have worked on the same case or a process instance. For instance, people who have worked in "case 1" belong to a "team 1". Therefore, team 1 is defined as in Figure 6.12.

```

axiom team1
  definedBy
    ?o memberOf team1
impliedBy
  ?e[has_name hasValue "case 1"] memberOf event
  and ?e[has_organator hasValue ?o] memberOf event
  and ?o[has_name hasValue ?name] memberOf organization_agent.

```

Figure 6.12: Definition of "team 1"

Query 2.8 *?organizationagent memberOf team1.*

Question 2.9 Who has much work experience?

Analysis 2.9 An employees has much work experience if he/she belongs to the high experience team. The concept *teamHighExperience* is defined by the axiom shown in Figure 6.6.

Query 2.9 *?organizationagent memberOf teamHighExperience.*

Question 2.10 Who works together in a failed team?

Analysis 2.10 A team is defined as failed if its members have worked in failed cases. The definition is described by the axiom in Figure 6.13.

```
axiom teamFail
  definedBy
    ?o memberOf teamFail
  impliedBy
    ?e[has_organator hasValue ?o] memberOf event
    and ?e[has_name hasValue ?name] and ?name memberOf fail_case_name.
```

Figure 6.13: Definition of a "Failed Team"

Query 2.10 *?organizationagent memberOf teamFail.*

Resource consumption

Question 3.1 Which resources are used for which activities?

Analysis 3.1 When an activity is carried out within an event, it needs to consume particular materials. For instance, to assemble a back of a chair, the materials are 2 ears, 1 top rial, 1 cross rail and 1 stile. The amount of each material and the consumption of the materials are included in the concept *use_resource*. This concept implies the consumed materials for performing activities. Moreover, the activities are part of the concept *event*. Therefore, a new property named *has_resource* is created in the concept *event* to imply the consumed materials or resources in each event. The property is represented by the predicate *has_resource(e,r)* and defined by the axiom shown in Figure 6.14

The axiom means that an event *?event* consumes a resource *?resource* if it is in the status *start* (i.e., the event is started) and it has an activity *?activity* and activity *?activity* uses resources *?resource*. In this axiom, the new property *has_resource* of the concept *activity* is used. This property is defined by the axiom shown in Figure 6.15

The property *has_resource* of the concept *activity* is defined via the concept *use_resource*.

```

axiom event_resource
  definedBy
    ?event[has_resource hasValue ?resource] memberOf event
  impliedBy
    ?event[has_activity hasValue ?activity] memberOf event
    and ?event[has_status hasValue "start"] memberOf event
    and ?activity[has_resource hasValue ?resource] memberOf activity.

```

Figure 6.14: The definition of the property *has_resource* of the concept *event*

```

axiom activity_use_resource
  definedBy
    ?activity[has_resource hasValue ?resource] memberOf activity
  impliedBy
    ?userresource[has_activity hasValue ?activity] memberOf use_resource
    and ?userresource[has_resource hasValue ?resource] memberOf use_resource.

```

Figure 6.15: The definition of the property *has_resource* of the concept *activity*

Query 3.1 *?activity[has_resource hasValue ?organizationagent] memberOf activity.*

Question 3.2 Which resources are used for the activity [assemble back]?

Analysis 3.2 This question can be answered directly via the property *has_resource* of the concept *activity*.

Query 3.2 *?activity[has_resource hasValue ?organizationagent] memberOf activity and ?activity[has_name hasValue "assemble back"].*

Question 3.3 Which resources are used by [Tim]?

Analysis 3.3 This question can be answered based on the properties *has_originator* and *has_resource* of the concept *activity* and the property *has_name* of the concept *organization_agent*.

Query 3.3 *?activity[has_originator hasValue ?originator] memberOf activity and ?activity[has_resource hasValue ?resource] memberOf activity and ?originator[has_name hasValue "Tim"].*

Question 3.4 Which resources are used by [SuccessTeam]?

Analysis 3.4 Resources used for successful teams are the resources consumed by activities which are performed by successful teams. The definition is represented by the axiom shown in Figure 6.16.

Query 3.4 *?resource memberOf resourceSuccessTeam.*

Costs

Question 4.1 Which activities have high costs?

```

axiom resourceSuccessTeam
  definedBy
    ?resource memberOf resourceSuccessTeam
  impliedBy
    ?e[has_resource hasValue ?resource]
    and ?e[has_originator hasValue ?o]
    and ?o memberOf teamSuccess.

```

Figure 6.16: Definition of Resource used for Successful Team

Analysis 4.1 To answer that question, domain experts need to create the definition of an activity considered high cost. For example, an expert of a wood furniture assembling domain claims that *"An activity is considered a high cost activity if the amount of resource items used for its performance is more than 2 and the price of each item is more than 20 money units"*. The definition is expressed by the axiom shown in Figure 6.17.

```

axiom activity_is_high_cost
  definedBy
    ?a memberOf activity_is_high_cost
  impliedBy
    ?ur[has_resource hasValue ?r] memberOf use_resource
    and ?ur[has_activity hasValue ?a] memberOf use_resource
    and ?ur[has_amount hasValue ?q] memberOf use_resource
    and ?r[has_cost hasValue ?c] memberOf resource
    and ?q > 2
    and ?c > 20.

```

Figure 6.17: A new concept high cost activity

Query 4.1 *?activity memberOf activity_is_high_cost.*

Question 4.2 How much does the labor cost for [Tim]?

Analysis 4.2 This question can be answered by the property *has_labour_fee* and *has_name* of the concept *organization_agent*.

Query 4.2 *?originator[has_labour_fee hasValue ?cost] memberOf organization_agent and ?originator[has_name hasValue "Tim"] memberOf organization_agent.*

Question 4.3 How much does the labor cost for performing activity [assemble support]?

Analysis 4.3 The labor cost for an activity is defined as the labor cost of the *organization_agent* who performs it. The property *has_cost_labor* of the concept *event* is created to define the *"labor cost"* of each activity involved in the event. Figure 6.18 shows the definition.

The question 4.3 can be answered by the properties *has_activity* and *has_cost_labor* of the concept *event* and the property *has_name* of the concept *activity*.

Query 4.3 *?event[has_activity hasValue ?activity] and ?event[has_cost_labor hasValue ?laborcost] and ?activity[has_name hasValue "assemble support"].*

```

axiom event_cost_labor
  definedBy
    ?event[has_cost_labor hasValue ?cost] memberOf event
impliedBy
  ?event[has_originator hasValue ?originator] memberOf event
  and ?originator[has_labour_fee hasValue ?cost] memberOf organization_agent.

```

Figure 6.18: Definition of the property *has_cost_labor* of the concept *event*

Question 4.4 How much does the labor cost for performing a success case?

Analysis 4.4 This question can be answered by the concept *success_case_name* and the properties *has_name* and *has_cost_labor* of the concept *event*.

Query 4.4 *?event[has_name hasValue ?name] memberOf event and ?name memberOf success_case_name and ?event[has_cost_labor hasValue ?cost] memberOf event.*

Question 4.5 How much does the labor cost for a high experienced team?

Analysis 4.5 This question can be answered by the concept *teamHighExperience* and the property *has_labour_fee* of the concept *organization_agent*.

Query 4.5 *?organizationagent[has_labour_fee hasValue ?labourcost] memberOf organization_agent and ?organizationagent memberOf teamHighExperience.*

Case study and Evaluation

This chapter introduces the implementation of this research. A use case is represented to illustrate the applicability of the implementation in forms of a prototype. The steps of the implementation are described and the results obtained from the research are shown in this chapter. The research is evaluated by the implementation and its results.

7.1 Case Study - An Order Process

Based on the scenario described in Chapter 1, this section represents a case study about an ordering process happened in a company providing services to assemble and sell wood furniture. A customer named Ted wants to order a wood side chair provided by the company. He sends the order to the company via email. Chris, who works in the customer management department of the company, receives Ted's email. Chris reads the email to analyze what kind of requests come from the customer, Ted. When Chris realizes the email from Ted is about ordering a wood side chair, he sends the order to Tom who works in the warehouse management department. Tom analyzes the order and checks whether the amount of components to produce the chair is sufficiently available in the warehouse. The analyzing of the order is supported by an application called Warehouse Control System. If the components are not enough to produce the chair as the customer requires, Tom will notify Chris that the order is not possible to be accomplished with the current state of the warehouse. In that case, Chris sends a notification to Ted that the product he orders is out of stock, and the ordering process has failed with the negative status. If the warehouse still has enough components to produce the chair of Ted's order, Tom will inform Chris that the order can be executed. Chris sends the confirmation of the order to Ted and the assembling of the chair is carried out by the staff of the warehouse management department. The Warehouse Control System manages the resource consumption in the assembling process. A chair is separated into three parts: back, seat and support. The process is started with the back part assembling handled by Tim, an employee of the warehouse manage-

ment department. After that, the seat and the support parts are assembled concurrently to the back part by Tom. When the assembling process is completed, the product is delivered to Ted by Brad who works in the delivering department. At the same time, the invoice of the order is sent to Ted by Adam who works in the accounting department. The customer Ted will send the payment to the company after he receives the chair as he ordered. When the payment is received by Amy, an employee of the accounting department, the ordering process is finished. Figure 7.1 shows the case study with the human interaction in the process model as defined as part of this thesis.

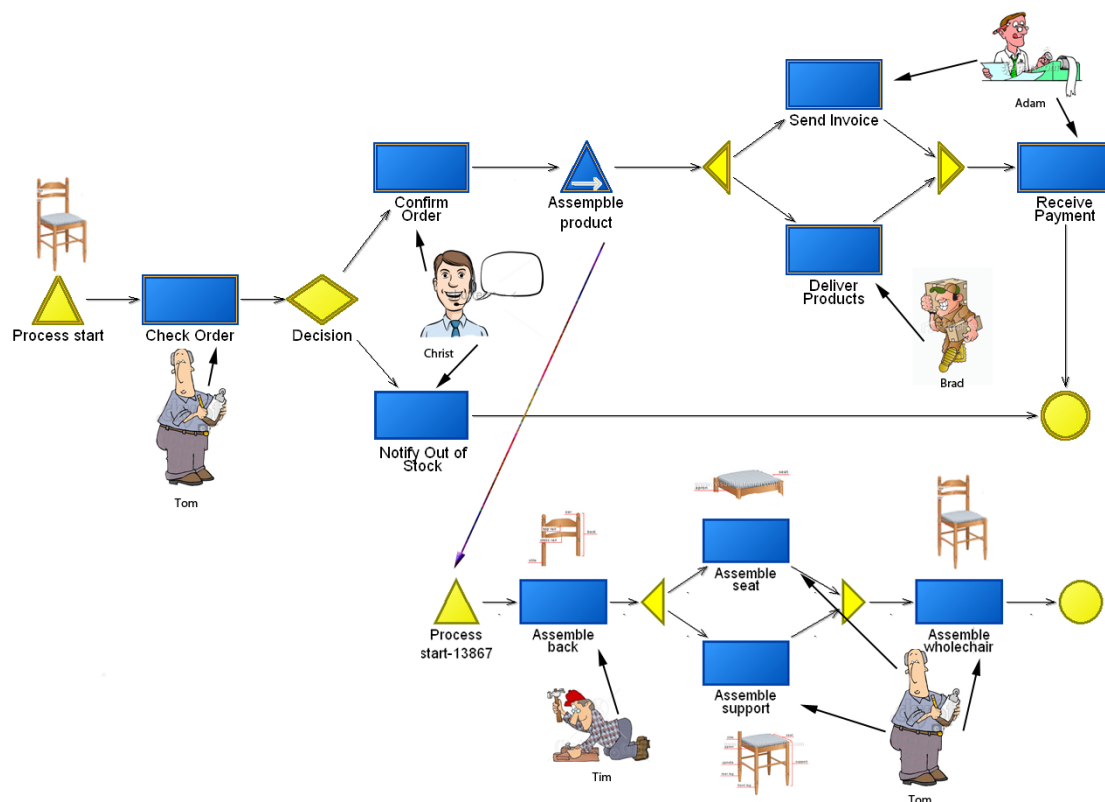


Figure 7.1: A use case of an order process

The order process described above takes place in a company and is executed by a Workflow management system. The generic architecture of Workflow management systems is represented in Chapter 2. In this section, the applications and the data sources related to the system are described to represent a concrete use case in which the implementation of the research can be executed and evaluated. In this case, the application involved in the order process operation is the Warehouse Control System. This application has the ability to manage the inventory and the consumption of items to assemble the company's products. Therefore, the Warehouse Control System has a database to maintain the data of

consumption resources used to assemble products. The amount of items used to assemble a chair and the cost of these items are managed by the system. Assume that it is possible to access the database of the systems, the essential data tables as shown in Figure 4.14 can be obtained by several filtering steps. The filter is out of scope of the thesis. Besides, the human factor plays an important role in the process operation. The characters Chris, Tom, Tim, Adam, Amy in the use case, who are the employees of the company, interact with the Workflow Management System to accomplish the business processes. Therefore, an application to manage the human resource is indispensable in the company. The application is able to maintain the personal information of the employee as well as their work experience. Another assumption is made for the existence of a database of the application which maintains the data of employees and the organization structure of the company. The database could cover all the perspectives of the company's organization management. However, to be suitable with the context of this research, the database is simplified as its schema shown in Figure 4.13. Basically, the organization data source supports the information of employees and the department/division of the company to which they belong. Beside the data sources obtained from the applications involved in the Workflow management system, there is event log data generated from the system itself. In this use case, the event log describes the process operation as shown in Figure 7.1.

Assume that Mr. Franklin is a manager of a company and he has the data sources. He wants to know the business process operation of the company where the data is from. For instance, what is the process model operating in the company? Which activities are happening in the company? When he knows the process model as shown in Figure 7.1 without the participation of humans, he wants to know who performs the activity Check Order? Who works with whom to accomplish an order? Which resources are used to assemble a product? How much do the resources cost to assemble a product? Which activities use the lowest price for resources? Which team works effectively?. Answering these kinds of questions can acquaint Mr. Franklin with the information of the company's business process operation. This is the case study of this research. The approach of this research could deal with the data source to produce the knowledge hidden in the data. Mr. Franklin would be provided or suggested questions and answers of the questions to have a comprehension about the business process operation of the company. The research is evaluated by the results gained from the implementation by testing with the case study.

The approach and the implementation of this research is represented in the previous chapters. In this chapter, the results of the approach are described with the use case above to evaluate the research.

7.2 Questions and Answers

The very first question can be raised by Mr Franklin is what is the process model of the company. To answer that question, the event log data is input to ProM, in particular, the Alpha algorithm plug-in in ProM, to get the process model as shown in Figure 5.11.

From the model, the questions are categorized into four perspectives: process operation,

organization structure, resource consumption, and operation costs. The following shows the questions in each category, the queries used to answer the questions and the results obtained from the knowledge base.

Process operation

Look at the process model, the activities are the significant objects which can have questions.

1. Who performs the activity "assemble support"?

Query: *?activity[has_originator hasValue ?organizationagent] memberOf activity and ?activity[has_name hasValue "assemble support"]*.

Result: Tom and Tim

2. Which process instances are successful in the way the order is accomplished with a product sold? How many of them?

Query: *?name memberOf success_case_name*.

Result: case 1, case 2, case 9. 3 cases are failed.

3. Which activities participate in successful cases?

Query: *?activity memberOf activity_belong_successcase*.

Result: check order, deliver products, assemble support, confirm order, assemble seat, assemble back, assemble whole chair, send invoice, receive payment.

4. Which process instances are failed? How many of them?

Query: *?name memberOf fail_case_name*.

Result: case 3, case 4. 2 cases are failed.

5. Which activities participate in failed cases?

Query: *?activity memberOf activity_belong_failcase*.

Result: check order, notify out of stock.

6. Which activities participate in a particular case, for example "case 1"?

Query: *?event[has_activity hasValue ?activity] memberOf event and ?event[has_name hasValue "case 1"] and ?event[has_status hasValue "end"]*.

Result: check order, confirm order, assemble back, assemble seat, assemble support, deliver products, send invoice, receive payment.

7. Which activities need to be performed by highly-experienced employees?

Query: *?activity memberOf activity_need_high_experience*.

Result: assemble seat, receive payment, send invoice, check order, assemble support.

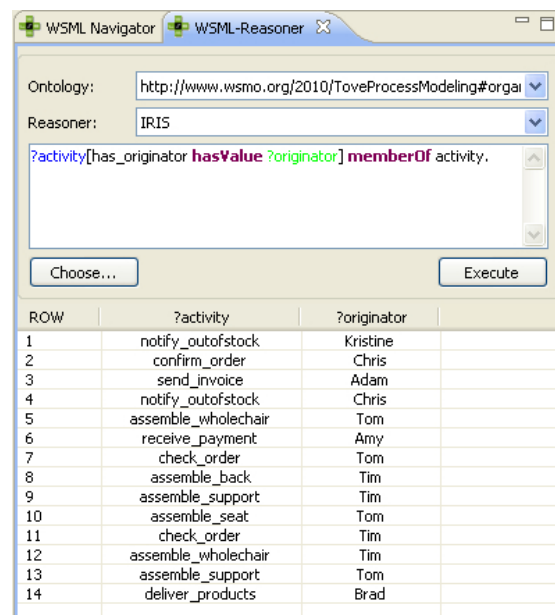
Organization structure

The second perspective of process operation is by who the activities performed. The performers who operate the activities are the members of the organization, or employees of the company. The questions related to the performance could be:

1. Who performs which activity?

Query: *?activity[has_organizer hasValue ?organizationagent] memberOf activity.*

Result: Figure 7.2



The screenshot shows the WSM Reasoner window with the following configuration:

- Ontology: <http://www.wsmo.org/2010/ToveProcessModeling#orga>
- Reasoner: IRIS
- Query: *?activity[has_organizer hasValue ?organizationagent] memberOf activity.*

The results are displayed in a table with the following data:

ROW	?activity	?organizer
1	notify_outofstock	Kristine
2	confirm_order	Chris
3	send_invoice	Adam
4	notify_outofstock	Chris
5	assemble_wholechair	Tom
6	receive_payment	Amy
7	check_order	Tom
8	assemble_back	Tim
9	assemble_support	Tim
10	assemble_seat	Tom
11	check_order	Tim
12	assemble_wholechair	Tim
13	assemble_support	Tom
14	deliver_products	Brad

Figure 7.2: Originators perform activities respectively in the same row of the result table

2. Who performs activity "assemble support"?

Query: *?activity[has_organizer hasValue ?organizationagent] memberOf activity and ?activity[has_name hasValue "assemble support"].*

Result: Tom and Tim

3. Which activities are performed by "Tom"?

Query: *?activity[has_organizer hasValue ?organizationagent] memberOf activity and ?organizationagent[has_name hasValue "Tom"].*

Result: check order, assemble support, assemble seat, assemble whole chair

4. Who works in which division?

Query: *?division[has_member hasValue ?organizationagent] memberOf division.*

Result: Figure 7.3

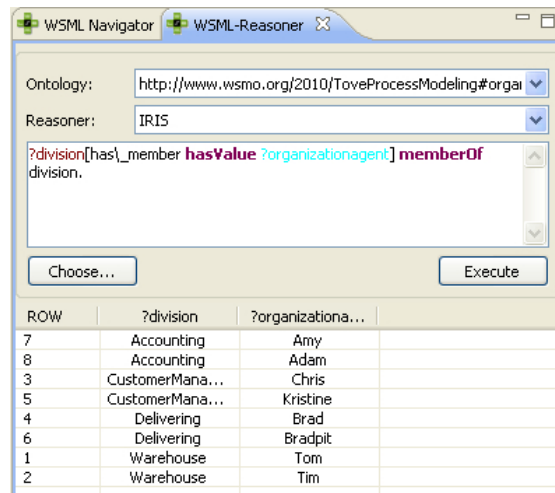


Figure 7.3: The answer is as shown in the result table

- Who works in the "Accounting Division"?

Query: *?organizationagent memberOf Accounting_division.*

Result: Adam and Amy

- Who has much work experience?

Query: *?organizationagent memberOf teamHighExperience.*

Result: Adam, Amy and Tom

- Who works together in "team 1"?

Query: *?organizationagent memberOf team1.*

Result: Adam, Amy, Brad, Chris, Tim and Tom

- Who works together in a successful team?

Query: *?organizationagent memberOf teamSuccess.*

Result: Adam, Amy, Brad, Chris, Tim and Tom

- Who works together in a failed team?

Query: *?organizationagent memberOf teamFail.*

Result: Chris, Kristine, Tim and Tom

- Does "Tim" work in a successful team?

Query: *?organizationagent memberOf teamSuccess and ?organizationagent[has_name hasValue "Tim"] memberOf organization_agent.*

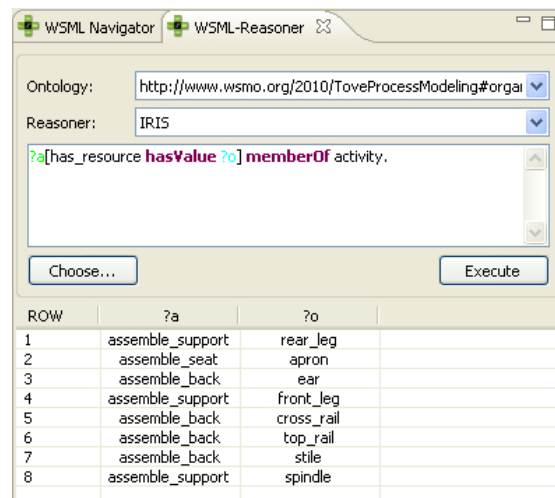
Result: Yes

Resource consumption

- Which resources are used for which activities?

Query: *?activity[has_resource hasValue ?organizationagent] memberOf activity.*

Result: Figure 7.4



The screenshot shows the WSML-Reasoner window. The 'Ontology' field is set to 'http://www.wsmo.org/2010/ToveProcessModeling#orgai' and the 'Reasoner' is 'IRIS'. The query entered is '?a[has_resource hasValue ?o] memberOf activity.'. Below the query area are 'Choose...' and 'Execute' buttons. The results are displayed in a table with columns 'ROW', '?a', and '?o'.

ROW	?a	?o
1	assemble_support	rear_leg
2	assemble_seat	apron
3	assemble_back	ear
4	assemble_support	front_leg
5	assemble_back	cross_rail
6	assemble_back	top_rail
7	assemble_back	stile
8	assemble_support	spindle

Figure 7.4: The answer is as shown in the result table

- Which resources are used for activity "assemble back"?

Query: *?activity[has_resource hasValue ?organizationagent] memberOf activity and ?activity[has_name hasValue "assemble back"].*

Result: ear, cross rail, top rail, stile.

- Which resources are used by a SuccessTeam?

Query: *?resource memberOf resourceSuccessTeam.*

Result: apron, cross rail, ear, front leg, rear leg, spindle, stile, top rail.

- Which resources are used by "Tim"?

Query: *?activity[has_originator hasValue ?originator] memberOf activity and ?activity[has_resource hasValue ?resource] memberOf activity and ?originator[has_name hasValue "Tim"].*

Result: stile, top rail, cross rail, ear, rear leg, front leg, spindle.

- How many items of resources are used for events?

Query: As shown in Figure 7.5

Result: 57

- How many items of resource "stile" are used?

Query: As shown in Figure 7.6

Result: 6

ROW	?event	?resource	?ur	?q
2	event301	stile	use4	2
4	event301	top_rail	use2	1
6	event301	ear	use1	2
16	event301	cross_rail	use3	1
10	event302	top_rail	use2	1
8	event302	ear	use1	2
17	event302	stile	use4	2
19	event302	cross_rail	use3	1
12	event309	stile	use4	2
20	event309	cross_rail	use3	1
22	event309	top_rail	use2	1
24	event309	ear	use1	2
15	event401	apron	use5	1
18	event402	apron	use5	1
23	event409	apron	use5	1
7	event501	rear_leg	use7	4
13	event501	spindle	use6	4
21	event501	front_leg	use8	4
5	event502	rear_leg	use7	4
11	event502	spindle	use6	4
14	event502	front_leg	use8	4
3	event509	rear_leg	use7	4
9	event509	spindle	use6	4
1	event509	front_leg	use8	4

Figure 7.5: The ?q column is the quantity of the resource items

ROW	?event	?resource	?ur	?q
2	event301	stile	use4	2
3	event302	stile	use4	2
1	event309	stile	use4	2

Figure 7.6: The ?q column is the quantity of the resource "stile" used.

Operation costs

1. How much does the labor cost for performing the activity "assemble support"?

Query: As shown in Figure 7.7

Result: 60

2. How much does the labor cost of the performer "Tim"?

Query: *?originator[has_labor_fee hasValue ?cost] memberOf organization_agent and ?originator[has_name hasValue "Tim"]*.

The screenshot shows the WSML-Reasoner window with the following configuration:

- Ontology: `http://www.wsmo.org/2010/ToveProcessModeling#orgai`
- Reasoner: IRIS
- Query: `?event[has_activity hasValue ?activity] and ?event[has_cost_labor hasValue ?laborcost] and ?activity[has_name hasValue "assemble support"]`

The results table is as follows:

ROW	?event	?activity	?laborcost
6	event501	assemble_support	8
4	event502	assemble_support	8
3	event509	assemble_support	14
5	event511	assemble_support	8
2	event512	assemble_support	8
1	event519	assemble_support	14

Figure 7.7: The ?laborcost column is the labor cost values for the events

Result: 8

- How much does the resource cost for performing activity "assemble support"?

Query: As shown in Figure 7.8

Result: 76

The screenshot shows the WSML-Reasoner window with the following configuration:

- Ontology: `http://www.wsmo.org/2010/ToveProcessModeling#orgai`
- Reasoner: IRIS
- Query: `?activity[has_name hasValue "assemble support"] and ?activity[has_resource hasValue ?resource] and ?resource[has_cost hasValue ?cost]`

The results table is as follows:

ROW	?activity	?resource	?cost
1	assemble_support	front_leg	33
2	assemble_support	spindle	21
3	assemble_support	rear_leg	22

Figure 7.8: The result is the sum of the numbers in the column ?cost

- How much does the resource cost for performing "case 1"?

Query: As shown in Figure 7.9

Result: 269

- How much does the resource cost for performing successful cases?

Query: As shown in Figure 7.10

Result: 807

WSML Navigator WSML-Reasoner

Ontology: <http://www.wsmo.org/2010/ToveProcessModeling#orgai>

Reasoner: IRIS

Query: `?event[has_name hasValue "case 1"] and ?event[has_cost_resource hasValue ?cost]`

Buttons: Choose... Execute

ROW	?event	?cost
4	event301	45
5	event301	12
7	event301	56
8	event301	46
6	event401	34
1	event501	22
2	event501	33
3	event501	21

Figure 7.9: The result is the sum of the numbers in the column ?cost

WSML Navigator WSML-Reasoner

Ontology: <http://www.wsmo.org/2010/ToveProcessModeling#orgai>

Reasoner: IRIS

Query: `?event[has_name hasValue ?name] and ?name memberOf success_case_name and ?event[has_cost_resource hasValue ?cost].`

Buttons: Choose... Execute

ROW	?event	?name	?cost
2	event301	"case 1"	45
8	event301	"case 1"	56
11	event501	"case 1"	22
13	event501	"case 1"	21
18	event401	"case 1"	34
21	event301	"case 1"	46
22	event301	"case 1"	12
23	event501	"case 1"	33
4	event502	"case 2"	33
6	event302	"case 2"	45
12	event302	"case 2"	46
24	event302	"case 2"	12
20	event502	"case 2"	22
15	event302	"case 2"	56
16	event502	"case 2"	21
19	event402	"case 2"	34
1	event509	"case 9"	21
3	event309	"case 9"	12
5	event309	"case 9"	46
7	event409	"case 9"	34
9	event509	"case 9"	33
14	event309	"case 9"	45
17	event509	"case 9"	22
10	event309	"case 9"	56

Figure 7.10: The result is the sum of the numbers in the column ?cost

6. How much does the labor cost for performing success cases?

Query: As shown in Figure 7.11

Result: Figure 7.11

The screenshot shows the WSM Reasoner window. The 'Ontology' field is set to 'http://www.wsmo.org/2010/ToveProcessModeling#orgai'. The 'Reasoner' field is set to 'IRIS'. The query is: `?event[has_name hasValue ?name] and ?name memberOf success_case_name and ?event[has_cost_labor hasValue ?cost]`. Below the query are 'Choose...' and 'Execute' buttons. The results are displayed in a table with columns: ROW, ?event, ?name, and ?cost.

ROW	?event	?name	?cost
2	event501	"case 1"	8
4	event711	"case 1"	12
6	event811	"case 1"	13
8	event401	"case 1"	14
11	event201	"case 1"	9
12	event301	"case 1"	8
15	event701	"case 1"	12
17	event511	"case 1"	8
21	event801	"case 1"	13
25	event211	"case 1"	9
27	event311	"case 1"	8
32	event901	"case 1"	13
37	event601	"case 1"	8
40	event111	"case 1"	8
47	event411	"case 1"	14
48	event911	"case 1"	13
51	event611	"case 1"	8
54	event101	"case 1"	8
14	event712	"case 2"	12
22	event502	"case 2"	8
26	event402	"case 2"	14
3	event812	"case 2"	13
13	event202	"case 2"	9
16	event512	"case 2"	8
5	event702	"case 2"	12
19	event802	"case 2"	13
7	event212	"case 2"	9
29	event312	"case 2"	8
1	event302	"case 2"	8
33	event902	"case 2"	13
39	event412	"case 2"	14
34	event602	"case 2"	8
49	event112	"case 2"	8
53	event612	"case 2"	8
50	event912	"case 2"	13
24	event102	"case 2"	8
43	event119	"case 9"	14
9	event519	"case 9"	14
10	event109	"case 9"	14
18	event609	"case 9"	14
20	event809	"case 9"	13
23	event719	"case 9"	12
52	event819	"case 9"	13
28	event909	"case 9"	13
30	event309	"case 9"	8
31	event209	"case 9"	9
35	event419	"case 9"	14
38	event709	"case 9"	12
36	event919	"case 9"	13
42	event219	"case 9"	9
41	event619	"case 9"	14
44	event409	"case 9"	14
45	event319	"case 9"	8
46	event509	"case 9"	14

Figure 7.11: The result is the sum of the numbers in the column ?cost

7. How much does the labor cost for highly-experienced team?

Query: As shown in Figure 7.12

Result: 40

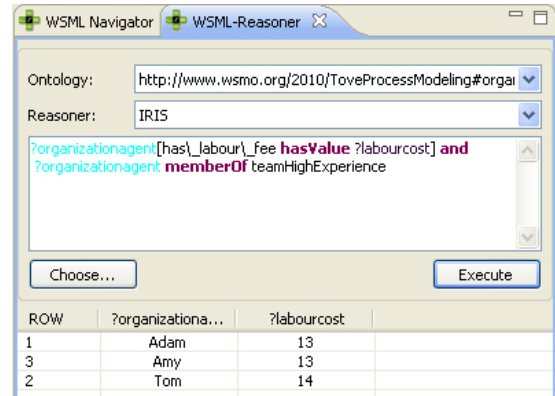


Figure 7.12: The result is the sum of the numbers in the column ?cost

8. Which activities have high costs?

Query: `?a memberOf activity_is_high_cost`

Result: assemble support

7.3 Evaluation

The current approach is evaluated by comparing the above results with the process mining and semantic process mining approach. Consider the question number 6 in the process operation part. "What activities need to be performed by highly-experienced employees?". The axiom *activity_need_high_experience* is defined to answer the question as shown in Figure 6.7. The axiom implies that an activity needs high experience when it is performed by a performer who belongs to the team *teamHighExperience*. The team *teamHighExperience* is defined by the axiom shown in Figure 6.6. A team is considered a *teamHighExperience* if it contains people who have more than 5 experience years. Therefore, to answer the question "What activities need to be performed by highly-experienced employees?", the reasoning is executed over the two axioms. To get the convenient observation, the questions and the related axioms are combined in Figure 7.13.

The analysis shows that the question is answered not only by querying but also reasoning. Moreover, the question is not able to be answered by process mining or semantic process mining because the two approaches do not contain the reasoning capability as shown above and the limitation of data sources.

```

axiom activity_need_high_experience
    definedBy
        ?a memberOf activity_need_high_experience
impliedBy
    ?a[has_organator hasValue ?o] memberOf activity
        and ?o memberOf teamHighExperience.

axiom teamHighExperience
    definedBy
        ?o memberOf teamHighExperience
impliedBy
    ?o[has_name hasValue ?name] memberOf organization_agent
        and ?o[has_experience_year hasValue ?t]
        and ?t > 5.

```

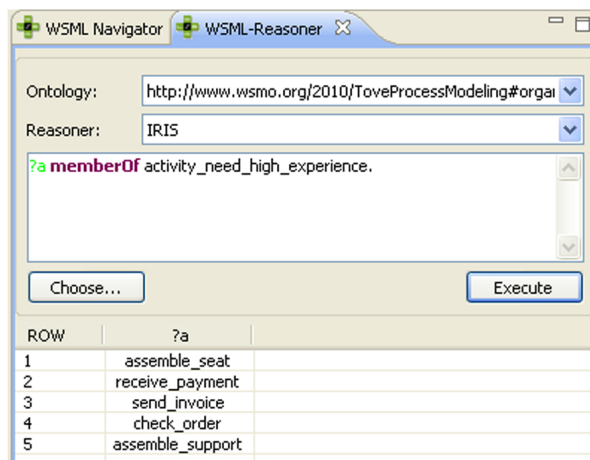


Figure 7.13: Querying and Reasoning

Conclusion and open issues

8.1 Summary

The research presented describes an approach for enriching information contained in event log data for answering questions related to business process operations. This allows answering the research question *"How can one combine existing approaches and extend their capabilities with the ultimate goal of providing an alternative solution for answering business-relevant questions?"* Therefore, the approach proposed integrates event log with relevant data sources utilizing and extending the ModTOVE ontology. The final result of the approach is the ModTOVE knowledge base serving as a basis for answering business-related questions. The evaluation of the research presented is carried out by conducting a case study and comparing the results with results of other relevant approaches. Finally, this chapter describes open issues for future works.

8.2 Contributions

The proposed approach takes advantage of ontology-based data integration to enhance the results delivered by semantic process mining. The detailed contributions are discussed in the following.

- **Enrich event log data.** The information that can be extracted by applying process mining on event logs is limited to the actual information contained in event logs. In other words, the information contained in event logs is still limited because of the dependency of the log data to the exporting systems. Applying semantic process mining aims at raising the semantic level by connecting the labels of data elements contained log data to corresponding concepts in ontologies. Although raising the level of abstraction, the scope of the information available is not sufficient for answering business-related questions. Therefore, the first contribution aims at identifying additional data sources which are beyond the scope of information that may

be extracted by solely analyzing event log data. For overcoming the heterogeneity of data sources, known concepts from ontology-based data integration are applied. In particular, this contribution derives the ModTOVE knowledge base by integrating log data with other data sources. To summarize, this contribution enables the integration of log data with additional information which allows extending the scope of potential business-related questions.

- **Extend the knowledge base.** For extending the scope of business-related questions that may be answered based on the ModTOVE knowledge base is extended with a set of axioms specifying additional information, resulting the in the second contribution of this thesis. In particular, the set of axioms eases reasoning and querying of the knowledge base. Having the extended knowledge base at hand allows querying the knowledge base for specific business-relevant questions beyond the scope of questions that may be answered by the application of semantic process mining only.

Commercial benefits

The goal of the research presented is taking full advantage of the data sources scattered in an enterprise system. The approach proposed in this research provides a company with the knowledge hidden in such scattered data sources. The knowledge relates to process operations of a company, thus it supports an internal observation of what really happens within a company.

The knowledge provided by this research could support business operations of a company in terms of understand what is happening in the company. The business-related questions provided by the approach cover all the perspectives of an organization and provide an overview of as well as insight into the company.

8.3 Limitations and Future Works

The research presented has several limitations which are subject to future work.

- The current approach utilizing the ModTOVE ontology is able to reduce the heterogeneity problem from a variety of data sources. However, the approach is currently not flexible enough for easily accommodating additional data sources. For this reason, the approach should be improved for being flexible enough for changing data sources on demand.
- The ontology-based data integration process, as utilized in the approach presented, currently requires the manual definition of mappings. Furthermore, users defining such mappings need to have knowledge on the domains that the different data sources stem from for successfully performing such mappings. As a next step, mapping concepts as part of the data integration should be done automatically or semi-automatically.

- Currently the case study of has been conducted on simulated data. In a consecutive step, data from real-world companies could be collected. Doing so would enable demonstrating the commercial benefits of the approach developed to organizations.

Bibliography

- [1] W M P Van Der Aalst and B F Van Dongen. Prom : The process mining toolkit. *Industrial Engineering*, 489:14, 2009.
- [2] Wil Van Der Aalst, Kees Van Hee, Prof. Dr. Kees, Max Hee, Remmert Remmerts De Vries, Jaap Rigter, Eric Verbeek, and Marc Voorhoeve. Workflow management: Models, methods, and systems, 2002.
- [3] Arya Adriansyah, Boudewijn F. van Dongen, and Wil M. P. van der Aalst. Towards robust conformance checking. In Michael zur Muehlen and Jianwen Su, editors, *Business Process Management Workshops*, volume 66 of *Lecture Notes in Business Information Processing*, pages 122–133. Springer, 2010.
- [4] Darko Anicic. Process ontology reasoner. <http://www.ip-super.org/>, 2007. SUPER Project, Deliverable 6.3.
- [5] Yigal Arens, Chin Chee, Chun nan Hsu, Hoh In, and Craig A. Knoblock. Query processing in an information mediator. In *In Proceedings of the ARPA/Rome Lab*, 1994.
- [6] Jess Barrasa, car Corcho, and Asuncimez-pz. R2o, an extensible and semantically based database-to-ontology mapping language. In *In Proceedings of the 2nd Workshop on Semantic Web and Databases(SWDB2004)*.
- [7] Christian Bizer. D2r map - a database to rdf mapping language. In *WWW (Posters)*, 2003.
- [8] Carmen Bratosin, Natalia Sidorova, and Wil M. P. van der Aalst. Discovering process models with genetic algorithms using sampling. In Rossitza Setchi, Ivan Jordanov, Robert J. Howlett, and Lakhmi C. Jain, editors, *KES (I)*, volume 6276 of *Lecture Notes in Computer Science*, pages 41–50. Springer, 2010.
- [9] J.C.A.M. Buijs. Mapping data sources to xes in a generic way. Master’s thesis, Eindhoven University of Technology, 2010.
- [10] Ana Karla A. de Medeiros, A. J. M. M. Weijters, and Wil M. P. van der Aalst. Genetic process mining: A basic approach and its challenges. In Christoph Bussler and Armin Haller, editors, *Business Process Management Workshops*, volume 3812, pages 203–215, 2005.

- [11] Ana Karla Alves de Medeiros. Process instance analysis environment. <http://www.ip-super.org/>, 2007. SUPER Project, Deliverable 5.2.
- [12] Ana Karla Alves de Medeiros. Semantic process mining prototype. <http://www.ip-super.org/>, 2007. SUPER Project, Deliverable 6.5.
- [13] Ana Karla Alves de Medeiros, Carlos Pedrinaci, Wil M. P. van der Aalst, John Domingue, Minseok Song, Anne Rozinat, Barry Norton, and Liliana Cabral. An outlook on semantic business process mining and monitoring. In Robert Meersman, Zahir Tari, and Pilar Herrero, editors, *OTM Workshops (2)*, volume 4806 of *Lecture Notes in Computer Science*, pages 1244–1255. Springer, 2007.
- [14] Ana Karla Alves de Medeiros, Wil M. P. van der Aalst, and Carlos Pedrinaci. Semantic process mining tools: Core building blocks. In Willie Golden, Thomas Acton, Kieran Conboy, Hans van der Heijden, and Virpi Kristiina Tuunainen, editors, *ECIS*, pages 1953–1964, 2008.
- [15] Marlon Dumas, Wil M. van der Aalst, and Arthur H. ter Hofstede. *Process-aware information systems: bridging people and software through process technology*. John Wiley & Sons, Inc., New York, NY, USA, 2005.
- [16] Marlon Dumas, Wil M. van der Aalst, and Arthur H. ter Hofstede. *Workflow market: A global perspective for 2000*. John Wiley & Sons, Inc., New York, NY, USA, 2005.
- [17] Eclipse. The eclipse foundation open source community website. <http://www.eclipse.org/>. Retrieved: January, 2012.
- [18] F G Fadel. A resource ontology for enterprise modelling. *Third Workshop on Enabling Technologies Infrastructures for Collaborative Enterprises*, 1994.
- [19] Mark S. Fox, Mihai Barbuceanu, Michael Grüninger, and Jinxin Lin. An organization ontology for enterprise modelling. In *Modeling, In: International Conference on Enterprise Integration Modelling Technology 97*. Springer, 1997.
- [20] Mark S. Fox and Michael Grüninger. Enterprise modeling. *AI Magazine*, 19(3):109–121, 1998.
- [21] Michel Gagnon. Ontology-based integration of data sources. *2007 10th International Conference on Information Fusion*, pages 1–8, 2007.
- [22] Raji Ghawi and Nadine Cullot. Database-to-Ontology Mapping Generation for Semantic Interoperability. In *Third International Workshop on Database Interoperability (InterDB 2007)*, 2007.
- [23] BOC Group. Boc group: Adonis - business process management. <http://www.boc-group.com/products/adonis/>. Retrieved: January, 2012.
- [24] BOC Group. Market research reports - business market research reports & industry analysis. <http://www.marketresearch.com/Infiniti-Research-Limited-v2680/Workflow-Management-Systems-6169512/>. Retrieved: January, 2012.

- [25] Thomas R. Gruber. A translation approach to portable ontology specifications. *Knowl. Acquis.*, 5(2):199–220, June 1993.
- [26] Michael Grüninger, Katy Atefi, and Mark S. Fox. Ontologies to support process integration in enterprise engineering. *Comput. Math. Organ. Theory*, 6(4):381–394, 2000.
- [27] Michael Grüninger and Mark S. Fox. An activity ontology for enterprise modelling. In *Submitted to: Workshop on Enabling Technologies - Infrastructures for Collaborative Enterprises*, West Virginia University, 1994.
- [28] Michael Grüninger and Mark S. Fox. The role of competency questions in enterprise engineering. In *Proceedings of the IFIP WG5.7 Workshop on Benchmarking - Theory and Practice*, 1994.
- [29] Paul Harmon. The bptrends 2010 bpm software tools report on boc’s adonis version 4.0, 2010. Vienna, Austria.
- [30] Alan R. Hevner, Salvatore T. March, Jinsoo Park, and Sudha Ram. Design science in information systems research. *MIS Quarterly*, 28(1):75–105, 2004.
- [31] Yannis Kalfoglou and W. Marco Schorlemmer. Information-flow-based ontology mapping. In Robert Meersman and Zahir Tari, editors, *CoopIS/DOA/ODBASE*, volume 2519 of *Lecture Notes in Computer Science*, pages 1132–1151. Springer, 2002.
- [32] Yannis Kalfoglou and W. Marco Schorlemmer. Ontology mapping: The state of the art. In Yannis Kalfoglou, W. Marco Schorlemmer, Amit P. Sheth, Steffen Staab, and Michael Uschold, editors, *Semantic Interoperability and Integration*, volume 04391 of *Dagstuhl Seminar Proceedings*. IBFI, Schloss Dagstuhl, Germany, 2005.
- [33] Mick Kerrigan and Adrian Mocan. The web service modeling toolkit. *Chaos*, pages 1–5, 2008.
- [34] Thanh Tran Thi Kim. An ontology based framework for enriching event log data. In *ISEMAPRO 2011, The Fifth International Conference on Advances in Semantic Processing*, pages 110 –115, Nov 2011. IARIA Conference.
- [35] Enterprise Integration Laboratory. Tove ontology project. <http://www.eil.utoronto.ca/enterprise-modelling/tove/>. Retrieved: August, 2011.
- [36] R. S. Mans, Helen Schonenberg, Minseok Song, Wil M. P. van der Aalst, and Piet J. M. Bakker. Application of process mining in healthcare - a case study in a dutch hospital. In *BIOSTEC (Selected Papers)*, pages 425–438, 2008.
- [37] R. S. Mans, M. H. Schonenberg, Minseok Song, Wil M. P. van der Aalst, and Piet J. M. Bakker. Process mining in healthcare - a case study. In *HEALTHINF (1)*, pages 118–125, 2008.
- [38] Ronny Mans, Helen Schonenberg, Giorgio Leonardi, Silvia Panzarasa, Anna Cavallini, Silvana Quaglini, and Wil M. P. van der Aalst. Process mining techniques: an

- application to stroke care. In Stig Kjær Andersen, Gunnar O. Klein, Stefan Schulz, and Jos Aarts, editors, *MIE*, volume 136 of *Studies in Health Technology and Informatics*, pages 573–578. IOS Press, 2008.
- [39] Eduardo Mena, Arantza Illarramendi, Vipul Kashyap, Amit Sheth, Interoperation Across Pre existing Ontologies, and Athman Bouguettaya. Observer: An approach for query processing in global information systems based on interoperation across pre-existing ontologies, 2000.
 - [40] Microsoft. The workflow engine model. [http://msdn.microsoft.com/en-us/library/aa188337\(office.10\).aspx](http://msdn.microsoft.com/en-us/library/aa188337(office.10).aspx). Retrieved: January, 2012.
 - [41] MySQL. Mysql:: The world’s most popular open source database. <http://www.mysql.com>. Retrieved: August, 2011.
 - [42] N. F. Noy and M. A. Musen. Smart: Automated support for ontology merging and alignment. In *Proceedings of the Twelfth Workshop on Knowledge Acquisition, Modeling and Management (KAW’99)*, Alberta, 1999.
 - [43] Natalya Fridman Noy and Mark A. Musen. Prompt: Algorithm and tool for automated ontology merging and alignment. In Henry A. Kautz and Bruce W. Porter, editors, *AAAI/IAAI*, pages 450–455. AAAI Press / The MIT Press, 2000.
 - [44] Alun Preece, Kit Hui, Alex Gray, Philippe Marti, Trevor Bench-capon, Dean Jones, and Zhan Cui. The kraft architecture for knowledge fusion and transformation, 1999.
 - [45] Eindhoven University of Technology Process Mining Group, Math&CS department. Process mining. <http://www.processmining.org/>. Retrieved: January, 2012.
 - [46] SUPER Project. Super integrated project. <http://www.ip-super.org/>. Retrieved: January, 2012.
 - [47] Liu Qiang. The design and implementation of process definition tool. In *Information Science and Engineering (ISISE), 2010 International Symposium on*, pages 283 – 286, dec. 2010.
 - [48] Anne Rozinat, R. S. Mans, Minseok Song, and Wil M. P. van der Aalst. Discovering colored petri nets from event logs. *STTT*, 10(1):57–74, 2008.
 - [49] Anne Rozinat and Wil M. P. van der Aalst. Conformance testing: Measuring the fit and appropriateness of event logs and process models. In Christoph Bussler and Armin Haller, editors, *Business Process Management Workshops*, volume 3812, pages 163–176, 2005.
 - [50] Anne Rozinat and Wil M. P. van der Aalst. Decision mining in prom. In Schahram Dustdar, José Luiz Fiadeiro, and Amit P. Sheth, editors, *Business Process Management*, volume 4102 of *Lecture Notes in Computer Science*, pages 420–425. Springer, 2006.
 - [51] Anne Rozinat and Wil M. P. van der Aalst. Conformance checking of processes based on monitoring real behavior. *Inf. Syst.*, 33(1):64–95, 2008.

- [52] Gerd Stumme and Alexander Maedche. Ontology merging for federated ontologies on the semantic web. In *In Proceedings of the International Workshop for Foundations of Models for Information Integration (FMII-2001)*.
- [53] D Tham, M S Fox, and M Grüninger. *A cost Ontology for Enterprise Modelling*, pages 197–210. IEEE Comput. Soc. Press, 1994.
- [54] Mike Uschold, Michael Gruninger, Mike Uschold, and Michael Gruninger. Ontologies: Principles, methods and applications. *Knowledge Engineering Review*, 11:93–136, 1996.
- [55] W. M. P. van der Aalst and A. J. M. M. Weijters. Process mining: a research agenda. *Comput. Ind.*, 53:231–244, April 2004.
- [56] Wil M. P. van der Aalst. Distributed process discovery and conformance checking. In Juan de Lara and Andrea Zisman, editors, *FASE*, volume 7212 of *Lecture Notes in Computer Science*, pages 1–25. Springer, 2012.
- [57] Wil M. P. van der Aalst, Hajo A. Reijers, and Minseok Song. Discovering social networks from event logs. *Computer Supported Cooperative Work*, 14(6):549–593, 2005.
- [58] Wil M. P. van der Aalst, Hajo A. Reijers, A. J. M. M. Weijters, Boudewijn F. van Dongen, Ana Karla Alves de Medeiros, Minseok Song, and H. M. W. (Eric) Verbeek. Business process mining: An industrial application. *Inf. Syst.*, 32(5):713–732, 2007.
- [59] W.M.P. van der Aalst, A.J.M.M. Weijter, and L. Maruster. Workflow mining: Discovering process models from event logs. *IEEE Transactions on Knowledge and Data Engineering*, 16:2004, 2003.
- [60] B. van Dongen, A. de Medeiros, H. Verbeek, A. Weijters, and W. van der Aalst. The prom framework: A new era in process mining tool support. In Gianfranco Ciardo and Philippe Darondeau, editors, *Applications and Theory of Petri Nets 2005*, volume 3536 of *Lecture Notes in Computer Science*, pages 1105–1116. Springer Berlin / Heidelberg, 2005.
- [61] H Wache, T Vgele, U Visser, H Stuckenschmidt, G Schuster, H Neumann, and S Hbner. *Ontology-based integration of information-a survey of existing approaches*, volume 2001, pages 108–117. Citeseer, 2001.
- [62] WfMC. Workflow management coalition terminology & glossary. *Management*, 39(3):1–65, 1999.
- [63] The ESSI WSMO working group. Web service modeling language. <http://www.wsmo.org/wsml/>. Retrieved: January, 2012.
- [64] Michael zur Mühlen. Process-driven management information systems - combining data warehouses and workflow technology. In B. Gavish, editor, *Fourth International Conference on Electronic Commerce Research*, pages 550–566, 2001. Dallas.