TNO Information and Communication Technology

Nederlandse Organisatie voor toegepast-natuurwetenschappelijk onderzoek / Netherlands Organisation for Applied Scientific Research

Enschede
Colosseum 27
7521 PV Enschede
Nederland

www.tno.nl

T +31 53 483 52 00
F +31 53 483 52 22

TNO report

Identifying and solving overlap between messaging standards with the help of domain models
illustrated with HR-XML's SIDES and SEP

Date	November 30, 2007
Author(s)	J.B.M. (Jasper) Roes BSc
Number of pages	113
Number of appendices	6

University of Twente
Enschede - The Netherlands

Master Thesis J.B.M. (Jasper) Roes BSc

Identifying and solving overlap between messaging standards with the help of domain models
illustrated with HR-XML's SIDES and SEP

Author	J.B.M. (Jasper) Roes BSc
Student number	s0021504
MSc programme	Computer Science
Track	Information Systems Engineering
Institute	University of Twente, Enschede, the Netherlands
Faculty	Electrical Engineering, Mathematics and Computer Science
Graduation company	TNO Information and Communication Technology Colosseum 27 7521 PV Enschede the Netherlands http://www.tno.nl/ict
Date	November 30, 2007
Graduation committee	prof. dr. R.J. Wieringa *faculty of Electrical Engineering, Mathematics and Computer Science*
	drs. L. Bodenstaff *faculty of Electrical Engineering, Mathematics and Computer Science*
	ir. D. Krukkert *TNO Information and Communication Technology*
	ir. E.J.A. Folmer *TNO Information and Communication Technology*

Abstract

In the Dutch staffing industry, the SIDES standard is widely used for the electronic exchange of information. Recent developments have indicated that in the nearby future the recruitment and matching processes may also be transformed from a complete paper process to an electronic process. Because of these developments, there is also interest in the SEP standard which can be used to exchange recruitment and matching information electronically. Therefore SEP may also be introduced in the Dutch staffing industry. SIDES and SEP have a certain overlap, which makes it unclear when to use what standard in certain cases. As TNO Information and Communication Technology performs work for the SETU (Stichting Elektronische Transacties Uitzendbranche / Foundation for electronic transactions in the staffing industry), which focuses on standards to be used in the Dutch Staffing Industry for the electronic exchange of information, it is important for TNO to know how SIDES and SEP overlap, and how this overlap can be handled for the SETU. The research presented in this paper looks into the entire standardisation process to find out how the overlap between SIDES and SEP can be identified and solved by TNO for the SETU.

All reasons for standardisation can be traced back to two main categories of reasons: technical and economical reasons. Technical because multiple companies want to have the possibility to easily exchange information, economic because standardisation can save money and the developing company can profit from standardisation. Standards can be divided into *de facto* standards, standards developed and maintained by the market, and *de jure* standards, standards developed and maintained by a formal institution or consortium. The *de jure* standard can be further split up into grey standards, maintained by consortia, and formal standards, maintained by a formal institution. To create good standards, TNO has created a definition of what a good electronic messaging standard should look like. A good electronic messaging standard should contain a context diagram that specifies which parties use the standard and a relationship diagram that specifies the relation between the objects (for example an ERD or a Class diagram). These two diagrams together form the static part. The standard should further contain a dynamic model that specifies the dynamic aspects of the standard (for example STDs, an event list or an object event table), a communication diagram that specifies how the objects interact (for example an Activity diagram and the messages) and a definition part that defines all objects and attributes. These models and diagrams together form the complete specification of an electronic messaging standard.

A problem with different standards that describe the same domain or parts of the same domain is that there is a risk of overlap between the standards. This overlap can make it difficult to choose which standard to use. Overlap between standards can be classified into three different types, the subset relation type, the equal relation type and the intersection relation type. Overlap and conflict between standards can be introduced through the institutional framework (if for example two institutions work on the same problem), the actors (if for example their interests or motives are different) and the technological foundation (if for example there is already a standard that should be incorporated in the new standard). Overlap can further be introduced in the decision-making process and in the final definitions of standards. Because standards are sometimes ambiguous, overlap and conflict can also be introduced in the implementation phase. Preventing overlap is possible if the standardisation institutions put effort into it.

The standardisation process that is currently used by many standardisation institutions seems to be missing one important part, the creation of a domain model on which the standard is based. A mapping to the Model Driven Architecture, a software developing

architecture, shows how the domain model can help in creating good standards. To create a domain model one needs a domain modelling technique. Three techniques, Merode, Fusion and NYAM, have been compared. Of these Merode is the most suitable technique to create domain models that are comparable and that TNO can use to identify overlap. Merode is therefore used in the research to identify and solve overlap between electronic messaging standards and to create domain models for SIDES and SEP. These domain models consist of a context diagram, an entity relationship diagram, a class diagram, an object event table and STDs for each class in the class diagram.

Identifying and solving overlap is important to create consistent and usable standards. To identify overlap one should look at the static part and the dynamic part of the created domain model. Each of these parts can independently indicate overlap and conflict. During the identification of overlap between two standards not only the models, but also the semantics of the models are important. The research presented in this paper identified eight methods for solving overlap:

1. Merge standards
2. Extend one standard and drop the other
3. Remove the overlap from one standard
4. Leave the standards as they are and create guidelines
5. Create a complete new standard
6. Remove the overlap from both standards and define a new standard for the gap
7. Define one standard prevailing
8. Define mappings

To choose the most appropriate solving method, two filters and five categories of suitability criteria were defined. The two filters are ‘type of overlap’ and ‘size of the overlapping part’ and these filters are used to do a first selection out of the eight possible solving methods. These filters result in six groups of solving methods that are possible for the combinations of filters. The five categories of suitability criteria that were defined all consist of one or more means to measure the suitability:

1. Number of resulting standards
2. Changes to be made in existing implementations
3. Satisfaction of current users with the overlapping standards
4. Political influences
5. Authority of the standardisation institution

The overlap of SIDES and SEP was in some way already clear when starting the research described in this thesis. That the overlap is of the intersection type became clear in the identification process based on the static and the dynamic part of the domain models of SIDES and SEP. While SEP at first seemed to be a subset of SIDES, it became very clear that the goals of the two standards are the same, but that the methods for reaching this are pretty different in both standards. Both the static and the dynamic part comparison showed the same type overlap, thereby reinforcing each other’s result. The best method for TNO to solve the overlap between SIDES and SEP for the SETU is to extend the SIDES standard with the extra parts of SEP. This keeps the changes to existing implementations as small as possible. To make the transition process from the old standards to the new standard TNO should define mappings from the old messages to the new messages.

Samenvatting

In de Nederlandse uitzendbranche wordt de SIDES standaard veelvuldig gebruik voor het elektronisch uitwisselen van informatie. Recente ontwikkelingen duiden erop dat in de toekomst ook het werven en matchen van kandidaten en vacatures meer en meer elektronisch uitgevoerd gaat worden. Door deze ontwikkelingen wordt er op dit moment door steeds meer partijen gekeken of de SEP standaard ook geïntroduceerd moet worden in de Nederlandse uitzendbranche. SIDES en SEP zijn twee standaarden die op een bepaalde manier overlappen wat het moeilijk maakt om voor één standaard te kiezen voor het uitwisselen van bepaalde informatie. Omdat TNO Informatie- en Communicatietechnologie om dit moment werkzaamheden uitvoert voor de SETU (de SETU houdt zich primair bezig met het gebruik van standaarden binnen de Nederlandse uitzendbranche) is het belangrijk voor TNO om te weten hoe SIDES en SEP precies overlappen en hoe deze overlap voor de SETU opgelost kan worden. Het onderzoek dat beschreven wordt in deze thesis bekijkt het hele standaardisatieproces om de overlap tussen SIDES en SEP te identificeren en om een advies te geven aan TNO hoe de overlap opgelost kan worden voor de SETU.

Er zijn een groot aantal redenen aan te wijzen waarom elektronische berichtenstandaarden gestandaardiseerd worden, maar uiteindelijk zijn deze allemaal terug te leiden tot twee hoofdcategorieën, namelijk technische en economische redenen. De technische redenen zijn gerelateerd aan het feit dat meerdere bedrijven een makkelijke manier willen om informatie uit te wisselen; de economische redenen liggen bijvoorbeeld in het besparen van geld door het elektronisch uitwisselen van informatie en het feit dat een bedrijf dat een standaard ontwikkelt winst kan halen uit deze standaard. Er zijn twee typen standaarden, namelijk *de facto* standaarden, deze worden ontwikkeld en onderhouden door de markt, en *de jure* standaarden, deze worden ontwikkeld en onderhouden door een formeel instituut of consortium. De *de jure* standaarden kunnen nog verder opgesplitst worden in grijze en formele standaarden. Grijze standaarden worden onderhouden door consortiums, formele standaarden worden onderhouden bij een formeel instituut. Om een goede standaard te creëren heeft TNO een definitie opgesteld van een goede elektronische berichtenstandaard. Een goede standaard moet een context diagram bevatten dat aangeeft welke partijen de standaard gebruiken. Er moet een relationship diagram aanwezig zijn dat de relaties tussen de objecten specificeert (bijvoorbeeld een ERD of een klassediagram). Deze twee diagrammen samen het statische gedeelte van het model. Verder moet er een dynamisch model aanwezig zijn dat de dynamische aspecten van de standaard definieert (bijvoorbeeld STD's, een event lijst of een object event tabel). Hiernaast moet er ook een communicatiediagram aanwezig zijn dat specificeert hoe de verschillende object interacteren (bijvoorbeeld een activity diagram en de berichten). Als laatste moet er ook een gegevenswoordenboek aanwezig zijn dat de definities van alle objecten en attributen opslaat. Al deze modellen en diagrammen samen vormen de complete specificatie van een standaard.

Een probleem dat voorkomt uit het feit dat er meerdere standaarden zijn die allemaal (een deel van) hetzelfde domein beschrijven is dat er een kans is op overlap tussen de standaarden. Deze overlap kan de keuze voor een standaard bemoeilijken. Overlap tussen standaarden kan onderverdeeld worden in drie types, het kan gebaseerd zijn op een subset relatie, een equal relatie en een intersection relatie. Overlap en conflicten tussen standaarden kunnen op een aantal moment geïntroduceerd worden. Overlap kan ontstaan binnen de standaardisatie instituten doordat twee instituten aan hetzelfde probleem werken, door de actoren omdat de interesses en motivaties kunnen verschillen

en door de technologische basis van de nieuwe standaard omdat er bijvoorbeeld een oude standaard meegenomen moet worden in de nieuwe standaard. Overlap kan ook geïntroduceerd worden in het beslissingsproces en bij het formaliseren van een nieuwe standaard. Zelfs in de implementatiefase van een standaard kan nog overlap ontstaan, deze overlap komt dan vaak voort uit het feit dat standaarden soms dubbelzinnig zijn.

In de standaardisatieprocessen die op dit moment worden uitgevoerd door de verschillende standaardisatie instituten, lijkt een belangrijk onderdeel te missen, namelijk het definiëren van een domeinmodel waarop de standaard gebaseerd kan worden. Door het standaardisatieproces af te beelden op de Model Driven Architecture, dat gebruikt wordt om software te ontwerpen, is getoond dat een domeinmodel belangrijk kan zijn in de ontwikkeling van een standaard. Drie technieken om een domeinmodel te specificeren zijn onderzocht: dit zijn de Merode, Fusion en NYAM technieken. Merode is de meest geschikte techniek om domeinmodellen te creëren die goed vergeleken kunnen worden. In het onderzoek wordt daarom Merode gebruikt om overlap te identificeren en op te lossen. Met de hulp van de Merode techniek zijn twee volledige domeinmodellen opgesteld die bestaan uit een context diagram, een entity relationship diagram, een klassendiagram, een object event tabel en een state transition diagram voor iedere klasse uit het klassendiagram.

Het identificeren en oplossen van overlap tussen standaarden is van belang om consistente en bruikbare standaarden te creëren. Om overlap te identificeren binnen standaarden moet er gekeken worden naar zowel het statische als het dynamische gedeelte van de domeinmodellen. De delen kunnen allebei losstaand gebruikt worden om overlap te identificeren. Tijdens het identificeren van overlap tussen twee standaarden zijn niet alleen de modellen belangrijk, ook de semantiek van de modellen is belangrijk. Het oplossen van overlap kan op een achttal manieren, deze manieren zijn:

1. Fuseer de standaarden
2. Breid één standaard uit en gooi de andere weg
3. Verwijder de overlap uit één standaard
4. Creëer richtlijnen en laat de standaarden zoals ze zijn
5. Creëer een volledige nieuwe standaard
6. Verwijder de overlap uit beide standaarden en creëer een nieuwe standaard voor het ontstane gat
7. Verklaar één standaard prevalerend over de andere
8. Definieer vertalingen van de ene naar de andere standaard

Om de meest geschikte manier te kiezen om overlap op te lossen zijn twee filters en vijf categorieën geschiktheidcriteria gedefinieerd. De twee filters zijn ‘type overlap’ en ‘grootte van het overlappende gedeelte’ en deze worden gebruikt om een eerste selectie te maken uit de acht mogelijke oplossingsmethoden. De filters resulteren in zes groepen van oplossingsmethoden, één groep voor iedere combinatie van filters. De vijf categorieën van geschiktheidcriteria die gedefinieerd zijn bevatten allemaal één of meerdere mogelijkheden om de geschiktheid te meten:

1. Het aantal resulterende standaarden
2. De veranderingen die nodig zijn in bestaande implementaties van de standaarden
3. De tevredenheid van de huidige gebruikers met de overlappende standaarden
4. Politieke invloeden
5. Het gezag van het standaardisatie instituut

Het feit dat er overlap is tussen de SIDES en SEP standaarden was de aanleiding voor het onderzoek dat in deze thesis beschreven wordt. Het onderzoek heeft aangetoond dat het vermoeden klopt en dat er sprake is van overlap van het intersection relation type. In eerste instantie leek het erop dat SEP een subset was van SIDES, maar het werd uiteindelijk heel duidelijk dat de doelen van de standaarden wel hetzelfde zijn, maar dat de methoden die gebruikt worden om dit doel te bereiken heel verschillend zijn. Zowel het statische gedeelte als het dynamische gedeelte van de domeinmodellen heeft hetzelfde type overlap opgeleverd, wat het type overlap nog meer bevestigt.

TNO kan de overlap tussen SIDES en SEP voor de SETU het beste oplossen door de SIDES standaard uit te breiden met de extra onderdelen uit SEP. Dit zorgt ervoor dat de wijzigingen binnen al bestaande implementaties zo klein mogelijk zijn. Om de overgang voor bestaande implementaties van de oude standaarden naar de nieuwe standaard zo gemakkelijk mogelijk te maken zou TNO ook vertalingen moeten definiëren van de oude berichten naar de nieuwe berichten.

Preface

This thesis concludes my study period at the University of Twente, a six year period which taught me quite a lot, fortunately not only in the field of Computer Science. With the start of my study, I moved from Terborg, the nicest little city in 'de Achterhoek' to 'big city' Enschede. The first four years of my study I lived at Calslaan 6, together with 19 housemates. That provided me with an environment that stimulated not only studying, but also other activities, ranging from playing football and playing darts, to organizing a 'flatfeest' every year. I would like to thank all my housemates from that period for providing me with a stimulating environment and ensuring that life not only consisted of studying. After these first four years I moved to an apartment in the city together with Chris Scheffers, a housemate from Calslaan 6 where I am living throughout the rest of my study. I want to thank him for his support and friendship during these years.

The university not only provided me with the opportunity to study Computer Science, after three and a halve year of study it also provided me with the opportunity to put my study on a halt for one year to join the board of the Sports Council. This year has given me an unforgettable experience and taught me many valuable things that I could have never gained by only following courses. I want to thank Jeroen, Marc, Barry, Erik and Benjamin for this unforgettable year. This addition to my study extended my study period with one year, but also gave me the opportunity to join the 'Kryptos' study tour to the United States to perform research in the field of IT security. This also was a great experience.

After returning from the study tour in the United States I started with my internship at Goed Wonen in Twello, in which I advised on and implemented a project- and risk management system. I would like to thank all my colleagues at Goed Wonen for providing me with this opportunity, and specially my daily supervisor Jan Roders for ensuring that my internship was great.

Upon finishing my internship at Goed Wonen, my graduation process at TNO Information and Communication Technology started. My colleagues at TNO provided an ideal environment to perform this research, not only with interesting discussions, but also by playing poker during breaks to put my mind at something different. Special thanks go out to Dennis Krukkert and Erwin Folmer who provided useful feedback and comments and offered me freedom to direct my own research. I also want to thank Menno Holtkamp for promising me cake when finishing this preface; it sure sped up the process!

Support from the university came from prof. dr. Roel Wieringa and drs. Lianne Bodenstaff who, even though they unfortunately missed the first part of my research, provided me with constructive comments to improve my research; thank you for that.

Last, but certainly not least, I want to thank by parents and my sister for supporting me throughout my study and for always believing in me. Without them I would never have come this far.

Enschede, November 30, 2007

Jasper Roes

“Life’s like a box of chocolates.
You never know what you're gonna get.”

- Forrest Gump, 1994

Contents

1 Introduction

This chapter presents the context, the research objectives, the scope and the approach of the performed research. After describing the background and motivation, the research objectives and the research approach are presented which are followed by the outline of this thesis.

1.1 Background

Standardisation in the Information and Communication Technology (ICT) area has always been a field of interest. Standards were regarded important to make interoperability possible and can therefore be seen as an enabler of electronic business-to-business (B2B) commerce. Furthermore the setting of, or at least influencing, standards has become one of the core strategic challenges in the information technology area. The main reason for this is that the competition takes a 'winner takes all' form. A firm that establishes a technical standard can therefore receive large returns, whereas competitors might effectively be locked out of the market [Fomin & Keil, 2000].

In the staffing industry, large amounts of data are collected and exchanged between staffing companies and staffing customers. This exchange of data has been dominated by paper exchange, in which employees had to fill in forms, which later would be digitized by hand. One problem of this approach is that faults can be made in every step. Because of the manual translation of paper forms to digital data there is a chance that extra work has to be done to correct errors. Furthermore, all of this paperwork requires extra labour, thus generating extra costs and possible higher workloads.

To reduce this extra work and to be able to work quicker and with less cost, a number of leading companies within the staffing industry began to work on a standard to improve the overall integration process. In this case, the staffing industry companies worked together to provide one standard and decided not to compete each other with numerous standards. This of course is related to the market at hand. There are quite a lot of staffing companies and their customers can very easily switch between them. If the companies would all create their own standards, they might prevent their own customers from switching to another company, but they might also prevent other customers from switching to them. It was therefore desirable to create one standard, to enable the market to continue working as before. The result of this work was the SIDES (Staffing Industry Data Exchange Standards) standard, which was donated to the HR-XML consortium. Next to the SIDES standard, the HR-XML consortium also worked on the SEP standard (Staffing Exchange Protocol) which was primarily focussed on the entire process of matching and exchanging information needed to come to an assignment. Both standards are formalized by the HR-XML consortium and are actively used all over the world.

1.2 Motivation

Large messaging standards often consist of numerous partial standards. Each of these partial standards consists of a certain number of messages. When developing (partial) standards, the most common approach is to translate an existing messaging structure to an electronic equivalent, without first defining an appropriate domain model. This working method often results in problems with overlapping messages between standards, for example the overlap between SIDES and SEP.

In the Dutch staffing industry, the SIDES standard is widely used. Recent developments have indicated that in the nearby future the recruitment and matching processes may also be transformed from a complete paper process to an electronic process. Because of these developments, there is interest in the SEP standard and SEP may also be introduced in the Dutch staffing industry. While SIDES focuses on back office processes concerning staffing, timecards and invoices, but also incorporates the matching process, SEP is fully focused on supplying (and matching) job opportunities and candidates.

SIDES and SEP have a certain overlap, which makes it unclear when to use which standard. For both SIDES and SEP domain models have not been created, which makes it difficult to identify the overlap and makes it difficult to judge what the consequences of this overlap are. These consequences can be difficult for two companies who both implemented another standard and want to exchange information, but can also cause problems with advising on which standard to use in a specific information exchange for which both standards could be suitable. As TNO Information and Communication Technology performs work for the SETU (Stichting Elektronische Transacties Uitzendbranche / Foundation for electronic transactions in the staffing industry) [SETU, 2007], which focuses on standards to be used in the Dutch Staffing Industry for the electronic exchange of information, it is important for TNO to know how SIDES and SEP overlap, and how this overlap can be handled for the SETU.

1.3 Research model and objectives

The final goal of this assignment is to identify the overlap between SIDES and SEP and to give TNO an advice on how to handle this overlap for the SETU. To be able to identify the overlap, research is performed into domain modelling to see whether this can help in identifying overlap between electronic messaging standards. The two main research questions on which the research is based are:

- *"How do SIDES and SEP overlap?"*
- *"Overlap between SIDES and SEP makes it difficult for the SETU to decide which standard to use for the exchange of information, how can this overlap be handled?"*

These research questions can be represented in the research model as depicted in Figure 1. This research model shows the different research objects that are used during the research. The research starts with a study into the theory of standardisation, overlap and conflict and also looks into domain modelling. The research makes it possible to define methods to identify, prevent and solve overlap situations between standards. The defined methods are then applied to the case of SIDES and SEP to identify the overlap and finally to deliver the result of this research: an advice for TNO on how to handle the overlap between SIDES and SEP for the SETU.

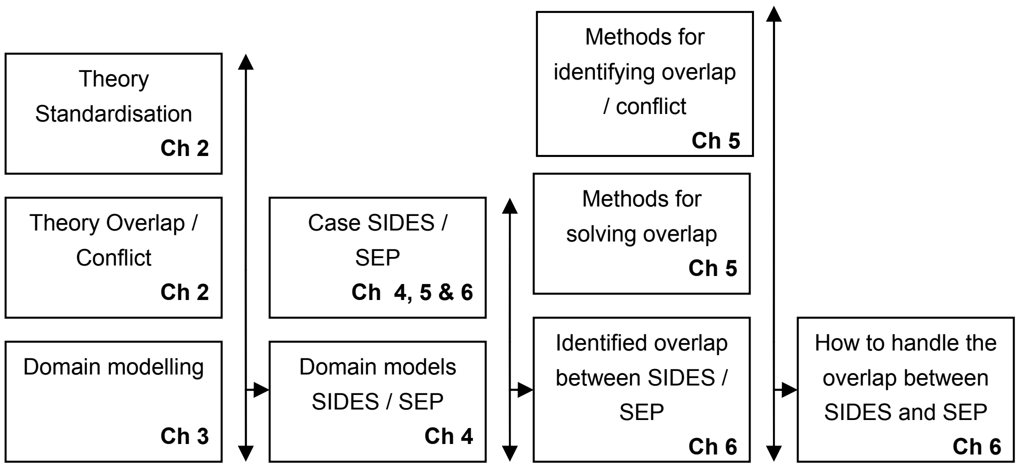

Theory Standardisation Ch 2
Theory Overlap / Conflict Ch 2
Domain modelling Ch 3
Case SIDES / SEP Ch 4, 5 & 6
Domain models SIDES / SEP Ch 4
Methods for identifying overlap / conflict Ch 5
Methods for solving overlap Ch 5
Identified overlap between SIDES / SEP Ch 6
How to handle the overlap between SIDES and SEP Ch 6

Figure 1: Research model

From the research model and the two main research questions two groups of sub questions arise that need to be answered to answer the main research questions. The first group of sub questions is related to the first main research question; the second group is related to the second main research question.

- What are, in the case of electronic messaging standards, the definitions of overlap and conflict?
    - What is the difference between the definitions?
- What are the possible types of overlap in electronic messaging standards?
    - What is the type of overlap between SIDES and SEP?
- What are the causes of overlap and conflict between electronic messaging standards?
    - Can standardization institutions prevent the creation of overlap between standards?
- How can domain modelling be used to identify overlap between electronic messaging standards?
    - Which domain modelling techniques are available?
    - What are the differences between the available domain modelling techniques?
    - Which technique is most suitable to identify overlap between electronic messaging standards?
        - What are the comparison criteria?
- What are the domain models of SIDES and SEP?

- What are the possibilities for handling overlap?
- What is the most suitable handling method that TNO can use to solve the overlap between SIDES and SEP for the SETU?
    - What are the comparison criteria?

1.4 Research scope

As standardisation is a relevant topic in the ICT world, numerous different products are standardized; this ranges from software products to hardware products. This research focuses on the standardisation of electronic messaging standards and specifically on standardisation of the electronic XML messaging standards SIDES and SEP. While some of the results from this research may also seem applicable to other forms of standardisation, this applicability has not been investigated during the research. When applying the result to other standardisation processes, the applicability should be investigated to prevent unwanted or non correct results.

1.5 Research approach

The research presented in this thesis can be described as a combination of a 'typical design research project' and a 'typical empirical research project' as described in [Oosterhuis & Pires, 2004]. The research starts with a literature study into overlap and conflict between standards and investigates methodologies to identify overlap and conflict. Besides this literature study, a literature study into domain modelling techniques is performed. This study is used to identify the most suitable domain modelling technique for identifying overlap between standards. The performed literature research creates the base for the rest of the research. In the rest of the research the domain models for both SIDES and SEP are designed. The overlap is identified according to the techniques defined and an advice is given about the way how TNO should handle the overlap between SIDES and SEP for the SETU.

1.6 Thesis outline

This thesis presents the research that was performed to be able to give an advice for TNO on how to handle the overlap between SIDES and SEP for the SETU. The reader who is only interested in the main results from the research should at least look at the domain models in chapter 4, the exact overlap as located and solved in chapter 6 and the main conclusions that are presented in section 7.1.1. The reader who is also interested in the methods to identify and solve overlap between electronic messaging standards should also read the chapters 3 and 4.

The remainder of this thesis is structured as presented in Figure 1: chapter 2 presents background information on standardisation, overlap and conflict, indicates what the origin of overlap and conflict between standards is and presents three methods that can be used by standardisation institutions to prevent overlap. Chapter 3 investigates the concept of domain modelling to see whether this can help in solving overlap, it presents three domain modelling techniques, Merode, Fusion and NYAM and discusses which of these three techniques is most suitable to create domain models that TNO can use to identify and solve overlap. Chapter 4 presents the domain models of SIDES and SEP that are created with Merode and chapter 5 dives into the subject of identifying and solving overlap and indicates what the most suitable solving method for solving overlap is. Chapter 6 identifies and solves the overlap between SIDES and SEP. Chapter 7 concludes this thesis with the conclusions and recommendations.

2 Standardisation, Overlap and Conflict

Overlap and conflicts between standards occur frequently, with different impact on the standardisation domain. Overlap and conflicts can exist between two or more independent standards, but it is also possible that they co-exist within a standard, as conflicts or overlap rise between two or more sub standards. Just as there are multiple possibilities of overlap and conflict between standards, there are also multiple origins for overlap and conflict.

Reasons for standardisation are presented in section 2.1. A definition created by TNO of a good model of a messaging standard is presented in section 2.2. The definitions of overlap and conflict are presented in section 2.3 and the origin of overlap and conflicts in standards are presented in section 2.4. Section 2.5 contains the definition of prevent and solve and section 2.6 presents methods to prevent overlap. Section 2.7 contains a summary of the chapter.

2.1 Standardisation

This section presents background information on standardisation. Reasons to standardize are presented first, followed by an explanation of the different ways to develop a standard. The last paragraph explains the standardisation process within a standardisation institution.

There are quite a few reasons to standardize, but they can al be traced back to two main categories of reasons: technical and economic. As [Schmidt & Werle, 1992] argued, standardisation is a response to technical and actor complexity. That is, as the domain of a technique becomes larger and larger, companies or standardisation institutes might envision problems with regard to compatibility with other techniques now and in the future. Therefore they might seek for standardisation of the technique, to ensure that there will not be a big increase in interfaces between companies and to ensure that it stays compatible in the future. Especially companies, but also standardisation institutions competing with other institutions, may have different reasons and interests in standardizing techniques developed internally. If a technique becomes standardized the developing company may have an advantage over the competitors or it is sure that its own products do not need any changes because of conflicts with a (new) standard. This motivation for standardisation is therefore more an economic reason [Fomin & Keil, 2000; Schilling, 1998].

Just like there is more than one reason to standardize, there is also more than one way to develop standards. The development of standards can be market driven, in that case the resulting standards are called *de facto* standards, but they can also be committee led. Examples of a *de facto* standard are Adobe's PDF and Microsoft Office documents. Standards resulting from a committee led development are called *de jure* standards [Fomin & Keil, 2000]. An example of a *de jure* standard is the HTTP standard. From the presented distinction between *de facto* and *de jure* it might look like one can directly identify a standard to be a *de facto* or a *de jure* standard. Unfortunately this is not always the case. The problem with this identification of standards as a *de facto* or a *de jure* standard mainly comes from the *de jure* standards. A *de jure* standard can be developed by a formal standards body; this is called formal standardisation, for example the SQL standard. A *de jure* standard can also be developed by a consortium and in that is called grey standardisation, for example the XML standard. Table 1 presents the distinction between the three types of standards. The grey standardisation makes the distinction between *de facto* and *de jure* standards difficult. The reason for this is that the grey standardisation groups (consortia and others) are not always clear on whether a standard is implementation-dependent or implementation-independent. At the time of writing this report the distinction between implementation-dependent and implementation-independent is important in Adobe's attempt to make their PDF standard a *de jure* standard and Microsoft's attempts to make their OOXML standard a *de jure* standard. If these two standards are accepted as *de jure* standards, Microsoft and Adobe are favoured as the standards are implementation-dependent. In the case of an implementation-dependent standard one or more companies might be favoured, and the standard might look like a *de facto* standard. In the case of an implementation-independent standard there is no company favoured, and the standard looks exactly like a formal *de jure* standard. These facts make it difficult to distinguish between *de facto* and *de jure* standards [Besen & Farrell, 1994; Egyedi & Dahanayake, 2003; Oksala et al, 1996; Schmidt & Werle, 1998].

Table 1: Types of standards

Type		Description
de facto		Market driven standard
de jure	formal	Standard maintained by a formal institution, legal consequences for market
	grey	Standard maintained by consortia, no legal consequences for market

Standardisation in consortia is often done by a committee, working under the responsibility of the consortia. Most committees that are active in standardisation efforts for the Information Technology (IT) industry are established and operate under the governance of a formal international standards organization. Organizations relevant, and active in the IT industry, are the International Telecommunication Union [ITU, 2007], International Organization for Standardisation ISO [ISO, 2007], the International Engineering Consortium [IEC, 2007] and the Institute of Electrical and Electronics Engineers [IEEE, 2007]. Furthermore there are numerous smaller organizations, often country based, that work in close cooperation with the international organizations. Besides these formal international organizations there are also different consortiums that create standards, of which the Word Wide Web Consortium [W3C, 2007] is one of the most commonly known organizations. When comparing the working methods of the different types of standardisation organizations, it becomes apparent that the methods for developing standards are pretty much the same. They all work with committees that develop the standards, negotiate over differences and difficulties and ultimately present a standard. The main difference is that the formal institutes are often governmental organizations, or sponsored by governments, and their standards are formal standards, while the consortia are often primarily sponsored by corporate industries and their standards are grey standards [Egyedi & Dahanayake, 2003].

With regard to the standards this research focuses on, one can see that both SIDES and SEP are grey *de jure* standards. Grey because the HR-XML consortium is a consortium, and not a formal institution, *de jure*, because the standards are not created by the market, but by an institution.

2.2 Messaging standard definition

This section presents a definition of a messaging standard that is developed and used within TNO. This message standard definition consists of five main parts and is used in the research that is presented in this thesis. The definition explains what, from the viewpoint of TNO, a good domain model for an electronic messaging standard should look like. The definition of a messaging standard makes a distinction between a static part and a dynamic part. In practice, most electronic messaging standards unfortunately do not adhere to this definition. This might be a reason overlap exists between standards.

The static part of a messaging standard describes the actors that are actively associated to the messaging standard and the objects that play a role in the subject domain of the messaging standard. The static part also identifies the relations between the objects. In the model overview in Figure 2 the static part is split up between a context part and a relationship part. The reason for this is that there is no direct link between the context diagram and the class diagram and that the split in these two parts makes the standard description more readable. For the research performed this distinction is not directly necessary and these parts are therefore taken together and are called the static part.

The dynamic part of a messaging standard identifies the events that have an effect in the messaging standard. This can be events that trigger the sending of a message, but it can also be events triggered by the reception of a message. The dynamic part also includes state transition diagrams (STDs) that indicate how an event affects the state of an object in the domain model.

In Figure 2 an overview of the models, that should be contained in a good messaging standard according to TNO, is given [Bastiaans, 2007]. Not all of the diagrams mentioned in this overview are part of the domain model that is used in this research to identify overlap; they are however needed to create a good standard and help to see how the different models relate to each other. The lines between the boxes indicate a linkage between two models:

1. Each actor has to be represented in the context diagram
2. Each object has to be present in the object event table
3. Each entity has to be represented in the class diagram
4. Each state change has to be a event in the event list
5. Each object has one STD
6. Each event has to be present in the object event table
7. Each object and each attribute is added to the dictionary
8. Each element in a message has to be an attribute in the class diagram
9. Each event in an activity diagram has to be present in the object event table
10. Each actor has to be represented in at least one activity diagram

When creating the different models the above mentioned rules help to ensure that the models created are in accordance with each other. The model overview is divided in five main parts: the context, relationship, dynamic, communication and definition part. The context, relationship and dynamic part together make up the domain model part that is used in this research to indicate overlap between two standards. For a complete standard the communication and definition parts should be added.

A complete domain model should therefore, according to the definition by TNO, exists of the following specifications:

- Context diagram - Actors + Roles - Class diagram - ERD	Static part
- State transition diagram - Event list - Object event table	Dynamic part

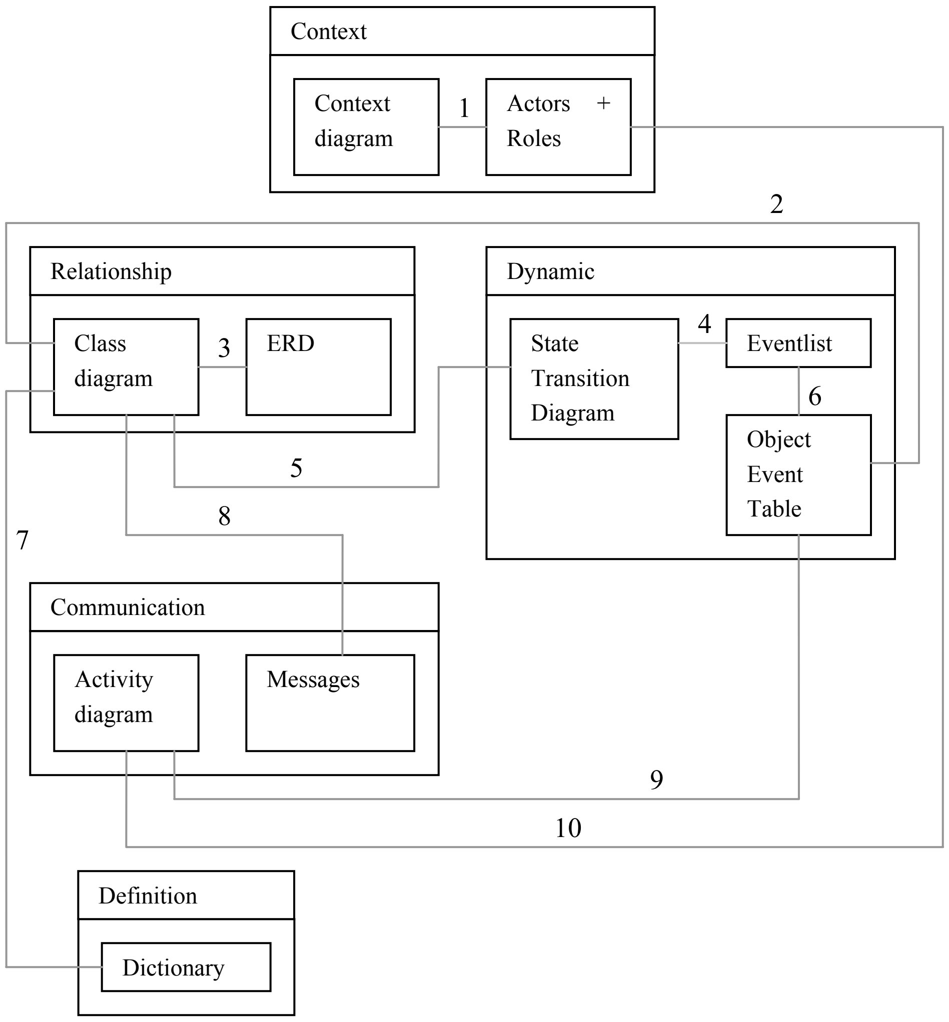


Figure 2: Model overview standard definition

2.3 Definition overlap and conflict

This section presents definitions of the terms overlap and conflict. The first subsection gives an overall definition of the two terms; the second subsection gives an explanation of the different types of overlap.

2.3.1 Formal definitions

To be able to compare problems within standardisation and problems between two standards it is important to have a good definition of what an overlap is and what a conflict is. This section therefore gives definitions of both notions with regard to standardisation and explains the differences.

In Table 2 the dictionary definitions of the terms overlap and conflict are presented.

Table 2: Dictionary definitions of overlap and conflict [Cambridge, 2007]

Overlap	**Conflict**
verb **-pp-** **1** [I or T] to cover something partly by going over its edge; to cover part of the same space: *The standard is made of messages which overlap (exchange the same information).* **2** If two or more activities, subjects or periods of time overlap, they have some parts which are the same: *The assignment message in SIDES does not overlap* ***with*** *SEP's at all.*	verb [I] **1** If beliefs, needs, or facts, etc. conflict, they are very different and cannot easily exist together or both be true: *The new standard seems to be in conflict* ***with*** *existing standards.* **2** to fight or disagree actively: *If SIDES and SEP conflict* ***with*** *each other again, this will be disastrous for the HR-XML consortium.*

The two definitions seem to be quite different, and it may look easy to make a separation between overlap and conflict. But when taking a closer look at the definition of conflict, one may notice that the first definition of the term conflict is 'a belief, need or fact about someone' and that the second definition of the term conflict is 'something that is different from another belief, need or fact'. Both of these definitions indicate that there must be something to have a conflict about. This also indicates that the two sides that are in conflict must have something in common, and thus these two sides must have an overlap, otherwise there would not be a conflict [Moffett & Sloman, 1993].

The rule that says that to have a conflict one has to have an overlap cannot be applied the other way around. One cannot say that to have an overlap, there must be a conflict. It is very well possible that there is some kind of overlap between two things, but that this is not a cause to a conflict. When referring this to standards and the standardisation processes, one can say that an overlap between two standards or sub standards can exist without it ever being a problem.

The above definitions of overlap and conflict, and the explanations regarding standardisation might give the impression that one only has a problem if there is some kind of overlap between standards <u>and</u> there is a conflict. This is not exactly true. It is true that one is in trouble when there is an overlap <u>and</u> a conflict, but one might also be in trouble when there is <u>only</u> overlap. One might be in trouble in case there is only overlap between two standards because in this case it is possible that two companies each implemented one of the overlapping standards, and still cannot communicate electronically because the messages exchanged are not understood by the other company.

2.3.2 *Types of overlap*

To indicate what types of overlap can be present between two overlapping electronic messaging standards, set theory is used. Three relations, the subset, the equal and the intersection relation describe the overlap situations that can exist between two electronic messaging standards.

The subset relation indicates a standard that is a complete subset of another standard; if this is the case between two standards it might very well be possible to replace the subset standard *A* for the standard *B*, thereby solving the overlap problem. For the equal relation the same can be said, but in this case the two standards both describe exactly the same domain. The standards are therefore equal and an example of a solution to solve this type of overlap is to make a choice for one of the two standards. The intersection relation is the form of overlap one will most likely find between standards. Both standards describe a part of the world that is equal, but both also describe distinct parts of the world. An example of a solution to this overlap type is to remove the overlap from one standard and to leave the other standard intact. Both standards can be used after this solution for their respective part of the world, but for the overlapping part one standard is defined to be the solution. Table 3 presents an overview of the three definitions and gives a short description of each overlap relation.

Table 3: Overlap relations standards

Overlap type	Description
Subset	One standard can completely be embedded into the other standard
Equal	Two standards exactly overlap
Intersection	Two standards partly overlap

One final remark has to be made about the set theory relations presented in this section. As will be described in the next section, to find overlap between standards one will look at different levels, from general to more specific. The set theory relations as described in this section can be applied on all these different levels, even though the types of overlap are described pretty high-level in this section.

2.4 Origin of overlapping and conflicting standards

The origin of overlap and conflict between standards cannot be directed back to one main cause; there is more than one source that can create overlap and conflicts. This makes it difficult to guide the process of standardisation to ensure that a new standard is not a cause to overlap and conflict. In this section different causes of overlap and conflict are presented and conflicting implementations of one and the same standard which may cause problems when modifying standards to new insights, are discussed. The strategies used by companies when implementing a standard in a product that is not fully compliant to the standard might also be of influence in the process of standardisation.

2.4.1 Causes of overlap and conflict

As already mentioned, there are many different causes to overlap or conflict between standards. These different causes are spread over the entire standardisation process and can be organized in three parts: the structural aspects, the process aspects and the output phase [Schmidt & Werle, 1998]. A graphical representation of these three parts can be found in Figure 3. The structural aspects are further split up in institutional framework aspects, actor aspects and technological foundations aspects.

The institutional framework aspects relate to rules inside an organisation and to the relations with other standards organizations. All of these aspects are related to how the institution works. The actor aspects are related to the persons that create a standard. What their interest in the subject of the standard is and what kind of resources they have available. This can be physical resources like computers, but also non-physical resources like time. Furthermore their perceptions on the world and the standard in specific, but also their motives can be of big influence on the standard. The technological foundation aspects relate to the technical implementation of the standards and problems that relate to the translation of a specification to a real standard.

The process aspects relate to the decision making process. Aspects that are relevant for this part are the complexity of the problem, how more complex the system is, how more difficult the decision-making process can become. Furthermore the actors, the coordination of the work and the strategies and/or coalitions between the actors may make the decision process more difficult, but may also prevent the best solution from being chosen.

Finally in the output phase there are also aspects that can influence a standard. A committee might have to choose from a number of options, the standards might not be consistent, but the architectural entrenchment may also influence the approval of a standard. A last big influence on a standard can be whether it is an ‘old’ standard or a ‘new’ standard. Old standards are those that have already gained standing as *de facto* standards or as recommendations of another standards committee which can easily be ratified ex post, while in some cases it may be easier to formulate a completely new standard than to try to agree on an already existing standard.

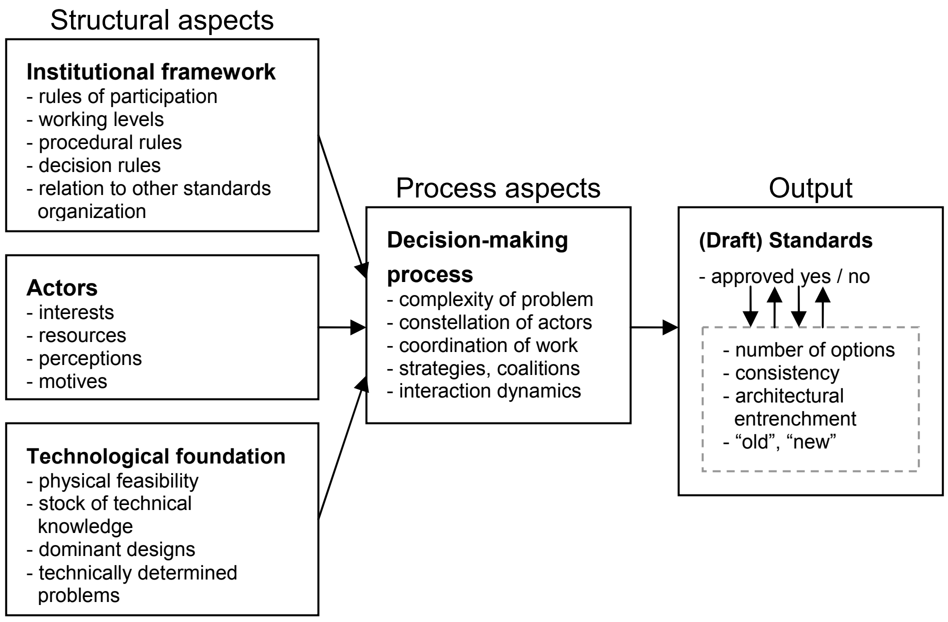


Figure 3: The standardisation process [Schmidt & Werle, 1998]

The causes that are relevant in creating overlap between standards in Figure 3 are depicted in Table 4. Each cause that is important in the creation of overlap is discussed in the paragraphs following the table.

Table 4: Causes of overlap and conflict in the standardisation process

Structural aspects	Process aspects	Output
Institutional framework	Coordination of work	Consistency
Actors - Motives		
Actors - Interests		
Technological foundation		

The overlap / conflict cause in the ‘output’ of the standardisation process is discussed in section 2.4.2 because this is not of direct influence to overlap in standards. The causes from the structural and process aspects of the standardisation process are discussed first. The causes found in the structural aspects part of the standardisation process can be divided in three categories: ‘the institutional framework’, the ‘actors’ and the ‘technological foundation’. The category ‘actors’ can be further subdivided in the categories ‘motives’ and ‘interests’. Causes of overlap and conflict coming from the ‘institutional framework’ are often rules and working procedures within an institution, but also the relation of the institution with other standardisation institutes. If this relation is good, overlap can be avoided by talking about the developments within the institutions. If the relation is not so good, it might happen that two different standardisation institutions are working on standards for the same problems, or are both defining equal standards. Another potential risk within standardisation institutions is that not everybody can participate in all institutions. Therefore there might be less consensus from the industry, and there will be more potential for overlapping standards.

Causes of overlap and conflict coming from the 'actors' part of the standardisation process are all related to interests and motives of both persons and organizations working in or with standardisation institutions in developing standards. If for example the motive is to create a standard that suits a companies need, there might be other companies that have different needs, therefore making it difficult to create a uniform standard. But personal motives can also play a role in causing overlap and conflict between standards. If persons involved are only interested in working on the standard, but do not have real interest in the results, it might be possible that nobody takes a good look into the real-world, thereby delivering a standard others are already working on, or have already developed. Economic interests might also introduce problems. Companies that have products that are within the scope of a new standard might prevent certain aspects of standards being standardized because this might result in their products being non-compliant when the standard is finalized. Furthermore political interests may also be a risk; governments might support standards created in their own country, thereby making it virtually impossible to create one overall standard.

The final cause of overlap and conflicts from the structural aspects part of the standardisation process comes from the technical foundation of the standard. As one can imagine, there are numerous standards that are not created from scratch, but that result from technologies and / or standards that are already on the market, or are about to hit the market. It might therefore not be possible to create a completely new standard in which everything that is already worked out is completely reinvented. One has to build on existing building blocks, which might give cause to overlap and conflicts.

In the process aspects part of the standardisation process, the coordination of the entire process is of influence to overlap and conflict between standards. If coordination is good, an institution might notice that two working groups are working on the same part of the real-world, and might prompt the two groups to talk to each other to prevent overlap and conflict between the standards being developed. If the coordination is not so good, these kinds of overlap and conflict might not be noticed until the time of publication by which it is far too late to change the developed standards. Therefore a good coordination of the standardisation procedures within standardisation institutions, and preferably also between institutions, is of uttermost importance to prevent overlapping and conflicting standards from being formalized. More information about preventing overlap and conflicts can be found in section 2.5 and 2.6.

2.4.2 *Overlap introduced through implementations*

Not only can overlap in standards be introduced in the development of standards, it can also be introduced in the implementation of standards, the 'output' phase of the standardisation process. Interoperability is therefore not only dependent on a standards' specification, but also on a consistent implementation of the standards. A consistent implementation is unfortunately not always straightforward because standards are sometimes ambiguous. Therefore, if standards are interpreted differently, incompatibility and lack of exchangeability between two or more implementations of standards are likely to occur [Egyedi & Dahanayake, 2003].

While the above mentioned implementation differences are not of direct influence to the standardisation process, their existence is inherent to an ambiguous standard, which might be caused by a not so good standardisation process but also by different uses of the same standard. It is therefore important to be aware of these implementation differences as they might shed light on where the standardisation process should be adapted to prevent ambiguous standards. Even though the implementation differences are not of direct influence to the standardisation process, it might be possible that a standard is modified after a while to incorporate new insights. These new insights might come from different implementations and the process to come to a new standard might therefore become cumbersome due to differences caused by differentiating implementations. It is therefore all the more important to try to minimize differentiation in the implementation process.

Differences in the implementation of standards and therefore in the standards used can be caused by ignorance of the users, but it may also be caused by an aggressive market strategy in which companies intentionally introduce deviant standards implementations. There are three well-known aggressive market strategies; these are the *embrace-and-extend*, the *embrace-and-omit* and the *embrace-and-adapt* strategies [Egyedi, 2007]. In all situations resulting from these strategies the integrity of the standard is at stake. In some cases the interoperability can be re-created, but it requires extra effort. Because of this extra work most cases result in market fragmentation. The three strategies that may cause these problems with interoperability, together with a real-world example, are described next. The real-world examples are taken from [Egyedi, 2007].

The *embrace-and-extend* strategy is one of the most well known strategies. This strategy introduces extra functionalities into standards implementation that are not compliant to the standard. This extra functionality therefore interferes with the intentions of standards; making products interoperable. A real-world example of the *embrace-and-extend* strategy is the case of SGML and XML. The initial idea of XML was to bring the 'structured data exchange' functionality of SGML to the web. The idea was that XML was the successor of SGML in the web environment. This was however only partly achieved because XML documents can not be processed by SGML tools. It can therefore be said that the developers of XML, W3C, embraced the SGML standard but extended it to the new XML standard that was not interoperable anymore with its origin standard, SGML.

The *embrace-and-omit* strategy is based on omitting things that are part of the standard in the final product. Therefore not the entire standard is implemented, which may be cause to a problem in interoperability if another product assumes that everything that is specified in the standard is present. A real-world example that presents the *embrace-and-omit* strategy is the Open System Interconnection (OSI) model case. The OSI is a standard reference framework. The framework identifies ICT services as consisting of a set of functions that are mapped onto seven layers. Base standards, which are generic building blocks specified for these layers, can contain options. The problems started to rise when OSI implementers omitted functionality that was part of the standard because they wanted to cut down costs. This resulted in OSI-compatible products, but there was only partial compliance because of the *embrace-and-omit* strategy of the OSI implementers.

The third and last well known strategy is the *embrace-and-adapt* strategy. In this approach a company implementing a standard introduces local adaptations to the implementation of the standard. This again raises problems with interoperability because the adaptations are not understood by other implementations. The Unified Modelling Language (UML) case is a real-world example of the *embrace-and-adapt* strategy. In November 1997 the Object Management Group (OMG) adopted the UML as a standard. The standard aimed at simplifying and consolidating the large number of Object Oriented (OO) software developing methods that had emerged. UML itself comprises several complementary and substitutive modelling techniques that use different terminology for similar things. UML is not intended to be a complete development method. In the beginning a complementary companion book, the Rational Unified Process (RUP), was proposed. However, this work was not seen as the best solution according to experts, thus giving other people and companies the challenge to propose better UML-based methodologies. Unfortunately, these methodologies are not interoperable, thus resulting in similar functional solutions that cannot be exchanged or integrated. Because of the lack of a good companion book to UML, different people and companies, using the *embrace-and-adapt* strategy invented new UML based methodologies that were not interoperable.

2.5 Definition prevent and solve

In this section formal definitions of the terms ‘prevent’ and ‘solve’ are presented and the relation between these two terms is discussed. The formal definitions of the terms prevent and solve come from the Cambridge dictionary and can be found in Table 5.

Table 5: Dictionary definitions of prevent and solve [Cambridge, 2007]

Prevent	**Solve**
verb [T] to stop something from happening or someone from doing something: *Study the standards SIDES and SEP to prevent overlap in the new standard you are creating*	verb [T] to find an answer to a problem: *to solve the overlap*

The dictionary definitions above show that the two definitions are totally different. Even though, they are closely related when talking about overlap and conflicts. If one prevents an overlap between standards, one does not have a problem with overlap; therefore one does not have to solve overlap. And the other way around; if one has to solve a problem with regard to an overlap, this shows that the overlap was not prevented in the standardisation process of the standard.

2.6 Preventing overlap

This section introduces and explains three methods that can be used to prevent overlap between electronic messaging standards. Trying to prevent overlap from happening should be in the minds of every designer of standards. There are several options to prevent overlap from happening. Each option to prevent overlap has its own characteristics and may be suitable for a specific standardisation world. The three methods that are mentioned can only be applied by the standardisation institutions that develop standards. The users of a standard cannot prevent overlap from happening.

Three methods that can prevent overlap from happening are:

1. Research into the standards’ domain
2. Coordination of standardisation process
3. Coordination between institutions

These options are discussed shortly in the following subsections.

2.6.1 Research into the standards' domain

The first method is to do research into the standards' domain. Many new standards are developed according to the demand for standards from the industry. In cases where this happens the developers might not always first look around in the domain of the requested new standard to see whether there are already standards present or in development, that are satisfactory for the demand from the industry. Therefore, to prevent overlap between standards, an institution should always do a thorough investigation into the domain of the requested standard. When performing this research, the institution should not contain the domain being researched too much. There might be standards in just slightly different domain that could suffice the request, or be used with some minor modifications.

2.6.2 Coordination of standardisation process

The second method is to improve the coordination of the standardisation process. If there is coordination within institutions, overlap between two or more standards in creation can be noticed and the working groups can be pointed to this overlap. They can then talk to each other to see whether the working groups should go further as one new group, or should set boundaries for each group to ensure that the resulting standards will not overlap.

2.6.3 Coordination between institutions

This section describes the third method that can prevent overlap; coordination between institutions. The second method focussed on multiple standards being developed by the same institution, but overlap and conflict can also be present in standards developed by two or more different institutions. In this case the coordination process is slightly more complicated because there is no overlapping authority for both institutions. To solve these cases of overlap and conflict, standardisation institutions should work closely together and not compete with one another to create standards quicker. Only a good relation between institutions, in which there are no secrets about which standards are developed can prevent overlap and conflict.

2.7 In summary

All reasons for standardisation can be traced back to two main categories of reasons: technical and economic reasons. Technical reasons because multiple companies want to have the possibilities to easily exchange information. Economic reasons because standardisation can save money and the developing company can gain profit from standardisation. Standards can be divided into *de facto* standards, standards developed and maintained by market, and *de jure* standards, standards developed and maintained by a formal institution or consortia. The *de jure* standard can be further split up into grey standards, maintained by consortia, and formal standards, maintained by a formal institution. To create good standards, TNO has created a definition of what a good electronic messaging standard should look like. A good electronic messaging standard should contain a context diagram that specifies which parties use the standard and a relationship diagram that specifies the relation between the objects (for example an ERD or a Class diagram). These diagrams together form the static part of the model. The standard should also contain a dynamic model that specifies the dynamic aspects of the standard (for example STDs, an event list or an object event table), a communication diagram that specifies how the objects interact (for example an Activity diagram and the messages) and a definition part that defines all objects and attributes.

These models and diagrams together form the complete specification of an electronic messaging standard.

A problem with different standards that describe the same domain or parts of the same domain is that there is a risk of overlap between the standards. This overlap can make the choice for a standard to use difficult. Overlap between standards can be classified into three different types, the subset relation type, the equal relation type and the intersection relation type. Overlap and conflict between standards can be introduced through the institutional framework (if for example two institutions work on the same problem), the actors (if for example their interests or motives are different) and the technological foundation (if for example there is already a standard that should be incorporated in the new standard). Overlap can further be introduced in the decision-making process and in the final definitions of standards. Even in the implementation phase of a standard overlap and conflict can be introduced, mainly because standards are sometimes ambiguous. Preventing overlap is possible if the standardisation institutions put effort into it.

3 Domain modelling

This chapter indicates which domain modelling techniques can be used for locating overlap between messaging standards. It starts off with an introduction into Model Driven Architecture (MDA) and a mapping of this architecture onto the standardisation process to indicate where the domain model can be placed in the standardisation process. Following this first section, three different domain modelling techniques are presented. The presentation of these domain modelling techniques is followed by a presentation of three suitability criteria to determine the most suitable domain modelling technique for TNO to handle the overlap between SIDES and SEP. The suitability criteria are then used in the final section to choose the most suitable domain modelling technique for TNO to identify overlap between SIDES and SEP.

3.1 Model Driven Architecture

In this section the Model Driven Architecture (MDA) is introduced and a mapping of this architecture onto the standardisation process is made to indicate where domain modelling can be used in the process.

The MDA is an approach to developing software in which business and application logic is separated from underlying platform technology. One might think that the MDA is not applicable to the subject of this thesis, because its subject has little to do with software engineering but all the more with electronic messaging standardisation. This is partially true, but the main idea behind MDA, the separation of business and application logic from the underlying platform technology is applicable to the domain of electronic messaging standardisation as well [Miller & Mukerji, 2003].

In the MDA three different models are depicted, all three with different perspectives and different uses:

1. the Computation Independent Model (CIM) contains the domain model and the requirements for the system. It further shows the environment in which the system will operate. The model is independent of how the system is implemented.
2. the Platform Independent Model (PIM) describes the system but does not show details of the platform used. The PIM will be suited for a particular architectural style, or several.
3. the Platform Specific Model (PSM) is a model specific for one or more platforms that are able to implement the system with the desired architectural qualities.

The MDA will then also define mapping steps to map the PIM into a PSM for a specific platform. The nature of the mapping will be determined by the platform model. If we map this architecture with the CIM, PIM and PSM models to the standardisation process, as depicted in figure 4, we see that in most standardisation processes the equivalent to the CIM in MDA is missing. The standard itself can be mapped to the PIM because a standard can be implemented on every available platform and in every programming language available. The implementation of a standard can be mapped to the PSM model as this is specific for the platform it is used on. An equivalent to the CIM, a domain model of the world to which the standard is projected, is often omitted in the standardisation processes. Most standardisation processes start with directly creating the paper version electronic, thereby omitting the CIM.

The reason for the CIM in the MDA approach is to create a high-level model that is not directly related to the final product. In this way a high-level view of the system is preserved which makes it easier to compare two different products or standards and prevent errors because of subjective models. As the authors [Pokraev et. al, 2005] conclude: "*Interoperability is about effective use of systems' services. The most important precondition to achieve interoperability is to ensure that the message sender and receiver share the same understanding of the data in the message and the same expectation of the effect of the message*". It therefore seems to be a good idea to create a clear domain model for each standard to ensure that users and developers understand the world they are working in and share the same thoughts about this world.

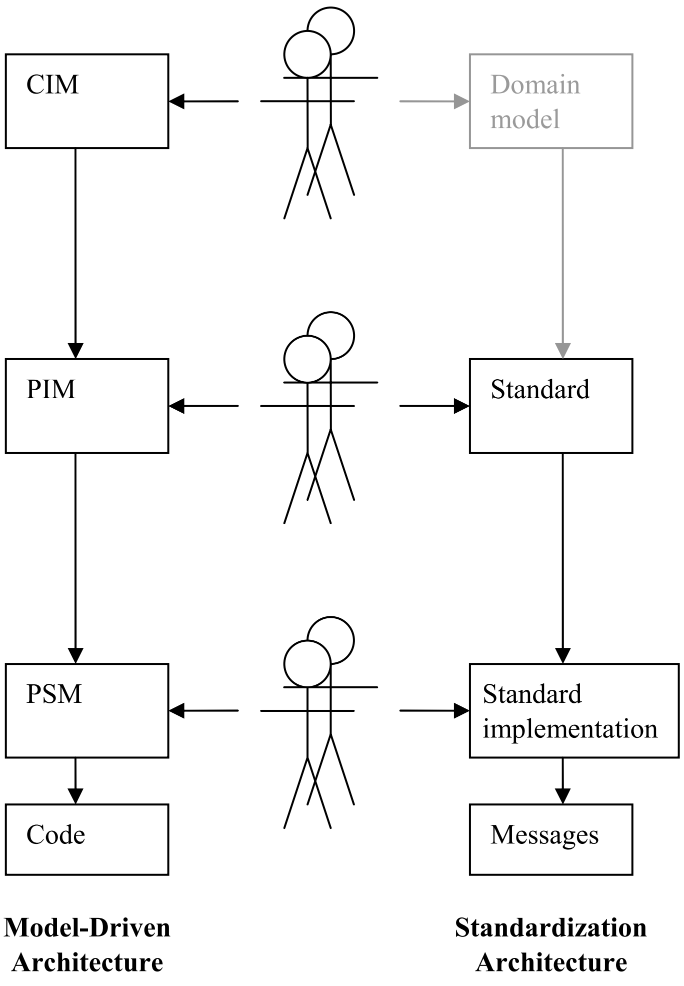


Figure 4: MDA mapped to standardisation architecture

As mentioned before, a domain model seems to be helpful in developing concise standards. It could help in preventing overlap because the developers of the standards know in which environment they are working, and standards developed with the same domain model as a base can be compared more easily and therefore possible overlap issues can be detected earlier. The following section presents three domain modelling techniques that could be used to create a domain model for a messaging standard.

3.2 Domain modelling techniques: Merode, Fusion and NYAM

This section presents three domain modelling techniques that can be used by TNO to create domain models for identifying overlap. The three techniques that are describe are all object oriented modelling techniques that are not directly focussed on creating models that specify a electronic messaging standard. The reason for choosing these three domain modelling techniques for the comparison is that within TNO the Merode domain modelling technique is already used to create domain models and was indicated as a possible interesting solution for domain modelling to identify overlap. The other two techniques, Fusion and NYAM, were chosen because they show some similarities to the Merode technique and might therefore be possible substitutes. The difference between Merode and Fusion/NYAM is that the Merode technique is completely focussed on domain modelling, while the Fusion and the NYAM techniques are software modelling techniques that include domain modelling.

The reason for choosing these three techniques for the comparison and not to include modelling techniques that focus more on the specification of standards, like UMM (UN/CEFACT Modeling and & Methodology) is that these techniques do not add extra weight to the more generic techniques for our research. In the research it is not the intention to develop standards (the standards are already present), the intention is to use domain models to identify overlap between existing standards.

The following three paragraphs each describe one of the three domain modelling techniques, first Merode is presented, then Fusion and finally NYAM.

3.2.1 Merode

Merode is an object oriented analysis method developed at the Katholieke Universiteit Leuven (Catholic University of Leuven). Its name is an abbreviation for Model-based Existence-dependency Relationship, Object-oriented Development. This abbreviation also indicates one of the most prominent features of the Merode method, the existence dependency relation. The existence dependency feature is explained in one of the following paragraphs.

The main phases of the Merode development process are the specification and the implementation phase. The system specification phase is further divided in enterprise, functionality and user interface modelling. The steps that are undertaken in the enterprise and functionality modelling phase are presented in Table 6.

Table 6: Merode system specification steps [Snoeck et. al, 1999]

1	Enterprise modelling phase
1.1	Modelling enterprise object types and business event types
	a. Identification of enterprise object types
	b. Identification of business event types
	c. Construction of the existence dependency graph
	d. Construction of the object-event table
1.2	Sequence constraint specification
1.3	Specification of state vectors and object-event methods
	a. Definition of state vectors
	b. Definition of object-event methods
	c. Definition of data constraints
2	Functionality modelling phase
2.1	Identification of all required services

2.2 For each service:

a. Identification of information object types
b. For input services: identification of the generated business events
c. Construction of the service specification diagram
d. Definition of the state vector
e. Definition of the methods

In the specification phase an object event table and an existence dependency graph are created. The columns of the object event table contain the different objects that are identified and the rows contain the identified events. Each event can perform three kinds of actions: it can create, modify or end an object. Another important fact is that every object should have a create event and an end event. The modify event is optional as there will always be objects that are not modified during their lifetime. The object event table is filled with create (C), end (E) and modify (M) actions, after which one can easily see if all objects have appropriate create and end actions. Table 7 is an example of an object event table.

Table 7: Merode object event table example

Event ↓ Object →	Position	Candidate	Assignment
Create_Requisitions	C		
Post_Requisitions	C		
Transfer_Applicant_Information	M	M	C
Transfer_New_Hire_Information	M	M	M
Candidate_Submit		C	
Post_Position	C		
Enter_Candidate		M	
Match_Position	M	M	C
Submit_Matched_Candidates	M	M	M
Candidate_Applies		C	
End_Assignment	E	E	E
Cancel_Position	E		
Accept_Assignment		E	

The existence dependency graph is another representation of the same reality. The distinguishing feature of an existence dependency graph in relation to other graphs is that relations can only exist if they are fully embedded in the lifetime span of their parent. An example of an existence dependency graph is presented in Figure 5.

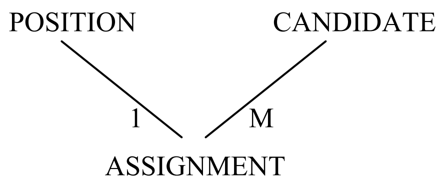


Figure 5: Existence Dependency Graph example

This existence dependency graph is based on the object-relationship diagram that can be found in Figure 6. The object-relationship diagram is completely valid, but the ‘applies for’ relation is not existent dependent. The reason for this is that the life of a candidate is not fully contained in the life of a position, and, the other way around, the life of a position is not fully contained in the life of a candidate. Therefore, in an existence dependency graph, the relationship between position and candidate must be instantiated to a new object type. In the example we call this object 'Assignment'.

This assignment object type captures all time related events in which a position and candidate object are related to each other. The assignment object type is existent dependent on both position and candidate. During its lifetime an assignment always references to the same position and the same candidate and information about these objects should not be removed as long as assignments are registered. The resulting existence dependency graph is depicted in Figure 5.

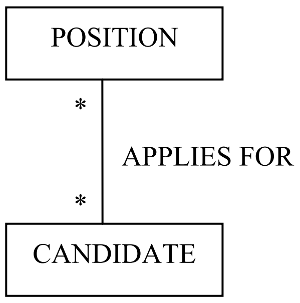


Figure 6: Object-Relationship schema example

The existence dependency graph notation is based on the notation of an Entity Relationship Diagram (ERD). It is therefore possible to use this notation to draw an existence dependency graph, but it should then be clear that every relation present represents existence dependency. As already mentioned, the existence dependency graph and the object event table are two representations of the same reality, they should therefore always be consistent with each other. The semantics of the existence dependency graph therefore must be in accordance with those of the object event table. Each object type that exists in the existence dependency graph should therefore have a column in the object event table, and vice-versa.

Based on the object event table and the existence dependency graph Merode also specifies the creation of state transition diagram (STD). And example of an STD can be found in Figure 7.

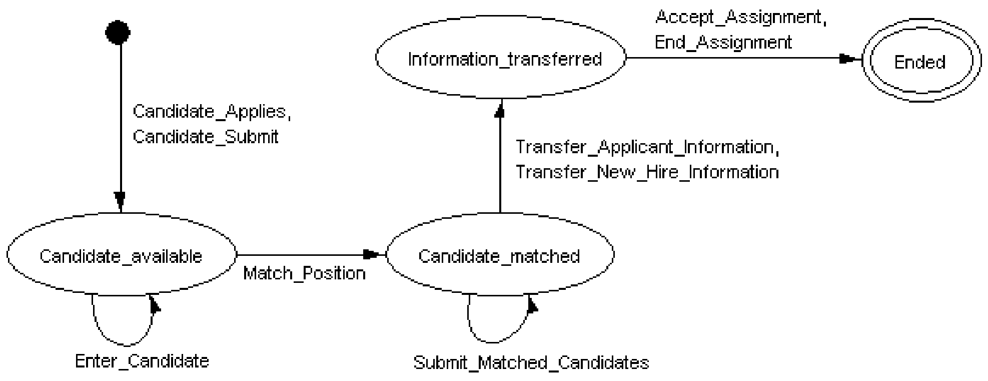


Figure 7: Example STD Candidate

As the STD's are, just like the object event table and the existence dependency graph, also another representation of the same world, the STD's should be consistent with the object event table and the existence dependency graph. Therefore each object should have an associated STD, and all events that influence the object should be represented in the STD. This includes all events inherited through existence dependency relations.

3.2.2 Fusion

The Fusion object-oriented technique is, as its name already suggests, a fusion of different techniques from other object-oriented techniques. The technique was developed to provide a systematic approach for developing software in an object-oriented fashion. It has integrated, but also extended, existing approaches to create a systematic technique from a requirements definition to a programming-language implementation. The technique is split up in different phases: the analysis phase, the design phase and the implementation phase. Each phase is then further split up in different steps which all describe the decisions that have to be made. The deliverables from each step serve as the input for the next step. The Fusion technique also provides rules for checking the consistency and completeness of the developing models [Coleman et. al, 1994].

The three phases in the Fusion technique: analysis, design and implementation all define different parts of the final product. Table 8 gives an overview of the contents of the three phases.

Table 8: Fusion system specification steps [Coleman et. al, 1994]

Analysis	The analyst defines the intended behaviour of the system. Models of the system are produced, which describe the following:
	- Classes of objects that exist in the system
	- Relationships between those classes
	- Operations that can be performed on the system
	- Allowable sequences of those operations
Design	The designer chooses how the system operations are to be implemented by the run-time behaviour of the interacting objects. Different ways of breaking up an operation into interactions can be tried. During this process, operations are attached to classes. The designer also chooses how objects refer to each other and what the appropriate inheritance relationships are between classes. The design phase delivers models that show the following:
	- How system operations are implemented by interacting objects
	- How classes refer one to another and how they are related by inheritance
	- Attributes of, and operations on, classes
Implementation	The implementer must turn the design into code in a particular programming language. Fusion gives guidance on how this is done.
	- Inheritance, reference, and class attributes are implemented in programming language classes
	- Object interactions are encoded as methods belonging to a selected class
	- The permitted sequences of operations are recognized by state machines
Fusion also offers advice on performance issues, and suggests how traditional inspection and testing techniques have to be modified for object-oriented software. At the end of each phase, Fusion provides consistency checks to make sure that the models constructed agree with each other and with the results from the previous phase.	

For domain modelling the design and implementation phase are not relevant, they are presented in this section to give a complete overview of the Fusion technique.

In the Analysis phase, which results in a model of the domain, two models are produced. These two models capture different aspects of a system. To understand the actions of the system all these aspects are important. The first model that is produced is de Object model. This model defines the static structure of the information in the system. The second model is the Interface model. This model defines the input and output communication of the system.

The object model in Fusion is based on extended entity relationship notations, in which it can represent classes, attributes and relationships between classes. The extensions that are used allow the use of aggregation and generalization. An example of an object model created within the Fusion technique which uses the aggregation extension can be found in Figure 8.

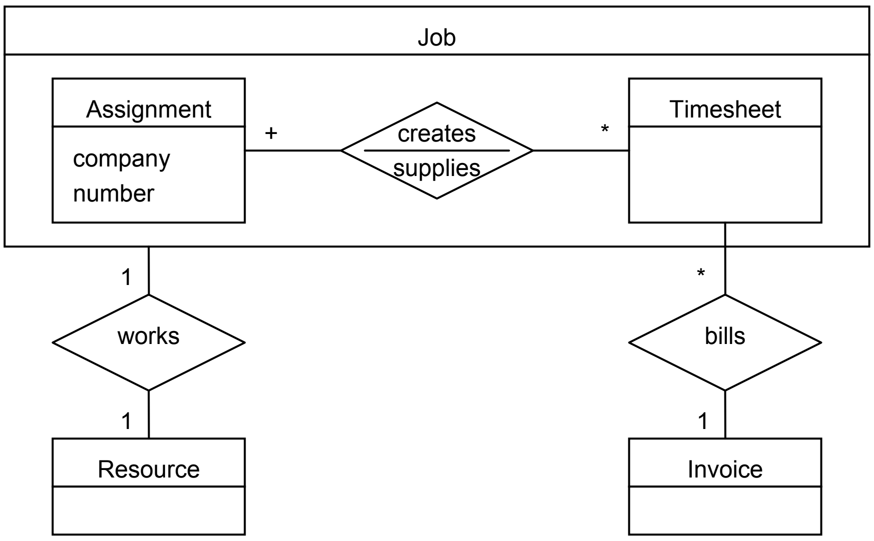


Figure 8: Fusion aggregation relation

An example of an object model that uses the generalization extension can be found in Figure 9.

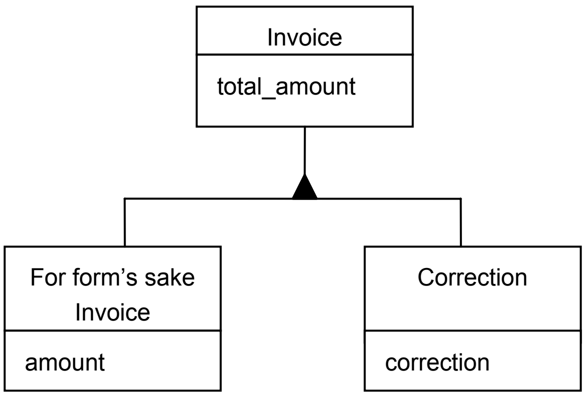


Figure 9: Fusion generalization relation

The second model that is created during the analysis phase of Fusion is the interface model. This model produces a description in terms of events and the change of state that these events cause. The interface model itself consists of two separate models. One is the operation model and the other is the life-cycle model. The operation model specifies the behaviour of the system operations declaratively by defining their effect in terms of change of state and the events that are output. An example of a specification of one event within an operation model can be found in Figure 10.

Operation:	submit_assignment
Description:	Submits an assignment to a staffing customer.
Reads:	**supplied** person: temporary_worker
Changes:	order
	resource
	new assignment
Sends:	staffing_customer: {assignment_description}
Assumes:	
Result:	An assignment has been created,
	order has been altered,
	resource has been altered,
	resource has been matches to assignment, and
	The assignment_description has been sent to the staffing customer

Figure 10: Fusion object model scheme example

The life-cycle model is the second model of the interface model. While the object model scheme describes the behaviour of an individual system operation, the life-cycle model describes the behaviour from the wider perspective of how the system communicates with its environment from creation until its death. The life-cycle model defines the allowable sequences of interactions that a system may participate in during its lifetime. An example of a life-cycle can be found in Figure 11.

```
lifecycle SIDES: (Order | Resource)*.Assignment.(Timesheet | Invoice)*
Order        =  submit_order . ((submit_assignment |
                end_assignment) | (generate_timesheet | modify_timesheet |
                approve_timesheet)*) . submit_acceptance
Resource     =  propose_staffing_resource . ((submit_assignment |
                end_assignment) | (generate_timesheet | modify_timesheet |
                approve_timesheet)*) . end_resource
Assignment   =  submit_assignment . (generate_timesheet | modify_timesheet |
                approve_timesheet)* . end_assignment
Timesheet    =  generate_timesheet . modify_timesheet*. approve_timesheet
Invoice      =  submit_invoice . (generate_timesheet | modify_timesheet |
                approve_timesheet)* . approve_invoice
```

Figure 11: Fusion life-cycle model scheme example

3.2.3 *Not Yet Another Method (NYAM)*

NYAM is, just as the Fusion technique, a combination of parts of existing techniques. The technique is described in [Wieringa, 2003] and describes a spectrum of notations that can be used from flyweight to heavyweight.

The entire technique is presented as a software engineering technique, whereby the modelling of the domain is part of the technique. Figure 12 gives an overview of the different descriptions that could be made according to the NYAM technique, whereby the design proceeds roughly from left to right. When working from the left to the right we make a progression from flyweight descriptions to descriptions of increasingly heavier weight. For the domain modelling part, only the top models, described as 'Use environment' are important. The other descriptions are relevant when one wants to perform the complete software engineering process [Wieringa, 2003].

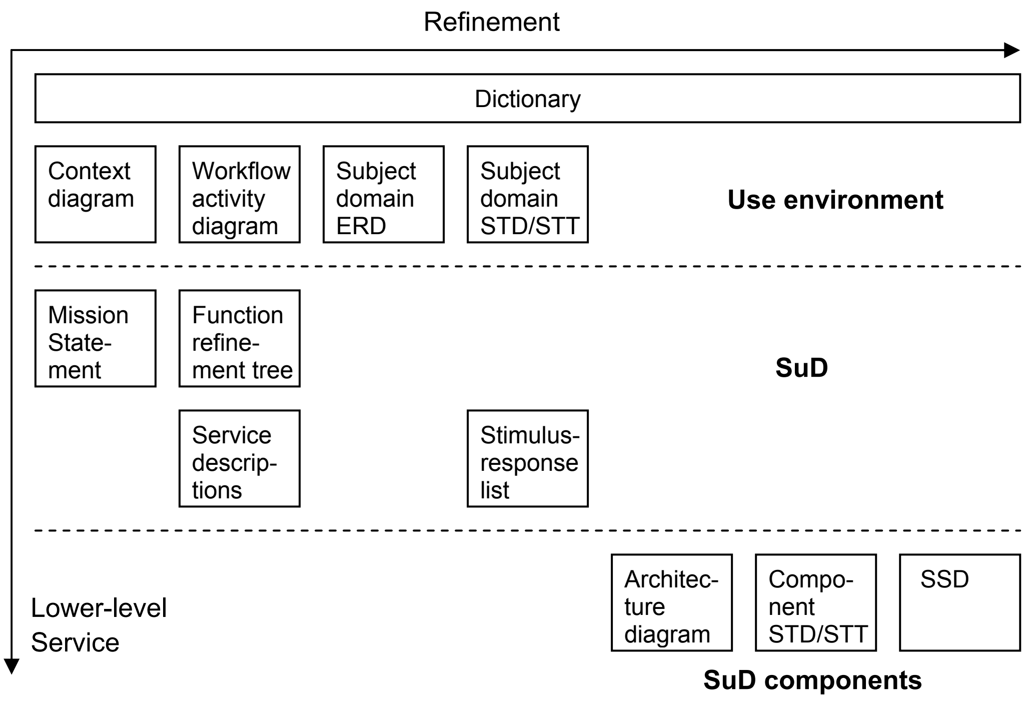


Figure 12: NYAM design process [Wieringa, 2003]

The four models that are included in the domain model are: a context diagram, a workflow activity diagram, a subject domain ERD and subject domain STDs/STTs. A short description of each of these models is given in the following paragraphs.

The context diagram is a communication diagram that represents the System under Development (SuD) and the entities and communications in its context. The entities that are represented in the diagram are the entities that are given to the designers. Different kinds of entities can be present in the environment of the system, these entities are:

– Physical entities, such as devices, natural objects, and people
– Conceptual entities, such as organization, promises, and vacation rights
– Lexical entities, such as contracts and specifications

An example of a context diagram can be found in Figure 13.

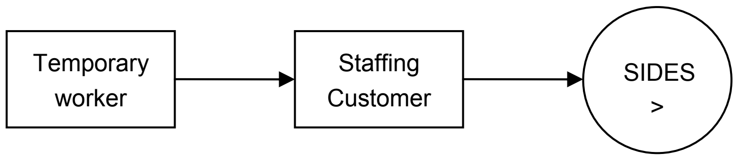


Figure 13: NYAM context diagram example

The workflow activity diagram is a model of the workflow in an environment model. It represents the workflow that is to be supported in the system. An example of a workflow activity diagram is given in Figure 14.

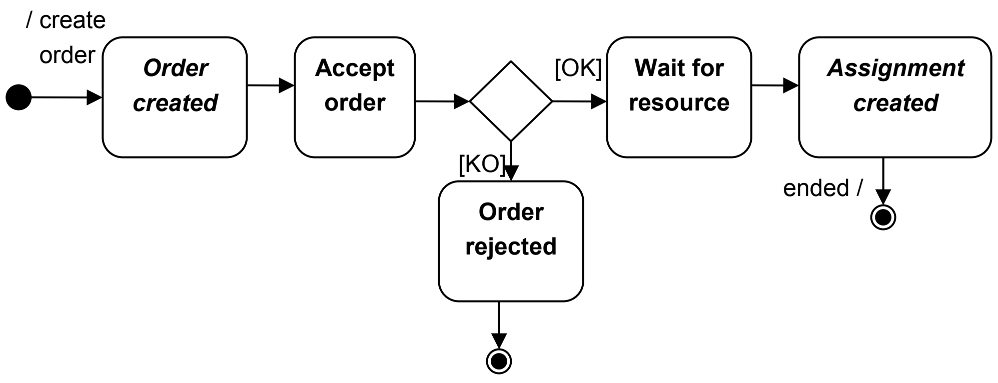


Figure 14: NYAM workflow activity diagram example

The subject domain ERD model is an ERD of the subject domain of the system under development. It is the union of the subject domains of all messages that cross its external interface. An example of a subject domain ERD model is presented in Figure 15.

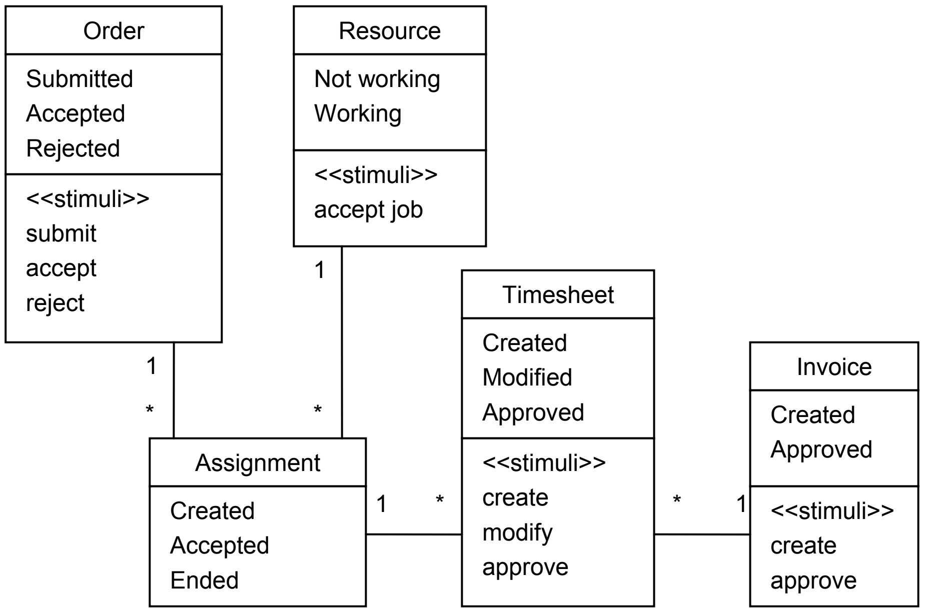


Figure 15: NYAM subject domain ERD example

The last descriptions that are part of the domain model in the NYAM are the subject domain STDs and State Transition Tables (STTs). The STD models present the state transitions in a picture, while the STT present them in a table. The transitions that are included in the STDs and STTs are all transitions that are relevant for the system under development. An example of an STD can be found in Figure 16.

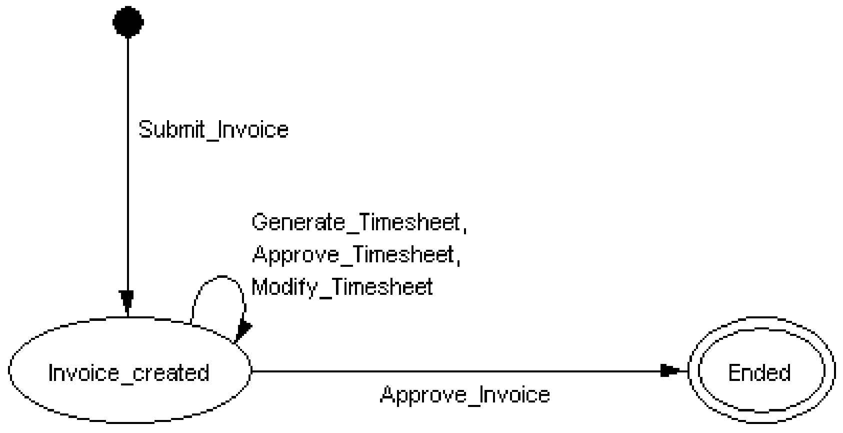


Figure 16: NYAM subject domain STD example

Table 9 gives an example of a STT that describes the same transitions as presented in the STD in Figure 16.

Table 9: NYAM subject domain STT example

Current timesheet state	Event	Next timesheet state
Created	modify	Modified
	approve	Approved
Modified	modify	Modified
	approve	Approved

3.3 Suitability criteria domain modelling technique

This section presents the suitability criteria that are defined to be able to choose the most suitable domain modelling technique. The domain modelling techniques that are compared have to be used by TNO to create domain models for identifying overlap between electronic messaging standards. The criteria that are defined are therefore deduced from meetings with researchers at TNO. Three suitability criteria are defined:

1. Created diagrams
2. Modelling rules
3. Match with existing techniques used at TNO

In the following three subsections the suitability criteria are presented.

3.3.1 Created diagrams

This subsection presents the 'created diagrams' criterion. This criterion is based on the definition of the models that should be contained in an electronic messaging standard as defined by TNO and explained in section 2.2. This definition specifies that the following six diagrams need to be present in a good domain model:

- Context diagram
- Class diagram

- ERD
- State transition diagram
- Event list
- Object event table

The most suitable method should therefore specify the creating of at least the biggest part of these models to be useable to create domain models that meet the requirements as defined by TNO.

3.3.2 *Modelling rules*

This subsection presents the 'modelling rules' criterion. Besides specifying the diagrams that are created with a domain modelling technique, a domain modelling technique can also describe rules and guidelines that should be followed when modelling. An example of this is the existence dependency rule when creating a ERD with Merode. This rule ensures that all relations that are specified are existence dependent and therefore helps the modelling user in creating complete and consistent models. Another example of this kind of rules are rules that specify constraints between models to ensure consistency. This kind of rules can help users in creating comparable and complete models and this is important if models need to be compared to other models.

3.3.3 *Match with existing techniques used at TNO*

This subsection presents the third criterion, 'match with existing techniques used at TNO'. The research performed is aimed at advising TNO on how to handle the overlap between SIDES and SEP, but the results from this research could also be used in the future to identify and handle overlap between other standards. It would therefore be ideal if the techniques used in this research match up with the techniques already used at TNO to enable a smooth introduction and use of the proposed domain modelling technique.

3.4 Comparison of domain modelling techniques

In section 3.2 three domain modelling techniques were introduced. Section 3.3 presented three criteria to decide which domain modelling technique is most suitable for TNO to use to identify the overlap between SIDES and SEP. This section compares the three domain modelling techniques with the help of the suitability criteria and concludes with a choice for the most suitable method for TNO to identify overlap between SIDES and SEP.

The first criterion that is used to compare the three domain modelling techniques is 'created diagrams'. This criterion is based on the definition of a good standard as defined by TNO and is used to determine which models should be available for a good standard. Part of these models make up the domain model, thus the following models should be created by the chosen domain modelling technique:

- Context diagram
- Class diagram
- ERD
- State transition diagram
- Event list
- Object event table

The creation of a context diagram is only specified in the NYAM technique, the creation of a class diagram is specified in the Merode technique and the Fusion technique, the creation of an ERD is specified in all three techniques. Even though the notation techniques are different, state transition diagram and event lists are also specified by each domain modelling technique. The creation of an object event table is a specific technique of Merode and is therefore not present in the other two domain modelling techniques.

The second criterion is 'modelling rules' and this criterion is about additional rules specified by the domain modelling technique besides the 'standard' diagram modelling rules. This can for instance be consistency checks between two different models, or extension on existing models to help the creator of the models to create comparable models. Merode specifies a number of modelling rules that help the creator of models to create consistent models and that help different creators to create comparable models of one system. One of the rules specified by Merode is the existence dependency relation that specifies that the life of one object should be embedded in the life of another object. If this is not the case between two objects, they may not be directly connected and a third object is then needed. An example of this is the hire of a temporary worker. The objects in this case are the temporary worker and the staffing agency which are not existence dependent; the temporary worker can be hired through another staffing agency, the staffing agency can hire another temporary worker. To create an existence dependency graph a third object, the assignment, is needed. This object is existence dependent on both the temporary worker and the staffing agency. Without one of these objects the assignment cannot exist. Other rules that are specified by Merode are rules to keep the Object Event Table consistent with the ERD. There is for instance a rule that specifies that each object should be a column in the object event table, and that for each object there should be at least an event that creates the object and an event that ends the object. Fusion and NYAM do not specify this kind of rules; they both specify which models to use and when to create which model, but they do not specify rules to compare the created models.

The third and final criterion is 'match with existing techniques used at TNO'. As TNO has already done work in the standardisation environment, there are already techniques that have been used and models that have been created. Most of the techniques used within TNO are object-oriented techniques and the three domain modelling techniques all match up. One of the techniques that is compared, Merode, has actually already been used before at TNO, and therefore has preference above the other two methods.

The three suitability criteria showed that the three techniques are quite similar and that they all can create a good domain model of a standard that can be used to compare two standards. A choice of the preferred method to be used in this research can therefore not be made on the diagrams for representing events and actions. The methods are too closely related and are based on the same ideas, only the way of representation is different. The main differences between the techniques are based on the modelling rules and the match with existing methods at TNO. Based on these differences the Merode technique is chosen to create the domain models. The existence dependency relation that is part of Merode can help the creator of the models to create comparable models and the checks between the different models ensure consistency. Furthermore, this technique has already been used at TNO before and the use of the same technique for identifying overlap between two standards ensures that the results from this research can easily be reused for other projects within TNO in the future.

3.5 In summary

The standardisation process that is currently used by many standardisation institutions seems to be missing one important part: the creation of a domain model on which the standard is based. A mapping to the Model Driven Architecture, a software developing architecture, shows how the domain model can help in creating good standards. To create a domain model one needs a domain modelling technique. Three techniques, Merode, Fusion and NYAM have been compared. Merode is the most suitable of these techniques to create domain models that are comparable and that TNO can use to identify overlap. Merode will therefore be used in this research to identify and solve overlap between electronic messaging standards.

4 Domain models SIDES and SEP

This chapter presents the domain models for SIDES and SEP. The domain models are reverse engineered from the specifications of the SIDES and SEP standards because no domain models were present for the standards. To be able to do the reverse engineering, the specifications of SIDES and SEP and accompanying information that is available from the HR-XML consortium and other sources were used [Enting et. al, 2006; HR-XML, 2002; SEP, 2007; SIDES, 2007]. The domain models presented here are created with the help of the Merode modelling technique in the tool Mermaid [Mermaid, 2007] and are in accordance with the TNO specification of a complete domain model in section 2.2. The use of the tool Mermaid to create Merode domain models is explained in Appendix IV.

4.1 Domain model SIDES

This section presents the domain model of the SIDES standard. The actors, the roles these actors can play, a context diagram, a class diagram and an event list are presented. Furthermore state transition diagrams for each class in the class diagram are also presented.

The actors that are present in the SIDES standard are:

- Staffing supplier
- Staffing customer
- Intermediary
- Temporary worker

The roles that can be played by these four actors are:

- Supplier
- Customer

The context diagram that relates the actors to the SIDES standard can be found in Figure 17. In this context diagram the staffing customer and the staffing supplier have a direct relation to the SIDES standard, they directly use the standard to exchange information. The temporary worker only has an indirect relation with the SIDES standard, information about the temporary worker is exchanged using the SIDES standard, but the temporary worker does not directly interact with the SIDES standard. The intermediary also has an indirect relation with the SIDES standard; it is only used as a means to exchange information between the staffing customer and staffing supplier.

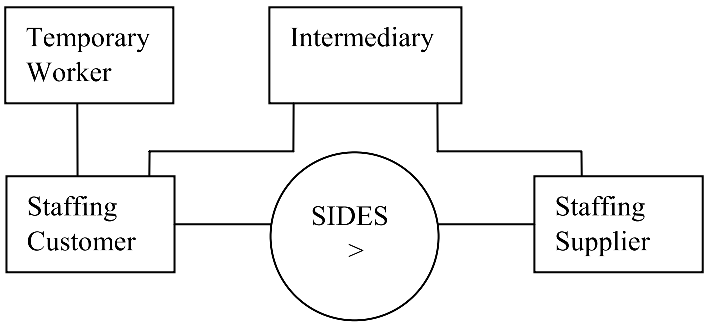


Figure 17: SIDES Context diagram

The entity relationship diagram that was created with Merode can be found in Figure 18. The legend for the Merode Entity Relationship diagram notation can be found in Figure 19. The classes that were identified are: Assignment, Order, Resource, Invoice and Timesheet.

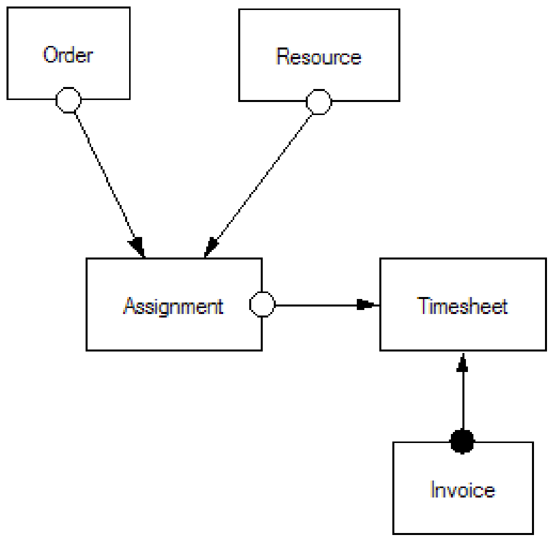


Figure 18: SIDES Entity Relationship diagram

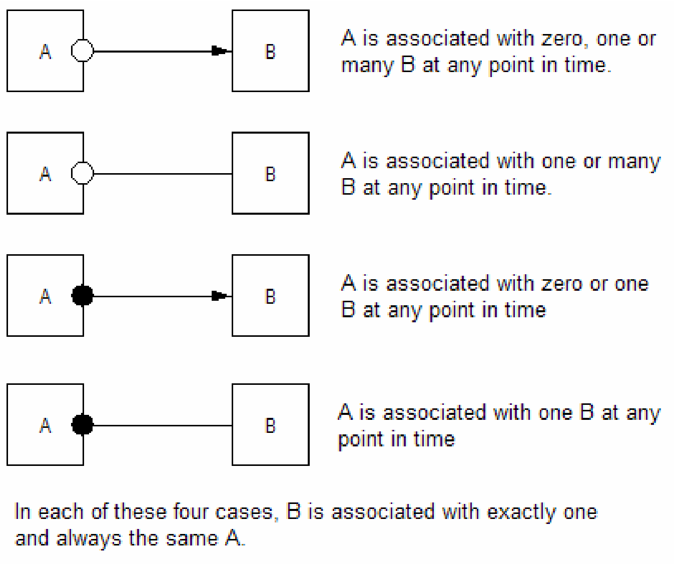

A
B
A is associated with zero, one or many B at any point in time.
A
B
A is associated with one or many B at any point in time.
A
B
A is associated with zero or one B at any point in time
A
B
A is associated with one B at any point in time
In each of these four cases, B is associated with exactly one and always the same A.

Figure 19: Merode ERD notation technique legend

The assignment object is existence dependant on the order and resource objects. The timesheet object is existence dependant on the assignment object and the invoice object is existence dependant on the timesheet object.

The complete class diagram of SIDES can be found in Appendix V. It is not presented in this chapter because it is too big to be readable at this place.

The next part of the SIDES domain model is the event list. This list is represented with an Object Event Table as described in Merode. The objects are presented on the columns of the table, the events are presented on the rows. A ‘C’, ‘M’ or an ‘E’ on a row-columns point of intersection indicates that this particular event creates, modifies or ends occurrences of the object. The ‘C’, ‘M’ and the ‘E’ can be preceded by an ‘O’ or an ‘A’. The ‘O’ indicates that it is an owned event of the object, the ‘A’ indicates that the event is acquired through existence dependency. For example, the O/C for the Submit_Order event means that this event is owned by the Order object and that the event creates the Order object. The A/M for the object Resource and the event End_Assignment indicate that the event is acquired by the Resource object, in this case from the assignment object and that the event modifies the Resource object. The Object Event Table (OET) for the SIDES standard can be found in Table 10.

Table 10: SIDES Object Event Table

Event ↓ Object →	Order	Timesheet	Assignment	Invoice	Resource
Submit_Order	O/C				
Submit_Acceptance	O/E				
Generate_Timesheet	A/M	O/C	A/M	A/M	A/M
Modify_Timesheet	A/M	O/M	A/M	A/M	A/M
Reject_Timesheet	A/M	O/M	A/M	A/M	A/M
Approve_Timesheet	A/M	O/E	A/M	A/M	A/M
Submit_Assignment	A/M		O/C		A/M
End_Assignment	A/M		O/E		A/M
Submit_Invoice				O/C	
Modify_Invoice				O/M	
Reject_Invoice				O/M	
Approve_Invoice				O/E	
Propose_Staffing_Resource					O/C
Reject_Staffing_Resource					O/M
End_Resource					O/E

From the entity relationship diagram and the object event table the state-transition diagrams for the five objects can also be deduced. These can be found in Figure 20, Figure 21, Figure 22, Figure 23 and Figure 24.

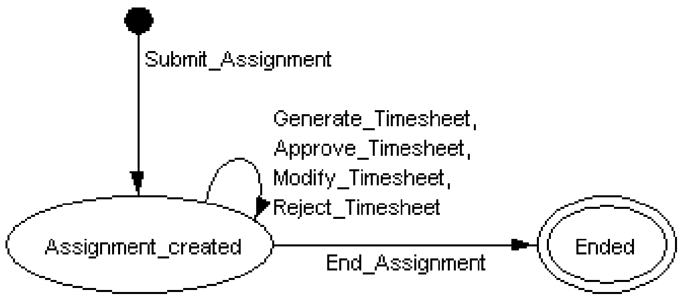


Figure 20: STD Assignment object SIDES

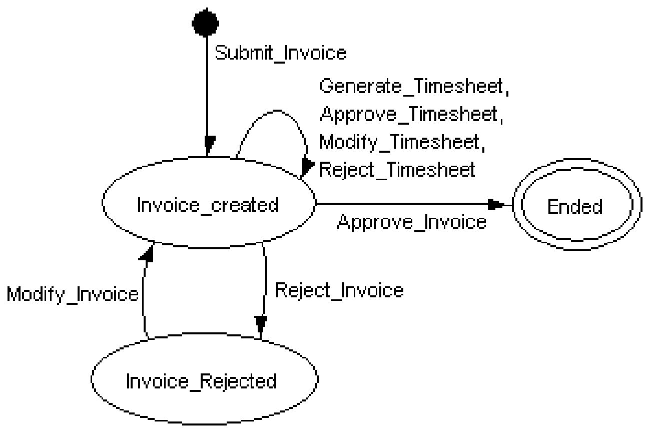


Figure 21: STD Invoice object SIDES

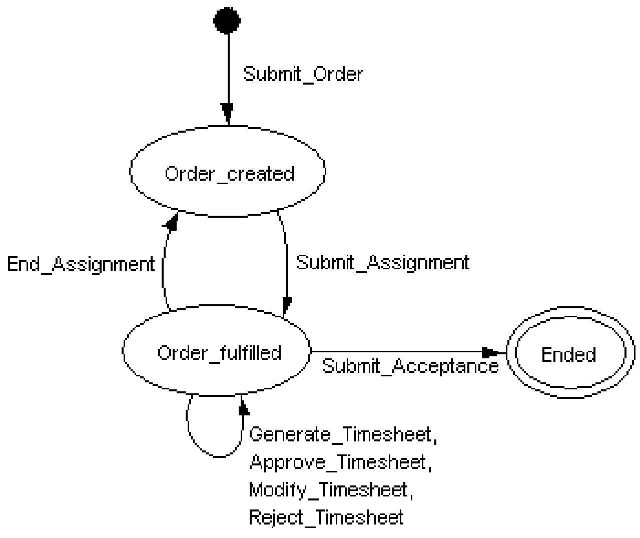


Figure 22: STD Order object SIDES

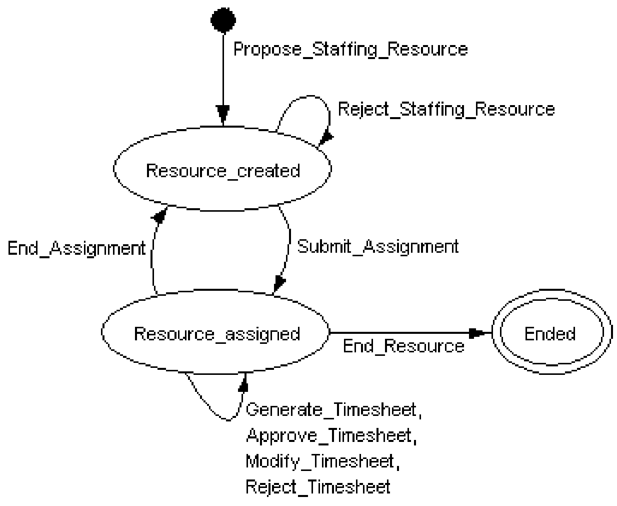


Figure 23: STD Resource object SIDES

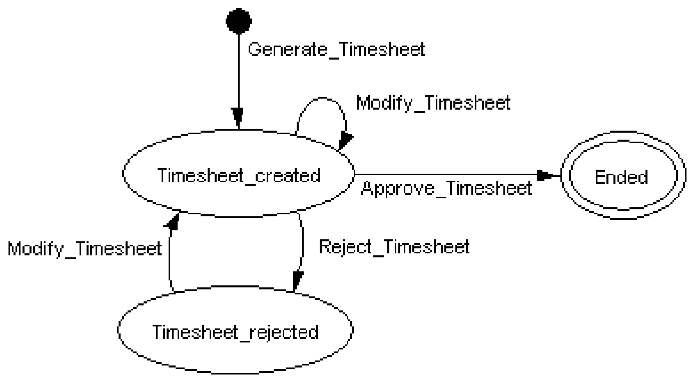


Figure 24: STD Timesheet object SIDES

4.2 Domain model SEP

This section presents the domain model of the SEP standard, just like the domain models of the SIDES standard in the previous section. The actors, the roles these actors can play, a context diagram, a class diagram and an event list are presented. The state transition diagrams for each class in the class diagram are also presented.

The actors that are present in the SEP standard are:

- Staffing supplier
- Staffing customer
- Temporary worker
- Job board

The roles that can be played by these four actors are:

- Supplier
- Customer

The context diagram that relates the actors to the SEP standard can be found in Figure 25. In this context diagram we see that the staffing customer, the staffing supplier and the job board have a direct relation to the SEP standard. They directly use the standard to exchange information. The temporary worker only has an indirect relation with the SEP standard; information about the temporary worker is exchanged with the SEP standard, but the temporary worker does not directly interact with the SEP standard. For instance, when a temporary worker applies for a job, the staffing supplier creates a message that is sent to the staffing customer. Information about the temporary worker is exchanged with the help of SEP, but the interaction is performed by the Staffing Supplier and Staffing Customer

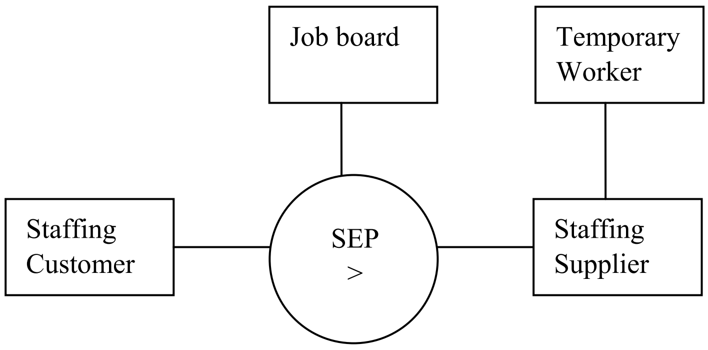


Figure 25: SEP Context diagram

The entity relationship diagram created with Merode can be found in Figure 26. The classes that were identified are: Position, Candidate and Assignment.

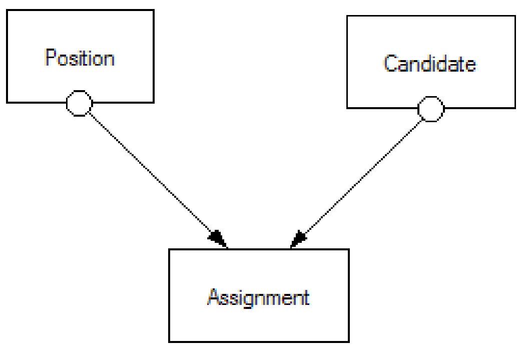


Figure 26: SEP Entity Relationship diagram

The assignment object is existence dependent on the objects position and candidate. The complete class diagram with all its attributes can be found in Appendix VI. The assignment class has no attributes assigned to it because it is created to ensure existence dependency in the entity relationship diagram. The attributes that are assigned to the classes are taken from the specification of the SEP standard, and do not include specific attributes for the assignment class.

The next part of the SEP domain model is the event list. This list is represented with an Object Event Table as described in the Merode method; the event list can be found in Table 11.

Table 11: SEP Object Event Table

Event ↓ Object →	Position	Candidate	Assignment
Create_Requisitions	O/C		
Post_Requisitions	O/C		
Transfer_Applicant_Information	A/M	A/M	O/C
Transfer_New_Hire_Information	A/M	A/M	O/M
Candidate_Submit		O/C	
Post_Position	O/C		
Enter_Candidate		O/M	
Match_Position	A/M	A/M	O/C
Submit_Matched_Candidates	A/M	A/M	O/M
Candidate_Applies		O/C	
End_Assignment	A/E	A/E	O/E
Cancel_Position	O/E		
Accept_Assignment		O/E	

From the entity relationship diagram and the object event table the state-transition diagrams for the five objects can also be deduced. These can be found in Figure 27, Figure 28, and Figure 29.

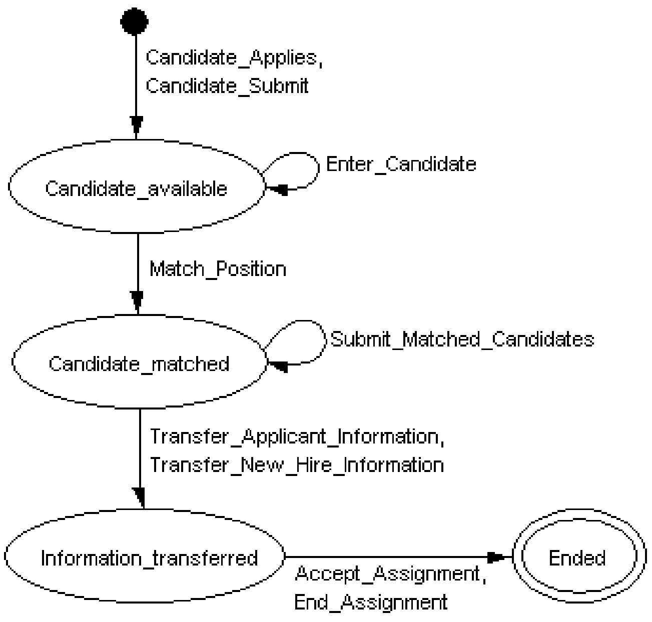


Figure 27: STD Candidate object SEP

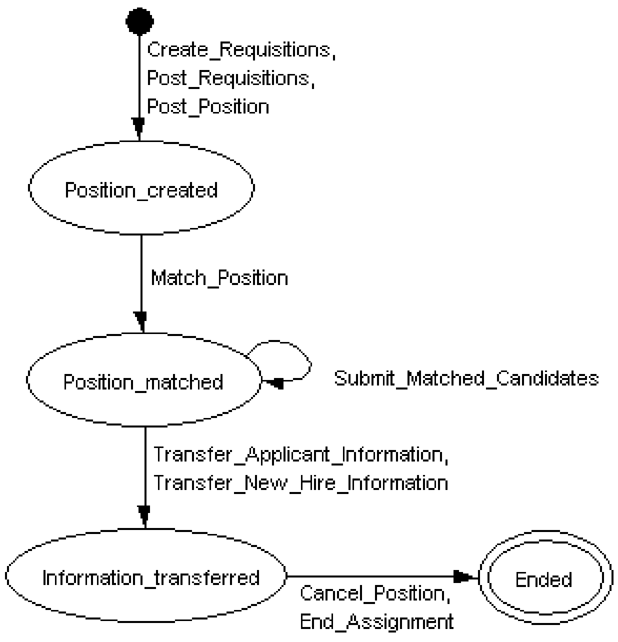


Figure 28: STD Position object SEP

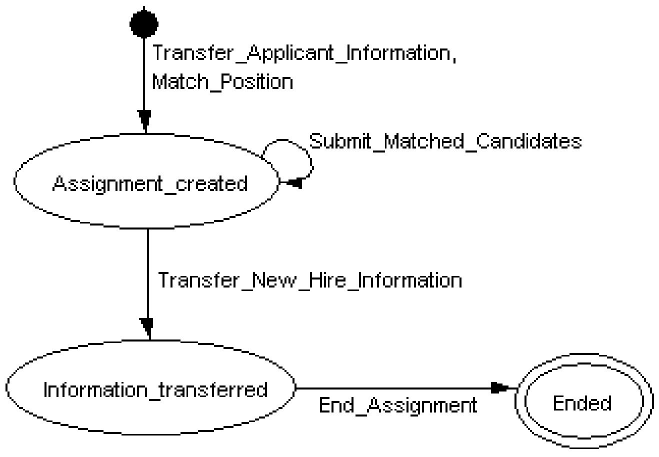

Transfer_Applicant_Information,
Match_Position
Submit_Matched_Candidates
Assignment_created
Transfer_New_Hire_Information
Information_transferred
End_Assignment
Ended

Figure 29: STD Assignment object SEP

4.3 In summary

Domain models consisting of a context diagram, an entity relationship diagram, a class diagram, an object event table and STDs for each class in the class diagram are created for both SIDES and SEP with the help of the Merode modelling technique. The use of Merode with its existence dependency requirement has created models that are comparable and can be used to identify overlap.

5 Identifying and solving overlap

This chapter shows how the domain models that are created with the help of Merode can be used to identify and solve overlap between standards. Because overlap cannot be identified and solved with only the created models, the semantics of the standards are also discussed in this chapter as they play an important role. The identification of overlap and the related semantics are discussed first, after which eight methods for solving overlap are presented and discussed. Suitability criteria to compare the solving methods are presented hereafter and the most suitable solving method for each overlap type is presented.

5.1 Identifying overlap

This section describes how overlap can be identified with the use of the domain models that were created. The identification of overlap between standards is split into two parts. The first part is based on the static part of the domain model, and is presented in subsection 5.1.1. The second part is based on the dynamic part of the domain model, and is presented in subsection 5.1.2. The third subsection indicates the importance of semantics during the identification process.

5.1.1 Standards static conflicts

Section 2.2 presented a definition of a messaging standard as created and used by TNO. It further presented a division between a static and a dynamic part. This section dives deeper into the static part of a standard and presents ways to classify and locate conflicts. The next section dives deeper into the dynamic part of a standard and presents a division in types of conflicts for that part. Guidelines are given that help to determine when to call a standard to be conflicting with another standard.

To find overlapping and conflicting situations in the static part, a top-down division is defined. A high-level overview of this top-down division is presented first. After this the use of this division in identifying overlap in the static part is discussed.

A three-level top-down approach is used to identify overlap and conflict in the static model. The three levels defined are:

1. Model level
2. Object level
3. Attribute level

The model-level consists only of an overall class diagram or ERD of which an example can be found in Figure 30. Only the relations of the objects with other objects are taken into consideration to identify whether there are parts of the two standards that are overlapping. When objects are found in the two standards that are overlapping, but not relate to the other objects in a comparable way, this is called a conflict. This thus results in a model conflict.

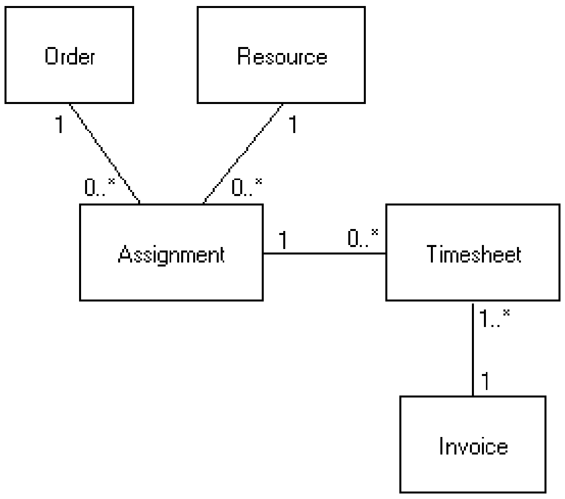


Figure 30: Example Entity Relationship Diagram

At the object-level the individual object themselves, with their attributes, are looked at to see whether objects are comparable, and to see if there are differences between the objects with regard to the attributes. Conflicts found on this level can be of different types. The objects can be identical in behaviour but define different attributes, or there are objects in the two standards that have identical attributes, but their behaviour does not match. An example of the diagram that is used for this comparison can be found in Figure 31.

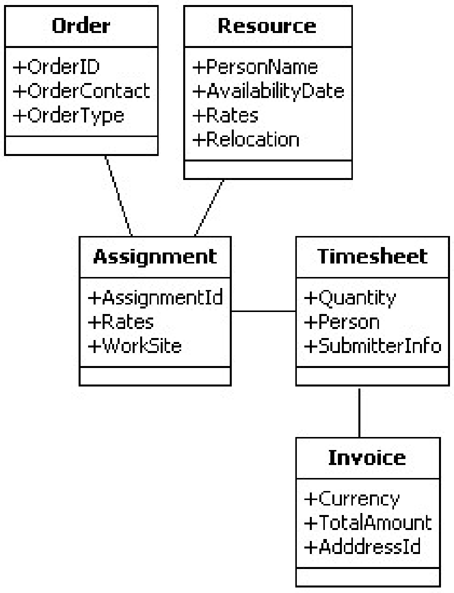


Figure 31: Example class diagram with attributes

Finally, at the attribute-level, the attributes themselves and the way in which they are represented are compared. This is a low-level view, and conflicts at this level might not look to troublesome, but in some cases conflicts at this level can cause problems when implementing or converting already existing systems. An example of the model that is used for this comparison, which is an extension of the model presented in Figure 31, can be found in Figure 32.

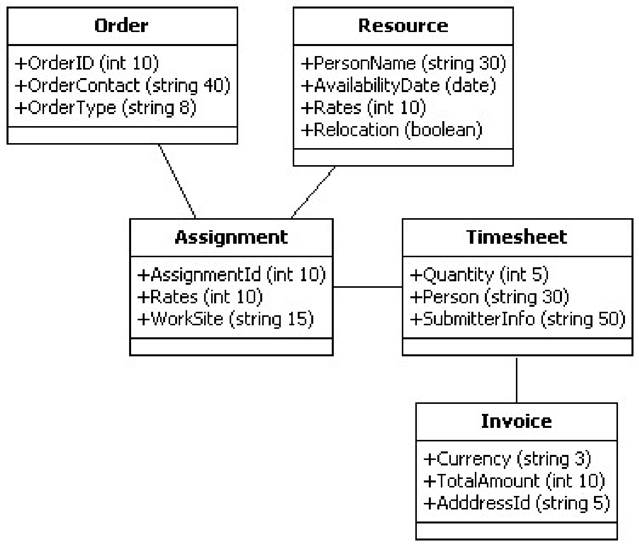


Figure 32: Example class diagram with attribute specifications

As already mentioned, the conflicts on the lowest level (the attribute-level) are the conflicts that are most likely the easiest to solve. It is therefore assumed that these conflicts are not problematic enough to cause real conflicts between standards. Therefore, two standards are called to be conflicting if in the static part a conflict arises at the object-level or at the model-level. In case that happens the two standards are said to be in conflict, and actions have to be taken if the standards are supposed to be working together. To define two standards to be in conflict judging on the static part of the domain model, the dynamic part of the domain models is not taken into consideration. Conflicts found at the top two levels are severe enough to call the standards in conflict. The next section presents guidelines indicating when to call a standard to be in conflict when looking at the dynamic part of the domain model.

5.1.2 Standards dynamic conflict

In the previous section a division between a model conflict, an object conflict and an attribute conflict in the static part of a domain model was presented. For the dynamic part of a standard, the events, this division cannot be used. In [Moffett & Sloman, 1993] the authors describe the possibilities of overlap between policies and give means to identify these conflicts. In this subsection the results from [Moffett & Sloman, 1993] for policy conflict are projected on the overlap between events in the standardisation process. Furthermore an adapted version of their division for the events in standards is presented.

Moffett and Sloman define four possibilities of overlap between policies (subject overlap, target overlap, double overlap and subjects-targets overlap). Only the first three of the mentioned four possibilities are also applicable to overlap between events in standards. The following sections refer to ‘subject’ and ‘target’ objects. In relation to events the ‘subject’ is the action that is performed on an object (create, modify or end); the ‘target’ is the object or action that is affected by the event. The three possibilities of overlap correspond to the various combinations of overlap between objects in the ‘subject’ and ‘target’ object sets of the events. The possibility ‘subjects – targets overlap’ that Moffett and Sloman defined is not applicable to overlap between events because the subjects and targets in the research are not identical entities; the subjects are actions and the targets are objects.

Table 12: Types of overlap between events

Type of overlap	Definition
Double overlap	Both the subjects and target objects of the two events overlap
Subject overlap	The subject objects of the two events overlap
Target overlap	The target objects of the two events overlap

The definitions of the three possibilities of overlap in Table 12 give the possibility to make a distinction between the different types of overlap. To put the three possibilities in some more perspective, the next three paragraphs give a description and an example for each of the possibilities.

Double overlap

A double overlap occurs when both the subjects and the targets of two events overlap. If that occurs, chances are big that the two events send a message with an identical meaning, which perhaps can be combined to resolve the overlap. This kind of overlap is important in locating overlap between messaging standards with the help of events.
An example of this conflict in SIDES and SEP is the posting of a position in SEP and the creation of an order in SIDES. They each intend to create a job vacancy that can be fulfilled by a job seeker. Both of the messages are create actions, thus their subjects overlap, and the objects they affect, position in SEP and order in SIDES, are also comparable.

Subject overlap

A subject overlap occurs when the subjects of two events overlap. When this occurs it implies that the same action is performed on the object related to the events. This kind of overlap is not directly important in locating overlap between standards because the objects that are affected can be very different. The reason for indicating this overlap type is that it is important to know how the subjects of events overlap to be able to locate a double overlap. An example of this conflict in SIDES and SEP is the modify action performed on a timesheet in SIDES and the modify action performed transferring new hire information in SEP. The actions performed are equal, both events do a modify action, but the objects affected (assignment in SEP and timesheet in SIDES) are totally different.

Target overlap

A target overlap conflict occurs when the target objects of two events overlap. When this occurs it implies that two events both do an action on the same target, the actions can be different. This kind of overlap is, just like the subject overlap, not directly important in locating overlap between standards. The reason for this is that the actions performed on the object can be very different. Just as for the subject overlap type, the reason for indicating this type of overlap is that it is important to know how the targets of events overlap to be able to locate double overlap. An example of this conflict in SIDES and SEP is the ending of an assignment in SIDES and the creation of an assignment in SEP. Both actions affect the assignment object, but the actions are completely different (end and create).

All three overlap possibilities help in classifying conflicts between events in different standards. However, it should be noted that most of the work has to be done by a human being. Furthermore, it is important to note that there are numerous of other influences that make it difficult to use an off-the-shelf solution when solving conflicts between events.

Only if a conflict of the type ‘double overlap’ occurs between events when comparing two standards, the standards themselves are defined to be in conflict. The static part of the domain model is not important for this decision and does not place any constraints on defining standards to be in conflict judging from the dynamic part of the domain models. Just like the conflicts in the static parts, the conflicts that can be identified in the dynamic part are severe enough to justify defining the standards to be in conflict.

5.1.3 *Semantics during identification*

The previous sections described how to identify overlap between standards. This section describes the semantics that are of influence on the identification of overlap.
During the identification of overlap between two standards with the help of their respective domain models, the semantics of the models are also important to be able to match objects and attributes. The reason for this is that with only comparing the models themselves, overlap between two standards can not be completely found. If the two standards both use building blocks that are the same, for instance a contact info block, then these blocks can easily be matched and overlap between the standards is found. But, if the standards also define their own elements, then only the semantics of both models can indicate whether two elements are equal. For instance the element PersonName can in two standards both represent the name of the temporary worker, but it is also possible that in one standard the element PersonName represents the name of the temporary worker, while in the other standard the element PersonName represent the name of an employee of the staffing agency. Even in one standard the PersonName element can be used multiple times, and every time be representing the name of another person. Only with the help of semantics it is possible to be sure what the element refers to and for the identification of overlap between standards the semantics of the elements and objects are therefore important factors.

5.2 Solving overlap

In this section eight methods for solving overlap between standards are presented and discussed. If overlap and conflict are not detected until the standards are publicized, the only good solution is to try to solve the overlap. Solving overlap between standards if it is cause to problems for the users is the only good solution. Unfortunately it is often the case that organizations pretend not to see the overlap and conflicts, and just continue as they were doing before. This might be advantageous in the beginning, as a non-overlapping standard might cost money in losing customers to other organization, but in the future it might be cause to serious problems for which the solution costs much more than the profit that was gained by ignoring overlap and conflict.

Because not dealing with overlap in the beginning of the standardisation process may lead to much higher costs later on in the process, it is smarter to invest a little bit more money in the implementation of standards to ensure that problems with overlap and conflict are solved, than to ignore the problems and just create implementations, that later on all have to be adapted by hand because the standards have changed to solve overlap and conflict.

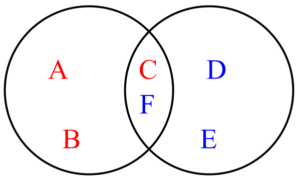
There are many different ways to solve overlap. Each way of solving has its specific characteristics and its advantages and disadvantages. In this section eight solving methods are described and for each method the number of resulting standards (assuming we start of with two partial overlapping standards), the impact of the solving method on existing implementations of the standards and the type of overlap (from set theory) for which the solving method works are indicated. The solving methods are deduced from meetings with researchers at TNO and have been defined with the SIDES and SEP case of the SETU in mind. They can however also be used in other overlap situation, but should be reviewed to ensure that they are general enough. The eight overlap solving methods defined are:

1. Merge standards
2. Extend one standard, drop the other
3. Remove the overlap from one standard
4. Leave the standards as they are, but create clear guidelines on when to use which standard
5. Create a complete new standard and drop the old ones
6. Remove the overlap from both standards and create a new standard for the created gap
7. Define one standard to be prevailing over the other
8. Define mappings for the different messages

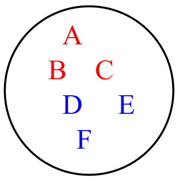
Because a number of these solving methods might look to have the same result, Figure 33 gives a graphical representation of the original standards, and presents the resulting standard(s) when applying the eight solving methods.

In the graphical explanation of the solving methods two standards are used. In the pictures the circles represent the standards, the letters denote the messages within the standards, colours are used to help clarify the different standards. Standard 1 contains the messages: A, B and C (indicated in red). Standard 2 contains the messages: D, E and F (indicated in blue). New messages that are created when using the handling methods are indicated in green.

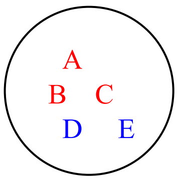
The overlap between the standards lies between message C in standard 1 and message F in standard 2 this are different messages with the same goal. The overlap is an example of an intersection relation. This is depicted in the following figure:



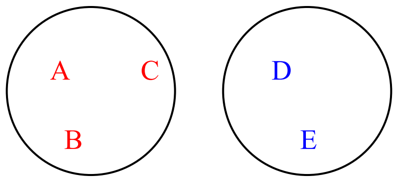
When performing solving method 1, 'merge standards', the resulting standard looks like the following figure:



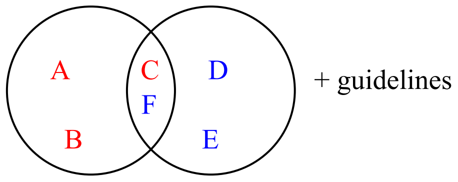
When performing solving method 2, 'extend one standard and drop the other one', the results look like this (message F is taken together with C, thereby resulting in a standard with 5 messages):



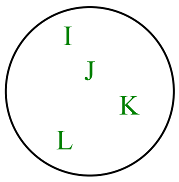
The third method that is described is 'remove the overlap from one standard', after applying this method the result looks like this:



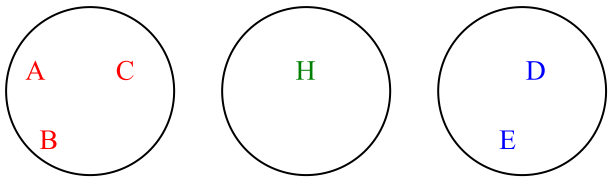
When applying solving method 4, 'leave the standards as they are, but create clear guidelines', the results look like this:



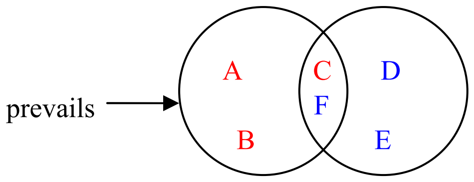
After performing solving method 5, 'create a complete new standard and drop the old ones', the result looks like the picture below:



The sixth method, 'remove the overlap from both standards and create a new standard for the created gap', result in the following standards:



The seventh method, 'define one standard to be prevailing over the other', gives the following result:



The last, and eighth method, 'define mappings for the different messages', gives the following result:

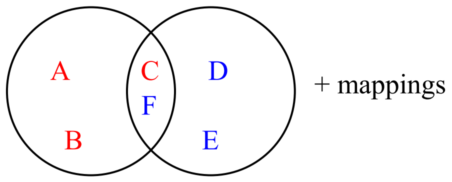


Figure 33: Overlap solving methods

Now that an indication is given of what the resulting standards will look like, and what the differences between the solving methods are, each of these solving methods is further described and discussed in the following seven subsections.

5.2.1 *Merge standards*

A solving method that may immediately pop in ones mind when thinking of solving the problem of overlap between standards is merging the two standards. This might be a good way of solving overlap, but there are also numerous problems that can arise when choosing this solving method. One of the first problems that may come up is that in case of messaging standards, the contents of the messages might be very different while the meaning of the messages is equal, making it difficult to take them together without creating a standard that is almost impossible to implement and use. Another problem is that when merging the two standards, the resulting standard will often be bigger than the original standards, it will contain more messages, or the messages are bigger. This can create problems for implementers of the standard, as they will also have to support all new and / or extended messages. An advantage of this method is that there will be one resulting standard that describes everything from the two original standards and thus removes the overlap. This solving method can be used for almost all relations defined in section 2.3.2, the section that presents three types of overlap. One can merge a subset from the subset relation with its superset, one can merge two standards that both describe the same domain (the equal relation) and one can merge two standards that only partly overlap (the intersection relation).

5.2.2 *Extend one standard, drop the other*

The second solving method defined is extending one of the two standards, and dropping the other one. The advantage of this method is that it results in one standard and that the overlap is removed. Another advantage, especially when comparing it with the first method we defined, is that the duplicate message is actually removed. In the merge operation all messages are taken together in one standard, in this approach the duplicate message is removed and the other messages are added to the standard. A disadvantage that is equal to one of the disadvantages of the first method is that the new standard will be bigger than the original standards, possibly resulting in problems for implementations of the standards. When looking at the types of overlap that were defined section 2.3.2, it becomes apparent that this solving method is best suited to handle the intersection relation. By extending one of the two standards with the messages of the other standard that not overlap, a new standard is created. This standard is capable of performing all the interactions that were possible with the original standards.

5.2.3 *Remove the overlap from one standard*

Removing the overlap from one standard and leaving the other standard intact is the third solving method that is defined. If one uses this method to handle the overlap, the result will be that there are still two standards, but that the overlap is removed and so the two standards are disjoint. An advantage of this method is that implementers of the standard that is untouched do not have to change their implementations. Only the implementers of the standard from which a part is removed have to implement the replacing message(s) from the other standard. A disadvantage of this method is that there will be two resulting standards which, most probably, keep their existing names. It is therefore important to explicitly point out to everyone who uses the standards that the changed standards are not backwards compatible, and that users should switch to the new standards to prevent overlapping situations from occurring. This solving method can be used for more than one type of overlap. It can be used with a subset relation, but then the overlap should be removed from the superset; otherwise the solving method will result in only one standard which is not the intention of this solving method. The solving method can also be used on the intersection relation. In this case one standard becomes a little bit smaller and the intended result of this solving method is reached.

5.2.4 *Leave the standards as they are and create guidelines*

The fourth solving method defined is to leave the standards as they are, but to create clear guidelines which describe when to use which standard. Evidently, this results in two standards plus a number of guidelines. An advantage of this method is that the standards themselves are not changed, thus implementers of a standard do not have to change their implementations to keep supporting the original standard. But, even though there is no direct change in the standards, it might very well be possible that implementations have to be changed because the guidelines prescribe another message for the action that is performed. These cases might earlier be handled with the original standard, but now have to be handled with the other standard. So, while this solving method might look ideal, it has some serious drawbacks that might result in not solving the problem, but making problem solving even harder. The reason for this is that guidelines will always be subject to a user's opinion, thereby resulting in different implementations that might for instance be incompatible, or again overlapping. The solving method can be used when the overlap can be typed as the subset relation, the equal relation and the intersection relation.

5.2.5 *Create a complete new standard*

Solving method five that is defined, is to drop both existing standards and create a complete new standard. The main advantages of this solving method is that a complete new standard is developed based on the old standards, which can incorporate all old messages within both standards, but can also improve messages or even the overall set of messages. A big disadvantage of this method is that every implementer of one of the old standards has to implement the new standard. This might not be a big problem if the new standard stays closely to the old standards, but if the new standard is very different because of new insights in the domain in which the standard operates, it might be possible that implementations have to be completely rewritten, something that most implementers will not appreciate. The solving method that creates a complete new standard can be used to solve overlap problems related to the subset relation, the equal relation and the intersection relation.

5.2.6 *Remove the overlap from both standards and create a new standard for the gap*

The sixth solving method defined is to remove the overlapping parts from both standards, and to create a new standard for the overlap. This results in three standards, from which two of them are almost the same as in the original situation, only the overlapping part is removed. An advantage of this method is that existing implementations will keep working on every part of the standard except the parts that are removed. A disadvantage is that every implementer of the original standards that wants to keep supporting the overlapping part has to implement the new standard. The new standard for the overlapping part can be completely redesigned, or it can be a modified version of the overlapping parts of the standards that created the overlap. This solving method is most appropriate to solve overlap problems that can be categorized in the intersection relation category. When using this solving method to solve overlap based on a subset relation, the subset relation will disappear and the superset plus a new standard will be left. When using this solving method in case of an equal relation, only the new standard will be left, thereby making this method equal to the fifth method: creating a complete new standard.

5.2.7 *Define one standard prevailing*

The seventh solving method for overlap between standards is the method of defining one standard to be prevailing over the other. In this case, the result will be that there are still two standards, but there is no discussion on which standard to use in the overlapping par. In the overlapping part one has to use the standard that is defined to be prevailing. An advantage of this approach is that only the implementers of the standard that is not declared prevailing have to modify the part of their implementation that implemented the overlapping part. A disadvantage of this solving method is that, while this may work in theory, in the practice this may only work if a government, or other organization that has authority, declares one standard prevailing. Otherwise it may be so that other organisations twist about which standard should be prevailing, and may never solve the conflict. This solving method works in case of an equal relation, a subset relation and an intersection relation. In all of these relations the standard that prevails, only prevails in the overlapping part, which implies for the equal relation the entire standards, but for the subset and intersection relation only a part of the standards.

5.2.8 *Define mappings for the different messages*

The eight, and last, solving method for overlap is the method of defining mappings for the different messages. When this method is applied, the result will be that there are still two standards, which are identical to the standards that existed before applying this method, but that there are clear mappings to translate a message from one standard to the equivalent message of the other standard. An advantage of this approach is, just like with the seventh solving method, that the implementers of the standard do not have to modify their implementation that implemented the overlapping part. A disadvantage of the method is that everyone that uses the standards has to implement an intermediate that can translate messages of one of the standards to messages of the other standard and vice versa. This thus creates work for almost everybody that uses the standards for sending and receiving messages. Only organisations that only send messages and never receive them do not have to implement extensions. This solving method works in case of an equal relation, a subset relation and an intersection relation.

5.3 Filtering solving methods

This section filters the solving methods based on two filters: the type of overlap, and the size of the overlapping part in the standards. The reason for this filtering is because the eight solving methods that are defined in the previous section are not all applicable to each overlap situation. By filtering out the solving methods that cannot be applied to the different situations, choosing the most suitable handling method becomes easier. The first subsection presents and explains the two filters; the second section presents the filtered lists for each combination of the two filters.

5.3.1 *Filters for solving methods*

In this section the two filters, the type of overlap and the size of the overlapping parts in the standards, are presented and discussed.

Type of overlap

The type of overlap is the first filter that is important in choosing a solving method. The reason for this is that not all solving methods are applicable to the three types of overlap (subset, equal and intersection relation). If the overlap is of the subset type the only solving methods that cannot be used are 'extend one standard and drop the other' and 'remove the overlap from both standards and create a new standard for the gap'. The first solving method would have the same result as merging the standards; the second method would drop the subset standards, which is not the intended result of this solving method. If the method 'remove the overlap from one standard' is applied, the removal should take place on the superset standard, otherwise the intended result of this method is not reached. If the overlap is of the equal type the solving methods 'extend one standard, drop the other', 'remove the overlap from one standard' and 'remove the overlap from both standards and create a new standard for the created gap' cannot be applied. The first and solving methods cannot be used because they would have the same result as the 'merge standards' method. The third method cannot be used because it would have the same effect as the method 'create a complete new standard and drop the old ones', which is not the effect that is intended with this solving method. If the overlap is based on the intersection relation, all solving methods can be applied to solve the overlap between the standards.

Table 13 gives an overview of the different solving methods and their applicability on the different types of overlap.

Table 13: Applicable solving methods for each type of overlap

Solving method ↓ Overlap type →	Subset	Equal	Intersection
Merge standards	X	X	X
Extend one standard, drop the other			X
Remove the overlap from one standard	X		X
Leave the standards as they are but create clear guidelines on when to use which standard	X	X	X
Create a complete new standard and drop the old ones	X	X	X
Remove the overlap from both standards and create a new standard for the created gap			X
Define one standard prevailing over the other	X	X	X
Define mappings for the different messages	X	X	X

Size of the overlapping parts in the standards

The size of the overlapping parts in the standards is the second filter for choosing the possible solving methods. The size of the overlapping parts in relation to the entire standards is measured for each of the standards individually. To measure the size of the overlapping part the ERD models of the standards are used. By counting the objects that are involved in the overlap and the total number of objects in the standard, the percentage of overlap can be determined. For example, in SIDES the total number of objects is five, the number of objects that are involved in the overlap is three. The percentage of overlap for the SIDES standards therefore is 60%.

If the overlapping parts of the standards make up the largest part of the standards, removing part of the standard can result in crippled standards that are not useable anymore. In such a case it might be wiser to look for another solving method. A crippled standard is a standard that misses essential parts that are needed for using a standard. For example: the removal of the 'Assignment' object in the SIDES standards cripples SIDES because the information contained in this object is needed in the other objects. If the overlapping parts of the standards only make up a small part of the standards, the removal of parts of the standard may not cause any problem in the use of the maintained part of the standard. In determining if the overlap is large or small, a guideline that can be used is that if the overlapping part is less than 40% of the entire standard the overlap is small. If the overlapping part is more than 70% of the entire standard the overlap is large. These percentages are based on current knowledge within TNO and should be refined in the future with the help of more overlap cases. Cases that are between these two percentages have to be judged individually. Table 14 gives an overview of the different solving methods and their applicability for overlap that makes up the smallest or largest part of the standards.

Table 14: Applicable solving methods based on the size of the overlapping parts

Solving method ↓ Size overlap →	Small	Large
Merge standards	X	X
Extend one standard, drop the other	X	X
Remove the overlap from one standard	X	
Leave the standards as they are but create clear guidelines on when to use which standard	X	
Create a complete new standard and drop the old ones	X	X
Remove the overlap from both standards and create a new standard for the created gap	X	
Define one standard prevailing over the other	X	
Define mappings for the different messages	X	X

If the overlap is small, all solving methods defined can be used to handle the overlap. None of the solving methods results in crippled standards. If the overlap is large, only three solving methods can be used to handle the overlap; these are: ‘merge standards’, ‘extend one standard, drop the other’ and ‘create a complete new standard and drop the old ones’. The solving methods ‘remove the overlap from one standard’, ‘remove the overlap from both standards’ and ‘create guidelines and define one standard prevailing’ result in crippled standards because at least one of the resulting standards becomes too small.

5.3.2 *Filtered solving methods*

This section presents the possible solving methods for each of the six combinations of filters, being subset & small, subset & large, equal & small, equal & large, intersection & small and intersection & large. The six situations and the possible solving methods are presented in Table 15. Each of the different combinations of filtering methods result in a number of solving methods that are applicable for each overlap type and size. To be able to determine the most suitable handling method for an overlap between two standards, criteria are needed. These suitability criteria are presented and discussed in the following section.

Table 15: Filtered solving methods

Solving method ↓	Overlap type filter → Overlap size filter →	Subset		Equal		Intersection	
		Small	Large	Small	Large	Small	Large
Merge standards		X	X	X	X	X	X
Extend one standard, drop the other						X	X
Remove the overlap from one standard		X				X	
Leave the standards as they are but create clear guidelines on when to use which standard		X		X		X	
Create a complete new standard and drop the old ones		X	X	X	X	X	X
Remove the overlap from both standards and create a new standard for the created gap						X	
Define one standard prevailing over the other		X		X		X	
Define mappings for the different messages		X	X	X	X	X	X

5.4 Suitability criteria solving methods

To be able to determine which of the defined solving methods is most suitable to solve overlap between standards, suitability criteria are needed. This section defines five categories of suitability criteria and indicates for each category how the criteria can be measured. It further describes for each category of criteria which solving methods can be excluded based on the criteria. The categories of suitability criteria that are presented here have been deduced from meetings with researchers at TNO and from reading reports about the SIDES and SEP standards. The categories not only apply tot the specific SIDES/SEP case but can also be used in other overlap cases to identify the most suitable solving method. The categories are:

1. Number of resulting standards
2. Changes to be made in existing implementations
3. Satisfaction of current users with the overlapping standards
4. Political influences
5. Authority of the standardisation institution

The following seven paragraphs describe the categories of suitability criteria, indicate how the criteria can be measured and indicate the exact way in which the suitability criteria influence the solving methods.
Table 16 presents an overview of the criteria categories with the measures.

Number of resulting standards
The category ‘number of resulting standards’ judges the solving methods based on the number of standards that the different solving methods create. The original standards are also counted to judge on this criterion, because it is possible that the original standards will also be used next to the proposed solution to solve the overlap. To judge on this criterion a maximum number of resulting standards has to be defined by the person solving the overlap. In case this maximum number is set to three, the solving methods that can be applied are:

– Merge standards
– Extend one standard, drop the other
– Leave the standards as they are but create clear guidelines on when to use which standard

- Create a complete new standard and drop the old ones
- Define one standard to be prevailing over the other
- Define mappings for the different messages

In case this maximum is set to four, the only solving method that cannot be applied is: 'Remove the overlap from both standards and create a new standard for the created gap' as this method creates five standards. In the case that the maximum is set to five or more all solving methods can be applied as they all do not create more than three new standards and therefore do not result in more than five standards.

Changes to be made in existing implementations
To judge the solving methods based on the category 'changes to be made in existing implementations' the solving methods are judged on the necessary changes to the attributes, to the messages and/or the domain models.

If changes to all of these three parts (attributes, messages, models) are allowed, all solving methods can be used to solve the overlap. If it is not allowed to change the attributes, the solving methods that should not be used are 'Extend one standard, drop the other', 'Remove the overlap from one standard' and 'Create a complete new standard and drop the old ones'. These solving methods need changes to the attributes of existing implementations because parts of the standard currently being used are replaced with another standard that defines other attributes. In the case it is not allowed to change the messages, the solving methods that are not allowed are 'Remove the overlap from one standard' and 'Create a complete new standard and drop the old ones' as these solving methods change the existing implementations in such a way that current messages have to be changed to be able to communicate according to the new standard. If it is not allowed to change the domain models of the standards the only solving method that can be used is 'Define mappings for the different messages' as this solving methods do not require a change in the domain models since the messages from the existing implementation can be translated to the messages required for the new standard.

Satisfaction of current users with the overlapping standards
For judging the solving methods based on the category 'satisfaction of current users with the overlapping standards', three judgment criteria are chosen that are appropriate for judging solving methods for overlap between standards. These are 'content', 'format' and 'ease of use' [Doll & Torkzadeh, 1988]. The content criterion is based on the satisfaction of the users with the current information that can be exchanged with the standards. If the users do not want to exchange extra information they are satisfied with the content. If the users want to exchange more information than possible with the standard they are not satisfied about the content. The format criterion is based on how the content is contained in the current standard. For XML based standards this criterion is based on the way the messages are built for example 'logical structure of the messages' and 'logical XML element names'. The final criterion in this category, 'ease of use', is based on the how easy users can work with the overlapping standards. If they have to change their entire work process to work with the standards working with the standards is not easy. If the standards match up with the work process, the standards are easy to use.

If the content is not satisfying, the solving methods 'Create a complete new standard and drop the old ones' and 'Remove the overlap from both standard and create a new standard for the created gap' could be applied because these methods change the content. If the content is satisfying no change to the content is needed and all other solving methods could be applied. If the format of the current standards is satisfying, the solving methods ‘Create a complete new standard and drop the old ones’ and ‘Remove the overlap from both standards and create a new standard for the created gap’ should not be applied as they are likely to change the formal of the standards. All other solving methods can be used to solve the overlap as they all do not need a change to the format of the standards. If the format is not satisfying, the two solving methods that can be used to solve the overlap are ‘Create a complete new standard and drop the old ones’ and ‘Remove the overlap from both standards and create a new standard for the created gap’ as these are the only two methods that are likely to change the formal of the standards. In case the ease of use is satisfying to the users, the two solving methods that should not be used are ‘Create a complete new standard and drop the old ones’ and ‘Remove the overlap from both standards and create a new standard for the created gap’ as they are likely to change the ease of use of the standard. The other solving methods do not drastically change the standards and the ease of use is not likely to change. In case the ease of use is not satisfying, the two methods that can be used are the methods that could not be used if the ease of use is satisfying; ‘Create a complete new standard and drop the old ones’ and ‘Remove the overlap from both standards and create a new standard for the created gap’ as they have the possibility to change the ease of use.

Political influences
Judging the solving methods in the category political influences is based on the criterion ‘preferences of contributors’. Contributors can have a preference for one of the standards because they are involved in the work on these standards or because they already implemented one of these standards.

If the contributors have preferences for one of the standards, the solving methods ‘Merge standards’, ‘Create a complete new standard and drop the old ones’ and ‘Remove the overlap from both standards and create a new standard for the created gap’ should not be used to solve the overlap because these standards modify both original standards, therefore also resulting in changes to the standard that has preference. The solving methods ‘Extend one standard, drop the other’, ‘Leave the standards as they are but create clear guidelines on when to use which standard’ and ‘Define one standard to be prevailing over the other’ should only be used if the extended standard is the preferred standard. The method ‘Remove the overlap from one standard’ should only be used if the standard of which the overlap is removed is not the preferred standard.

If the contributors do not have a preference for one of the standards, all solving methods can be used to solve the overlap.

Authority of the standardisation institution
The final criteria category is the authority of the standardisation institution. The criteria that are used in this category to judge the solving methods are ‘right to decide’ and ‘effective control over decisions’ [Aghion & Tirole, 1997]. The first criterion is an example of formal authority and judges whether the standardisation institution can decide to change the standards and to throw out a standard. The second criterion is an example of real authority and judges whether the standardisation institution can control the use of the standards and can therefore enforce the use of the new standard(s).

If the standardisation institution has the right to decide, all solving methods can be used to solve the overlap. If the standardisation institution does not have the right to decide, the only solving method possible is ‘Define mappings for the different messages’.

In case the standardisation institution has effective control over decisions, all solving methods can be used to solve the overlap as they all can be controlled and checked. In case the standardisation institution does not have effective control over their decisions, the only solving method possible is ‘Define mappings for the different messages’. The standardisation institution can then only hope that the solution is used as they cannot control this.

Table 16: Possible solving methods according to the suitability criteria

Solving method → Category ↓	Measure ↓	Answer ↓	Merge standards	Extend one standard, drop the other	Remove the overlap from one standard	Leave the standards as they are but create clear guidelines on when to use which standard	Create a complete new standard and drop the old ones	Remove the overlap from both standards and create a new standard for the created gap	Define one standard to be prevailing over the other	Define mappings for the different messages
Number of resulting standards		3	X	X		X	X		X	X
		4	X	X	X	X	X		X	X
		≥5	X	X	X	X	X	X	X	X
Open for changes in existing implementations	Attributes	yes	X	X	X	X	X	X	X	X
		no	X			X		X	X	X
	Messages	yes	X	X	X	X	X	X	X	X
		no	X	X		X		X	X	X
	Model	yes	X	X	X	X	X	X	X	X
		no								X
Satisfaction of users with the current (overlapping) standards	Content	yes	X	X	X	X			X	X
		no					X	X		
	Format	yes	X	X	X	X			X	X
		no					X	X		
	Ease of use	yes	X	X	X	X			X	X
		no					X	X		
Political influences	Preferences contributors	yes		X	X	X			X	X
		no	X	X	X	X	X	X	X	X
Authority of the standardisation institution	Right to decide	yes	X	X	X	X	X	X	X	X
		no								X
	Control over decisions	yes	X	X	X	X	X	X	X	X
		no								X

5.5 In summary

Identifying and solving overlap is important to create consistent and useable standards. To identify overlap one should look at the static part and the dynamic part of the created domain model. Each of these parts can independently indicate overlap and conflict. During the identification of overlap between two standards not only the models, but also the semantics of the models are important. The research identified eight methods for solving overlap:

1. Merge standards
2. Extend one standard and drop the other
3. Remove the overlap from one standard
4. Leave the standards as they are and create guidelines
5. Create a complete new standard
6. Remove the overlap from both standards and define a new standard for the gap
7. Define one standard prevailing
8. Define mappings

To choose the most appropriate solving method, two filters and five categories of suitability criteria were defined. The two filters are ‘type of overlap’ and ‘size of the overlapping part’ and these filters are used to do a first selection out of the eight possible solving methods. These filters result in six groups of solving methods that are possible for the combinations of filters. The five categories of suitability criteria that were defined all consist of one or more means to measure the suitability:

1. Number of resulting standards
2. Changes to be made in existing implementations
3. Satisfaction of current users with the overlapping standards
4. Political influences
5. Authority of the standardisation institution

6 Overlap between SIDES and SEP

This chapter presents the research into the overlap between SIDES and SEP. The first section goes into the overlap identified with the help of the static part of the domain models. The second section goes into the overlap identified with the help of the dynamic part of the domain models. The third section indicates how the overlap between SIDES and SEP can be solved.

6.1 Overlap in the domain models static part

This section presents the overlap between SIDES and SEP that is identified with the help of the static part of the domain model. The identification of overlap is done at three levels: the model, object and attribute level; each of these levels is described in the following subsections.

6.1.1 Overlap identified at the model level

This section identifies overlap between SIDES and SEP at the top-level of the domain models, the model level. Only the objects themselves and their respective relations are used to identify overlap. The comparison at the model level starts with a comparison of the context diagrams of SIDES and SEP. As indicated in Figure 34 the context diagrams for both standards are almost the same, and can therefore be melted into one new context diagram. The only difference between the context diagrams of SIDES and SEP is that 'Job board' is only an actor in SEP and not in SIDES and that 'Intermediary' is only an actor in SIDES (they are therefore grey)

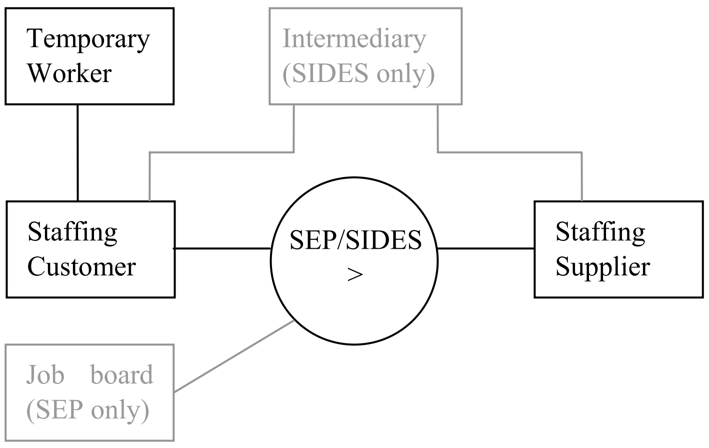


Figure 34: SIDES/SEP Context diagram

Since both context diagrams are almost equal, both standards seem to operate in the same field. This also indicates that they are both connected to the real world in the same manner. It does not mean that the standards are therefore equal or that they can replace each other; the intentions of the relations and the interactions between two objects can differ.

The static part not only contains the context diagram, it also contains an ERD of the model level. When taking a look at the ERD diagrams it becomes apparent that the complete SEP diagram can be matched to part of the SIDES diagram, this is indicated in Figure 35.

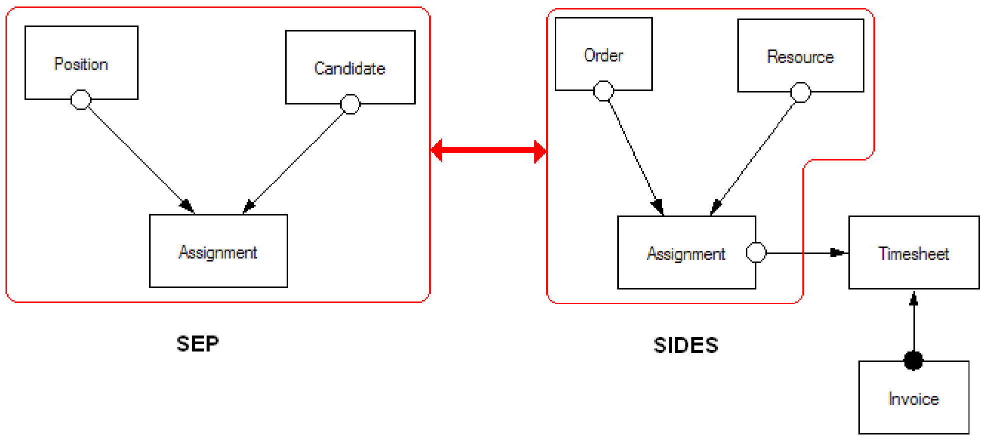


Figure 35: SIDES/SEP overlap ERD

This match seems to be a one-on-one match. Assignment is in both diagrams existence dependent on two other objects and it looks like the SEP standard can be fully contained in the SIDES standard. When taking a closer look at the ERD's, it becomes apparent that the names of the objects on which the assignment object in SEP and SIDES are existence dependent are not the same. In SEP the objects are called 'Position' and 'Candidate' while in SIDES they are called 'Order' and 'Resource'. Even though these objects may still seem to match based on the semantics of the objects (a position is a logical match with an order and a candidate is a logical match with a resource), the differences in the diagrams indicate that the overlap is not as clear as it may look like.

6.1.2 Overlap identified at the object level

The next level that is used to locate overlap in the static part of the domain model is the object level. At this level the objects with their attributes are compared to locate overlap. This comparison is done with the help of the class diagram of SIDES and SEP which can be found in Appendix V and Appendix VI. When looking at these models the differences between the different models become more apparent. Locating the overlap at the object level starts with a comparison between the objects 'Resource' and 'Candidate'. In Table 17 the matching attributes are presented. Every match is discussed in numerical order. Some matches are pretty clear and do not need much discussion, while others are more complicated or not directly logical.

1. Both objects have a 'PersonName' attribute that contains the name of the person the object describes
2. To describe how to contact a person, both objects have a 'ContactMethod' attribute
3. Both objects have an attribute that can contain the ID of the person the object is about
4. Both objects have an attribute to indicate which staffing supplier supplies the person. In Resource this is only an id, in Candidate more extensive information can be contained in the object
5. Both objects have the possibility to indicate when a person is available. Candidate can contain some more information, namely the term of notice for the current contract can be contained

6. Both objects can contain information about the rates asked, but in Resource this information is more extensive. In Candidate the PositionProfile can be used to indicate what kind of work a person wants and what the rate for that function should be
7. Both objects have a ‘Profile’ in which structured profile information can be contained
8. Both objects have the possibility to include a complete resume; the information that can be contained in both objects is equal
9. Information about a screening can only be contained in a Resource object; a Candidate object does not have a place for this information
10. Both objects can contain information about the travelling preferences of the person they describe
11. Both objects can contain information about whether the person is willing to relocate
12. Both objects can contain distribution rules for the distribution of the information in the object. In Resource this are restrictions, one describes where not to distribute to. In Candidate guidelines are applied that indicate who may receive the information
13. Organization information in Resource cannot be contained in Candidate; this information describes the staffing organization that supplies the person.

Table 17: Matching attributes Resource (SIDES) and Candidate (SEP)

	Resource (SIDES)		Candidate (SEP)	
1	PersonName		PersonName	
2	ContactMethod		ContactMethod	
3	HumanResourceId		PersonId	
4	StaffingSupplierId		CandidateSupplier	
5	AvailabilityDate		AvailabilityInfo AvailabilityDate TermOfNotice	StartDate EndDate
6	Rates		PrefferedPosition	
7	Profile		CandidateProfile	
8	Resume		Resume ProfileName ContactMethod ResumeId Qualifications Languages Achievements References TextResume LinkToResume EffectiveDate ContactInfo Objective	SecurityCredentials ProfessionalAssociations EmploymentHistory EducationHistory MilitaryHistory Associations SupportingMaterials DistributionGuidelines ExecutiveSummary LicensesAndCertification PatentHistory PublicationHistory SpeakingEventsHistory
9	screeningType	StaffingScreeningType ResourceScreening KnownScreeningType		

10	TimeMax DistanceMax willingToTravel	PercentageTravel TravelConsiderations CommuteComments	Commute TimeMax DistanceMax	Applicable TravelFrequency TravelConsiderations
11	Relocation	willingToRelocate	relocationConsidered	
12	DistributionRestrictions		DistributionGuidelines	
13	Organization PaymentInfo VATRate OrderId PositionId BilltoEntityId AssingmentId TimeCardId InvoiceId Amount CompanyName CompanyInfo ContactType ContactInfo SiteName SiteContact BankAccountInfo OrganizationId IntermediaryId ReferenceIdInfo	typeOfOrganization StaffingOrganization StaffingCustomerId StaffingSupplierId HumanResourceId PaymentInfoType OrganizationalUnitId PaymentCondition PaymentTimeAllowed PaymentEvent. PaymentDay FinancialGuarantee CollectiveAgreement StaffingCustomer- OrgUnitId StaffingSupplier- OrgUnitId StaffingReference- IdType		

Locating the overlap on the object level is continued with a comparison between the objects 'Order' and 'Position'. The matches are presented in Table 18. The matching attributes in 'Order' and 'Position' are discussed in numerical order.

1. Both objects can contain information about the staffing customer
2. Both objects have an attribute that can contain the 'Industrycode' which uniquely identifies the industry in which the staffing customer is active
3. A position title can be contained in both objects; in the order object it is part of the attribute 'PositionHeader'
4. Information about start dates and end dates can be contained in both objects; in the Position object it is part of the 'PostionDateInfo' attribute
5. Both objects can contain information about the rates that are applicable to the vacancy they describe
6. Information about the worksite can be described in both object, only the names of the attributes are different
7. Both objects have an attribute that can contain information about shifts that apply tot the vacancy
8. Information about the competencies that are needed to fulfil the vacancy can be contained in both objects

Table 18: Matching attributes Order (SIDES) and Position (SEP)

	Order (SIDES)		Position (SEP)	
1	StaffingCustomerType StaffingStaffingCustomerId MasterCustomerInformation StaffingCustomerName StaffingCustomerContactInfo StaffingCustomerIdentifiers StaffingCustomerOrgUnit CustomerIndustry		Company CompanyScale	
2	IndustryCode		IndustryCode	
3	PositionHeader		PositionTitle	
4	MaxStartDate StartDate ActualEndDate	PositionDateRange MaxNeedEndDate StartAsSoonAs-Possible	PositionDateInfo	
5	Rates		BasePay OtherPay Benefits otherInterval	OtherPayAmountMin OtherPayAmountMax OtherPayCalculation BasePayAmountMax BasePayAmountMin
6	WorkSite		Area PostalAddress SpatialLocation	PhysicalLocation TravelDirections
7	StaffingShift		Shift	
8	Competency		Competency	

The final objects that have to be compared are the Assignment objects. Since the SEP Assignment object does not contain any attributes, there are no matching attributes with the SIDES Assignment object. The reason for the empty SEP Assignment object is that this object was created to ensure existence dependency in the SEP domain model. The SEP standard itself has no message that can be used to exchange information about assignments. Even though the objects cannot be compared with the help of the attributes, it might be possible to use the SIDES Assignment object in the SEP standard. It can for instance be used to extend the SEP standard with the possibility to exchange information about assignments. The Assignment object might also be used as a coupling point between the two standards when integrating them into one standard.

6.1.3 *Overlap identified at the attribute level*

This section explains why the identification of overlap on the attribute level is not performed for the SIDES and SEP case. In section 5.1.1 a three-level top-down approach was described to identify overlap in the static part of a domain model. The previous two sections described the identification of overlap on the model and the object level. The identification of overlap on the attribute level is not performed since the domain models that were created for SIDES and SEP do not contain enough information to compare the models at this level. The detailed information about the attributes is not present in the domain models because these models have been reverse engineered from the XML specifications of the standards. These XML specifications do not contain this detailed information and therefore this information could not be added to the domain model. Identifying overlap based on the attribute level of the domain models is therefore not possible. One could argue that the information should be added to be able to perform the identification of overlap on all levels as explained in section 5.1.1. Because the results from the identification of overlap in the static part on the model and object level give enough information about the overlap, and because the identification of overlap on the attribute level is not likely to presents new insight, the choice was made not to extend the domain models and therefore not to perform the identification of overlap on the attribute level for the SIDES and SEP case.

6.2 Overlap in the domain models dynamic part

This section describes the identification of overlap between SIDES and SEP based on the dynamic part of the domain models. In section 5.1.2 a method was defined to identify overlap in the domain models dynamic part. This method makes use of the events that were distinguished with the help of Merode and that were represented in the Object Event tables. The method classified the possible overlap situations with the help of subject and target objects. For example, when creating an assignment in SIDES, the subject is 'create' and the target is 'assignment'. The possible overlap is categorized in three categories: subject overlap, target overlap or double overlap. An example of a subject overlap is creation of a vacancy and the creation of a timecard by a staffing customer. An example of a target overlap is the creation of an assignment and the ending of an assignment. The case in which in both SIDES and SEP a vacancy is created is an example of double overlap.

To be able to determine whether there is overlap, the subject and target for each event in the Object event tables of both SIDES and SEP are defined. The list for SIDES can be found in Table 19, the list for SEP can be found in Table 20. The information gathered in the comparison on the static part of the models is used in the comparison of the events in the dynamic part. We therefore know that the target 'Order' in SIDES matches with the target 'Position' in SEP, furthermore we know that the target 'Resource' in SIDES matches with the target 'Candidate' in SEP. The targets 'Assignment' in both SIDES and SEP also match.

Table 19: Subject and target of SIDES events

Event	Subject	Target
Submit_Order	Create	Order
Submit_Acceptance	End	Order
Generate_Timesheet	Create	Timesheet
Modify_Timesheet	Modify	Timesheet
Reject_Timesheet	Modify	Timesheet
Approve_Timesheet	End	Timesheet
Submit_Assignment	Create	Assignment
End_Assignment	End	Assignment
Submit_Invoice	Create	Invoice
Modify_Invoice	Modify	Invoice
Reject_Invoice	Modify	Invoice
Approve_Invoice	End	Invoice
Propose_Staffing_Resource	Create	Resource
Reject_Staffing_Resource	Modify	Resource
End_Resource	End	Resource

Table 20: Subject and target of SEP events

Event	Subject	Target
Create_Requisitions	Create	Position
Post_Requisitions	Create	Position
Transfer_Applicant_Information	Create	Assignment
Transfer_New_Hire_Information	Modify	Assignment
Candidate_Submit	Create	Candidate
Post_Position	Create	Position
Enter_Candidate	Modify	Candidate
Match_Position	Create	Assignment
Submit_Matched_Candidates	Modify	Assignment
Candidate_Applies	Create	Candidate
End_Assignment	End	Assignment
Cancel_Position	End	Position
Accept_Assignment	End	Candidate

Because it is pretty difficult to locate all possible overlap situations with the help of the above tables, the events from both SIDES and SEP are combined into Table 21. On the horizontal axes the events from SIDES are represented, on the vertical axes the events from SEP are represented. For each combination of events an indication is given whether there is overlap, and what the type of overlap is. To indicate the type of overlap we use an 'S' in case of subject overlap, a 'T' in case of target overlap and a 'D' in case of double overlap. If a cell is left empty, there is no case of overlap between the relating events.

Table 21: Type of event overlap dynamic part domain models

SEP event ↓ SIDES event →	Submit_Order	Submit_Acceptance	Generate_Timesheet	Modify_Timesheet	Reject_Timesheet	Approve_Timesheet	Submit_Assignment	End_Assignment	Submit_Invoice	Modify_Invoice	Reject_Invoice	Approve_Invoice	Propose_Staffing_Resource	Reject_Staffing_Resource	End_Resource
Create_Requisitions	D	T	S				S		S				S		
Post_Requisitions	D	T	S				S		S				S		
Transfer_Applicant_Information	S		S				D	T	S				S		
Transfer_New_Hire_Information				S	S		T	T		S	S			S	
Candidate_Submit	S		S				S		S				D	T	T
Post_Position	D	T	S						S						
Enter_Candidate				S	S					S	S		T	D	T
Match_Position	S		S				D	T	S				S		
Submit_Matched_Candidates				S	S		T	T						S	
Candidate_Applies	S		S				S		S	S	S		D	T	T
End_Assignment		S				S	T	D				S			S
Cancel_Position	T	D				S		S				S			S
Accept_Assignment		S				S		S				S	T	T	D

As Table 21 shows, there are quite a number of possible overlap situations. The reason for marking all subject and target overlap situations with a grey 'S' respectively a grey 'T' is that these overlap situations cannot be cause to conflict between the standards as explained in paragraph 5.1.2. All other overlap situations that are marked with a 'D' may be cause to a conflict. The use of the distinction between three different types of overlap (subject, target and double overlap) that can be found in the dynamic part of the domain model, resulted in the overview in Table 22. The overview presents which events match and what the corresponding targets are. Not all events of both SIDES and SEP have a matching event in the other standard, but there are quite a few events that do have a match and are thus overlapping.

Table 22: Overlapping objects according to event overlap SIDES and SEP

SEP event	SIDES event	SEP target	SIDES target
Create_Requisitions	Submit_Order	Position	Order
Post_Requisitions	Submit_Order	Position	Order
Transfer_Applicant_Information	Submit_Assignment	Assignment	Assignment
Candidate_Submit	Propose_Staffing_Resource	Candidate	Resource
Post_Position	Submit_Order	Position	Order
Enter_Candidate	Reject_Staffing_Resource	Candidate	Resource
Match_Position	Submit_Assignment	Assignment	Assignment
Candidate_Applies	Propose_Staffing_Resource	Candidate	Resource
End_Assignment	End_Assignment	Assignment	Assignment
Cancel_Position	Submit_Acceptance	Position	Order
Accept_Assignment	End_Resource	Candidate	Resource

6.3 Solving the overlap between SIDES and SEP

This section presents the possible solving methods for solving the overlap between SIDES and SEP with the help of the results of the previous two paragraphs. First the specific type of overlap is determined and with the help of the filters defined in section 5.3, the possible solving methods are determined. The suitability criteria defined in section 5.4 are used in the second subsection to choose the most suitable handling method from the list of handling methods that resulted from the filtering of the methods.

6.3.1 Possible solving methods

In this section the results of the previous two sections are combined with the filters that have been defined in section 5.3 to decide which solving methods can be used by TNO to handle the overlap between SIDES and SEP for the SETU.

The first filter defined is to filter the possible solving methods on the type of overlap. In section 2.3.2 three types of overlap were determined: the subset, equal and intersection type. Based on the overlap identified in the static and dynamic part of the standards in the previous two paragraphs, the type of overlap between SIDES and SEP can be determined. The comparison in the static part on the model level indicates that the objects match and are closely related, but that they are not equal. The comparison on the object level presented the same results; it became clearer that the standards definitely overlap, but that the overlap is only partly. From the comparison on the static part it followed that the objects Position (SEP) and Order (SIDES) are comparable just like the objects Candidate (SEP) and Resource (SIDES). This information was used in the comparison of the models in the dynamic part. This comparison showed that SIDES and SEP have messages that have the same behaviour in both of the standards.

Based on the information presented in the previous paragraph that was collected by comparisons in the static and the dynamic part of the models of SIDES and SEP, it can be concluded that the overlap between SIDES and SEP is of the intersection type. The overlap is of the intersection type as the standards clearly show overlap, but they both also define parts that are not present in the other standards. This could also indicate that the overlap is of the subset type, SEP being a subset of SIDES. There is however a difference between the standards that indicates that the overlap is not of the subset type. This difference is that while both standards function in comparable domains, and both perform a process of which the result is equal, the assignment of a temporary worker to a vacancy, the way both standards implement the process is different. SEP specifies a candidate and a job opening that can be matched, while SIDES specifies a human resource (that contains more information than a candidate), and an order (that is more general than the job opening in SEP). This difference in the process not only indicates that the overlap is not of the subset type, but also that it is not of the equal type. The models differ too much to make it possible for one of the standards to completely replace the other standard while still having the possibility to exchange the same information. Since the two standards share the same domain, but also define a part of a domain that is distinct, the overlap is of the intersection type.

The second filter that is defined to filter the possible solving methods is the size of the overlapping parts in the standards. The size of the overlapping parts in SIDES and SEP are therefore determined. In section 5.3.1 guidelines were presented to determine the size of the overlapping parts. In case the overlapping parts were less than 40% of the entire standard the overlap is called small. In case the overlapping parts are more than 70% of the entire standard the overlap is called large. Cases that are between these two boundaries have to be judged individually. When looking at the number of objects that are involved in the overlap in relation to the total number of objects in the entire ERD model of the SEP standard, one sees that all objects in the SEP domain model are involved in the overlap. This leads to the conclusion that the overlapping objects in the SEP domain model comprise 100% of the entire domain model, thereby making the size of the overlapping parts in the SEP standard large according to the guidelines.

When looking at the objects involved in the overlap in SIDES standard in relation to the total number of objects in the ERD, one sees that three of the five objects in the ERD are involved. Therefore the overlap in the SIDES standard is 60%. Based on this percentage it is not possible to directly conclude whether the overlap is small or large because this percentage lies within the non-deterministic part (between 40% and 70%) and therefore has to be judged individually. If one takes a closer look at the overlapping part in the SIDES standard one sees that the 'Assignment' object is also part of the overlap. This object is important for the entire standards, as it contains information which is reused in all other objects. Because this important object is also part of the overlap, the conclusion is drawn that the overlapping part in the SIDES standard can be classified as a large overlap.

Now that it is known that the overlap is of the intersection type, and that the size of the overlapping parts is large in both the SIDES and SEP standards, the results in section 5.3.2 can be used to decide on the possible solving methods that can be applied by TNO to solve the overlap between SIDES and SEP for the SETU. Namely:

- Merge standards
- Extend one standard, drop the other
- Create a complete new standard and drop the old ones
- Define mappings for the different messages

6.3.2 *Most suitable solving method*

To determine the most suitable solving method, the criteria defined in section 5.4 are applied to the four solving methods. To be able to do this each criterion has to be supplied with an answer that is applicable to the SIDES/SEP case.

Based on experiences that TNO has with other standards and to make the maintenance of the new standard easier, the total number of resulting standards should not be more than three. This implies that only one new standard should be created. If the Dutch staffing industry prefers two standards, one for the matching process, and one for the timesheet/invoice process, the new standard could split in two non-overlapping sub standards.

The SETU has the ability to change the entire standard. The new standards are defined for the Dutch staffing industry exclusively and although it would be a good idea to match up with existing standards, this is not an obligation. The answers for the 'open for changes in existing implementations' criteria category are therefore 'yes'.

The answers to the criteria in the category 'satisfaction of users with the current (overlapping) standards are also all 'yes'. The reason for this is that the current content of the standards and the current format are both good, and these do not have to change. The ease of use of both existing standards is also good for the current users, but would become better if the overlap was removed.

The political influences are measured by means of the preferences of the current contributors of the existing standards. In the case of the SETU there is a preference for the SIDES standards as this standard has already been implemented by a number of staffing agencies. The SEP standard has not been implemented by a staffing agency in the Netherlands.

The final criteria category is 'authority of the standardisation institutions'. To measure this category two criteria have been defined: 'right to decide' and 'control over decisions'. The answers to these two criteria are for the SETU both 'yes'. The SETU has the right to decide what the new standards for the Dutch staffing industry will look like, and the SETU also has control over these decisions as they will also maintain the new standard(s).

An overview of these criteria and the specific answers for the SETU SIDES/SEP case can be found in Table 23.

Table 23: Answers to measure criteria SETU SIDES/SEP case

Category	Measure	Answer
Number of resulting standards		3
Open for changes in existing implementations	Attributes	yes
	Messages	yes
	Model	yes
Satisfaction of users with the current (overlapping) standards	Content	yes
	Format	yes
	Ease of use	yes
Political influences	Preferences contributors	yes
Authority of the standardisation institution	Right to decide	yes
	Control over decisions	yes

Now that the criteria are known, it is possible to use Table 16 to choose the most suitable solving method for the SETU SIDES/SEP case. With the help of the filters the number of possible solving methods was reduced to four:

- Merge standards
- Extend one standard, drop the other
- Create a complete new standard and drop the old ones
- Define mappings for the different messages

The criterion 'number of resulting standards' and the criteria 'changes allowed to the attributes, messages and models' do not change this list. The criteria 'content' in the category 'satisfaction of users with the current (overlapping) standards' changes the number of possible solving methods to three. The solving method 'Create a complete new standard and drop the old ones' is not possible anymore. The criteria 'format' and 'ease of use' of the same category do not change the list any further.

- Merge standards
- Extend one standard, drop the other
- Define mappings for the different messages

The answer to the criteria 'preferences contributors' is yes, thereby reducing the above list to two. Merging the two standards is not a possibility any more. As the SETU has the right to decide and has control over the decisions, these two criteria do not change the list of possible solving methods any more. The final list of most suitable solving methods for the TNO to solve the overlap between SIDES and SEP for the SETU therefore is:

- Extend one standard, drop the other
- Define mappings for the different messages

Since filtering the solving methods based on the criteria results in two possible solving methods they can both be used to handle the overlap. There is however a difference between the two solving methods. The first method actually changes the existing standards, the second leaves them intact and defines mappings. As the intention of the SETU is to define a new standard that solves the overlap, choosing the solving method 'extend one standard, drop the other' is the best choice for TNO to solve the overlap. As the other solving method can also be used, this method could be used to make the transition from the current standards to the new standard easier and give implementers the time to change their current implementation without loosing the possibility to electronically exchange information. By using these solving methods in combination TNO creates an extended SIDES standard for the SETU and solves the overlap between SIDES and SEP, while offering the SETU the possibility to make the transition process from the old standard to the new standard as smooth as possible for existing implementations.

A high-level ERD of the proposed new standard that TNO should implement for the SETU to solve the overlap between SIDES and SEP, can be found in Figure 36. The ERD of the new model is pretty similar to the ERD of the original SIDES ERD, the main difference is the extension of the model with the entities 'position' and 'candidate'.

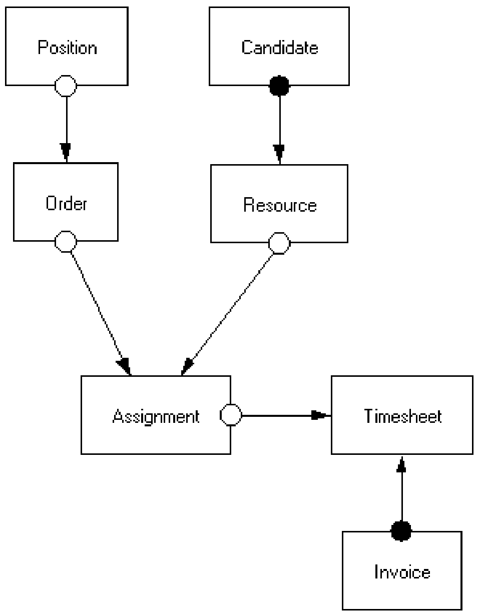


Figure 36: SIDES/SEP combination domain model

6.4 In summary

The overlap of SIDES and SEP was in some way already clear when starting the research described in this thesis. That the overlap is of the intersection type became clear in the identification process based on the static and the dynamic part of the domain models of SIDES and SEP. While SEP at first seemed to be a subset of SIDES, it became very clear that the goals of the two standards are the same, but that the methods for reaching this are pretty different in both standards. Both the static and the dynamic part comparison showed the same type overlap, thereby reinforcing each other's result. The best method for TNO to solve the overlap between SIDES and SEP for the SETU is to extend the SIDES standard with the extra parts of SEP. This keeps the changes to existing implementations as small as possible. To make the transition process from the old standards to the new standard easier TNO should define mappings from the old messages to the new messages.

7 Conclusions and recommendations

The research presented in this thesis is triggered by the overlap between SIDES and SEP and the question on how to solve this overlap. To be able to answer this question, research was performed into identifying, preventing and solving overlap, thereby focussing on using domain modelling to perform these parts. This chapter presents the conclusions and recommendations that follow from the performed research. A roadmap that helps to solve overlap is presented as part of the recommendations.

7.1 Conclusions

This section presents the conclusions in two parts: the first subsection answers the two main research questions and presents two extra conclusions drawn from the research, the second subsection answers all other research questions.

7.1.1 Main conclusions

The research presented in this thesis is based on two main research questions; the answers to these questions are given first, after that the two extra conclusions are presented.

How do SIDES and SEP overlap?
The overlap between SIDES and SEP is of the intersection type. This means that the standards partly overlap. Both standards also contain parts that are not available in the other standard. Overlapping parts in SIDES and SEP based on the created domain models are the objects Order (SIDES) and Position (SEP), the objects Resource (SIDES) and Candidate (SEP) and the objects Assignment (SIDES) and Assignment (SEP).

Overlap between SIDES and SEP makes it difficult for the SETU to decide which standard to use for the exchange of information, how can this overlap be handled?
The advice for TNO on how to handle the overlap between SIDES and SEP for the SETU is to extend the SIDES standard with the parts from the SEP standard that are not already present in SIDES and then to drop the SEP standard. This creates a extended SIDES standard that can handle all interactions that could be handled with the two separate standards. It further solves the problem of deciding which standard to use for the exchange of information. To make the change process from the old standards to the new standards as easy as possible for current implementations, TNO should define mappings that make it possible to translate messages of the old standard to messages of the new standard.

Domain modelling
Three domain modelling techniques Merode, Fusion and NYAM were compared to find the most suitable technique to identify overlap between two electronic messaging standards. The Merode domain modelling technique is for TNO the most suitable technique for modelling the domain models of the standards because of the existence dependency relation that is defined and the cross-model checks that are defined to ensure that the different models that are created (ERD, OET, STD) are consistent.

Semantics

During the identification of overlap between two standards with the help of their respective domain models, the semantics of the models are also important to be able to match objects and attributes. The reason for this is that with only comparing the models themselves, overlap between two standards can not be completely found. Only with the help of semantics it is possible to be sure what the element refers to and to decide whether it overlaps with another element.

7.1.2 *Answers to the research questions*

This section presents the answers to all sub research questions that were defined. The answers to the sub questions are presented in two blocks; each block is related to one of the two main research questions. We start with the answers to the sub questions that are related to the first main research question.

- *What are, in the case of electronic messaging standards, the definitions of overlap and conflict?*
  Overlap in electronic messaging standards is a situation in which two different standards describe the same domain. A conflict is the situation in which two different standards overlap and define different things for the same domain.
  - *What is the difference between the definitions?*
    The definitions of overlap and conflict show that to have a conflict one has to have an overlap. This rule cannot be applied the other way around; an overlap can exist without a conflict being present. The difference between overlap and conflict therefore is that a conflict is a special kind of overlap.
- *What are the possible types of overlap in electronic messaging standards?*
  Three different types of overlap can be distinguished, based on set theory. These types are:
  - equal relation type
  - subset relation type
  - intersection relation type

  In case the overlap is of the equal relation type, both standards describe the same domain and can therefore replace each other. In case the overlap is of the subset relation type, one of the standards is completely contained in the other standard, the superset. In this case the superset could replace the subset standard. In case the overlap is based on the intersection relation type, only a part of both standards is overlapping, both standards also define parts that are not overlapping.
  - *What is the type of overlap between SIDES and SEP?*
    The type of overlap between SIDES and SEP is the intersection relation type. SIDES and SEP partly overlap, but both define also parts that are not overlapping.
- *What are the causes of overlap and conflict between electronic messaging standards?*
  Overlap and conflict between electronic messaging standards can have many causes. They can be introduced at the following places:
  - during the standardisation process
    - due to the institutional framework
    - due to the actors
    - due to the technological foundations
  - during the implementation process
    - due to the embrace-and-extend strategy
    - due to the embrace-and-omit strategy
    - due to the embrace-and-adapt strategy

The institutional framework can be a cause of overlap and conflict because different working groups within one institute do not communicate, but a greater cause is the lack of communication about the standard development between different standardisation institutions. The actors that are working on new standards are also a cause of overlap and conflict between standards. Actors all have different interests and motivations to work on creating a new standard and these differing interests and motivations are often cause to differentiating standards that in the end are overlapping or in conflict. The final cause to overlap and conflict in the standardisation process is based on the technological foundation of a standard. Numerous standards are not created from scratch, but are based on already existing standards. This often makes it difficult to prevent overlap and conflicts from emerging because a standard cannot be too different from older standards, or has to keep supporting the old standards.

In the implementation process of standards, overlap and conflict can also arise. In this case the original definitions of a standard may not overlap or be in conflict with another standard, but the implementation specific definitions of the same standard may be in conflict with the original standard, or with other implementation specific definitions. Three well-known strategies were presented that are cause to overlap and conflict in the implementation process. All three strategies modify the definition of the original standard in such a way that they are not equal any more, thereby often creating big overlaps and often also big conflicts that are not easy to solve in the future.

- o *Can standardization institutions prevent the creation of overlap between standards?*

  Standardization institutions can prevent overlap. Three methods to prevent overlap from happening are:
  - research into the standards' domain
  - coordination of standardisation process
  - coordination between institutions

  The first method prescribes that one has to do a thorough research into the domain of the new standard to find out whether a new standard is needed, or that there is an existing standard that can be used. The second method prescribes a better coordination of the standardisation process within standardisation institutions, while the third standard prescribes a better coordination between standardisation institutions. They both describe a mayor problem, namely the coordination between different groups that are working on standards. To prevent overlap from happening this coordination should be improved.

- *How can domain modelling be used to identify overlap between electronic messaging standards?*

  Domain modelling can be used to identify overlap by creating comparable domain models which consist of the following diagrams and models. This list is based on a definition of how a good standard should look that was developed at TNO.
  - Context diagram
  - Class diagram
  - ERD
  - State transition diagram
  - Event list
  - Object event table

  With the help of these models the standards that overlap can be compared to identify the overlap.

- *Which domain modelling techniques are available?*
  Three domain modelling techniques were identified that can be used to create a domain model from an electronic messaging standard. The three domain modelling techniques that were identified are:
  - Merode
  - Fusion
  - NYAM

  Merode, which stands for Model-based, Existence-dependency Relationship, Object-oriented, Development, is developed at the Catholic University of Leuven and uses existence dependency to create entity relationship diagrams.

  The Fusion technique is a fusion of different techniques from other object-oriented techniques. The technique was developed to provide a systematic approach for developing software in an object-oriented fashion, and the first phase of the entire technique describes the development of a domain model.

  NYAM, which stands for Not Yet Another Method, is just like the Fusion technique a combination of parts of existing techniques. This technique is also a software engineering technique, whereby the modelling of the domain is part of the technique.
- *What are the differences between the available domain modelling techniques?*
  The three techniques do not differ that much; they all prescribe the creation of an ERD and they all prescribe the creation of an overview of the events or actions that happen in a system. The differences between the techniques are not visible at this level, but are more subtle. The ERD created by the three techniques is for instance is not equal. Merode extends the ERD notation with the concept of existence dependency, an entity can only be part of the diagram if it is fully contained in the life cycle of its parents. Another difference is that Merode defines constraints between the different models that ensure that the models are consistent, something that Fusion and NYAM do not specify.
- *Which technique is most suitable to identify overlap between electronic messaging standards?*
  The domain modelling technique that is most suitable for TNO to locate overlap between SIDES and SEP is Merode. The reason that Merode is the most suitable technique is primarily based on the existence dependency extension to the standard ERD notation and the constraints between the different models that the developers of Merode defined.
  - *What are the comparison criteria?*
    The criteria for determining what the most suitable modelling technique is to locate the overlap between standards are:
    - Created diagrams
    - Modelling rules
    - Match with existing techniques used at TNO

    These criteria are deduced from meetings with researchers at TNO.

- *What are the domain models of SIDES and SEP?*
  The domain models for SIDES and SEP have been created by reverse engineering the standards. The domain models can be found in chapter 4.

The following paragraph presents the answers to the sub questions that are related to the second main research question.

- *What are the possibilities for handling overlap?*
  Overlap can be handled by applying one of the eight solving methods that are defined in this thesis and which were deduced from meetings with researchers at TNO. Each method has specific characteristics and works on specific types of overlap:
  - Merge standards
  - Extend one standard, drop the other
  - Remove the overlap from one standard
  - Leave the standards as they are and create guidelines
  - Create a complete new standard
  - Remove the overlap from both standards and create a new standard for the gap
  - Define one standard to be prevailing
  - Define mappings for the different messages
- *What is the most suitable handling method that TNO can use to solve the overlap between SIDES and SEP for the SETU?*
  The most suitable handling method that TNO can use to solve the overlap between SIDES and SEP is the method 'extend one standard, drop the other'. To make the transition process from the old standards to the new standards easier the solving method 'define mapping for the different messages' should also be applied.
  - *What are the comparison criteria?*
    There are five criteria categories for determining what the most suitable method to handle overlap is. They all have one or more criteria to measure the suitability:
    - Number of resulting standards
    - Changes to be made in existing implementations
      - Changes to attributes
      - Changes to messages
      - Changes to models
    - Satisfaction of current users with the overlapping standards
      - Content
      - Format
      - Ease of use
    - Political influences
      - Preferences of contributors
    - Authority of the standardisation institution
      - Right to decide
      - Control over decisions

7.2 Recommendations for TNO

Based on the conclusions some recommendations are presented that can help in further improving the process of identifying and solving overlap between electronic messaging standards within TNO.

From the research a roadmap can be extracted that TNO can use to identify and handle overlap between standards. The roadmap helps in using domain models to identify and handle overlap and gives guidelines on what order to follow to locate overlap. It should be noted that the roadmap should be examined, tested and formalized in the future to ensure that it is complete.

1. Create domain models of both standards with the help of the Merode modelling technique
    a) Define the actors that are present in the standard
    b) Define the roles that the actors can play
    c) Create a context diagram
    d) Create a ERD with existence dependency
    e) Create a class diagram with attributes
    f) Define all events
    g) Fill in the Object Event Table
    h) Draw STD's for each entity
2. Compare the domain models static part
    a) Compare the context diagram
    b) Compare the ERD's
    c) Define which entities are overlapping
    d) Define which classes are overlapping
    e) Compare the classes that are overlapping
3. Compare the domain models dynamic part
    a) Compare the events with the help of their subjects and targets
    b) Locate all cases of subjects and targets overlap
    c) Define which classes are overlapping
4. Define the type of overlap with the help of the comparisons
5. Define the size of the overlapping parts in relation to the size of the standards
6. Define the most appropriate solving method
    a) With the help of the defined filters, filter the list of solving methods
    b) Define the maximum number of standards that should result after applying the solving method
    c) Define how each criteria is of influence by choosing an answer for each criteria
    d) With the help of the answers to the criteria locate the most appropriate solving method from the filtered list of solving methods
7. Solve the overlap

In the research the semantics of the standards and related domain models played an important role in identifying overlap between standards. These semantics proved very difficult to model and this makes a comparison based on only the models impossible. The dictionaries that are defined for a standard could be useful in comparing standards based on the semantics. Unfortunately, as the dictionaries are in natural language, it is difficult to compare these dictionaries. For this reason it would be interesting to perform research into the semantics of standards and related domain models to see whether the semantics can also be modelled so that the comparison between standards is less dependant on the knowledge of the standards by the person performing the comparison.

A final recommendation is to use the proposed solving method to solve the overlap between SIDES and SEP to see whether the proposed solution is acceptable for the Dutch staffing industry.

References

[Aghion & Tirole, 1997]	Aghion, P., Tirole, J. (1997), "Formal and Real Authority in Organizations", The Journal of Political Economy, Vol. 105, No. 1 (Feb., 1997), p. 1-29.
[Bastiaans, 2007]	Bastiaans, G.J.A. (2007), "Standardizing electronic transactions based on state-of-the-art concepts", TNO Information and Communication Technology, Graduation thesis, forthcoming.
[Besen & Farrell, 1994]	Besen, S.M., Farrell, J. (1994), "Choosing How to Compete: Strategies and Tactics in Standardisation" In: *The Journal of Economic Perspectives*, Vol. 8, No. 2. (Spring, 1994), p. 117-131.
[Cambridge, 2007]	Cambridge Advanced Learner's Dictionary, Cambridge University Press, Visited April 19, 2007 <http://dictionary.cambridge.org/>
[Coleman et. al, 1994]	Coleman, D., Arnold, P., Bodoff, S., Dollin, C., Gilchrist, H., Hayes, F., Jeremaes, P. (1994). "Object-Oriented Development, The Fusion method", Prentice Hall, Englewood Cliffs, New Jersey 07632, ISBN 0-133-38823-9
[Doll & Torkzadeh, 1988]	Doll, W. J., Torkzadeh, G. (1988), "The Measurement of End-User Computing Satisfaction", MIS Quarterly (12:2), June 1988, p. 259-274.
[Egyedi & Dahanayake, 2003]	Egyedi, T.M., Dahanayake, A. (2003) "Difficulties Implementing Standards" In: Egyedi, T.M., Krechmer, K., & K. Jakobs (Eds.), *Proceedings of the 3rd IEEE Conference on Standardisation and Innovation in Information Technology*, SIIT 2003, Delft, the Netherlands, p.75-84.
[Egyedi, 2007]	Egyedi, T.M. (2007), "Standard-Compliant, but Incompatible?!" In: *Computer Standards & Interface,* forthcoming.
[Enting et. al, 2006]	Enting, H.D., Bekkum, M.A. van, Folmer, E.J.A. (2006), "Implementatierichtlijnen hr-XML SIDES in Nederland versie 1.1", TNO Informatie- en Communicatietechnologie.
[Fomin & Keil, 2000]	Fomin, V., Keil, T. (2000), "Standardisation: Bridging the gap between economic and social theory" In: *Proceedings of the twenty first international conference on Information systems*, Brisbane, Queensland, Australia, p. 206-217.
[HR-XML, 2002]	HR-XML Consortium Inc (2002), "Guide to SIDES".
[HR-XML, 2007]	HR-XML Consortium, Visited April 16, 2007 <http://www.hr-xml.org>
[IEC, 2007]	International Engineering Consortium (IEC), Visited April 16, 2007 <http://www.iec.org>

[IEEE, 2007]	Institute of Electrical and Electronics Engineers (IEEE), Visited April 16, 2007 <http://www.ieee.org>
[ISO, 2007]	International Organization for Standardisation (ISO), Visited April 16, 2007 <http://www.iso.org>
[ITU, 2007]	International Telecommunication Union (ITU), Visited April 16, 2007 <http://www.itu.int>
[Magic Draw, 2007]	Magic Draw architecture tool, Visited September 6, 2007 <http://www.magicdraw.com>
[Mermaid, 2007]	Mermaid (MERODE Modeling Aid), Visited August 17, 2007 <http://merode.econ.kuleuven.ac.be/mermaid.aspx>
[Miller & Mukerji, 2003]	Miller, J., and Mukerji, J. (Editors), MDA Guide Version 1.0.1, Object Management Group, Inc.
[Moffett & Sloman, 1993]	Moffett, J.D. Sloman, M.S. (1993), "Policy Conflict Analysis in Distributed System Management" In: Journal of Organizational Computing, Vol. 4, No. 1 (1994), p. 1-22.
[Oksala et al, 1996]	Oksala, S., Rutkowski, A. Spring, M. O'Donnel, J. (1996) "The structure of IT standardisation" In: *StandardView*, Vol. 4, No. 1 (March, 1996), p. 9-22.
[Oosterhuis & Pires, 2004]	Oosterhuis-Geers, J., Ferreira Pires, L. (2004) "Final Project Guide", Master of Sciences Programmes, Department of Computer Science, University of Twente, version 2004-07-13.
[Pokraev, et.al, 2005]	Pokraev, S., Reichert, M.U., Steen, M.W.A., Wieringa, R.J. (2005) "Semantic and Pragmatic Interoperability: A Model for Understanding" In: Proceedings of the Open Interop Workshop on Enterprise Modelling and Ontologies for Interoperability (EMOI - INTEROP'05), 13 - 14 Jun 2005, Porto, Portugal. pp. 1-5. CEUR Workshop Proceedings 160. CEUR-WS.org. ISSN 1613-0073.
[Schilling, 1998]	Schilling, M.A. (1998), "Technological Lockout: An Integrative Model of the Economic and Strategic Factors Driving Technology Success and Failure" In: *The Academy of Management Review*, Vol. 23, No. 2 (Apr., 1998), p. 267-284.
[Schmidt & Werle, 1992]	Schmidt, S.K. Werle, R. (1992), "The Development of Compatibility Standards in Telecommunications: Conceptual Framework and Theoretical Perspective" In: Meinolf Dierkes / Ute Hoffmann (eds.), *New Technology at the Outset*. Frankfurt a.M: Campus, ISBN 3-593-24611-7 (Campus Verlag), ISBN 0-8133-8620-0 (Westview Press), p. 301-326.

[Schmidt & Werle, 1998]	Schmidt, S.K., Werle, R. (1998), "Standard Setting" In: *Coordinating Technology: Studies in the International Standardisation of Telecommunications.* The MIT Press, Cambridge, Massachusetts, ISBN 0-262-19393-0, p. 85-120.
[SEP, 2007]	Staffing Exchange Protocol (SEP), Visited April 16, 2007 <http://ns.hr-xml.org/2_4/HR-XML-2_4/SEP/StaffingExchangeProtocol.html>
[SETU, 2007]	Stichting Elektronische Transacties Uitzendbranche (SETU), Visited September 20, 2007 <http://www.setu.nl>
[SIDES, 2007]	Staffing Industry Data Exchange Standards (SIDES), Visited April 16, 2007 <http://ns.hr-xml.org/2_4/HR-XML-2_4/SIDES/SIDES.html>
[Snoeck et. al, 1999]	Snoeck, M., Dedene, G., Verhelst, M., Depuydt, A. (1999), "Object-Oriented Enterprise Modelling with MERODE", Leuven University Press, ISBN 9-061-86977-3.
[W3C, 2007]	World Wide Web Consortium (W3C), Visited April 16, 2007 <http://www.w3c.org>
[Wieringa, 2003]	Wieringa, R.J. (2003), "Design methods for reactive systems, Yourdon, Statemate and the UML, Morgan Kaufmann Publishers, ISBN 1-558-60755-2.

Appendix I. Glossary

B2B	Business-to-business
CIM	Computation Independent Model
de facto	"in fact" or "in practice", but not spelled out by law
de jure	"based on law", spelled out by law
EDG	Existence Dependency Graph
ERD	Entity Relationship Diagram
HR-XML Consortium	The HR-XML Consortium is an independent, non-profit organization that is dedicated to the development and promotion of a standard suite of XML specifications to enable e-business and the automation of human resources-related data exchanges
HTTP	HyperText Transfer Protocol
ICT	Information and Communication Technology
IEC	International Engineering Consortium
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardisation
ITU	International Telecommunication Union
MDA	Model Driven Architecture
Merode	Model-based Existence-dependency Relationship, Object-oriented Development
NYAM	Not Yet Another Method
OET	Object Event Table (Merode)
OMG	Object Management Group
OO	Object Oriented
OOXML	Office Open XML
OSI	Open System Interconnection
PDF	Portable Document Format

PIM	Platform Independent Model
PSM	Platform Specific Model
RUP	Rational Unified Process
SEP	Staffing Exchange Protocol
SETU	Stichting Elektronische Transacties Uitzendbranche / Foundation for electronic transactions in the staffing industry
SGML	Standard Generalized Markup Language
SIDES	Staffing Industry Data Exchange Standards
SQL	Structured Query Language
STD	State Transition Diagram
STT	State Transition Tables
SuD	System under Development
TNO	Nederlandse Organisatie voor toegepast-natuurwetenschappelijk onderzoek / Netherlands Organisation for Applied Scientific Research
UML	Unified Modelling Language
UMM	UN/CEFACT Modeling and & Methodology
W3C	Word Wide Web Consortium
XML	eXtensible Markup Language

Appendix II. List of figures

Appendix III. List of tables

Appendix IV. Mermaid domain modelling tool

In the research the Merode modelling technique was used to create domain models. The authors of the Merode technique not only described the modelling technique but also defined an accompanying tool, Mermaid. In this tool all of Merode's checks between models are incorporated and the tool can indicate missing events and point out inconsistencies between the entity relationship diagrams, the object-event table and the state transition diagrams. The ERD's, the object event tables and the STD's presented in this thesis were all created in Mermaid and the checks between the models surely helped to create consistent and complete models.

In Figure 37 the interface of Mermaid can be found with an ERD diagram. In this figure the ERD makes use of the specific ERD notation as defined for Merode, but it is also possible to use regular UML notations as can be seen in Figure 38. This makes it possible to exchange the diagrams with everybody who can understand UML diagrams and not only with people that understand the Merode notation.

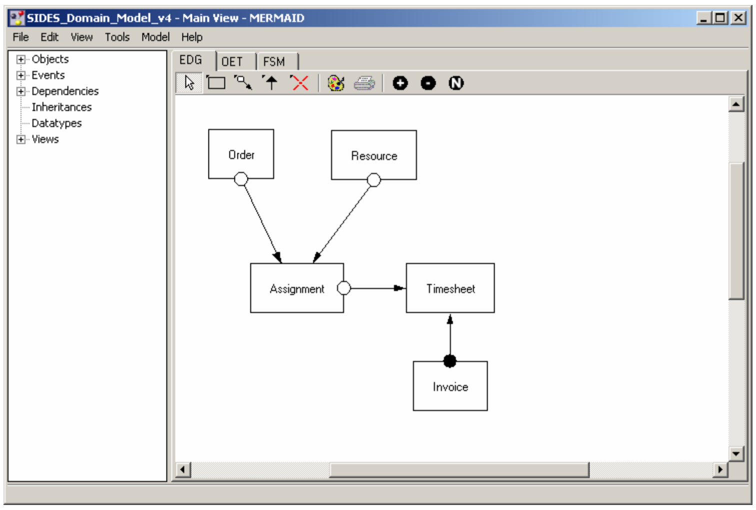


Figure 37: Mermaid screenshot EDG Merode notation

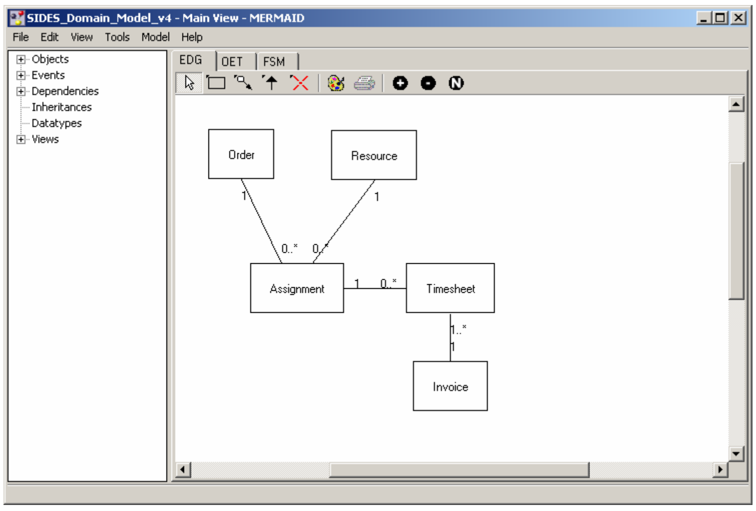
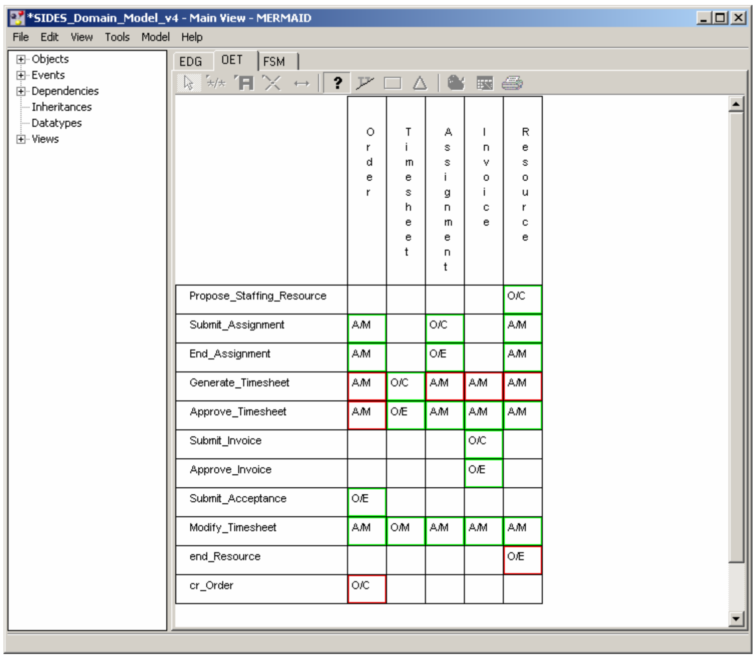


Figure 38: Mermaid screenshot EDG UML notation

The columns in the object-event table in Mermaid are automatically filled in with the entities that are defined in the existence dependency graph (EDG). This ensures that no entity is forgotten. When filling in the events, Mermaid can automatically check whether each event has a ‘create’ and an ‘end’ event. Mermaid furthermore checks whether all events are propagated up in the EDG. The events of the existence dependant object should be modifying events for their parents. In Figure 39 an object-event table created in Mermaid can be found and the missing events and propagated events are marked with a red box. All green boxes indicate correct events.


*SIDES_Domain_Model_v4 - Main View - MERMAID
File Edit View Tools Model Help
Objects
Events
Dependencies
Inheritances
Datatypes
Views
EDG
OET
FSM

	Order	Timesheet	Assignment	Invoice	Resource
Propose_Staffing_Resource					O/C
Submit_Assignment	A/M		O/C		A/M
End_Assignment	A/M		O/E		A/M
Generate_Timesheet	A/M	O/C	A/M	A/M	A/M
Approve_Timesheet	A/M	O/E	A/M	A/M	A/M
Submit_Invoice				O/C	
Approve_Invoice				O/E	
Submit_Acceptance	O/E				
Modify_Timesheet	A/M	O/M	A/M	A/M	A/M
end_Resource					O/E
cr_Order	O/C				

Figure 39: Mermaid screenshot Object-Event table

The final feature of Merode that is incorporated into Mermaid is the possibility to create state transition diagrams that can automatically be checked to see whether the STD is in accordance to the object-event table. A screenshot that shows this possibility can be found in Figure 40.

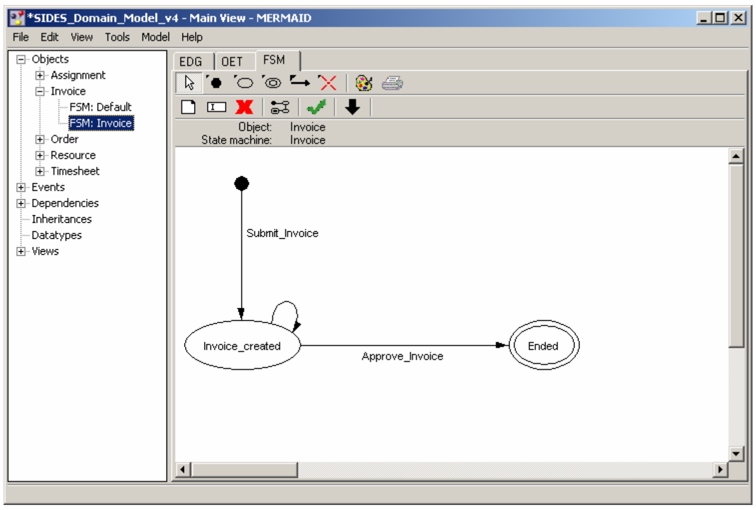
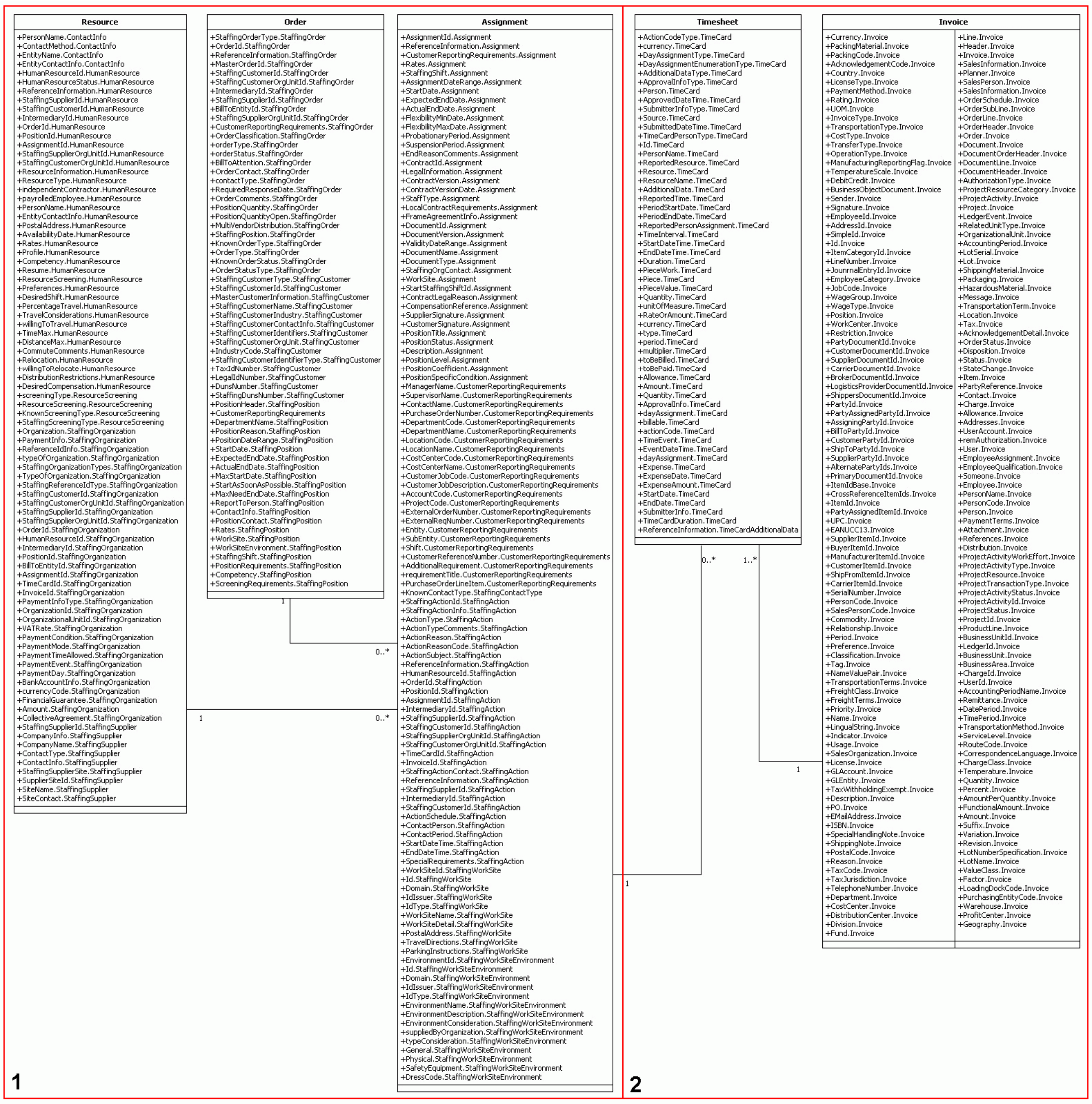
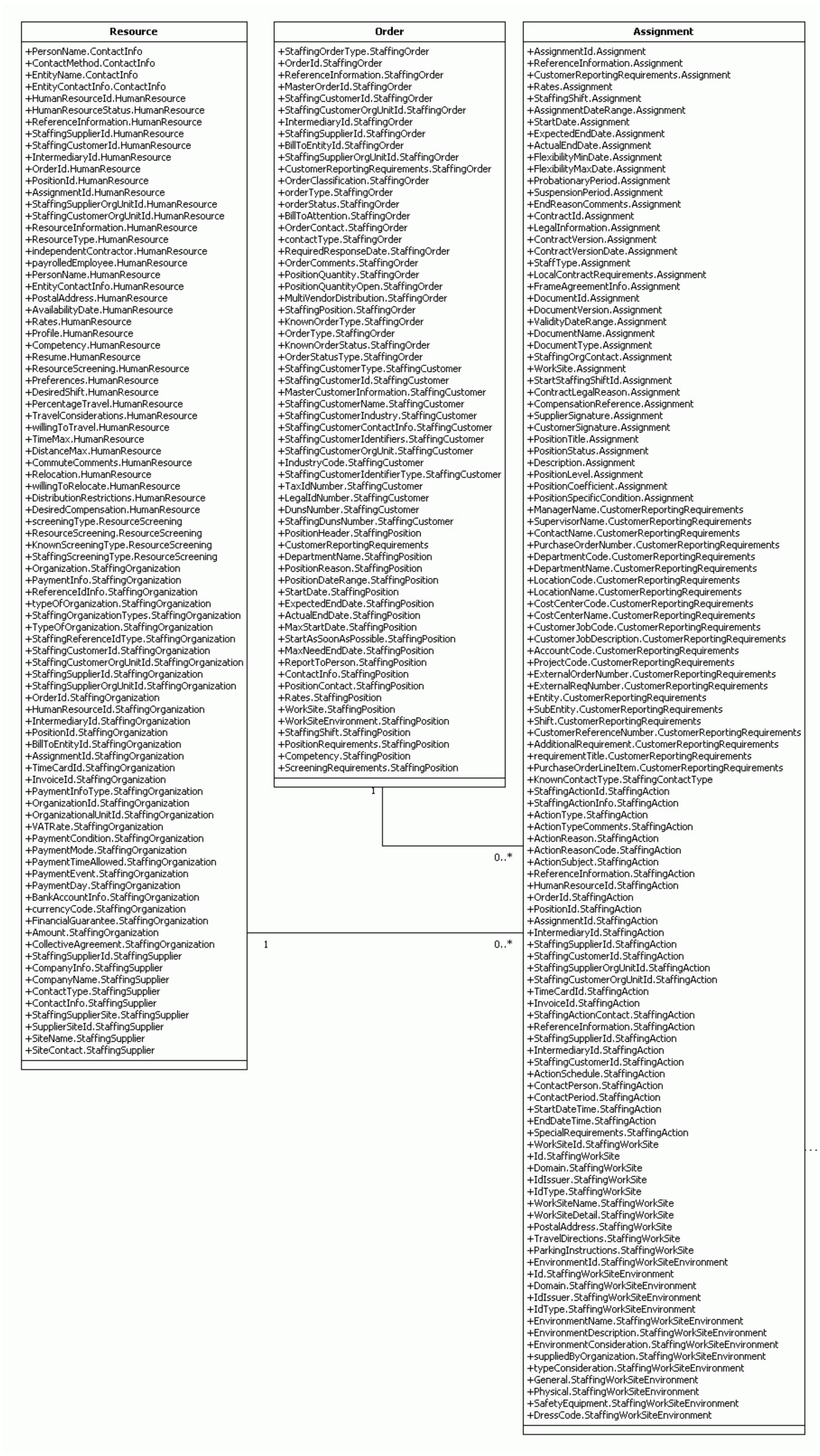


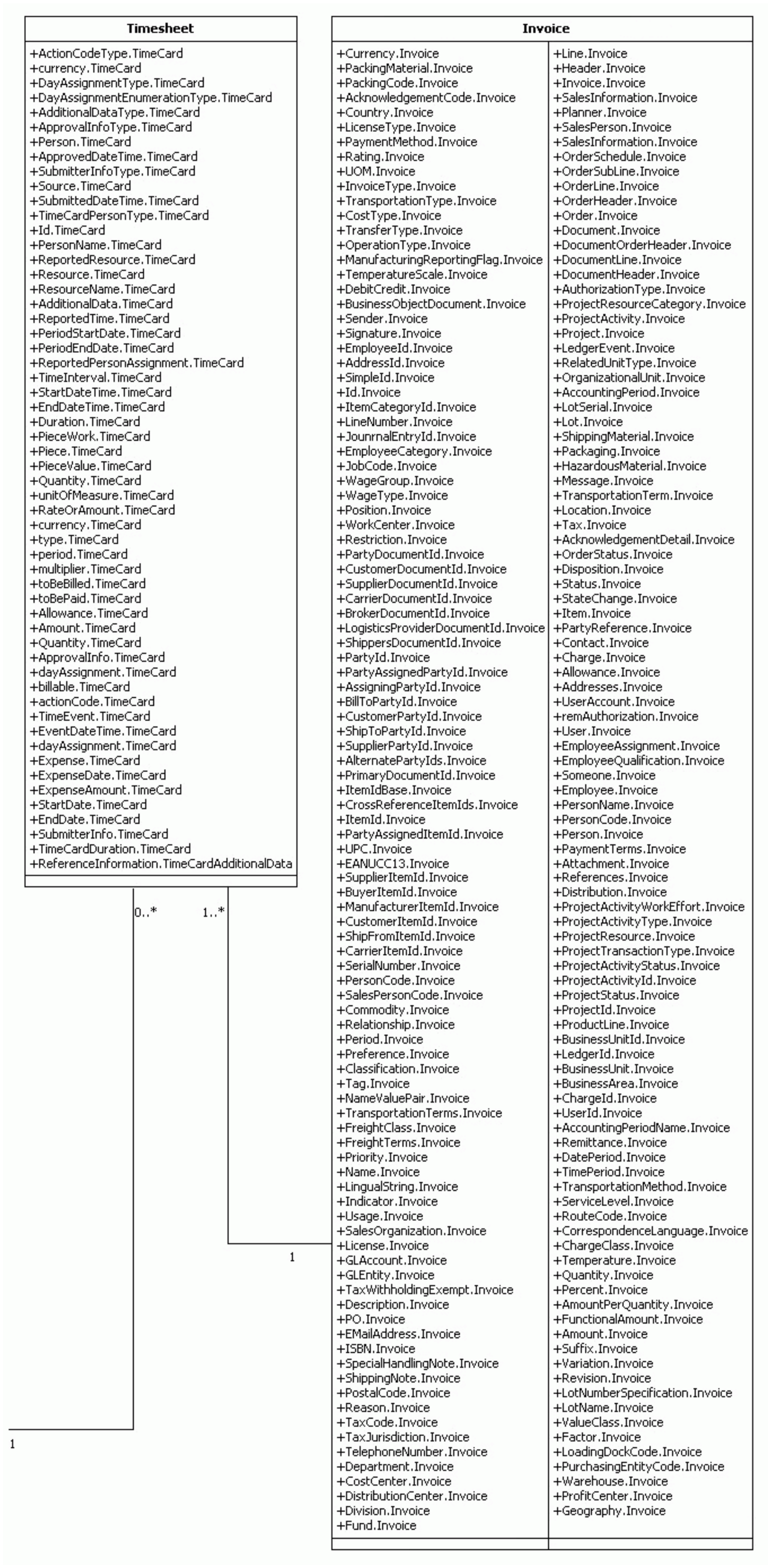
Figure 40: Mermaid screenshot STD

The Mermaid tool is a nice and handy addition to the Merode modelling technique that ensures that the method is used correctly and ensures correct models. The tool could be improved by also incorporating the more advanced features of Merode, like attributes, constraints and methods. Generalization, specialization and inheritance would also be nice additions to the EDG. Instead of improving the Mermaid tool, it might also be a good idea to implement the Merode domain modelling technique into other tools like for instance Magic Draw [Magic Draw, 2007] to simplify the use of the domain modelling technique.

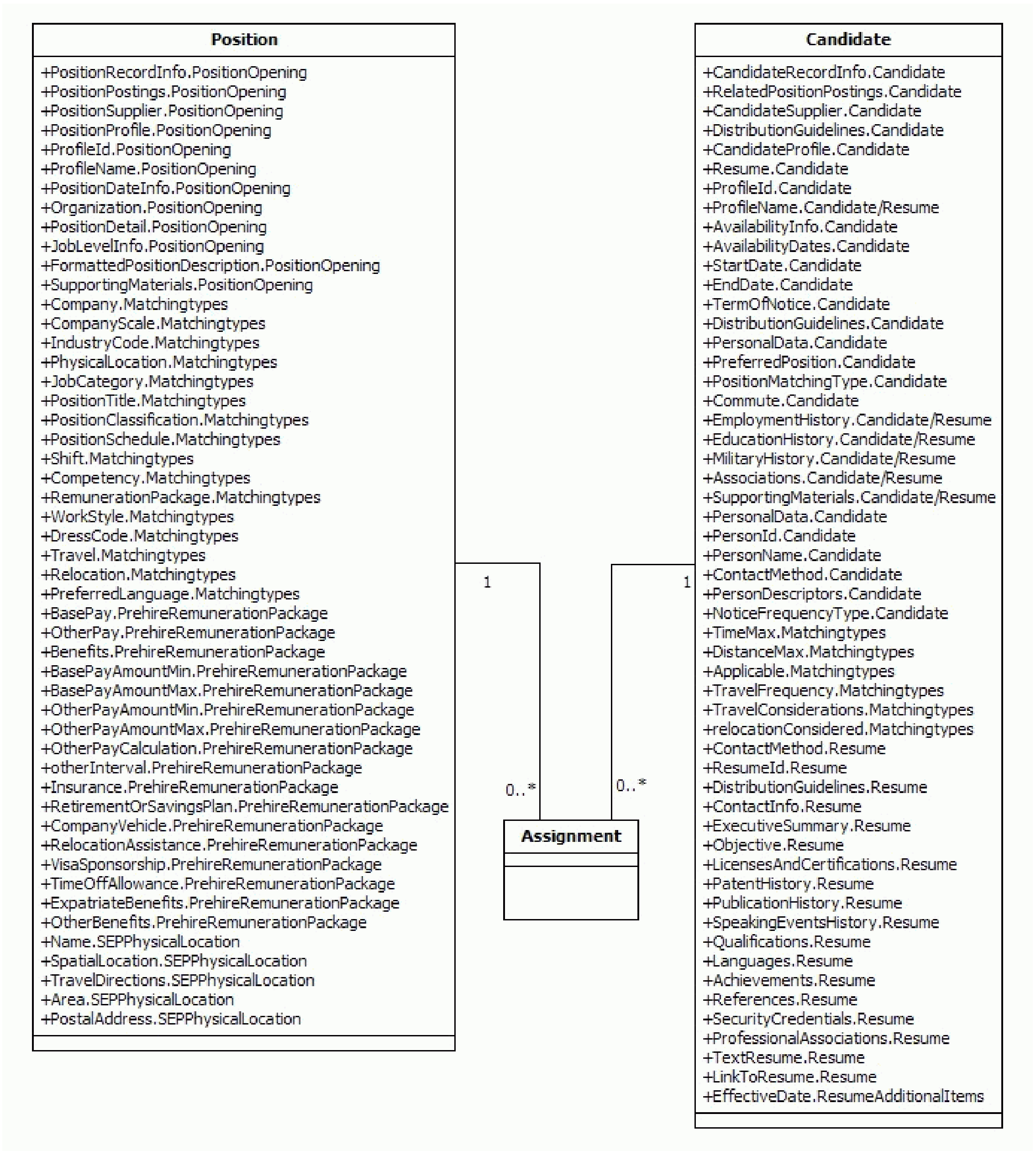
Appendix V. Class diagram SIDES




Resource
+PersonName.ContactInfo
+ContactMethod.ContactInfo
+EntityName.ContactInfo
+EntityContactInfo.ContactInfo
+HumanResourceId.HumanResource
+HumanResourceStatus.HumanResource
+ReferenceInformation.HumanResource
+StaffingSupplierId.HumanResource
+StaffingCustomerId.HumanResource
+IntermediaryId.HumanResource
+OrderId.HumanResource
+PositionId.HumanResource
+AssignmentId.HumanResource
+StaffingSupplierOrgUnitId.HumanResource
+StaffingCustomerOrgUnitId.HumanResource
+ResourceInformation.HumanResource
+ResourceType.HumanResource
+independentContractor.HumanResource
+payrolledEmployee.HumanResource
+PersonName.HumanResource
+EntityContactInfo.HumanResource
+PostalAddress.HumanResource
+AvailabilityDate.HumanResource
+Rates.HumanResource
+Profile.HumanResource
+Competency.HumanResource
+Resume.HumanResource
+ResourceScreening.HumanResource
+Preferences.HumanResource
+DesiredShift.HumanResource
+PercentageTravel.HumanResource
+TravelConsiderations.HumanResource
+willingToTravel.HumanResource
+TimeMax.HumanResource
+DistanceMax.HumanResource
+CommuteComments.HumanResource
+Relocation.HumanResource
+willingToRelocate.HumanResource
+DistributionRestrictions.HumanResource
+DesiredCompensation.HumanResource
+screeningType.ResourceScreening
+ResourceScreening.ResourceScreening
+KnownScreeningType.ResourceScreening
+StaffingScreeningType.ResourceScreening
+Organization.StaffingOrganization
+PaymentInfo.StaffingOrganization
+ReferenceIdInfo.StaffingOrganization
+typeOfOrganization.StaffingOrganization
+StaffingOrganizationTypes.StaffingOrganization
+TypeOfOrganization.StaffingOrganization
+StaffingReferenceIdType.StaffingOrganization
+StaffingCustomerId.StaffingOrganization
+StaffingCustomerOrgUnitId.StaffingOrganization
+StaffingSupplierId.StaffingOrganization
+StaffingSupplierOrgUnitId.StaffingOrganization
+OrderId.StaffingOrganization
+HumanResourceId.StaffingOrganization
+IntermediaryId.StaffingOrganization
+PositionId.StaffingOrganization
+BillToEntityId.StaffingOrganization
+AssignmentId.StaffingOrganization
+TimeCardId.StaffingOrganization
+InvoiceId.StaffingOrganization
+PaymentInfoType.StaffingOrganization
+OrganizationId.StaffingOrganization
+OrganizationalUnitId.StaffingOrganization
+VATRate.StaffingOrganization
+PaymentCondition.StaffingOrganization
+PaymentMode.StaffingOrganization
+PaymentTimeAllowed.StaffingOrganization
+PaymentEvent.StaffingOrganization
+PaymentDay.StaffingOrganization
+BankAccountInfo.StaffingOrganization
+currencyCode.StaffingOrganization
+FinancialGuarantee.StaffingOrganization
+Amount.StaffingOrganization
+CollectiveAgreement.StaffingOrganization
+StaffingSupplierId.StaffingSupplier
+CompanyInfo.StaffingSupplier
+CompanyName.StaffingSupplier
+ContactType.StaffingSupplier
+ContactInfo.StaffingSupplier
+StaffingSupplierSite.StaffingSupplier
+SupplierSiteId.StaffingSupplier
+SiteName.StaffingSupplier
+SiteContact.StaffingSupplier
Order
+StaffingOrderType.StaffingOrder
+OrderId.StaffingOrder
+ReferenceInformation.StaffingOrder
+MasterOrderId.StaffingOrder
+StaffingCustomerId.StaffingOrder
+StaffingCustomerOrgUnitId.StaffingOrder
+IntermediaryId.StaffingOrder
+StaffingSupplierId.StaffingOrder
+BillToEntityId.StaffingOrder
+StaffingSupplierOrgUnitId.StaffingOrder
+CustomerReportingRequirements.StaffingOrder
+OrderClassification.StaffingOrder
+orderType.StaffingOrder
+orderStatus.StaffingOrder
+BillToAttention.StaffingOrder
+OrderContact.StaffingOrder
+contactType.StaffingOrder
+RequiredResponseDate.StaffingOrder
+OrderComments.StaffingOrder
+PositionQuantity.StaffingOrder
+PositionQuantityOpen.StaffingOrder
+MultiVendorDistribution.StaffingOrder
+StaffingPosition.StaffingOrder
+KnownOrderType.StaffingOrder
+OrderType.StaffingOrder
+KnownOrderStatus.StaffingOrder
+OrderStatusType.StaffingOrder
+StaffingCustomerType.StaffingCustomer
+StaffingCustomerId.StaffingCustomer
+MasterCustomerInformation.StaffingCustomer
+StaffingCustomerName.StaffingCustomer
+StaffingCustomerIndustry.StaffingCustomer
+StaffingCustomerContactInfo.StaffingCustomer
+StaffingCustomerIdentifiers.StaffingCustomer
+StaffingCustomerOrgUnit.StaffingCustomer
+IndustryCode.StaffingCustomer
+StaffingCustomerIdentifierType.StaffingCustomer
+TaxIdNumber.StaffingCustomer
+LegalIdNumber.StaffingCustomer
+DunsNumber.StaffingCustomer
+StaffingDunsNumber.StaffingCustomer
+PositionHeader.StaffingPosition
+CustomerReportingRequirements
+DepartmentName.StaffingPosition
+PositionReason.StaffingPosition
+PositionDateRange.StaffingPosition
+StartDate.StaffingPosition
+ExpectedEndDate.StaffingPosition
+ActualEndDate.StaffingPosition
+MaxStartDate.StaffingPosition
+StartAsSoonAsPossible.StaffingPosition
+MaxNeedEndDate.StaffingPosition
+ReportToPerson.StaffingPosition
+ContactInfo.StaffingPosition
+PositionContact.StaffingPosition
+Rates.StaffingPosition
+WorkSite.StaffingPosition
+WorkSiteEnvironment.StaffingPosition
+StaffingShift.StaffingPosition
+PositionRequirements.StaffingPosition
+Competency.StaffingPosition
+ScreeningRequirements.StaffingPosition
Assignment
+AssignmentId.Assignment
+ReferenceInformation.Assignment
+CustomerReportingRequirements.Assignment
+Rates.Assignment
+StaffingShift.Assignment
+AssignmentDateRange.Assignment
+StartDate.Assignment
+ExpectedEndDate.Assignment
+ActualEndDate.Assignment
+FlexibilityMinDate.Assignment
+FlexibilityMaxDate.Assignment
+ProbationaryPeriod.Assignment
+SuspensionPeriod.Assignment
+EndReasonComments.Assignment
+ContractId.Assignment
+LegalInformation.Assignment
+ContractVersion.Assignment
+ContractVersionDate.Assignment
+StaffType.Assignment
+LocalContractRequirements.Assignment
+FrameAgreementInfo.Assignment
+DocumentId.Assignment
+DocumentVersion.Assignment
+ValidityDateRange.Assignment
+DocumentName.Assignment
+DocumentType.Assignment
+StaffingOrgContact.Assignment
+WorkSite.Assignment
+StartStaffingShiftId.Assignment
+ContractLegalReason.Assignment
+CompensationReference.Assignment
+SupplierSignature.Assignment
+CustomerSignature.Assignment
+PositionTitle.Assignment
+PositionStatus.Assignment
+Description.Assignment
+PositionLevel.Assignment
+PositionCoefficient.Assignment
+PositionSpecificCondition.Assignment
+ManagerName.CustomerReportingRequirements
+SupervisorName.CustomerReportingRequirements
+ContactName.CustomerReportingRequirements
+PurchaseOrderNumber.CustomerReportingRequirements
+DepartmentCode.CustomerReportingRequirements
+DepartmentName.CustomerReportingRequirements
+LocationCode.CustomerReportingRequirements
+LocationName.CustomerReportingRequirements
+CostCenterCode.CustomerReportingRequirements
+CostCenterName.CustomerReportingRequirements
+CustomerJobCode.CustomerReportingRequirements
+CustomerJobDescription.CustomerReportingRequirements
+AccountCode.CustomerReportingRequirements
+ProjectCode.CustomerReportingRequirements
+ExternalOrderNumber.CustomerReportingRequirements
+ExternalReqNumber.CustomerReportingRequirements
+Entity.CustomerReportingRequirements
+SubEntity.CustomerReportingRequirements
+Shift.CustomerReportingRequirements
+CustomerReferenceNumber.CustomerReportingRequirements
+AdditionalRequirement.CustomerReportingRequirements
+requirementTitle.CustomerReportingRequirements
+PurchaseOrderLineItem.CustomerReportingRequirements
+KnownContactType.StaffingContactType
+StaffingActionId.StaffingAction
+StaffingActionInfo.StaffingAction
+ActionType.StaffingAction
+ActionTypeComments.StaffingAction
+ActionReason.StaffingAction
+ActionReasonCode.StaffingAction
+ActionSubject.StaffingAction
+ReferenceInformation.StaffingAction
+HumanResourceId.StaffingAction
+OrderId.StaffingAction
+PositionId.StaffingAction
+AssignmentId.StaffingAction
+IntermediaryId.StaffingAction
+StaffingSupplierId.StaffingAction
+StaffingCustomerId.StaffingAction
+StaffingSupplierOrgUnitId.StaffingAction
+StaffingCustomerOrgUnitId.StaffingAction
+TimeCardId.StaffingAction
+InvoiceId.StaffingAction
+StaffingActionContact.StaffingAction
+ReferenceInformation.StaffingAction
+StaffingSupplierId.StaffingAction
+IntermediaryId.StaffingAction
+StaffingCustomerId.StaffingAction
+ActionSchedule.StaffingAction
+ContactPerson.StaffingAction
+ContactPeriod.StaffingAction
+StartDateTime.StaffingAction
+EndDateTime.StaffingAction
+SpecialRequirements.StaffingAction
+WorkSiteId.StaffingWorkSite
+Id.StaffingWorkSite
+Domain.StaffingWorkSite
+IdIssuer.StaffingWorkSite
+IdType.StaffingWorkSite
+WorkSiteName.StaffingWorkSite
+WorkSiteDetail.StaffingWorkSite
+PostalAddress.StaffingWorkSite
+TravelDirections.StaffingWorkSite
+ParkingInstructions.StaffingWorkSite
+EnvironmentId.StaffingWorkSiteEnvironment
+Id.StaffingWorkSiteEnvironment
+Domain.StaffingWorkSiteEnvironment
+IdIssuer.StaffingWorkSiteEnvironment
+IdType.StaffingWorkSiteEnvironment
+EnvironmentName.StaffingWorkSiteEnvironment
+EnvironmentDescription.StaffingWorkSiteEnvironment
+EnvironmentConsideration.StaffingWorkSiteEnvironment
+suppliedByOrganization.StaffingWorkSiteEnvironment
+typeConsideration.StaffingWorkSiteEnvironment
+General.StaffingWorkSiteEnvironment
+Physical.StaffingWorkSiteEnvironment
+SafetyEquipment.StaffingWorkSiteEnvironment
+DressCode.StaffingWorkSiteEnvironment
1
0..*
1
0..*


Timesheet
+ActionCodeType.TimeCard
+currency.TimeCard
+DayAssignmentType.TimeCard
+DayAssignmentEnumerationType.TimeCard
+AdditionalDataType.TimeCard
+ApprovalInfoType.TimeCard
+Person.TimeCard
+ApprovedDateTime.TimeCard
+SubmitterInfoType.TimeCard
+Source.TimeCard
+SubmittedDateTime.TimeCard
+TimeCardPersonType.TimeCard
+Id.TimeCard
+PersonName.TimeCard
+ReportedResource.TimeCard
+Resource.TimeCard
+ResourceName.TimeCard
+AdditionalData.TimeCard
+ReportedTime.TimeCard
+PeriodStartDate.TimeCard
+PeriodEndDate.TimeCard
+ReportedPersonAssignment.TimeCard
+TimeInterval.TimeCard
+StartDateTime.TimeCard
+EndDateTime.TimeCard
+Duration.TimeCard
+PieceWork.TimeCard
+Piece.TimeCard
+PieceValue.TimeCard
+Quantity.TimeCard
+unitOfMeasure.TimeCard
+RateOrAmount.TimeCard
+currency.TimeCard
+type.TimeCard
+period.TimeCard
+multiplier.TimeCard
+toBeBilled.TimeCard
+toBePaid.TimeCard
+Allowance.TimeCard
+Amount.TimeCard
+Quantity.TimeCard
+ApprovalInfo.TimeCard
+dayAssignment.TimeCard
+billable.TimeCard
+actionCode.TimeCard
+TimeEvent.TimeCard
+EventDateTime.TimeCard
+dayAssignment.TimeCard
+Expense.TimeCard
+ExpenseDate.TimeCard
+ExpenseAmount.TimeCard
+StartDate.TimeCard
+EndDate.TimeCard
+SubmitterInfo.TimeCard
+TimeCardDuration.TimeCard
+ReferenceInformation.TimeCardAdditionalData
0..*
1..*
1
1
Invoice
+Currency.Invoice
+PackingMaterial.Invoice
+PackingCode.Invoice
+AcknowledgementCode.Invoice
+Country.Invoice
+LicenseType.Invoice
+PaymentMethod.Invoice
+Rating.Invoice
+UOM.Invoice
+InvoiceType.Invoice
+TransportationType.Invoice
+CostType.Invoice
+TransferType.Invoice
+OperationType.Invoice
+ManufacturingReportingFlag.Invoice
+TemperatureScale.Invoice
+DebitCredit.Invoice
+BusinessObjectDocument.Invoice
+Sender.Invoice
+Signature.Invoice
+EmployeeId.Invoice
+AddressId.Invoice
+SimpleId.Invoice
+Id.Invoice
+ItemCategoryId.Invoice
+LineNumber.Invoice
+JounrnalEntryId.Invoice
+EmployeeCategory.Invoice
+JobCode.Invoice
+WageGroup.Invoice
+WageType.Invoice
+Position.Invoice
+WorkCenter.Invoice
+Restriction.Invoice
+PartyDocumentId.Invoice
+CustomerDocumentId.Invoice
+SupplierDocumentId.Invoice
+CarrierDocumentId.Invoice
+BrokerDocumentId.Invoice
+LogisticsProviderDocumentId.Invoice
+ShippersDocumentId.Invoice
+PartyId.Invoice
+PartyAssignedPartyId.Invoice
+AssigningPartyId.Invoice
+BillToPartyId.Invoice
+CustomerPartyId.Invoice
+ShipToPartyId.Invoice
+SupplierPartyId.Invoice
+AlternatePartyIds.Invoice
+PrimaryDocumentId.Invoice
+ItemIdBase.Invoice
+CrossReferenceItemIds.Invoice
+ItemId.Invoice
+PartyAssignedItemId.Invoice
+UPC.Invoice
+EANUCC13.Invoice
+SupplierItemId.Invoice
+BuyerItemId.Invoice
+ManufacturerItemId.Invoice
+CustomerItemId.Invoice
+ShipFromItemId.Invoice
+CarrierItemId.Invoice
+SerialNumber.Invoice
+PersonCode.Invoice
+SalesPersonCode.Invoice
+Commodity.Invoice
+Relationship.Invoice
+Period.Invoice
+Preference.Invoice
+Classification.Invoice
+Tag.Invoice
+NameValuePair.Invoice
+TransportationTerms.Invoice
+FreightClass.Invoice
+FreightTerms.Invoice
+Priority.Invoice
+Name.Invoice
+LingualString.Invoice
+Indicator.Invoice
+Usage.Invoice
+SalesOrganization.Invoice
+License.Invoice
+GLAccount.Invoice
+GLEntity.Invoice
+TaxWithholdingExempt.Invoice
+Description.Invoice
+PO.Invoice
+EMailAddress.Invoice
+ISBN.Invoice
+SpecialHandlingNote.Invoice
+ShippingNote.Invoice
+PostalCode.Invoice
+Reason.Invoice
+TaxCode.Invoice
+TaxJurisdiction.Invoice
+TelephoneNumber.Invoice
+Department.Invoice
+CostCenter.Invoice
+DistributionCenter.Invoice
+Division.Invoice
+Fund.Invoice
+Line.Invoice
+Header.Invoice
+Invoice.Invoice
+SalesInformation.Invoice
+Planner.Invoice
+SalesPerson.Invoice
+SalesInformation.Invoice
+OrderSchedule.Invoice
+OrderSubLine.Invoice
+OrderLine.Invoice
+OrderHeader.Invoice
+Order.Invoice
+Document.Invoice
+DocumentOrderHeader.Invoice
+DocumentLine.Invoice
+DocumentHeader.Invoice
+AuthorizationType.Invoice
+ProjectResourceCategory.Invoice
+ProjectActivity.Invoice
+Project.Invoice
+LedgerEvent.Invoice
+RelatedUnitType.Invoice
+OrganizationalUnit.Invoice
+AccountingPeriod.Invoice
+LotSerial.Invoice
+Lot.Invoice
+ShippingMaterial.Invoice
+Packaging.Invoice
+HazardousMaterial.Invoice
+Message.Invoice
+TransportationTerm.Invoice
+Location.Invoice
+Tax.Invoice
+AcknowledgementDetail.Invoice
+OrderStatus.Invoice
+Disposition.Invoice
+Status.Invoice
+StateChange.Invoice
+Item.Invoice
+PartyReference.Invoice
+Contact.Invoice
+Charge.Invoice
+Allowance.Invoice
+Addresses.Invoice
+UserAccount.Invoice
+remAuthorization.Invoice
+User.Invoice
+EmployeeAssignment.Invoice
+EmployeeQualification.Invoice
+Someone.Invoice
+Employee.Invoice
+PersonName.Invoice
+PersonCode.Invoice
+Person.Invoice
+PaymentTerms.Invoice
+Attachment.Invoice
+References.Invoice
+Distribution.Invoice
+ProjectActivityWorkEffort.Invoice
+ProjectActivityType.Invoice
+ProjectResource.Invoice
+ProjectTransactionType.Invoice
+ProjectActivityStatus.Invoice
+ProjectActivityId.Invoice
+ProjectStatus.Invoice
+ProjectId.Invoice
+ProductLine.Invoice
+BusinessUnitId.Invoice
+LedgerId.Invoice
+BusinessUnit.Invoice
+BusinessArea.Invoice
+ChargeId.Invoice
+UserId.Invoice
+AccountingPeriodName.Invoice
+Remittance.Invoice
+DatePeriod.Invoice
+TimePeriod.Invoice
+TransportationMethod.Invoice
+ServiceLevel.Invoice
+RouteCode.Invoice
+CorrespondenceLanguage.Invoice
+ChargeClass.Invoice
+Temperature.Invoice
+Quantity.Invoice
+Percent.Invoice
+AmountPerQuantity.Invoice
+FunctionalAmount.Invoice
+Amount.Invoice
+Suffix.Invoice
+Variation.Invoice
+Revision.Invoice
+LotNumberSpecification.Invoice
+LotName.Invoice
+ValueClass.Invoice
+Factor.Invoice
+LoadingDockCode.Invoice
+PurchasingEntityCode.Invoice
+Warehouse.Invoice
+ProfitCenter.Invoice
+Geography.Invoice

Appendix VI. Class Diagram SEP


Position
+PositionRecordInfo.PositionOpening
+PositionPostings.PositionOpening
+PositionSupplier.PositionOpening
+PositionProfile.PositionOpening
+ProfileId.PositionOpening
+ProfileName.PositionOpening
+PositionDateInfo.PositionOpening
+Organization.PositionOpening
+PositionDetail.PositionOpening
+JobLevelInfo.PositionOpening
+FormattedPositionDescription.PositionOpening
+SupportingMaterials.PositionOpening
+Company.Matchingtypes
+CompanyScale.Matchingtypes
+IndustryCode.Matchingtypes
+PhysicalLocation.Matchingtypes
+JobCategory.Matchingtypes
+PositionTitle.Matchingtypes
+PositionClassification.Matchingtypes
+PositionSchedule.Matchingtypes
+Shift.Matchingtypes
+Competency.Matchingtypes
+RemunerationPackage.Matchingtypes
+WorkStyle.Matchingtypes
+DressCode.Matchingtypes
+Travel.Matchingtypes
+Relocation.Matchingtypes
+PreferredLanguage.Matchingtypes
+BasePay.PrehireRemunerationPackage
+OtherPay.PrehireRemunerationPackage
+Benefits.PrehireRemunerationPackage
+BasePayAmountMin.PrehireRemunerationPackage
+BasePayAmountMax.PrehireRemunerationPackage
+OtherPayAmountMin.PrehireRemunerationPackage
+OtherPayAmountMax.PrehireRemunerationPackage
+OtherPayCalculation.PrehireRemunerationPackage
+otherInterval.PrehireRemunerationPackage
+Insurance.PrehireRemunerationPackage
+RetirementOrSavingsPlan.PrehireRemunerationPackage
+CompanyVehicle.PrehireRemunerationPackage
+RelocationAssistance.PrehireRemunerationPackage
+VisaSponsorship.PrehireRemunerationPackage
+TimeOffAllowance.PrehireRemunerationPackage
+ExpatriateBenefits.PrehireRemunerationPackage
+OtherBenefits.PrehireRemunerationPackage
+Name.SEPPhysicalLocation
+SpatialLocation.SEPPhysicalLocation
+TravelDirections.SEPPhysicalLocation
+Area.SEPPhysicalLocation
+PostalAddress.SEPPhysicalLocation
1
0..*
Assignment
0..*
1
Candidate
+CandidateRecordInfo.Candidate
+RelatedPositionPostings.Candidate
+CandidateSupplier.Candidate
+DistributionGuidelines.Candidate
+CandidateProfile.Candidate
+Resume.Candidate
+ProfileId.Candidate
+ProfileName.Candidate/Resume
+AvailabilityInfo.Candidate
+AvailabilityDates.Candidate
+StartDate.Candidate
+EndDate.Candidate
+TermOfNotice.Candidate
+DistributionGuidelines.Candidate
+PersonalData.Candidate
+PreferredPosition.Candidate
+PositionMatchingType.Candidate
+Commute.Candidate
+EmploymentHistory.Candidate/Resume
+EducationHistory.Candidate/Resume
+MilitaryHistory.Candidate/Resume
+Associations.Candidate/Resume
+SupportingMaterials.Candidate/Resume
+PersonalData.Candidate
+PersonId.Candidate
+PersonName.Candidate
+ContactMethod.Candidate
+PersonDescriptors.Candidate
+NoticeFrequencyType.Candidate
+TimeMax.Matchingtypes
+DistanceMax.Matchingtypes
+Applicable.Matchingtypes
+TravelFrequency.Matchingtypes
+TravelConsiderations.Matchingtypes
+relocationConsidered.Matchingtypes
+ContactMethod.Resume
+ResumeId.Resume
+DistributionGuidelines.Resume
+ContactInfo.Resume
+ExecutiveSummary.Resume
+Objective.Resume
+LicensesAndCertifications.Resume
+PatentHistory.Resume
+PublicationHistory.Resume
+SpeakingEventsHistory.Resume
+Qualifications.Resume
+Languages.Resume
+Achievements.Resume
+References.Resume
+SecurityCredentials.Resume
+ProfessionalAssociations.Resume
+TextResume.Resume
+LinkToResume.Resume
+EffectiveDate.ResumeAdditionalItems