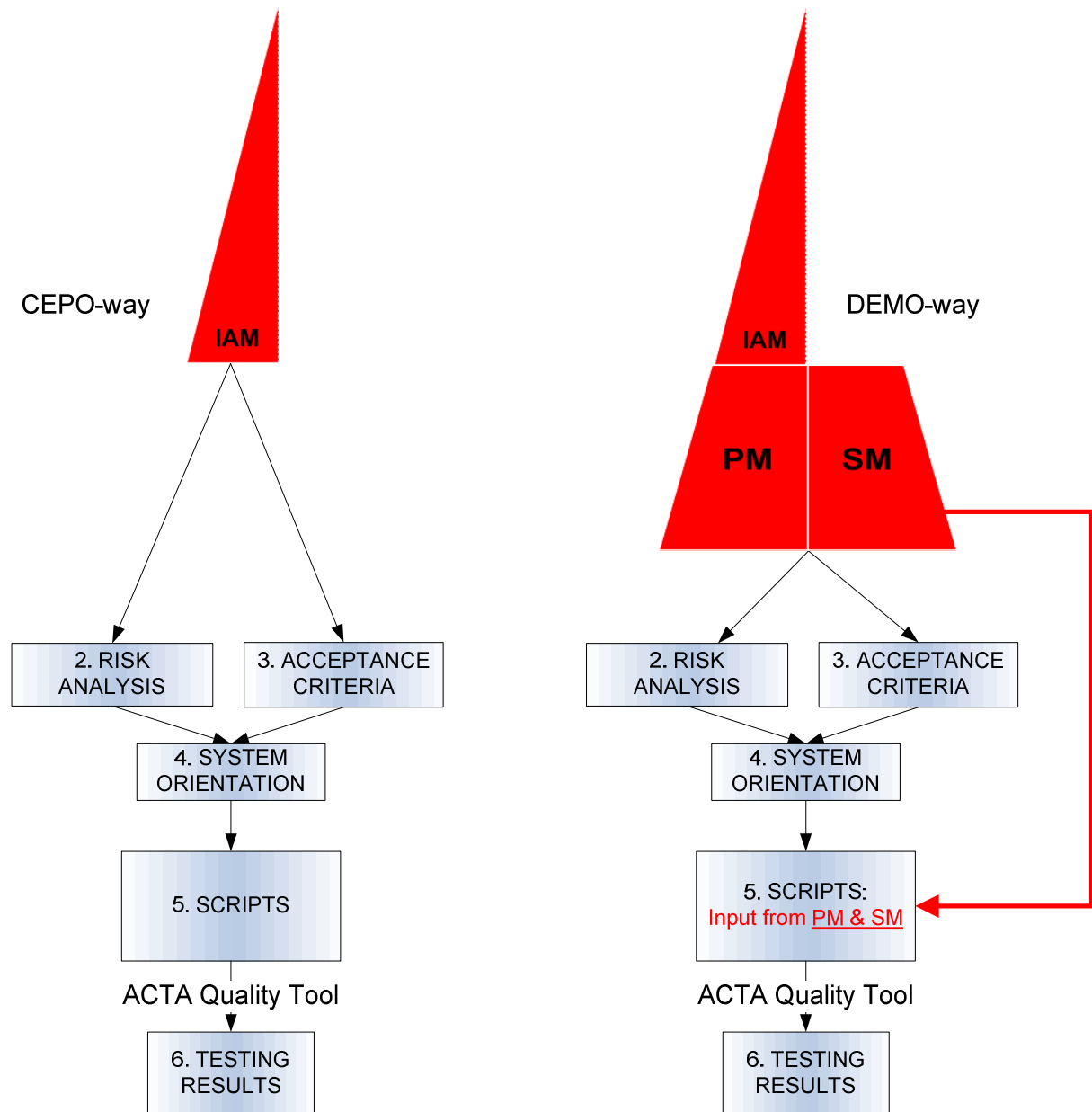


A Testing Framework Based on DEMO

'How the DEMO methodology can be a valuable addition to a testing framework'



Date: 06/07/2008
Author: L.J. Metz
Supervisor TU Delft: Prof. Dr. ir. J.L.G. Dietz
Supervisor CEPO: Drs. R. Ceelen

Executive Summary

Because many errors still exist in information systems for organizations, professional companies like CEPO are needed for testing information systems on their fit to the organization. To do so, one usually applies a testing framework. The existing (CEPO-way) framework is the result of ten years of experience.

The DEMO methodology (Dietz [1]) is a methodology that has proven itself successful in modelling organizations. DEMO is a powerful tool for identifying all important transactions of an organization (as well as communication with external actors). From this point of view, it could be possible to use DEMO as an aid to check the completeness of an information system, i.e. whether it covers the essential business processes. Note that the coverage need not be full; in particular for small and medium-sized enterprises it may be partial.

The existing CEPO-way framework is based on the DEMO methodology, but on only one model. In this study two models were added. The new, so-called DEMO-way, framework was discussed and one test case was used to demonstrate the added value.

The new framework is supported by a tool that was built by the author, DEMO-way Script Creator (DSC tool). This tool is a valuable addition to the derivation of testing scripts, because the process model as well as the state model can be placed into the scripts with little effort.

Several criteria were determined at the start of this study in order to evaluate the use of extra DEMO models in the framework: development time, walkthrough time, understandability, reproduction possibilities and lacking functionality / error detection. Ultimately, the DEMO-way scored better for many of these criteria, especially the detection of lacking functionality / errors in information systems. Although the DEMO testing script encompasses more testing instructions than the CEPO testing script, the testers did not spend more time walking through the scripts, due to more detailed navigation. Therefore, the recommendation is to start using the two extra DEMO models in combination with DSC tool.

Management Samenvatting

Vanwege het feit dat er nog altijd veel fouten in informatie systemen zitten, zijn er professionele bedrijven zoals CEPO nodig om de systemen te testen of ze wel of niet goed op organisaties aansluiten. Om dat goed uit te voeren gebruikt men een test framework. Het bestaande CEPO-framework is dan ook het resultaat van tien jaar ervaring.

De DEMO methodologie (Dietz [1]) is een methodology die zichzelf bewezen heeft in het succesvol modelleren van organisaties. DEMO is een krachtige tool om de belangrijkste transacties van een organisatie te identificeren (evenals de communicatie met externe actoren). Vanuit dit perspectief is het mogelijk om DEMO als hulpmiddel te gebruiken om de volledigheid van een informatie systeem te toetsen. De dekking hoeft niet volledig te zijn, voor midden en kleine bedrijven mag het ook deels zijn.

Het bestaande CEPO-framework is deels gebaseerd op de DEMO methodologie, maar het gebruikt slechts één model. In dit onderzoek zijn hier twee modellen aan toegevoegd. Het nieuwe DEMO-framework wordt besproken en er is een test case gebruikt om de toegevoegde waarde aan te tonen.

Het nieuwe framework wordt ondersteund door een programma dat door de auteur is gemaakt, de *DEMO-way Script Creator*. Deze tool is een geschikte toevoeging op het maken van test scripts, omdat zowel het procesmodel als het toestandsmodel met weinig moeite aan de scripts toegevoegd kunnen worden.

Een aantal criteria is aan het begin van dit onderzoek bepaald, te weten: ontwikkelingstijd, doorlooptijd, begrijpbaarheid, mogelijkheid tot reproductie van fouten en het vinden van deze fouten. Het bleek uiteindelijk dat het DEMO-framework beter scoorde op veel van deze criteria, vooral in het ontdekken van gebreken in het informatie systeem. Ondanks het feit dat het DEMO-framework uit meer test instructies bestaat, kostte het de tester evenveel tijd om hier doorheen te komen, dankzij de meer gedetailleerde navigatie instructies. Vandaar dat de aanbeveling is om voortaan de extra DEMO modellen te gebruiken met behulp van de DEMO-way Script Creator.

Sommaire Exécutif

Comme il existe toujours pas mal d'erreurs dans les systèmes informatiques, l'on a besoin d'entreprises telles que CEPO afin de tester que ces systèmes soient adaptés aux besoins des organisations. Pour ce faire, l'on se sert d'ordinaire d'un dispositif de test. Le dispositif CEPO actuel est le résultat de dix années d'expérience.

La méthodologie DEMO (Dietz [1]) est une méthodologie qui a fait ses preuves dans le domaine du modelage des organisations. DEMO est un outil puissant pour l'identification de toutes les transactions importantes d'une organisation (y compris la communication avec des acteurs extérieurs). De ce point de vue, il pourrait être possible de se servir de DEMO en tant qu'auxiliaire pour vérifier le caractère exhaustif d'un système informatique, c'est-à-dire pour savoir s'il couvre tous les processus essentiels de l'entreprise. La couverture n'a pas toujours besoin d'être complète, surtout pour les petites et moyennes entreprises une couverture partielle pourrait suffire.

Le dispositif CEPO actuel est partiellement basé sur la méthodologie DEMO, mais se sert seulement d'un modèle. Dans cette étude deux modèles ont été ajoutés. Le nouveau dispositif DEMO a été débattu et mis à l'épreuve afin de démontrer la valeur ajoutée.

Le dispositif nouveau est supporté par un programme développé par l'auteur lui-même, le DEMO-way Script Creator (DSC tool). Cet outil est un atout précieux pour la dérivation des scripts de test, parce que le modèle de processus aussi bien que le modèle d'état peuvent être ajoutés au script facilement.

Au début de cette étude plusieurs critères ont été identifiés, à savoir: la durée de développement, le temps requis pour parcourir les scripts, la compréhensibilité, les possibilités de reproduction d'anomalies et la détection de ces anomalies. Il s'est avéré dans cette étude que le dispositif DEMO réalisait un score plus performant pour plusieurs de ces critères, notamment en ce qui concerne la détection des anomalies. Quoique le dispositif DEMO comprenne plus d'instructions de test, le testeur n'a pas eu besoin de plus de temps pour parcourir les scripts, grâce à des instructions de navigation plus détaillées. De ce fait, il est recommandable de se servir dorénavant des deux modèles DEMO supplémentaires en combinaison avec le DEMO-way Script Creator.

Preface

This document provides the report for the thesis project at CEPO, located in Gorinchem, the Netherlands. It is intended for actors directly or indirectly involved with testing frameworks. It is recommended that the reader has knowledge of the DEMO methodology [1].

This project was carried out by a master's degree student from Delft University of Technology (TU Delft). It was conducted from November 2007 to July 2008 as part of the TU Delft Thesis Project, IN5000, as part of the Computer Science master's programme variant Information Architecture.

I would like to give many thanks to Jan Dietz for the use of a room at the EEMCS building, for his clear scientific support during this study and for arranging my internship at CEPO.

Rene Ceelen, the CEO of CEPO, also managed to provide valuable additions to this study. Thanks to his fast thinking, experience and result-oriented attitude, I was able to lift the study to a higher level. I am grateful for the fact that Paul Kastelijn helped me a lot in the analysis phase and during the days of testing.

Marijn Janssen, member of the thesis committee, is thanked for providing feedback at an early stage of the writing phase to increase the scientific level of this study.

Hans Tonino, member of the thesis committee, gave his criticism that proved very valuable. I would like to thank him as well for his support.

Hetty Splinter, Carlijn Janson and Karel van Dorp were the testers who helped me during the days of testing. If they had not been available for two days in their spare time and if they had not provided such excellent feedback on the information system, this study would not have the solid foundation as it has now.

My thanks also go to Jan van den Berg and Michel Meulpolder for providing objective feedback on the scientific foundation.

I hope you enjoy reading this paper.

Leiden, 6 July, 2008

Lauwerens J. Metz

Contents

Executive Summary	2
Management Samenvatting	3
Sommaire Exécutif.....	4
Preface.....	5
Acronyms.....	8
List of tables	9
List of figures	10
1 Introduction.....	13
1.1 Background.....	14
1.2 Research Goal.....	17
1.3 Research Approach.....	18
1.3.1 Document Structure	19
1.3.2 Methodologies, Methods and Tools Used	19
1.4 Organizational Structure	23
2 Testing Frameworks	27
2.1 Existing CEPO-way vs. New DEMO-way	27
2.2 Existing CEPO-way Framework.....	28
2.3 New Framework DEMO-way	31
3 Applying a Testing Framework	33
3.1 Interaction Model.....	35
3.2 Process Model	37
3.3 State Model	38
3.4 Transaction Groups and Business Processes.....	41
3.5 Risk Analysis.....	42
3.5.1 Scheduling Care and Freelancers	42
3.5.2 Payment Management.....	43
3.6 Acceptance Criteria	45
3.7 System Orientation with Documentation of Care Arrangement	46
3.8 Derivation of Testing Scripts	47
3.8.1 CEPO-way Definitions.....	47
3.8.2 CEPO-way Scripting	49
3.8.3 DEMO-way Definitions	50
3.8.4 DEMO-way Scripting.....	52

3.9	Testing Results.....	56
4	Comparison of Testing Frameworks.....	59
4.1	Quantification of Testing Results	60
4.2	Comparison Based on Acceptance Criteria	61
4.3	Comparison Based on Transactions	62
4.4	Comparison Based on Additional Criteria	63
5	Conclusion	65
6	Recommendations.....	67
7	Reflection.....	69
	References.....	71
	Appendix A DEMO description	73
	Appendix B Testing Plan of Action (in Dutch).....	75
	Appendix C Transaction Process Diagrams	79
	Appendix D Testing Scripts CEPO-way (in Dutch).....	92
	Appendix E Testing Scripts DEMO-way (in Dutch)	99
	Appendix F Client Report (in Dutch).....	111

Acronyms

DEMO – Design and Engineering Methodology for Organizations

TU Delft – Delft University of Technology

WSD – Wilhelmina Senioren Diensten

AQT – Acta Quality Tool

DSC tool – DEMO-way Script Creator

CM – Construction Model

PM – Process Model

AM – Action Model

SM – State Model

List of tables

Table 1: TMap next vs. CEPO

Table 2: Document structure

Table 3: Group composition and schedule

Table 4: Stakeholder descriptions

Table 5: Expectation matrix, an explanation of the numbers can be found below

Table 6: Enumeration of added value per testing step in the CEPO-way

Table 7: Symbols used for reporting findings

Table 8: Overview of the characteristics of DEMO models

Table 9: Enumeration of added value per testing step in DEMO-way

Table 10: Transaction Result Table from ATD of WSD

Table 11: OPL of WSD

Table 12: Time-OPL of WSD

Table 13: IUT of WSD

Table 14: Business Processes / Groups of Transactions

Table 15: Standard care query process mapped to transactions

Table 16: Payment management process mapped to transactions

Table 17: Acceptance criteria enumerated

Table 18: One example of a testing script with 3 intertwined scenarios consisting of 10 testing instructions

Table 19: Several scenarios in CEPO-way

Table 20: Scripting layout CEPO-way

Table 21: Definitions of abbreviations

Table 22: Final testing results based on findings

Table 23: Overview of CEPO versus DEMO

List of figures

Figure 1: The communication problem between software supplier and client

Figure 2: Position of CEPO in organizational & IT strategy

Figure 3: Seven phases of TMap next

Figure 4: Mapping from CEPO's testing framework

Figure 5: Overview of the research approach

Figure 6: The DEMO methodology and its aspect models

Figure 7: Screenshot of DEMO-way Script Creator

Figure 8: Screenshot of AQT

Figure 9: Overview of connections between organizations

Figure 10: Stakeholders overview

Figure 11: Both testing frameworks illustrated

Figure 12: Example of risk analysis

Figure 13: Screenshot of Care Arrangement

Figure 14: WSD expressed in an Actor Transaction Diagram

Figure 15: WSD Business Process Diagram

Figure 16: State Model (OFD)

Figure 17: Risk analysis of WSD

Figure 18: Financial flows of WSD

Figure 19: Document flows of WSD

Figure 20: Decision group

Figure 21: Definitions and relations used in CEPO-way

Figure 22: Parallel scenarios

Figure 23: DEMO-way vs. CEPO-way definitions including a transaction from the Interaction Model

Figure 24: Single scenario DEMO-way

Figure 25: Entering the State Model

Figure 26: DSC tool's relationship to a Transaction Process Diagram

Figure 27: Adding State Model to testing scripts

Figure 28: Adding cement between transactions

Figure 29: Example of a radar chart

Figure 30: CEPO-way with acceptance criteria on its axes

Figure 31: DEMO-way with acceptance criteria on its axes

Figure 32: CEPO-way with business processes on its axes

Figure 33: DEMO-way with business processes on its axes

Figure 34: DEMO transaction

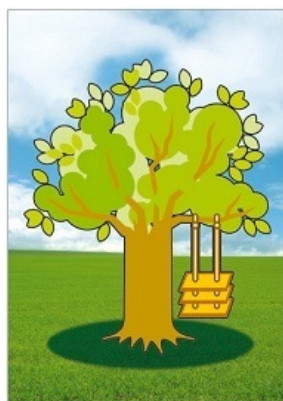
Figure 35: Ontological, Infological and Datalogical

Part A Introduction and Background

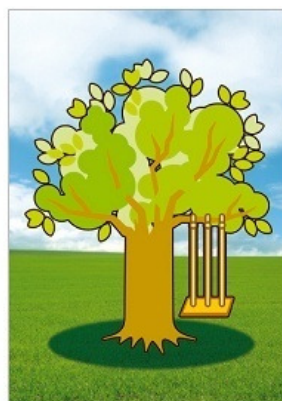
1 Introduction

Back in the early days of software development software projects failed, and today this is still the case. Software projects are cancelled or cost almost twice the estimated budget [2, 3]. There is an enormous amount of projects that fail [4, 5], one example being projects of the Dutch government [6, 7]. Much has been written about preventing these failures [8-12], however, software projects still fail far too often. If one takes a look at a random website about information technology, many problems within this sector can be found: failed software projects, software leaks [13] and software patches to correct all kinds of errors are in the daily news. There are many reasons for the failure of an implementation: political, functional, technical, budgetary and so forth [14].

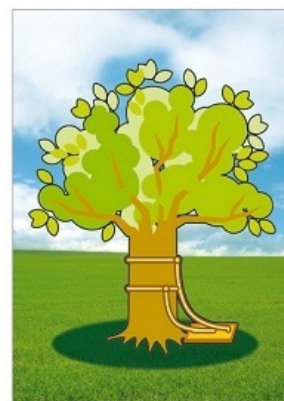
One of the most important factors within this problem was detected by *CEPO beyond ICT* (CEPO [15]); it is a lack of communication. The communication between software supplier and client is definitely not an easy task due to the fact that a software supplier must have insight into the business processes of the client, and the client has no knowledge of the capabilities of information technology. Therefore, CEPO acts as an intermediary between two parties to prevent well-known situations as depicted in Figure 1.



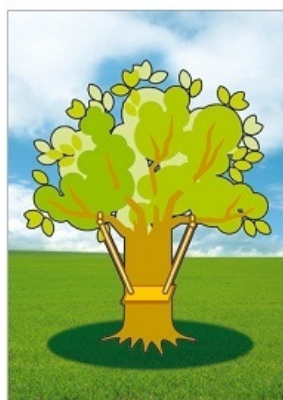
What the client asked for..



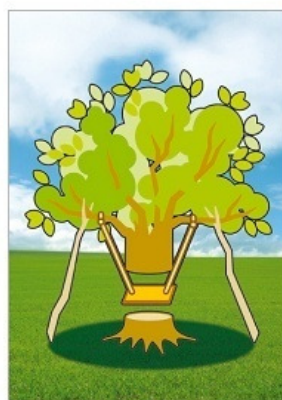
What was stated in the contract...



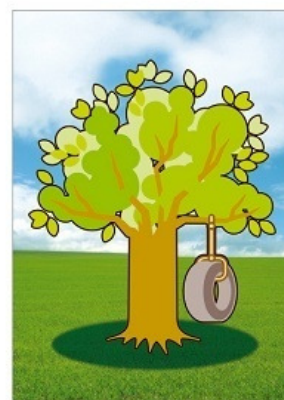
What the analyst designed...



What the programmer built...



What the software supplier delivered...



What the actual need of the client was...

Figure 1: The communication problem between software supplier and client

1.1 Background

The core business of CEPO is testing software. Figure 2 shows the position of CEPO using the terminology common in the field such as IT and organizational strategy and business processes.

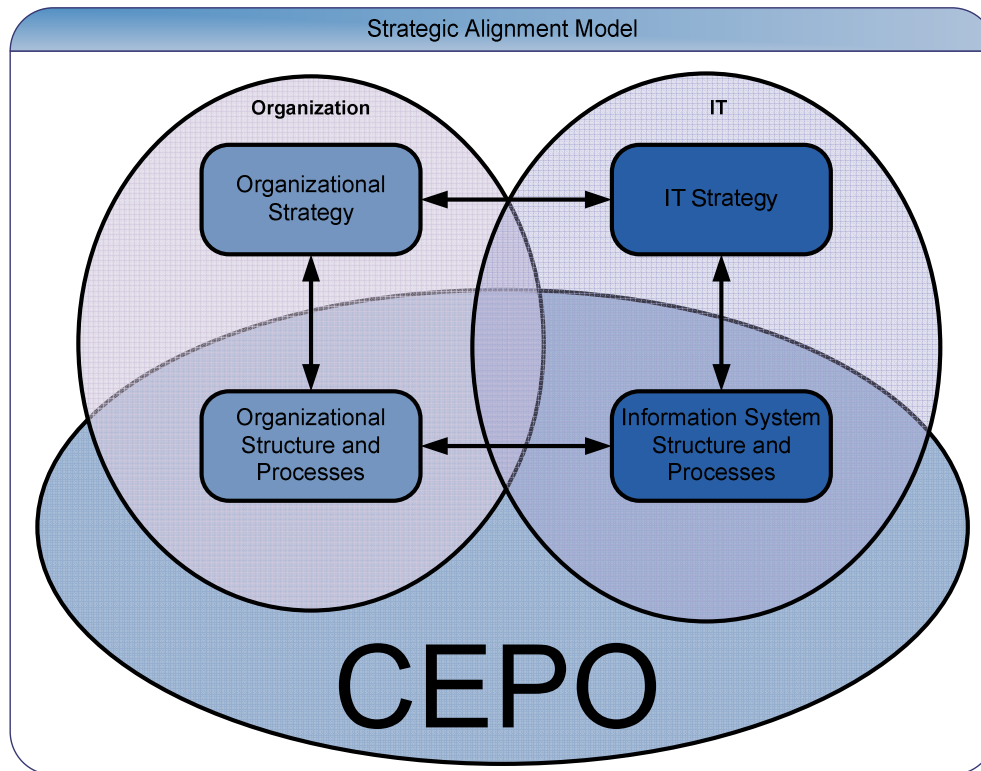


Figure 2: Position of CEPO in organizational & IT strategy

CEPO does acceptance testing, they perform research on a software package that has already been selected and provide advice and recommendations derived from its orientation. Confirmed by several sources, acceptance testing is defined as [16]:

In engineering and its various sub-disciplines, acceptance testing is black-box testing performed on a system (e.g. software, lots of manufactured mechanical parts, or batches of chemical products) before its delivery. In some engineering sub-disciplines, it is known as functional testing, black-box testing, release acceptance, QA testing, application testing, confidence testing, final testing, validation testing, usability testing, or factory acceptance testing.

Although the definition states *before its delivery*, CEPO also sends consultants to validate the information system after the deployment phase. The focus of CEPO lies more on validation than on verification, respectively meaning ‘you built the correct thing’ and ‘you built it correctly’.

A testing framework to perform tests is used to detect misunderstandings and to measure the quality of software. A framework is a basic conceptual structure used to solve or address complex issues [17]. A testing framework consists of separate phases or steps that the testing consultant has to follow. Sogeti [18] created a well-structured framework, which includes acceptance testing, called *TMap next*. *TMap next* is a widely used testing framework in the Netherlands, Belgium and Germany [19]. The testing framework of CEPO is partly derived from the framework that *TMap next* describes

[20, 21]. *TMap next* includes process and functional testing while CEPO's focus lies more on process testing.

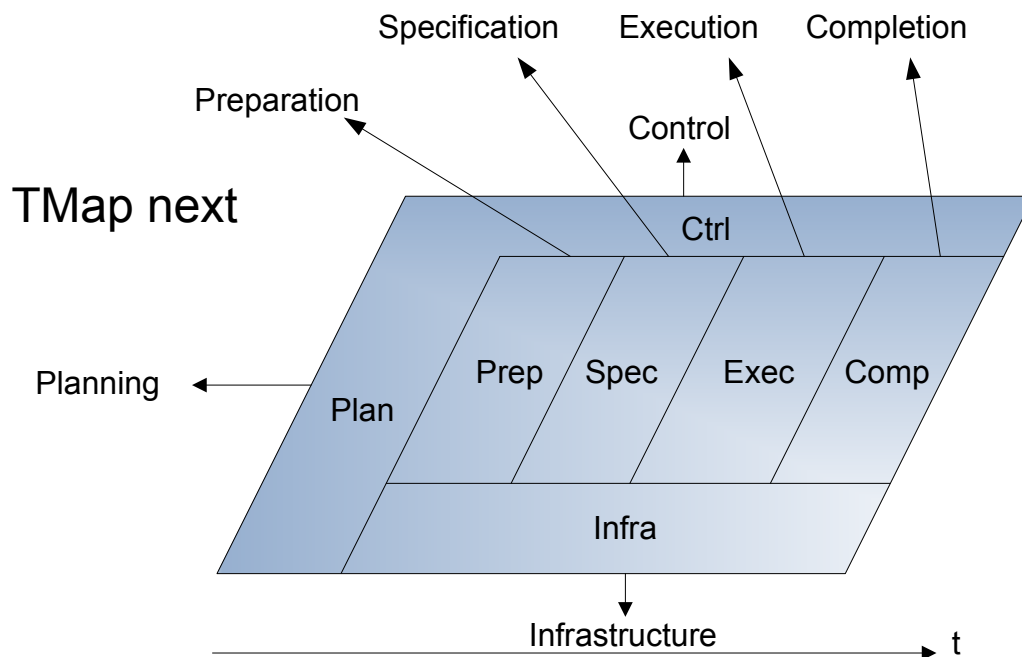


Figure 3: Seven phases of TMap next

The *TMap next* testing framework [22] is divided into several phases of which the first, second and third (planning, control and infrastructure) are continuous, see Figure 3.

1. *Planning*: a testing plan is built to inform every involved actor about the testing approach, the planning, the budget, the activities and the deliverables.
2. *Control*: this phase monitors the progress, quality and risks of the project. Predetermined deliverables are used to communicate this to the client.
3. *Infrastructure*: describes the infrastructure that is needed during the various phases and activities (e.g. available testing room, supporting testing tools, etc.)
4. *Preparation*: to obtain, with the client's approval, a testing basis that is of sufficient quality for designing the test cases. In order to determine this, a testability review of the testing basis is carried out during this phase, which will provide insight into the testability of the system.
5. *Specification*: during this phase, the required tests and starting points are specified. The aim is to prepare as much as possible, in order to be able to run the test as quickly as possible when the developers deliver the test object.
6. *Execution*: to obtain insight into the quality of the test object through the execution of the agreed tests.
7. *Completion*: to learn from experience gained during this test and to preserve testing ware for reuse in a future test.

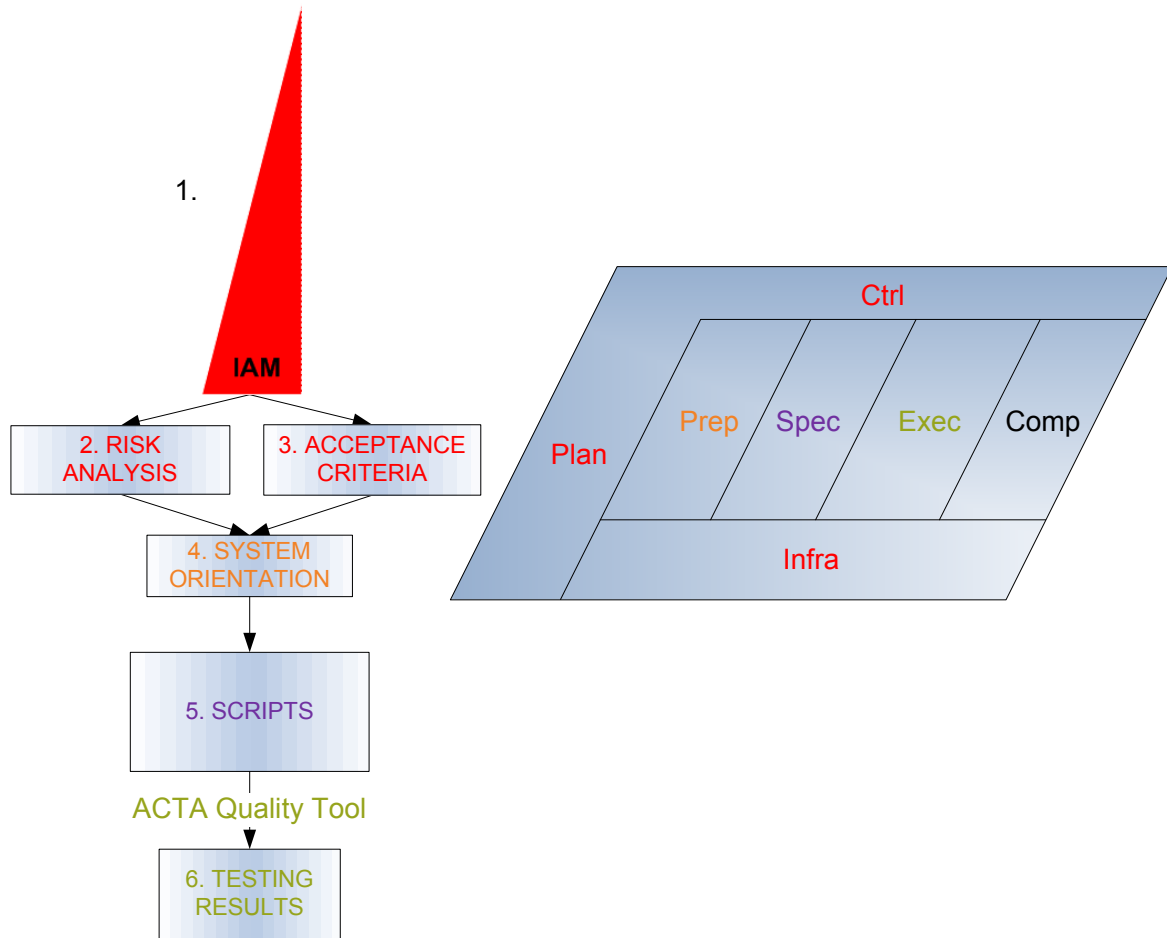


Figure 4: Mapping from CEPO's testing framework onto TMap

As stated before, CEPO has used parts of *TMap next* to build its own testing framework. Figure 4 illustrates the phases of CEPO's framework and their mapping onto *TMap next*, as in Table 1.

Table 1: TMap next vs. CEPO

TMap next	CEPO
Planning	Testing plan, Risk analysis (2) and Acceptance criteria (3)
Control	Testing plan, Risk analysis (2) and Acceptance criteria (3)
Infrastructure	Testing plan, Risk analysis (2) and Acceptance criteria (3)
Preparation	System orientation (4)
Specification	Testing scripts (5)
Execution	Pretesting, testing execution (between 5 and 6) and Final client report (6)
Completion	Save findings and testing scripts for retesting

Phase 1 from CEPO's framework is not present in *TMap next*, because DEMO modelling is an addition to CEPO. Today, CEPO only uses a part of DEMO [1] to model the client's organization. DEMO has been proven to be successful in modelling organizations [23]. DEMO is a powerful tool to identify all important transactions of an organization (as well as communication with external actors). CEPO saw opportunities to expand the use of DEMO models in its testing framework. The way in which DEMO could provide improvement was not clear to CEPO, therefore this study was set up.

DEMO supports people in understanding the essence of an organization. DEMO uses different conceptual models to present an organization. Important elements in these models are actor roles and transactions. Transactions between actor roles can be seen as the molecules of an organization. Once a transaction is completed, something new has been created, called a production fact. Different people in an organization can fulfil one or more actor roles, but one actor role is responsible for a single production fact. Transactions consist of process steps which can be seen as atoms, to continue the metaphor. The process steps *request*, *promise*, *execute*, *state* and *accept* together form the basis of a transaction. When a process exception occurs during a process step, it is called a cancellation for which patterns are defined.

DEMO encompasses four aspect models. The Construction Model contains the transactions and the associated actor roles, so by whom the transaction is initiated and which actor role is responsible for the production fact (the execution) of that transaction. The Process Model contains the order in which the transactions are started. The Action Model specifies conditions that must apply in order to continue to the next process step, and the State Model gives an overview of the production facts and the relationships between these facts. To make this methodology more tangible, refer to Chapter 3 for the test case models.

1.2 Research Goal

The following main hypothesis of this project was formulated:

ADDING THE PROCESS MODEL AND THE STATE MODEL FROM THE DEMO METHODOLOGY TO THE EXISTING TESTING FRAMEWORK (WHICH ONLY USES THE INTERACTION MODEL) CONSIDERABLY INCREASES THE DETECTION OF LACKING FUNCTIONALITY, ERRORS AND WEAKNESSES OF INFORMATION SYSTEMS

CEPO developed its own testing framework, which uses only the Interaction Model. The idea was to develop a new testing framework that would be capable of finding more errors and weaknesses of an information system while not taking much more time than the existing framework. DEMO encompasses two models that seemed suitable for this new framework: the Process Model and the State Model. The Transaction Process Diagram, which is a representation of a part of the Process Model, already goes into more detail than a basic transaction pattern so that it would probably be able to detect more problems. Thus, by applying the Transaction Process Diagram pattern it would be easier to check all possible trajectories within a transaction. The State Model is added, because this model seems promising to present objects that are created by the organization. This study includes one test case that will be used to demonstrate the benefits of adding the two new models. The new framework was tested on a number of criteria that were determined beforehand:

- Efficiency
 - Development time; the development time must be known to decide whether this framework is efficient. It is not desirable that the new framework takes much more time than the existing framework.
 - Walk-through time; if testing scripts take too much time to walk through, testers may become bored and distracted. In that case, the results would no longer be reliable.
 - Understandability; testers must be available to perform instructions with as little assistance as possible from the testing coordinator or consultant.

- Effectiveness
 - Lacking functionality, detection of errors and weaknesses; this is the final result that is delivered to the client.
 - Reproduction possibilities; in order to correct errors and weaknesses the software supplier should be able to reproduce these.

The following section describes the way in which this study was conducted and describes the structure of this report.

1.3 Research Approach

The approach used for this research is illustrated in Figure 5. It consists of three phases: the analysis, the scripting and the result phase. Every phase is subdivided into two to four sub-phases to construct a clear plan during the project and to get a clear overview of what was performed in these separate phases. First, an environment analysis at CEPO and at test case WSD was carried out that was used to construct the thesis proposal. The next sub-phase that belonged to the analysis was researching the CEPO framework [24-26]. The focus of this study was on the scripting phase, because this is the part where input from the researcher is required. The sub-phases are presented in Figure 5.

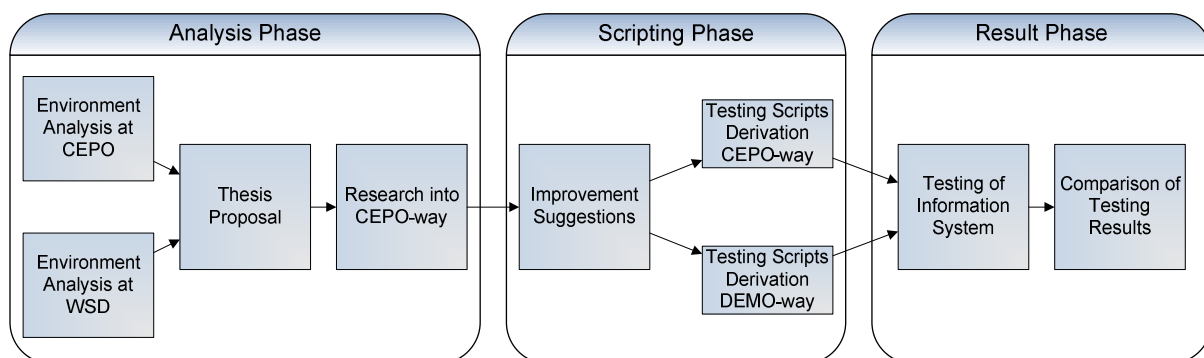


Figure 5: Overview of the research approach

Environment analysis at CEPO

The orientation at CEPO took place, recurrent appointments and an outline plan were made. It was determined when different resources like tools and employee's availabilities would be needed.

Environment analysis at Wilhelmina Senioren Diensten (test case WSD)

The orientation at WSD, which is the test case, took place, recurrent appointments and an outline plan were made. The employees involved at WSD were informed about the upcoming events so that WSD could make a plan for when resources would be needed.

Thesis proposal

The goal of this research was determined and the research approach was built. This was then confirmed by the author's supervisors.

Research into CEPO-way

The existing CEPO-way framework was investigated in order to understand the different phases and to increase the quality of the improvement suggestions.

Improvement suggestions

The crux of the studies lies in this phase. A proposal for a new framework was made which used two additional models from the DEMO methodology.

Testing scripts derivation CEPO-way

The testing scripts based on the CEPO-way were derived by an employee of CEPO. The CEPO scripts were not disclosed until the DEMO scripts had also been derived.

Testing scripts derivation DEMO-way

The testing scripts based on the DEMO-way were derived by the author.

Testing of information system

The actual testing which uses the testing scripts to walk through the information system was performed by the testers in this sub-phase.

Comparison of testing results

The testing results of both frameworks were compared against the criteria that had been determined at the outset.

1.3.1 Document Structure

The report has four distinct parts that are presented in Table 2.

Table 2: Document structure

Part	Contents
A: Introduction and Background	Position of this study + The approach
B: Testing Frameworks	Existing framework + Proposal for a new framework
C: Applying Testing Frameworks	Application of both frameworks to a test case
D: Conclusion	Comparison of testing results + Recommendations

1.3.2 Methodologies, Methods and Tools Used

Several tools, a method and a methodology were needed to improve the usability, reliability and correctness of the research conducted. A methodology is a set of tools, techniques, procedures and investigative methods, used to collect, store, analyze and present information [27]. DEMO was chosen as methodology for this research, because it seemed promising for use in the testing framework of CEPO due to its modelling possibilities. All DEMO models were built with a plug-in for MS Visio.

A method prescribes a way or how to do something which can be supported by a tool to perform this with less effort. The derivation of testing scripts is a method which can be supported by DEMO-way Script Creator. After trying several representations of a mapping from DEMO onto the testing scripts unsuccessfully, DEMO-way Script Creator was built by the author.

The registration during testing was supported by the ACTA Quality Tool that was created by CEPO.

The evaluation setup was based on a scientific control method [28] with which it was possible to isolate variable(s) that needed to be evaluated. The following paragraphs describe all these aids and explain their added value.

Summary of the DEMO methodology

DEMO effectively enhances understanding and managing the complexity of organizations. It reduces this complexity enormously in two ways, while maintaining solid connections with the actual realization and implementation. First, it provides a distinction between coordination and production acts, while clustering them into generic transactions and business processes. Second, it provides a distinction between three human abilities, which leads to distinguishing between three organizational aspects. The *forma* ability in coordination means being able to speak, write, listen and read (formative acts); regarding production, it means being able to perform *datalogical acts*, like storing, transmitting, and copying data or documents. The *informa* ability in coordination means being able to formulate and interpret thoughts (informative acts); regarding production, it means being able to perform *infological acts*, like reasoning and computing. The *performa* ability in coordination means being able to feel committed as performer or addressee of a coordination act; regarding production, it means being able to perform *ontological acts*, like deciding and judging.

The DEMO methodology consists of 4 aspect models: the Construction Model (CM), Process Model (PM), Action Model (AM) and State Model (SM), see Figure 6.

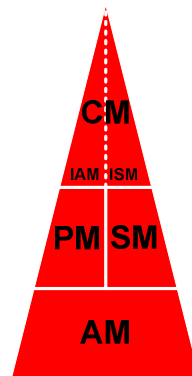


Figure 6: The DEMO methodology and its aspect models

The Construction Model specifies the identified transaction types and the associated actor roles, as well as the information links between the actor roles and the information banks (the collective name for production banks and coordination banks); in short, the Construction Model specifies the construction of an organization. This is the most concise ontological aspect model, comprising only a few sheets of paper for an average organization.

For every transaction type in the Construction Model, the Process Model contains the specific transaction pattern of the transaction type. The Process Model also contains the causal and conditional relationships between transactions. These relationships determine the possible trajectories in between transactions.

The Action Model specifies the action rules that serve as guidelines for the actors in dealing with their agenda. It contains one or more action rules for every process step: *request*, *promise*, *execute*, *state* and *accept*.

The State Model specifies production facts: the object classes and fact types, the result types, and the ontological coexistence rules.

The models collectively form the complete ontological model of an organization as in Figure 6. For a more detailed explanation refer to Appendix A DEMO description.

DEMO Add-in for MS Visio

MS Visio is widely used to draw diagrams. J. Muns, a former student in Information Architecture, implemented a DEMO plug-in for MS Visio during his thesis project [29]. After the installation of this tool it is easy to represent DEMO models in MS Visio. All DEMO models created in this report were created using this add-in.

DEMO-way Script Creator (DSC tool)

DSC tool is a tool that helps users to stay within the borders of a transaction process with the purpose of creating testing scripts. The author constructed this tool to improve the overview of transaction process diagrams. Every field is meant for one testing instruction.

Figure 7: Screenshot of DEMO-way Script Creator

If one enters a transaction into the tool, it also provides the option to fill in process exceptions. For every step from the standard transaction pattern it is possible to insert objects from the State Model. They can be used during the construction of a testing instruction. More information about the DEMO-way derivation of scripts is provided in 3.8.4.

ACTA Quality Tool (AQT)

AQT is a testing registration tool that was developed in-house by CEPO. It serves to save results of the testers.

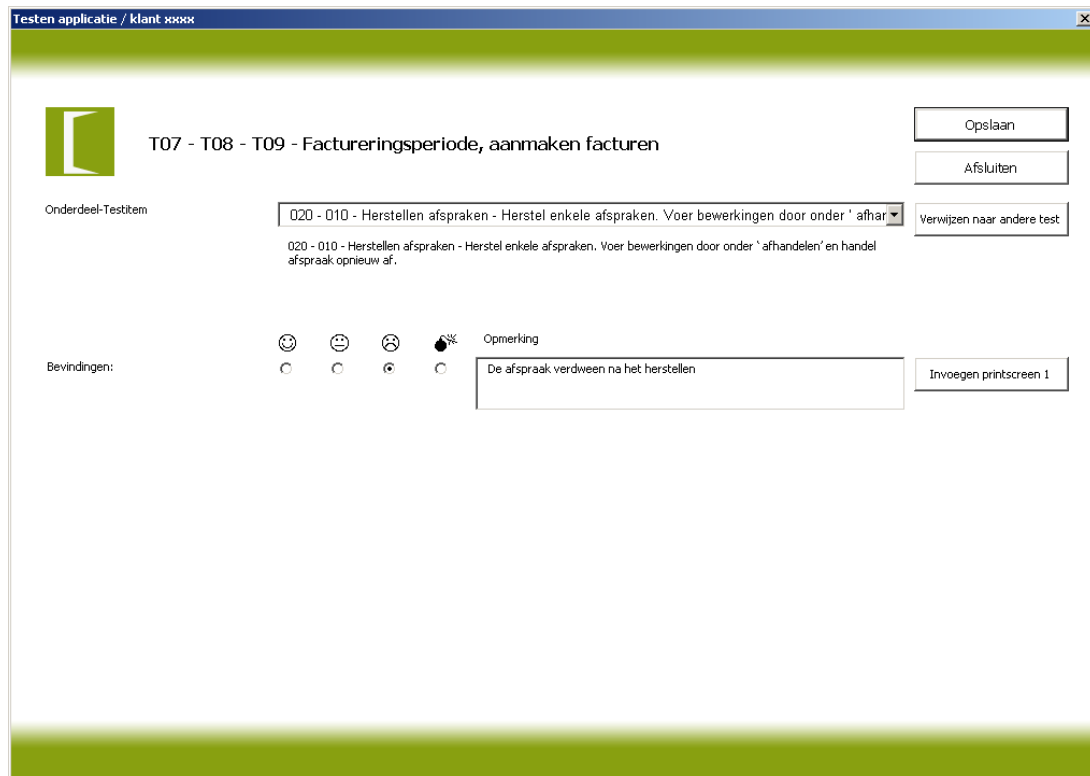


Figure 8: Screenshot of AQT

This tool is capable of presenting a testing instruction that needs to be followed and of collecting and storing the tester's feedback. AQT also provides the functionality to insert screenshots of the application that is tested. The goal of this tool is to create a report of the most important findings for the software supplier. The captured screenshots are reported, so that it is immediately clear what finding is given in which screen. When the expert testers have finished their testing scripts, AQT has another useful functionality: it is then able to generate a report for the testing consultant. This report is then used to build the final report for the client.

Evaluation Setup

The goal of this study was to determine which framework is best suited to the testing of CEPO. To improve the reliability of such a scientific experiment, two groups were made: one group that started with the CEPO script and ended with the DEMO script and another group that worked vice versa. This approach isolated the variables that are the foundation of this study [28]. Both groups were to work on the same computer system (e.g. speed and memory size). To prevent or keep the same level of disturbance or distraction during testing, the testing room was the same as well. The groups were composed by the CEO of the client company as in Table 3, one group with future regular users and one with a process expert. The process expert has knowledge of all business processes (a business process is defined as a collection of interrelated tasks that solve a particular issue [30]) within the client company and is able to provide constructive feedback during modelling. The testers needed to

have knowledge of the organization and were the ones who would become regular users of the information system (if it is accepted, of course).

Table 3: Group composition and schedule

Group	Day 1: 28-04-2008 (8:00-17:00)	Day 2: 02-05-2008 (9:00-16:00)
A: Future regular users	CEPO scripts	DEMO scripts
B: Process expert	DEMO scripts	CEPO scripts

The schedule and instruction for the testing day is set down in Appendix B Testing Plan of Action.

1.4 Organizational Structure

The test case presented in this section serves as a basis to prove that extra models from the DEMO methodology are appropriate for a testing framework.

Wilhelmina Senioren Diensten as a Test Case

WSD (Wilhelmina Senioren Diensten [31]) is a company that was founded in 2002. The company mediates between freelancers in the care sector and clients who apply for a “persoonsgebonden budget” [32]. WSD’s main tasks are to extract demands from clients, to schedule freelancers and care and to perform administration for clients and freelancers.

Situation As Is

WSD decided to start working with a standard software package that is called Care Arrangement, developed by Company X. This company already had several years of experience in developing software for care institutes. Company X claimed that all the essential business processes were covered by its software package. The names for the software supplier and its package are fictitious to make this study anonymous in order to prevent any possible harm to external parties.

This study used several business processes from this test case to evaluate the claim of Company X. The processes were tested with the CEPO-way and the DEMO-way. Figure 9 presents an overview of the parties involved in this project.

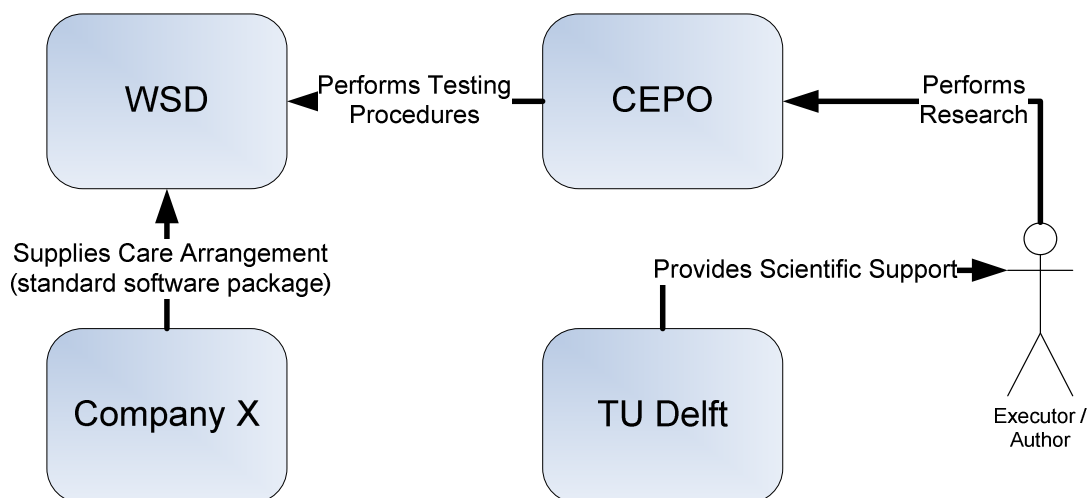


Figure 9: Overview of connections between organizations

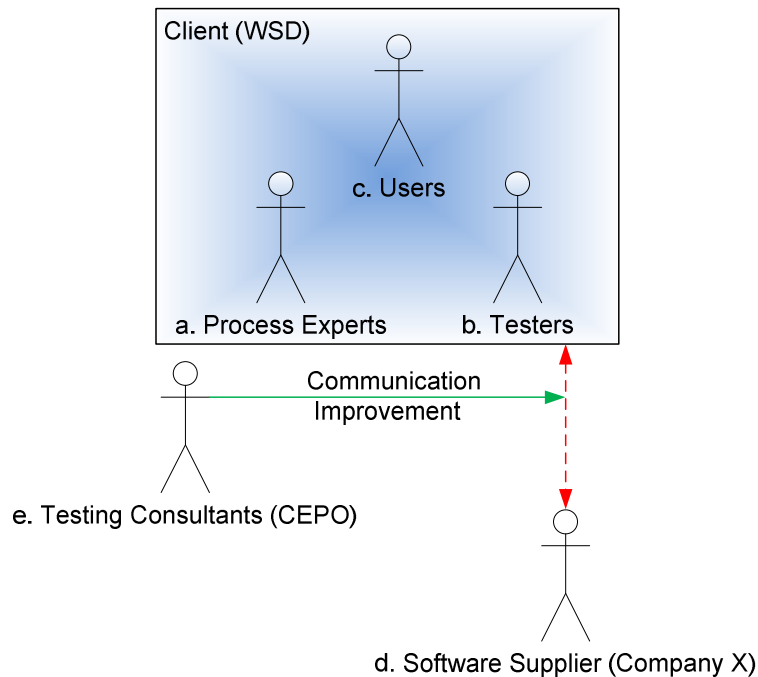


Figure 10: Stakeholders overview

For a description of all stakeholders, refer to Table 4.

Table 4: Stakeholder descriptions

Stakeholder	Description
a. Process experts	The process expert has knowledge of all business processes within the client company and is able to provide constructive feedback during modelling.
b. Testers	A tester is someone who will perform the actual acceptance tests. The goal is to let the tester find and register the shortcomings in the information system.
c. Users	The user is or will be using the information system in the future.
d. Software Supplier	The software supplier has built the software and is able to make corrections if shortcomings are detected.
e. Testing consultants	The testing consultants of CEPO are capable of leading all involved stakeholders through the process.

Company X could have been suspicious about the fact that a professional testing company was going to test Care Arrangement. To prevent possible tensions between the stakeholders depicted in Figure 10, an expectation matrix [33] is given in Table 5. An expectation matrix is capable of providing an overview of the range of all expectations, from one stakeholder to another.

The expectation matrix that is given below is the one that must be taken into account during the testing process.

Table 5: Expectation matrix, an explanation of the numbers can be found below

<i>From / To</i>	a. Process Experts	b. Testers	c. Users	d. Software Supplier	e. Testing Consultants
a. Process Experts		To carry out testing scripts	N.A.	To support processes with the information system	To detect existing gaps
b. Testers	N.A.		N.A.	To support processes with the information system	To deliver understandable testing scripts
c. Users	To support the processes with the information system	To represent the user		To support the processes with the information system	To detect existing gaps
d. Software Supplier	To deliver descriptions of the processes (1)	N.A.	To know various workarounds to make the information system work well (2)		To come up with blind spots
e. Testing Consultants	To deliver high-quality descriptions of the processes (1)	To carry out testing scripts	N.A.	To accept intervention (3)	

Several of the expectations need some clarification of their description and corresponding importance.

(1)

The expectation from the software supplier and testing consultants to the process experts differs only in the adjective 'high-quality'. This is done because CEPO thinks it is of the utmost importance that the processes are described correctly when the software supplier just wants to sell its software. This is the starting point where the communication between software supplier and client is lacking.

(2)

The software supplier built the software, so they will know various workarounds to make the information system work well or more efficiently. Nonetheless, this is often not communicated to the client and the client is therefore inclined to state that this information system is not user friendly or that it does not work properly.

(3)

If testing consultants from a professional company start testing the software, it is good to inform, and possibly involve, the software supplier as well. This is done to prevent a defensive attitude on the part of the software supplier and turn it into a cooperative attitude. Criticism of the software will then be seen as constructive.

The issues that are mentioned above give an insight into the complexity of a testing process. Communication improvement starts with informing all stakeholders about these expectations, so that unnecessary tensions can be prevented.

Part B Testing Frameworks

2 Testing Frameworks

The following chapter sets out the way of working using the testing frameworks. Two types of frameworks are distinguished: the CEPO-way and the DEMO-way. The CEPO-way is a framework developed in-house by CEPO. The DEMO-way is characterized by its intensive DEMO usage. However, several phases intersect. The overlap of phases and several differences are described in the following sections. Note that these descriptions are high-level and introductory, Chapter 3 provides more detail of every phase because then a test case will be used to demonstrate these phases.

2.1 Existing CEPO-way vs. New DEMO-way

CEPO is specialized in testing, therefore CEPO has made its own framework for that purpose. This chapter will set out the way CEPO's framework supports testing and a proposal for a new testing framework will be made.

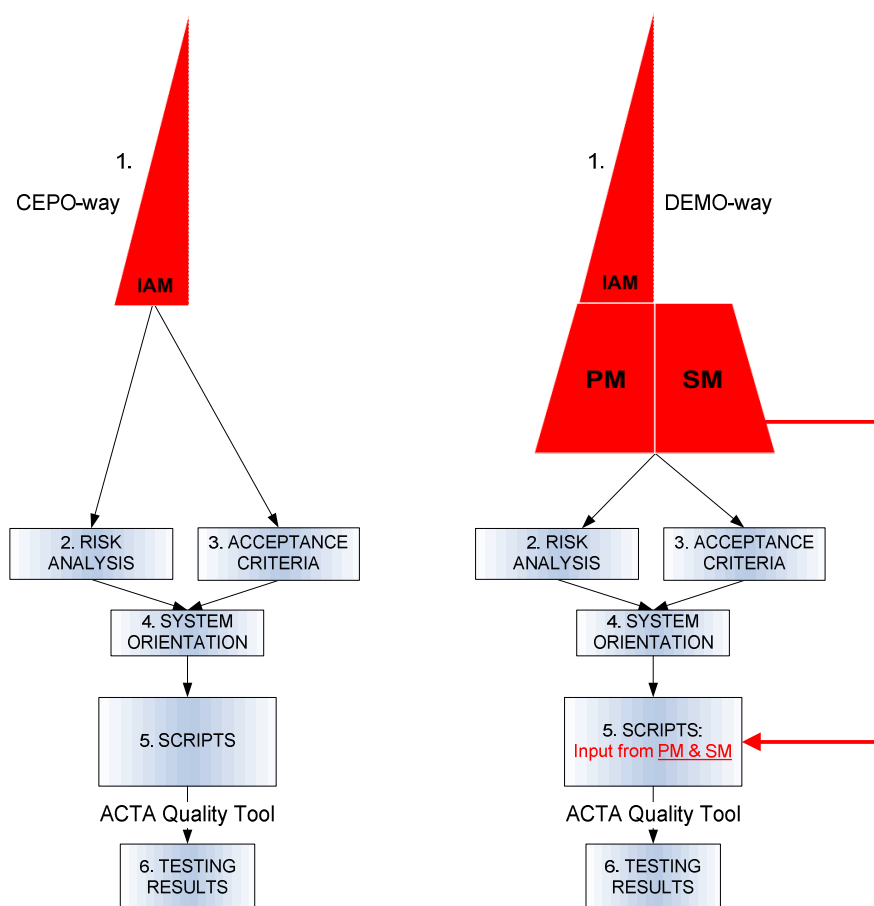


Figure 11: Both testing frameworks illustrated

Note that steps 2. Risk analysis, 3. Acceptance criteria and 4. System orientation are equal in both frameworks. The main differences are step 1. *Modelling*, because additional DEMO models are built, and step 5. *Scripts*, in which the Process Model and State Model are also used as input to derive testing scripts.

2.2 Existing CEPO-way Framework

In order to get a quick overview of the testing frameworks, the added value of every step is briefly described in Table 6 and a description of every testing step is provided.

Table 6: Enumeration of added value per testing step in the CEPO-way

Testing Step	Added Value
1. Interaction Model	This model provides a quick overview of essential business processes.
2. Risk analysis	Possibility to focus on important business processes.
3. Determination of acceptance criteria	The determination of these criteria is necessary because then a well thought out decision can be made after the evaluation of the testing results.
4. System orientation	Testing instructions that give clear directions in the information system support navigation which saves a significant amount of time during testing. It also gives more insight into the capabilities of the information system.
5. Testing scripts	Testing scripts lead the tester through the testing.
6. Testing results	The results are sent to the client and an overview of the shortcomings or values of the information system is given.

1. Interaction Model of an Organization

The Interaction Model is part of the DEMO methodology. In fact, this is the first phase of the complete modelling procedure. Only this model is created in the CEPO-way, an important difference between the two testing frameworks.

2. Risk Analysis

Risk analysis is an important testing step, because this step identifies the business processes that are more important and need more focus during the derivation of testing scripts. The risk analysis is based on groups of transactions that are found during the modelling of the Interaction Model.

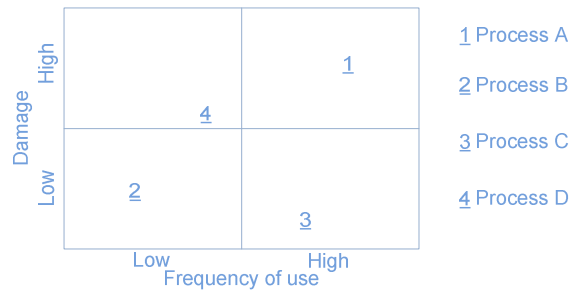


Figure 12: Example of risk analysis

Example of risk Process A:

Monthly (frequency of use) Financial operations should be performed correctly: WSD needs income from and the trust of its clients to prolong its existence (damage) and cannot allow more money than necessary to be paid by its clients.

3. Acceptance Criteria

The acceptance criteria on which the information system is tested are determined in cooperation with the client. This step ultimately provides a feedback mechanism because the norms that are determined here are used in the end to support the decision whether to continue working with the information system or not.

Acceptance criteria to be applied

The ensuing paragraphs describe the way criteria are found that will be used for the acceptance of an information system. They are determined beforehand to create a feedback mechanism to evaluate the decision of whether the organization should continue using the information system or not. During this criteria determination process several actions have to be carried out. It is important to describe what an acceptance criterion is, who will accept the information system, how it is accepted and which approach must be followed for determining these criteria.

Definition

Within CEPO, acceptance criteria are defined as:

“Performance requirements and essential conditions that have to apply in order to get the object to test accepted by the involved stakeholders.”

This definition is the one that will be used throughout this report.

Who accepts

At the start of acceptance testing it is necessary to assign a decision group of stakeholders from within the client company that will make the final decision mainly based on the testing results produced.

How it is accepted

The decision group receives an overview of testing results from the testing consultants. The evaluation report is based on findings that use the ordinal scale provided in Table 7.

Table 7: Symbols used for reporting findings

Symbol	Term	Definition
	Blocking	Process is blocked
	Bad	Interruption of a process, but the process can continue in another way
	Average	Finding that does not disturb the process, but is desirable to change (mainly cosmetic issues)
	Good	According to expectation

This small scale is used to sketch a quick overview of the testing results and to make it easier to understand for people without expertise in the subject matter.

Determination approach

In order to identify the criteria, four distinct steps were established by CEPO:

- a. Drawing up of draft report to send to the decision group; this is a standard list of criteria.
- b. Discussion within the decision group to add or remove criteria; this is done without CEPO involvement to make use of the naïve approach which means that almost no direction is given before discussion.
- c. Determination of acceptable testing results (norm); e.g. the amount of negative findings allowed per symbol.
- d. Assessment of testing scripts; CEPO and the client must be sure that all criteria are tested in the testing scripts that will be used.

Note that the testing scripts are derived between step *c* and *d*, cf. step 5.

4. System Orientation

In this step, the documentation of the information system is explored (e.g. user manual and functional specification). This step gives insight into the requirements and capabilities of the information system.

5. Derivation of Testing Scripts

Testing scripts are derived to be given to the expert tester. The tester uses these scripts to fill in information in the AQT as described in 1.3.2. Testing scripts consist of testing instructions.

Example of a testing instruction:

“Check via module Client-Registration whether a client already exists in the system based on postal code, phone number or address.”

6. Testing Results

The final deliverable for the client is built. This report can then be used in discussions with software supplier and the client company.

2.3 New Framework DEMO-way

The testing frameworks, DEMO and CEPO, both consist of six main steps, and steps 2, 3 and 4 are similar. The focus of this study is on the first and fifth phase of these frameworks. This section describes the motivation for the use of extra DEMO models and presents the framework with extra models. The DEMO models are based on DEMO 3.0 [34].

The CEPO-way only uses the interaction model and the goal of this study was to prove that one of the two frameworks can formulate better conclusions based on the criteria that were given in section 1.2. The decision that had to be made was which models were to be used for the DEMO-way. Several options were available, but first some characteristics of every aspect model within this type of study had to be clarified.

Table 8: Overview of the characteristics of DEMO models

Model	Characteristics
Construction Model	• This is the most concise ontological model • Suitable for strategic alignment discussions • Provides an organizational function overview • Represents the whole enterprise
Process Model	• Shows the dependencies of processes and all possible trajectories • Useful for requirements engineering • Suitable starting point for use cases
Action Model	• Defines all action rules for every actor role • Time-consuming compared to other models
State Model	• Serves as a data model • Is able to show which object classes may exist during the process • Defines data ownership

Motivation for Adding the Process Model

In order to apply a testing framework well and maintain the quality level that CEPO provides, all business processes that need to be tested are modelled beforehand. Otherwise, testing whether IT supports business processes makes little sense. Since the Process Model is capable of showing the interdependencies between transactions and business processes, this is precisely the type of model to use for a testing framework that performs acceptance testing on process level. The Process Model also describes the cancellation patterns. These patterns are capable of describing process exceptions. Process exceptions oblige the person who derives the testing scripts to use more functionalities of the information system in the script. Taking these two arguments taken into account, the Process Model is used to derive testing scripts.

Motivation for Adding the State Model without the Action Model

Before a State Model was made all the transaction result types were written down as object classes; this is the link between the Interaction Model and the State Model. After this step, a part of the object classes is linked to others by drawing relationships between these classes. Sometimes other object classes must be added to make this model complete. The Action Model is not constructed and

therefore the State Model cannot be complete. Although missing information in the State Model is not desirable, a small or medium-sized enterprise like WSD does not have the budget to have the whole enterprise modelled in DEMO by a consultant. Especially not the Action Model, which compared to other DEMO models, consumes a lot more time. During conversations with the client, enough information was obtained to construct a simple, small Action Model that did not have to be written down on paper. The State Model was then constructed with the 'simple and small' Action Model and was ready to use for the derivation of testing scripts because it contained all the important object classes and gave a clear overview of the existential laws. The State Model is also able to show which actor role made which object class and from which this object class is. This means the model is able to determine which actor role is responsible for the consistence of these object classes.

Motivation for Leaving out the Interstriction Model

The Interstriction Model contains the locations of different kinds of information that are necessary for an organization. Since the availability of information is more important than where the information comes from, this model was not created. In addition, making more models than just the Process Model and State Model would again take more time, which would negatively influence the development time criteria that were considered when the models were chosen.

Table 9: Enumeration of added value per testing step in DEMO-way

Testing Step	Added Value
1. Interaction Model	This model provides a quick overview of essential business processes.
a. Process Model	This model shows the interdependencies between transactions and the possible trajectories within a transaction.
b. State Model	This model illustrates all important object classes, the relations between them and the existential laws.
2. Risk analysis	Possibility to focus on important system aspects.
3. Determination of acceptance criteria	The determination of these criteria is necessary because a well thought out decision can be made after the evaluation of the testing results.
4. System orientation	Testing instructions that give clear directions in the information system support navigation which saves a significant amount of time during testing. It also gives more insight into the capabilities of the information system.
5. Testing scripts	Testing scripts lead the tester through the testing.
6. Testing results	The results are sent to the client and an overview of the shortcomings or values of the information system is given.

Steps 1, 1.a and 1.b are executed in sections 3.1, 3.2 and 3.3 respectively (DEMO models) and step 5 (Derivation of DEMO Testing Scripts) in section 3.8.4.

Part C Practice of Testing

3 Applying a Testing Framework

The ensuing sections describe the approach for achieving final testing results through both frameworks. First, the Care Arrangement case is described and then both testing frameworks are applied. Although this is one test case which does not provide enough data to validate the new framework on all organizations and information systems, this study was set up in a transparent way so that it can be reproduced completely. Analyzing WSD and the CEPO framework already took several months, so evaluating more than one test case would take considerably more time than was reserved for this project. If these frameworks had been compared on another (larger) information system, the relative results would most likely have been more or less the same due to the theoretical foundation in section 2.3 (unless an ideal information system were to exist, in which case no errors and / or weakness would be found).

Figure 13: Screenshot of Care Arrangement

The Case Care Arrangement

Care Arrangement (screenshot in Figure 13) is a standard software package that is used by a number of companies. These are companies that provide care at people's homes and do their administration. WSD performs the same kind of tasks, although it works with a different kind of financial administration, namely PGB instead of 'care in kind'. With PGB, people can buy their own care without being bound by a single organization's availability or specialization. One important difference

is that WSD uses freelancers instead of contracted employees. This is the main difference between the financial administrations. This care concept is relatively new in the Netherlands and is applied nationwide. The strategy of WSD is to start many of these franchise organizations [35]. All these activities should be supported by an information system to keep track of these franchise organizations. But first the essential business processes of a single organization have to be supported by this information system. At the moment, WSD can be positioned in this phase. Now that WSD and Care Arrangement have been presented briefly, phase one can be started: the DEMO modelling.

3.1 Interaction Model

The Interaction Model is expressed in an Actor Transaction Diagram (ATD) and a Transaction Result Table (TRT) and is the most concise ontological model of the methodology.

Actor Transaction Diagram (ATD)

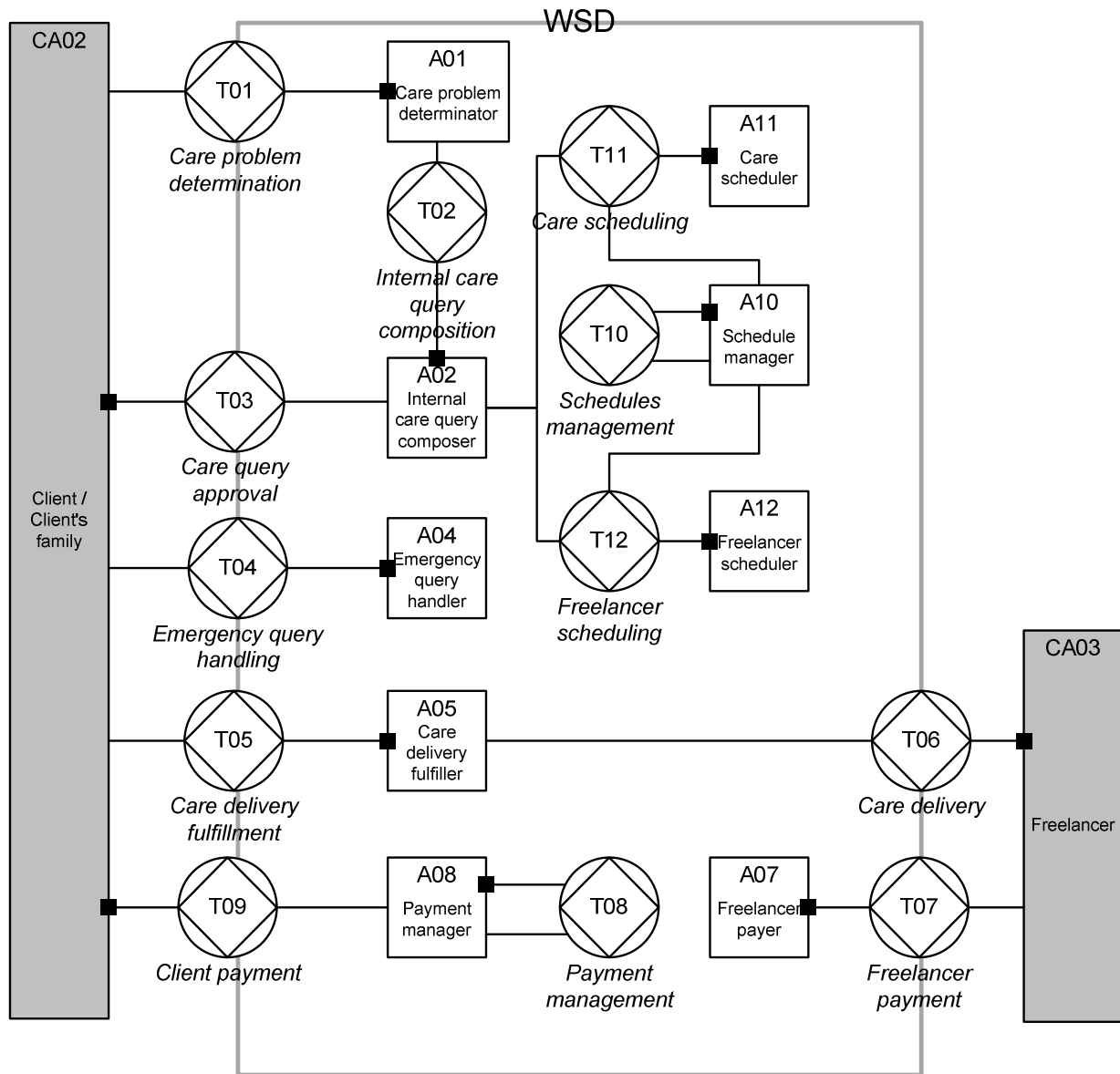


Figure 14: WSD expressed in an Actor Transaction Diagram

Example:

T01 is initiated by CA02 – Client / Client's family and executed by the actor role A01 – Care problem determinator. If this transaction is completed successfully, the care problem of the client has been determined.

For all the transaction results, refer to the Transaction Result Table in Table 10.

Transaction Result Table (TRT)

Every transaction is enumerated below and its corresponding result type, the production fact that is created after the completion of a transaction.

Table 10: Transaction Result Table from ATD of WSD

T#	Transaction name	R#	Result type formulation
T01	Care problem determination	R01	Care problem CP has been determined
T02	Internal care query composition	R02	Care query CQ has been composed
T03	Care query approval	R03	Care query CQ has been approved
T04	Emergency query handling	R04	Emergency query ECQ has been fulfilled
T05	Care delivering fulfilment	R05	Care query CQ has been fulfilled
T06	Care delivery	R06	Care C has been delivered
T07	Freelancer payment	R07	Freelancer payment FPAY has been done
T08	Payment management	R08	Payment management for period P has been done
T09	Client payment	R09	Client payment CPAY has been paid
T10	Schedules management	R10	Schedule management for period P has been done
T11	Care scheduling	R11	Care C has been scheduled
T12	Freelancer scheduling	R12	Care assignment CA has been done

The result types from the Transaction Result Table can be found in the State Model in which they are all connected to an object class.

The CEPO-way is based on these two models only. The next two aspect models are the ones that provide the DEMO-way with additional information.

3.2 Process Model

The Process Model is expressed in the Business Process Diagram (BPD) and Transaction Process Diagrams (TPDs). All Transaction Process Diagrams can be found in Appendix C Transaction Process Diagrams.

Business Process Diagram (BPD)

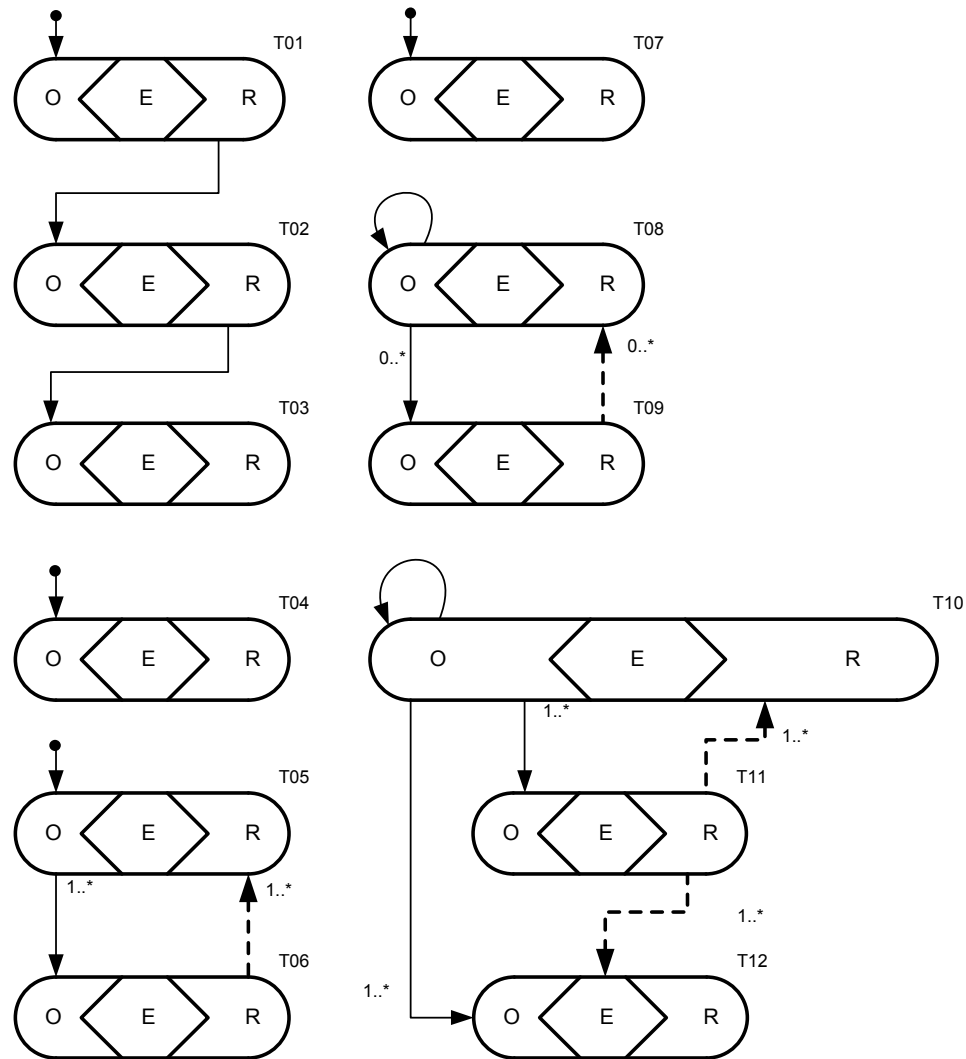


Figure 15: WSD Business Process Diagram

Three abbreviations are used within these models: O for Order, E for Execution and R for Result.

Example:

Transaction T02 can only start once transaction T01 has ended.

The Business Process Diagram is capable of showing the interdependencies between the transactions. This makes it possible to paste transactions behind one another when scripting. The

Transaction Process Diagrams in the appendix show the possible trajectories in the cancellation patterns that can be inserted into DEMO-way Script Creator.

3.3 State Model

The State Model is expressed in three models: the Object Fact Diagram (OFD), the Object Property List (OPL) and the Time Object Property List (Time-OPL). The latter was constructed by the author and is not part of the DEMO methodology. Usually, these attributes have already been defined for every object class. This is caught by the DEMO theory that all production facts have their creation time. In this case, it was necessary to insert these time-attributes into DEMO-way Script Creator because time is an important property to be tested for an object class in an information system.

Object Property List (OPL)

Table 11: OPL of WSD

property type	object class	scale
ecq_level	EMERGENCY CARE QUERY	DIPLOMA LEVEL
CIZ_indication_amount	CARE PROBLEM	EURO
CIZ_indication_start_date	CARE PROBLEM	JULIAN DATE
CIZ_indication_end_date	CARE PROBLEM	JULIAN DATE
problem_level	CARE QUERY	DIPLOMA LEVEL
care_query_duration	CARE QUERY	#MONTHS
care_query_hour_tariff	CARE QUERY	EURO
care_duration	CARE	#HOURS
sum_client_payment	CLIENT PAYMENT	EURO
freelancer_level	FREELANCER	DIPLOMA LEVEL
sum_freelancer_payment	FREELANCER PAYMENT	EURO

N.B. DIPLOMA LEVEL = (A,B,C,D)

Time-Object Property List (Time-OPL)

Table 12: Time-OPL of WSD

property type	object class	scale
ecq_start_time	EMERGENCY CARE QUERY	JULIAN DATE
ecq_end_time	EMERGENCY CARE QUERY	JULIAN DATE
care_start_time	CARE	JULIAN DATE
care_end_time (*)	CARE	JULIAN DATE
care_query_start_date	CARE QUERY	JULIAN DATE
care_query_end_date (*)	CARE QUERY	JULIAN DATE
date_client_payment	CLIENT PAYMENT	JULIAN DATE
date_freelancer_payment	FREELANCER PAYMENT	JULIAN DATE

Object Fact Diagram (OFD)

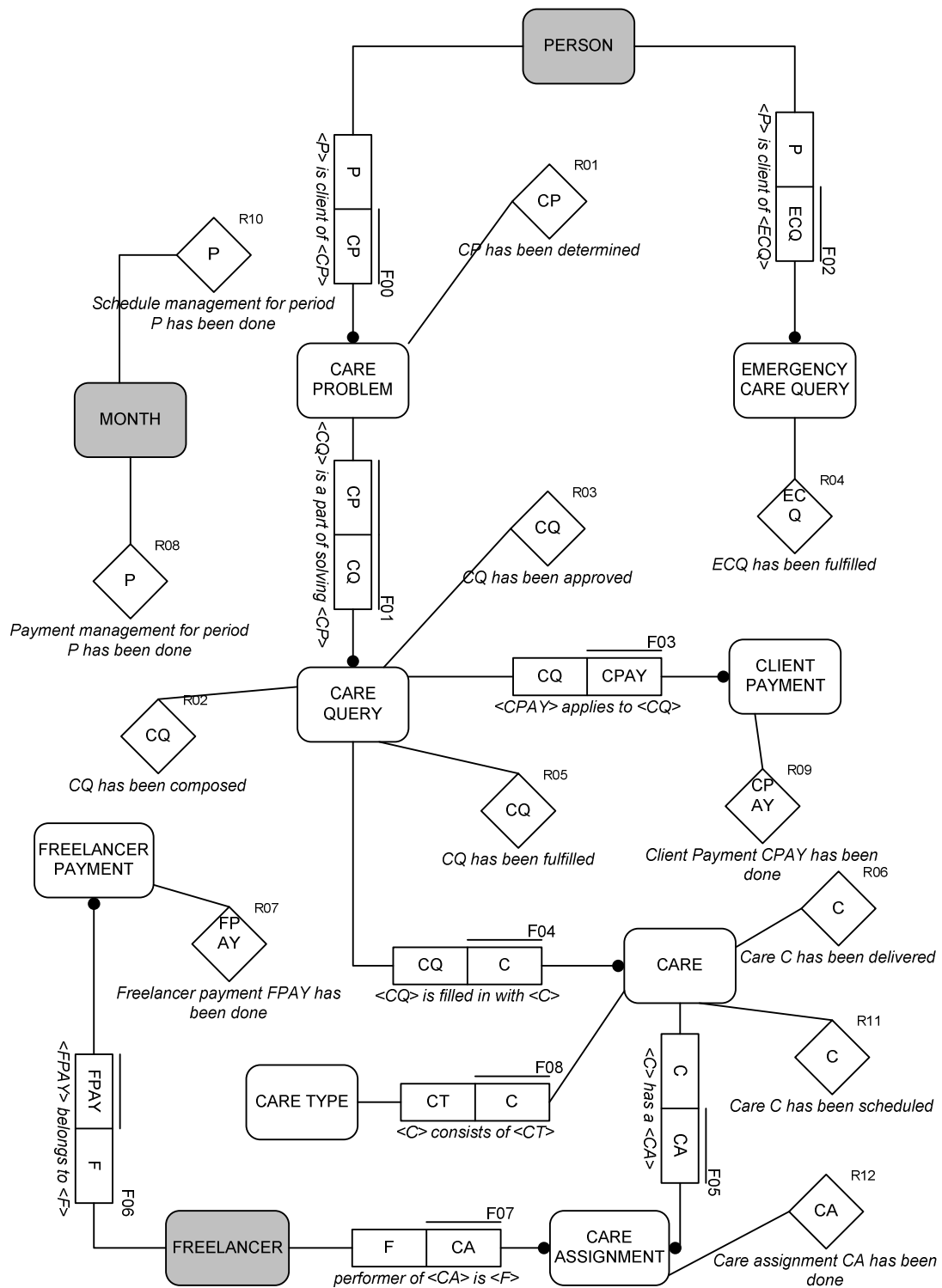


Figure 16: State Model (OFD)

Information Use Table (IUT)

An additional model is provided, which is called the Information Use Table (IUT). This model is the link between the Process and the State Model and contains the coupling of an object class (or fact type) to a process step(s).

Table 13: IUT of WSD

object class, fact type, or result type	process steps
PERSON	T01/rq T04/rq
person P is client of care problem CP	T01/rq
CARE PROBLEM	T01/ex
EMERGENCY CARE QUERY	T04/rq
person P is client of care query CQ	T04/rq
CARE QUERY	T02/ex T03/rq T05/rq
care query CQ is part of solving care problem CP	T02/ex
care query CQ is filled in with care C	T11/rq
CLIENT PAYMENT	T09/rq
client payment P applies to care query CQ	T09/ac
CARE	T06/rq T11/ac
care C has care assignment CA	T11/ac
CARE ASSIGNMENT	T12/ex
performer of care assignment CA is freelancer F	T12/ac
FREELANCER	T06/rq
FREELANCER PAYMENT	T07/rq
freelancer payment P belongs to freelancer F	T07/rq
MONTH	T08/rq T10/rq

3.4 Transaction Groups and Business Processes

In this chapter, the transactions are grouped so that they become separate business processes. These separate business processes are used later in this report to evaluate the results. All transactions in Table 14 are obtained from Figure 14 (Interaction Model). Transactions 5 and 6 are left out because these transactions are not directly supported by an information system. Although Object Class Removal is not a transaction, it is important that an information system supports the functionality to remove objects, like the deletion of a client.

Table 14: Business Processes / Groups of Transactions

Transaction number	Transaction Group / Business Process	Variable
T01(X) T02(X) T03(X)	<i>Client Registration</i>	X = Client
T11(X)	<i>Scheduling Care</i>	
T12(F1)+T12(F2)+T12(F3)	<i>Scheduling Freelancers</i>	F1 = Freelancer 1 F2 = Freelancer 2 F3 = Freelancer 3
T04(X)	<i>Emergency Care Query</i>	
T07(F1)+T07(F2)+T07(F3) T08(P) T09(X)	<i>Financial Transactions</i>	P = Period (Month)
Object Class removal	<i>Termination of Objects</i>	

3.5 Risk Analysis

The risk analysis will show the most important business processes in a chart based on two axes: the frequency of use and the damage if this process fails. The processes from Table 14 will be mapped on these axes determined by the client company, so that one can see which processes need to be tested more thoroughly. The chart finally consists of the risk analysis as shown in Figure 17.

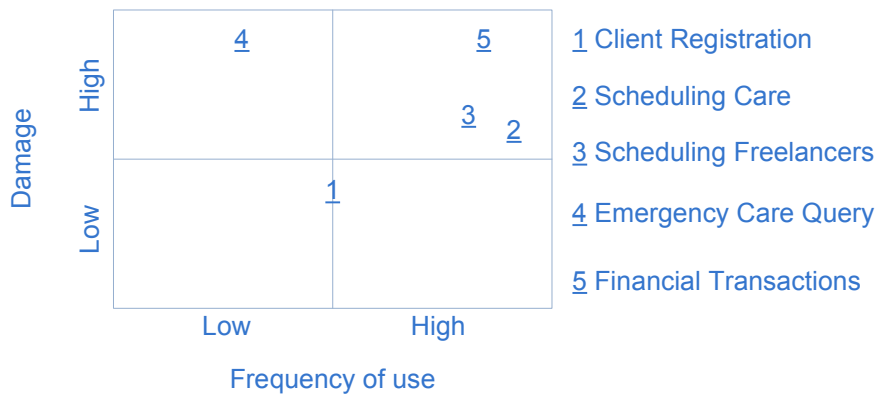


Figure 17: Risk analysis of WSD

The client, WSD, marked three processes as being most relevant: 2. *Scheduling Care*, 3. *Scheduling Freelancers* and 5. *Financial Transactions*. Therefore these processes are described in more detail below.

3.5.1 Scheduling Care and Freelancers

WSD is an organization that deals with and manages the lives of human beings. Therefore people are dependent on the schedule of care and its available freelancers. When care scheduling is not performed in the right way clients become unsatisfied, or worse, are left alone for several days while they need care twenty-four hours per day. This process is divided into two sub-processes: a standard care query and an emergency care query, because they consist of separate transactions and require different actions.

Standard Care Query Process

Table 15: Standard care query process mapped to transactions

Activity	Transactions
1. Care query is constructed beforehand in cooperation with the client and/or his/her family.	T01 + T02 + T03
2. The client's file is opened.	T01 + T02 + T03
3. The client schedule is opened and the care is scheduled.	T11
4. The freelancer schedule is opened and the freelancer is bound to care.	T12
5. Freelancers are now bound to the care queries.	T12

3.5.2 Payment Management

Payment management is one of the most relevant business processes, it deals with the financial flows within WSD. Clearly, income means continuity of existence.

Table 16: Payment management process mapped to transactions

Activity	Transactions
1. Monthly payment period has elapsed.	T08
2. Work sheets constructed by freelancers are collected.	T07 + T09
3. Confirmation takes place as to whether the amount of care corresponds to the care query.	T07 + T09
4. Every hour of work per client is processed.	T07 + T09
5. Three stacks of invoices are generated: one for WSD, one for the client and one for the freelancers.	T07 + T09
6. The invoices are printed and the so-called collection invoice is sent to the client. The client receives a copy of this invoice with the mediation fee for WSD.	T09
7. The freelancers also receive their invoices and are paid.	T07
8. If clients do not pay their invoice, a call is made or a letter is sent.	T09

This process is divided into two diagrams, the financial flows in Figure 18 and the document flows in Figure 19. Pars Pro Toto in both figures is an administrative organization that has been placed between the other actors because WSD is not allowed to have a tax relationship with its freelancers.

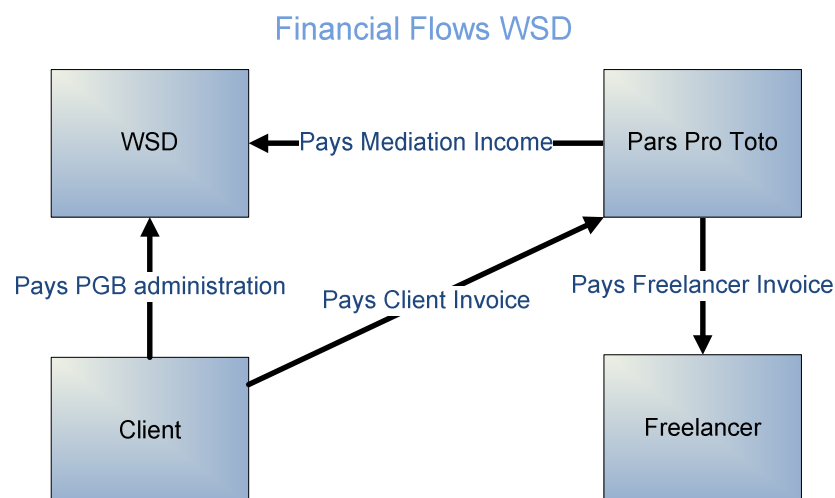


Figure 18: Financial flows of WSD

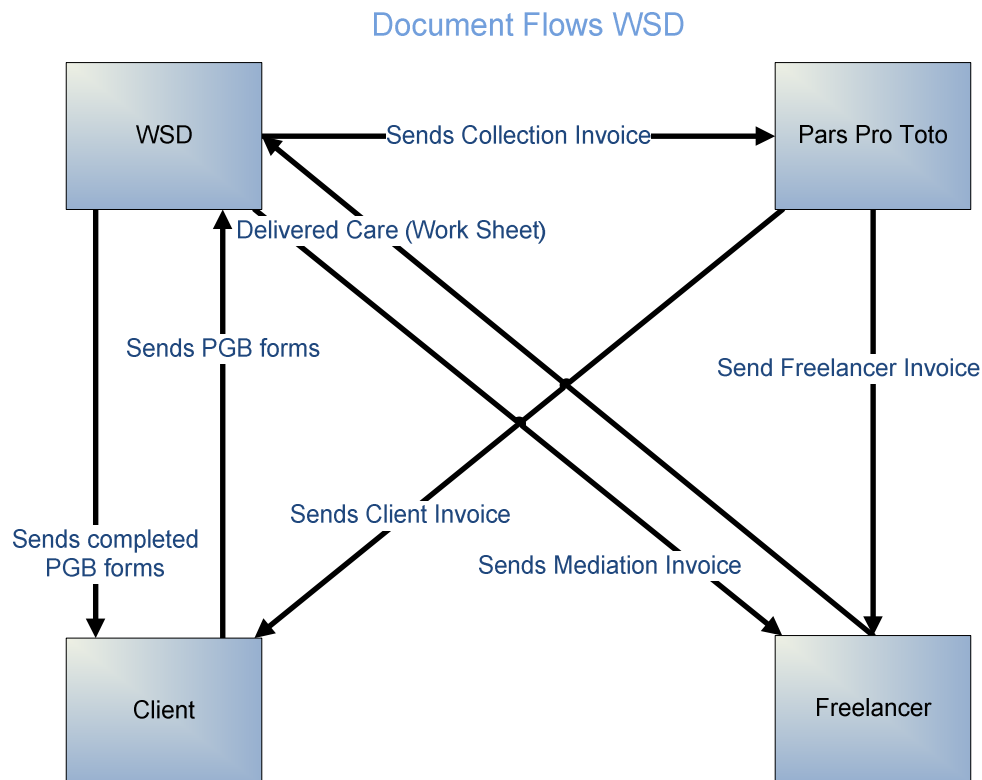


Figure 19: Document flows of WSD

From the figures above it can be concluded that the financial flows significantly increase in complexity due to the fact that freelancers are being handled in a different way to standard employees. Nonetheless, Company X still insisted on total coverage of these business processes from Care Arrangement.

3.6 Acceptance Criteria

The acceptance criteria are based on the experiences of CEPO, the ISO 9126 [36] standard and a book about testing [37] and they are categorized in Table 17. Finally, in discussion with client WSD these criteria were set down as the acceptance criteria.

Table 17: Acceptance criteria enumerated

Criterion	Description of criterion
Stability	The application did not cancel itself or crashed during the testing period as a result of a regular user action and the information is consistent through the system.
Understandability	Buttons, texts, and signals in the application are easy to understand and recognizable for internal and external users. They understand the consequences of a every action.
User Friendliness	Every possible action needs to be completed with a maximum of 8 mouse clicks or other actions and the consecutive actions are intuitive to perform.
Financial Functionality	The level of certainty that financial data is processed correctly and completely and conforms to the system's documentation.
Operational Functionality	The level of certainty that operational data is processed correctly and completely and conforms to the system's documentation.
Traceability	Every change made is traceable by the user and an undo option is available most of the time.

There is one more question that need to be answered with the aim of making a well thought out decision regarding the future of Care Arrangement and WSD.

Who accepts Care Arrangement?

For WSD, the decision was made by employees, who were to become regular users and the process expert. Together they constituted the decision group (cf. Figure 20 for their position within WSD).

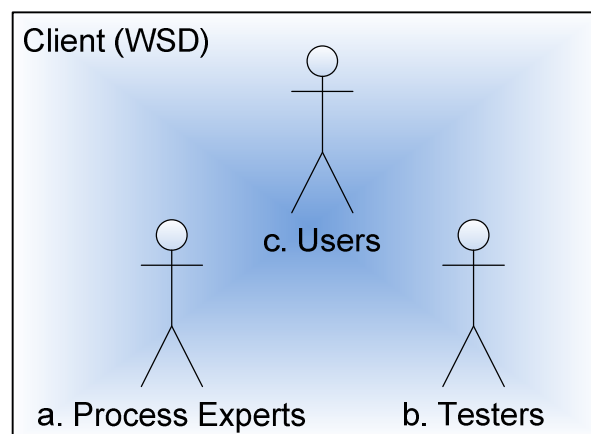


Figure 20: Decision group

3.7 System Orientation with Documentation of Care Arrangement

The next phase of the testing framework to be described is known as System Orientation; a phase in which the documentation of the information system is studied.

Four documents were used in this phase. The 'Functional specifications' of Care Arrangement [38] were used to investigate the functionalities of the information system. An 'Introduction course of Care Arrangement' [39] was also provided to understand the basics. In another document, the production options [40] (or parameterization) for WSD were described. The last document was a user manual [41] that was unfortunately not yet complete.

The added value of this orientation is that it will be easier to derive testing scripts, because specific actions can be described for navigation through the system (note that this is the first phase in which the information system is opened).

3.8 Derivation of Testing Scripts

The construction of testing scripts is considered the most difficult phase in the testing framework. A mapping has to be made of the phases leading to this phase. This is the most important step where all the elements that have been marked as relevant should return. This section starts with defining all the terms that are used in this phase for the CEPO-way.

3.8.1 CEPO-way Definitions

Three terms have to be defined before this phase can be started, namely testing scenario, testing script and testing instruction. Transaction and business process were already defined at an earlier stage. Figure 21 visualizes their relationships and Table 18 provides an example of a testing script. One should keep in mind that, although Figure 21 does not necessarily show the order in which these objects have to be constructed, it does illustrate the relationships.

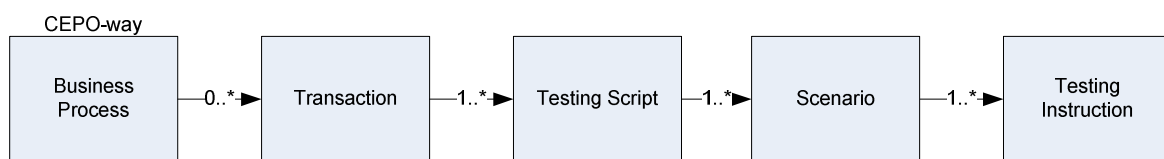


Figure 21: Definitions and relations used in CEPO-way

A scenario [42] is a synthetic description of an event or series of actions and events, i.e. occurrences in a particular business process. A testing scenario is a scenario that is used for the derivation of testing scripts. A testing script consists of one or more testing instructions. The semantics of a testing instruction consist of three parts: a description of input (what does the tester need to start), throughput (what should be done within the testing step) and expected output (what result is expected).

Table 18: One example of a testing script with 3 intertwined scenarios consisting of 10 testing instructions

Nr.	Instruction	Scenario
1.	Register client A with all personal information and a starting date of today.	A
2.	Register client B with all personal information and a starting date of one month ago.	B
3.	Register client A again in the form of client C, alert?	C
4.	Fill in the previous care program of client B.	B
5.	Delete the previous care program of client B.	B
6.	Schedule care for client A.	A
7.	Schedule care for client B.	B
8.	Adjust client A's birth date to the future.	C
9.	Schedule freelancer to care for client A.	A
10.	Schedule freelancer to care for client B.	B
11.	Calculate price for client A.	A
12.	Change care of client B.	B

The general idea of Table 18 is visualized in Figure 22. All the testing instructions are part of one testing script in this figure. Testing instructions that are used multiple times in this figure are, for example, instructions 6 and 7 from Table 18.

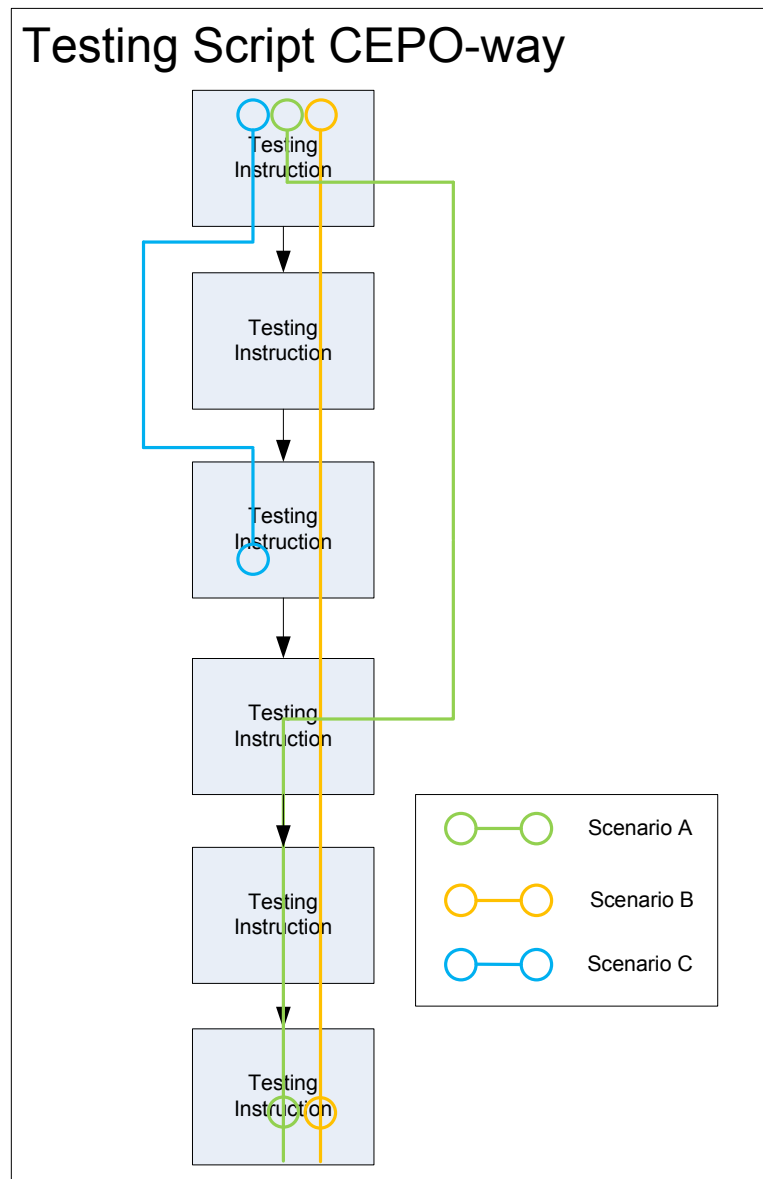


Figure 22: Parallel scenarios

In Figure 22 scenario A prescribes the success trail for this testing script, while scenario B takes one or more exceptions into account and scenario C stops before the end of the script. This is a short and simple example of a testing script, in practice testing scripts can have more than one hundred testing instructions.

The art of deriving testing scripts is to discover exceptions that are demonstrated by scenarios B and C, because then it is possible to go deeper into the information system than only the – already known and probably tested – success trails.

3.8.2 CEPO-way Scripting

The first thing that had to be done was to define a starting point from which these testing scripts would be derived. CEPO had used the Interaction Model in the past in this respect and in several cases it had been proven to be an appropriate guideline for deriving testing scripts from transactions.

Scenario Development

In practice, scenarios are developed in consultation with key users of the system and with process experts. All transactions are evaluated and many possible events are described. This process does take some time but it provides good insight into the number of scenarios that should be developed. One stops developing scenarios when both parties (client and CEPO employee) are satisfied or when the client company has depleted its budget.

Table 19: Several scenarios in CEPO-way

Scenario	Instruction(s)
Standard case	"Open the previously created 'Client' and go to the system path "Cases". Create one case, with a start date of today and leave the end date open. A case has been made."
Alternative 1 Serial cases	"Create one case with a start date of one month ago and an end date of last week." "Create another case with a start date of today and no end date. Two serial cases have been created."
Alternative 2 Parallel cases	"Create one case with a start date of one month ago and no end date. Create another case with a start date of last week and no end date. Two parallel cases have been created."

Subsequently, the transactions are placed in a table with three columns: transaction, testing subject and testing instruction. The layout as in Table 20 for the testing scripts is then ready to use.

Table 20: Scripting layout CEPO-way

Transaction	Testing subject	Testing instruction
T01	Make client file	1. Make client file with birth date in the future. 2. Save this client. 3. Check whether the system warns you or not.
...	...	...

Next, the scripting starts with opening Care Arrangement. Although several scripts contain testing instructions that are bound to this system, the key to derivation from transactions is thinking in an abstract way. Initially, one has to define the essence (or subject) of every testing script. This sounds odd because ontology is already about the essence of things but for testing scripts a lower level of abstraction is needed. For instance, in the Interaction Model from WSD, transaction T11 of

scheduling care – when thinking at the process level – actually consists of ‘define care’ (from a hard or soft copy care query), ‘divide care’ and ‘check care’. These actions can be itemized respectively as identification, processing and verification. This recurring pattern is typical of the selected transactions. When the steps of the pattern are clear in a transaction, testing instructions should be made for these scripts. The purpose of a testing instruction is that the tester is able to provide objective and clear feedback on that particular instruction in the form as depicted in Table 7, the definition of findings.

Section 3.8.1 states that instructions consist of three parts: input, throughput and expected output. These three parts must be transformed into concrete instructions as is done in the examples below.

1. Input: This part of the instruction describes what is needed to start the testing instruction.
Example: *“Open the previously created ‘Client’ and go to the system path “Cases”.*
2. Throughput: This part of the instruction describes what the tester should perform on the system. This part should lead to the outcome (output) of the testing instruction.
Example: *“Create one case, with a start date of today and leave the end date open”.*
3. Output: This part of the instruction describes the ensuing outcome of the testing instruction and what is expected. The tester should be able to check if the expected output is a result of the throughput.
Example: *“The case is created”.*

Then, point by point, the acceptance criteria and the risk analysis are checked. If all the risks and criteria that have been specified at the beginning are covered by the testing scripts, the scripts are ready for the next feedback moment. They are evaluated by a process expert who goes through the scripts to provide feedback to ensure that no important aspects of the information system are missing. Final adjustments are processed by the testing coordinators and preconditions are determined before testing. For instance, two preconditions are:

1. *Make sure you are using Internet Explorer 6 or 7.*
2. *Create a random client and fill in his/her name, address and phone number.*

The two scenarios are formatted in such a way that they can be entered into the tool that is provided by CEPO; the testing registration tool AQT. The testing scripts are then ready to be used by expert testers. The CEPO-way derived scripts for WSD can be found in Appendix D Testing Scripts CEPO-way (in Dutch).

3.8.3 DEMO-way Definitions

Although the DEMO-way makes use of the same definitions as the CEPO-way, the relationships between them are different. As can be seen in Figure 23, there are two substantial differences. If different testing subjects are defined per transaction, the CEPO-way can have more than one testing script per transaction. In CEPO scripts, scenarios are intertwined while DEMO scripts only describe one scenario. Because the Interaction, Process and State Model were used for scripting, the complexity increased significantly and the overview was lost. To use DEMO models consistently, two relations of definitions were changed into one to one relations and *testing script* and *scenario* were swapped.

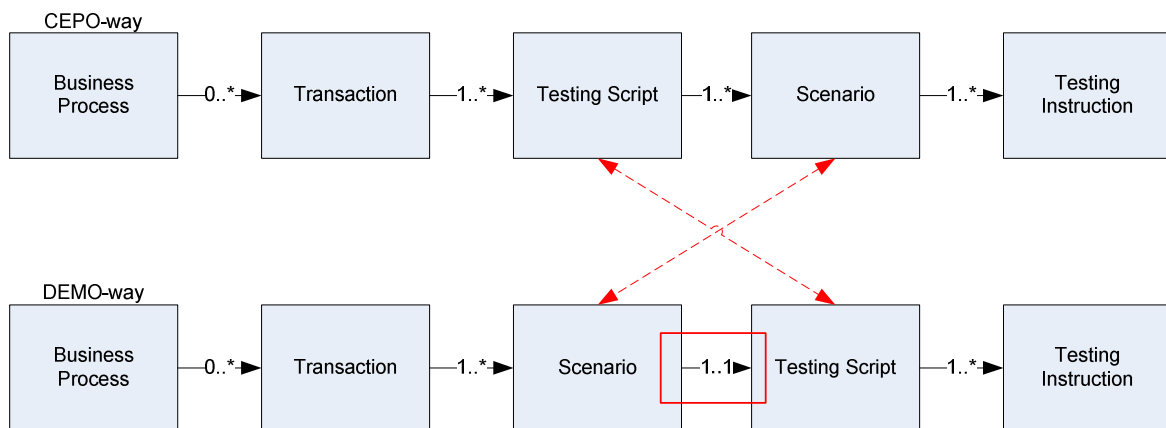


Figure 23: DEMO-way vs. CEPO-way definitions including a transaction from the Interaction Model

In the DEMO-way, every transaction will have one or more scenarios and for each scenario a testing script is written as in Figure 24. Note that this increases tracking possibilities when an error is found, because only one scenario needs to be followed in order to reproduce the error. It also helps to gain better insight into the relationship between the DEMO methodology and the derived testing scripts.

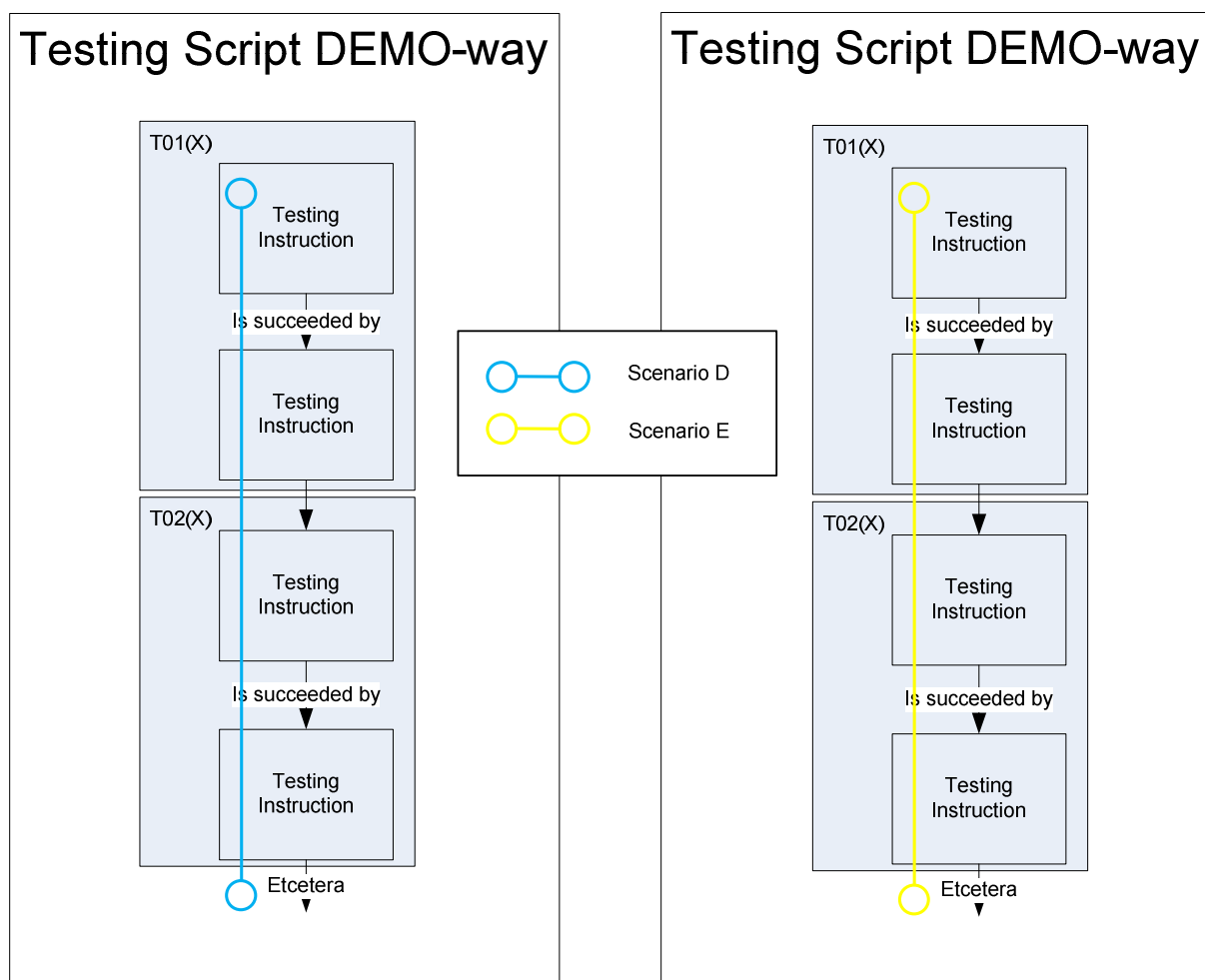


Figure 24: Single scenario DEMO-way

3.8.4 DEMO-way Scripting

This DEMO phase will use extra DEMO models when deriving the testing scripts. This section will describe the approach in which the DEMO models are used to create the DEMO scripts.

In the CEPO-way, one first starts to create testing scripts based on every transaction from the Interaction Model. The DEMO-way starts differently. One has to connect all transactions to each other in exactly the same order as in the Business Process Diagram, cf. Figure 24. In this case, it would become:

- T01(X) where X = Client
- T02(X)
- T03(X)
- T11(X)
- T12(X)
- T04(X)
- T08(P) where P = MONTH
- T09(X)
- T07(F1) + T07(F2) + T07(F3) where (F1, F2, F3) = Freelancer

Next, DEMO-way Script Creator is used for the creation of scenarios. This tool has proven to be very useful for staying within the borders of a transaction. To demonstrate the use of this tool several screenshots are presented in the ensuing figures.

Before following the order in which the transactions proceed, the State Model must be put into DSC tool as shown in Figure 25. One can add fact types and object classes from the Object Fact Diagram including its properties from the Object Property List and the time-Object Property List.

Then, when the button “Save and Close” is pressed, the information is saved and the tool will return to the start screen. Once the State Model is added, the user (testing consultant) can start scripting.

Figure 25: Entering the State Model

Figure 26 contains ten abbreviations that stand for the possible process steps within a transaction. Table 21 lists the abbreviation definitions.

Table 21: Definitions of abbreviations

Abbreviation	Definition	Abbreviation	Definition
rq	Request	c_rq	cancellation of request
dc	Decline	c_pm	cancellation of promise
pm	Promise	c_st	cancellation of state
ex	Execute	c_ac	cancellation of accept
st	State		
rj	Reject		
ac	Accept		

DSC tool and the Transaction Process Diagram were already presented. Therefore Figure 26 only serves to illustrate the link between DSC tool and the Transaction Process Diagram. One has to insert one testing instruction per field.

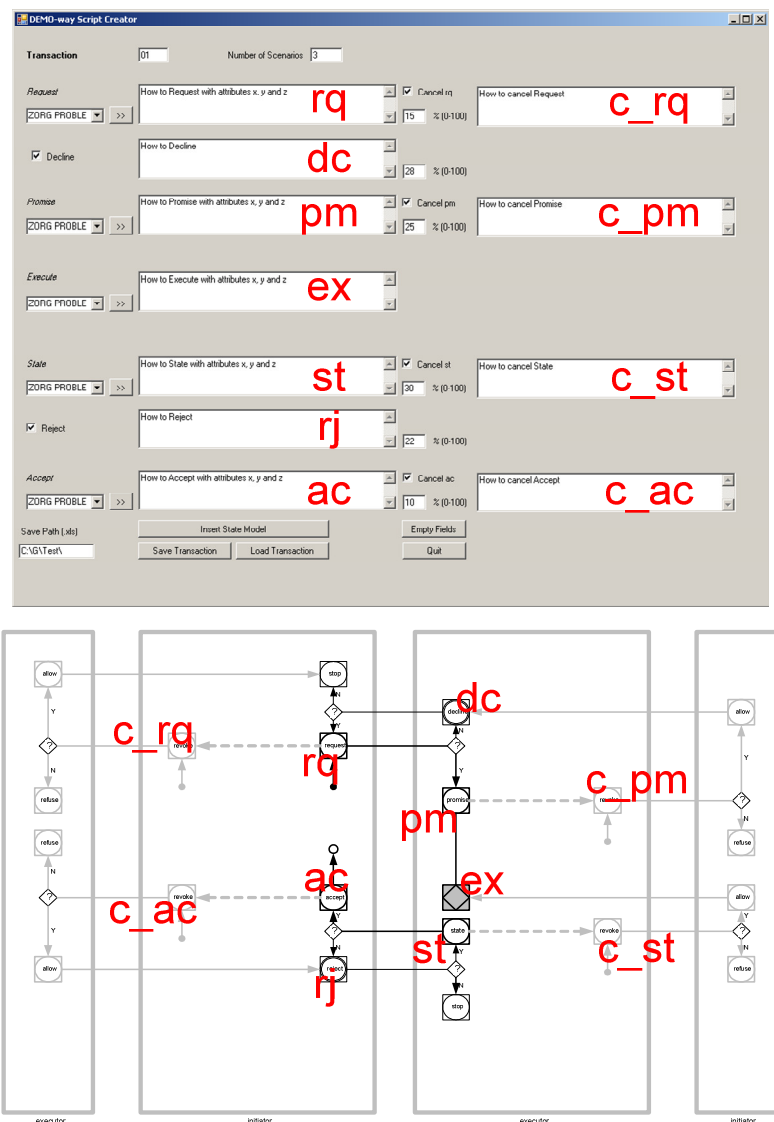


Figure 26: DSC tool's relationship to a Transaction Process Diagram

The process steps decline and reject, and all cancellation patterns (process exceptions) have a certain probability of occurrence determined by the client. The tool provides the option to fill in these values and uses them at a later stage.

When entering a Transaction Process Diagram into DSC tool, one has the option to add object classes and fact types from the previously entered State Model to the process steps.

Example:

The process step 'st' (state) needs the object class 'CARE QUERY'. In the drop-down box all object classes and fact types from the Object Fact Diagram are given that were entered before. This is where the Information Use Table (Table 13) comes into play. Here, 'CARE QUERY' is used during the 'st'-step. When 'CARE QUERY' is selected, one should press the '>>' button to place the object class with its properties into the 'state'-field. These variables can now be used to support the testing instruction.

The screenshot shows the 'DEMO-way Script Creator' window. It contains several sections for configuring a transaction process diagram. At the top, there are fields for 'Transaction' (01) and 'Number of Scenarios' (3). Below this, there are sections for different process steps: 'Request', 'Decline', 'Promise', 'Execute', 'State', and 'Accept'. Each section has a dropdown menu for selecting a process step, a text field for a description, a checkbox for 'Cancel', a probability value, and a text field for a cancellation pattern. The 'State' section is currently selected, and a dropdown menu is open showing a list of object classes and fact types, including 'ZORG PROBLEM', 'ZORG', 'ACUTE ZORG', 'ZORG VRAAG', 'CLIENT BETALING', 'ZZP BETALING', 'ZZPER', and 'Persoon is de cliënt van Acute Zorg'. At the bottom, there are buttons for 'Save Path (.xls)', 'Insert State Model', 'Empty Fields', 'Save Transaction', 'Load Transaction', and 'Quit'.

Figure 27: Adding State Model to testing scripts

When all transactions are entered into DSC tool and saved, the tool generates and saves three kinds of scenarios: one success trail, one trail with any kind of exception and one with the two exceptions of the highest probability. If the probability is higher than 40% the exception will always fire, which means that even when a success trail is generated it will belong to the testing script.

In this test case, two scenarios were used: one scenario is the success trail only and one scenario includes any process exception that may occur during a transaction. The tool outputs testing instructions per transaction. These transactions have to be assembled and merged in the order of the Business Process Diagram. Sometimes, when two transactions are merged and there are some steps missing, extra testing instructions have to be added.

Example (see Figure 28):

If one merges T11(X) and T12(F1), no freelancer has been entered into the information system yet. Therefore, between the testing instructions of these transactions, extra instructions or ‘cement’ have to be added to fill in this gap, such as:

“Create three freelancers with care level B.” (testing instruction to be placed at the red triangle)

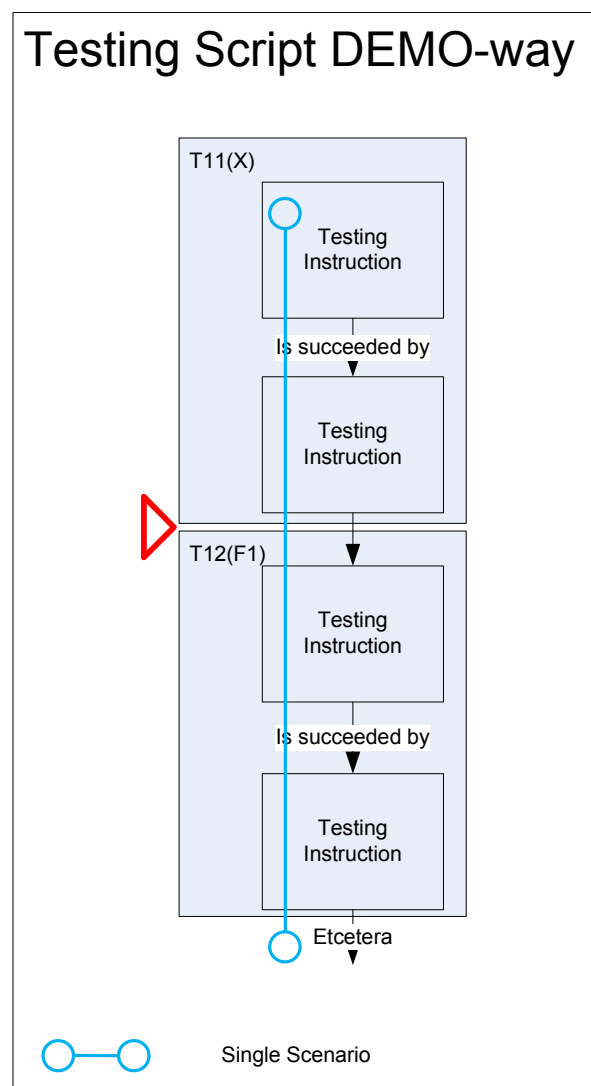


Figure 28: Adding cement between transactions

There are two scenarios that have created objects in the information system. The process of terminating these objects also belongs to a testing script (e.g. deletion of a care query). This makes it possible to test whether objects can be removed. One has to check in a testing script which objects

have been created and place the removal instructions after the existing scripts, this is the process called OC (Object Class) removal.

Then, point by point, the acceptance criteria and the risk analysis are checked. If all risks and criteria that have been specified in the beginning are covered by the testing scripts, they are ready for the next feedback moment. The scripts are evaluated by a process expert who goes through the scripts to provide feedback and to ensure that no important aspects of the information system are missing. Final adjustments are processed by the testing coordinators and preconditions are determined before testing. Two example preconditions are:

1. *Make sure you are using Internet Explorer 6 or 7.*
2. *Create a random client and fill in his/her name, address and phone number.*

Note that these final steps are equivalent to those of the CEPO-way. The two scenarios are formatted in such a way that they can be entered into the tool that is provided by CEPO; the testing registration tool AQT. The testing scripts are then ready to be used by the expert tester. The DEMO testing scripts can be found in Appendix E Testing Scripts DEMO-way (in Dutch).

3.9 Testing Results

After the testing was completed, all the results were assembled per testing script. The subsequent step is the coupling of the acceptance criteria and transactions to the testing instructions.

Explanation of Transaction and Acceptance Criterion Coupling

Section 3.6 explained how the acceptance criteria came about. Before the actual testing started, every testing instruction was linked to one (and sometimes two) of the criteria and to predefined groups of transactions as provided in section 3.4. After testing, there were some findings for testing instructions that were not linked to a testing instruction beforehand. To resolve this issue, these testing instructions were counted twice so that the number of findings would never be higher than the number of testing instructions. The best way to clarify this is with an example.

Example: Testing instructions and the way acceptance criteria were bound to them

Acceptance criterion	Testing instruction
Financial functionality / <i>Understandability</i>	The freelancer delivered care to the client. Several appointments must be confirmed through Module Roster – Confirmation. Do this for two appointments.
Traceability	An appointment was not performed according the actual schedule. Go to Module Roster – Appointments – Restore. Restore an appointment that needs to be restored.

The non-italic quality criteria were linked to testing instructions beforehand, but during testing some testers thought that it was not understandable enough to perform the first instruction. Therefore the additional italic criterion *Understandability* was linked afterwards. This testing instruction now counts as two instructions, because two findings are associated with it.

The transaction groups were connected beforehand and do not change afterwards, although more than one finding per testing instruction also counts as the amount of findings that is given.

Example: Testing instructions and the way transactions were bound to them

Transaction group	Testing instruction
T07 / T08 / T09	The freelancer delivered care to the client. Several appointments must be confirmed through Module Roster – Confirmation. Do this for two appointments.
T07 / T08 / T09	An appointment was not performed according the actual schedule. Go to Module Roster – Appointments – Restore. Restore an appointment that needs to be restored.

Results

The final testing results are given in Table 22 and show that the DEMO-way found significantly more lacking functionalities, errors and weaknesses than the CEPO-way.

Table 22: Final testing results based on findings

				
CEPO-way	39	16	15	8
DEMO-way	41	17	20	20

The number of neutral findings is almost equal, because in most cases these findings represent cosmetic findings. Since those do not comprise functionality, this also proves the consistency of the testing. The DEMO-way scored much higher on negative and blocking findings.

For example, only the DEMO-way discovered that freelancers could be connected to a client that dislikes them. When one wants to shift the care delivery to another day, the care disappeared and was not traceable anymore. This problem was also not detected in the CEPO-way. The next chapter will go into more detail about the differences of the frameworks.

The testing results are presented in a report for the client. These findings are used to set up the report and recommend the client to continue working with the information system or not. The client report for this case can be found in Appendix F Client Report (in Dutch).

Part D Argumentation for the use of DEMO

4 Comparison of Testing Frameworks

A number of criteria were determined at the beginning of this study in section 1.2. These criteria will return in this section. The comparison is based on quantified testing results, development time, time to walk through testing scripts, understandability and reproduction possibilities. In this chapter, every criterion is dealt with in the presented order.

The results of the frameworks are presented in a radar (or spider) chart [37, 43] to give an overview of the testing results. The first two charts are based on the same acceptance criteria and its norms that were set up at the beginning, the latter two show transactions from the Interaction Model on their axes and an additional axis *Object Class Removal*.

The differences in the outcome of the frameworks were illustrated in two ways, because both diagrams are capable of adding information to the final research goal. These charts are completely based on the results of the testing frameworks; user experiences and time to develop are left out of these charts. They will be taken into account at a later stage.

Note that the ensuing charts are not a complete measure for Care Arrangement's quality, they only serve to illustrate the difference between the two frameworks.

4.1 Quantification of Testing Results

Section 2.2 lists what findings can be given for a testing instruction. All the findings of the testing instructions are brought together in charts as shown in Figure 29. To explain the calculation for the values on the axes of the radar charts an example is given below.

Example: Findings of acceptance criterion Financial Functionality

Finding	😊 (pos)	😐 (neutral)	😞 (neg)	💣 (block)
Frequency	4	1	0	8

$$\# pos + \# neutral + \# neg + \# block = \# total$$

$$\text{Calculation for percentage light grey: } \frac{\# pos + \# neutral}{\# total} = \frac{4 + 1}{4 + 1 + 0 + 8} = 38\%$$

$$\text{Calculation for percentage dark grey: } \frac{\# pos}{\# total} = \frac{4}{4 + 1 + 0 + 8} = 31\%$$

□ % gap (negative and bombs) ▒ % fit (positive and neutral) ■ % fit (positive only)

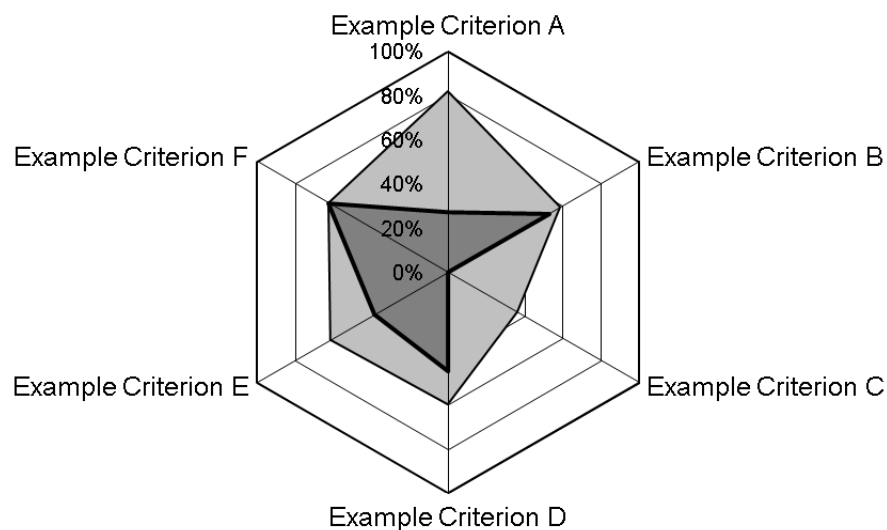


Figure 29: Example of a radar chart

The surfaces of the grey areas in the charts now depict the comparison of results at a glance. The grey areas stand for the relative amount of positive findings. If the grey areas are large, a lot of testing instructions resulted in the expected output. The smaller the grey surface, the fewer the number of test instructions that resulted in the expected output.

4.2 Comparison Based on Acceptance Criteria

Figure 30 shows the CEPO-way results and Figure 31 the DEMO-way results. One remarkable characteristic that can be derived from these diagrams is that the major difference is the criterion Traceability: 25% of all instructions scored positive for DEMO versus 89% for CEPO. The DEMO-way has an important focus on undoing and traceability, especially because of its cancellation patterns. The scores for the other criteria are more or less the same, they only differ from 6% to 14%. These findings are depicted relatively, more were registered during the DEMO scripts, in other words, the DEMO scripts found more lacking functionalities, errors and weaknesses. So if these problems were communicated to Company X, more mistakes could be corrected.

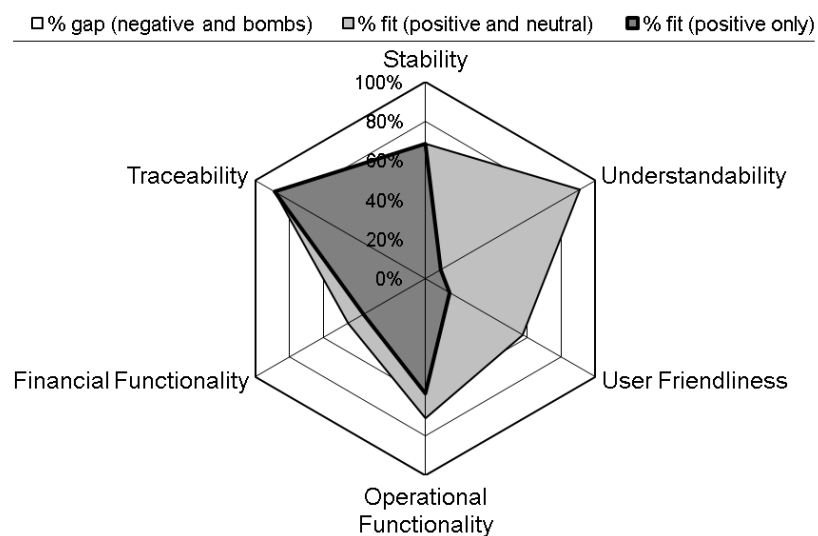


Figure 30: CEPO-way with acceptance criteria on its axes

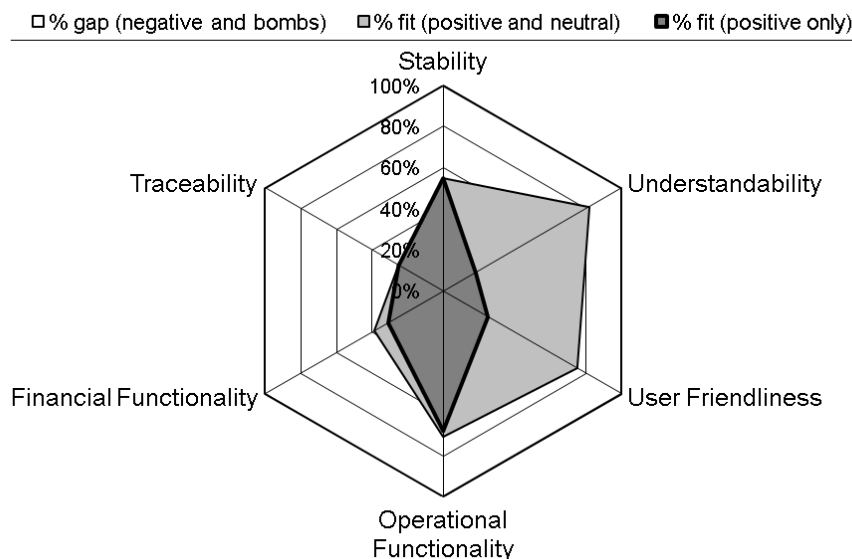


Figure 31: DEMO-way with acceptance criteria on its axes

The next method to represent testing results is to use transactions as axes. One should again keep in mind that there were more DEMO findings than CEPO findings.

4.3 Comparison Based on Transactions

Figure 32 and Figure 33 illustrate the differences between the two frameworks based on transactions that are grouped in processes. Both grey areas in these figures have approximately the same shape although the DEMO-way's surface is perceptibly smaller. It can again be concluded from this that the DEMO-way found more problems in the information system than the CEPO-way. OC removal scores clearly higher in the CEPO-way than in the DEMO-way. Client Registration does have a lower percentage in the CEPO-way, this can be explained by the fact that the CEPO-scripts required the creation of more clients than the DEMO scripts. Derived from the risk analysis, WSD's priorities are scheduling care (CEPO: 33%, DEMO:40%), freelancers (CEPO: 35%, DEMO: 15%) and performing financial transactions (CEPO: 54%, DEMO: 27%) correctly. These are significant differences.

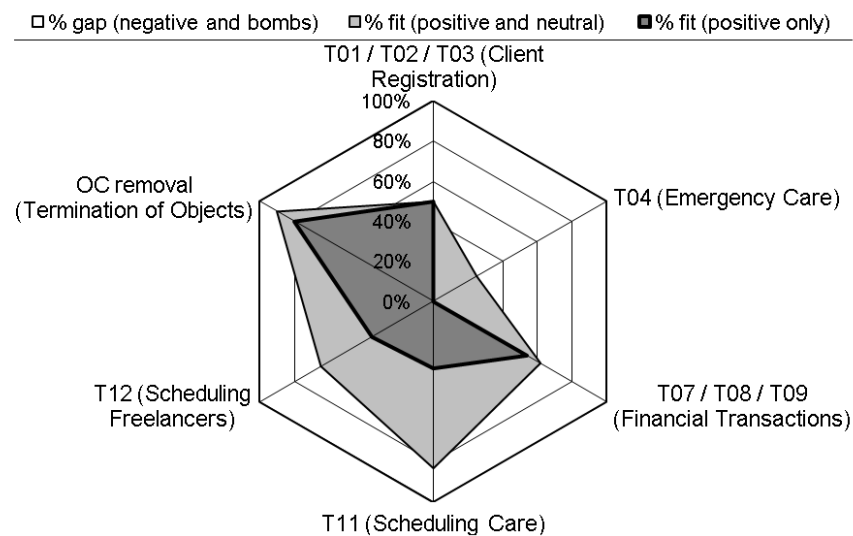


Figure 32: CEPO-way with business processes on its axes

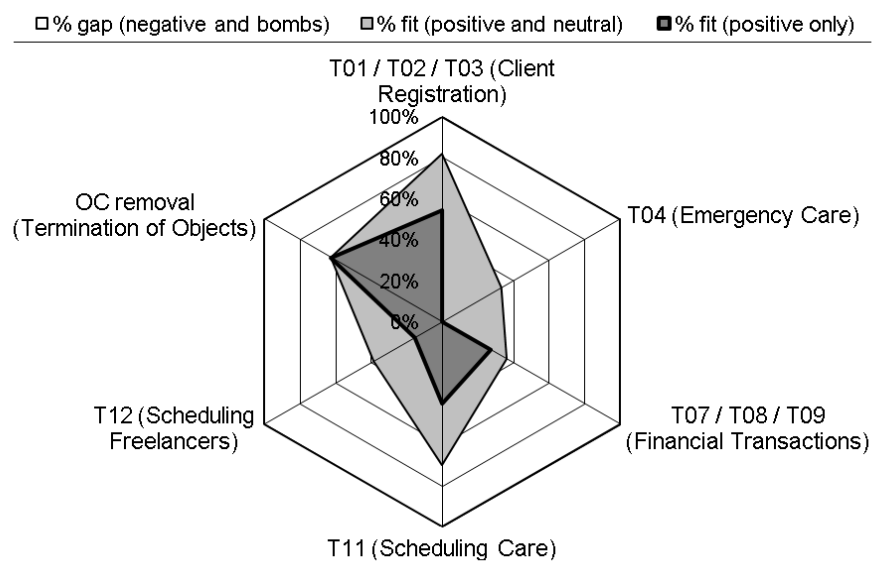


Figure 33: DEMO-way with business processes on its axes

The next section will discuss additional criteria that cannot be derived (directly) from testing results.

4.4 Comparison Based on Additional Criteria

Now that the testing results have been presented, several other criteria should be described as well. The development time for testing scripts is an important factor, especially when the frameworks are used by a small or medium-sized company. Their budget or the number of man-hours available per project is not always as much as desired. Therefore the time to develop testing scripts was also monitored. It turned out that the CEPO-way took 40 hours and the DEMO-way 54 hours to develop. The Process Model and the State Model both took six hours to build. The DEMO scripting took longer than the CEPO scripting, because more information needed to be added to the scripts. However, thanks to DSC tool the scripting time only increased by two hours.

The actual testing took two days using two teams. As can be seen in Table 3 on page 23, group A first went through the CEPO-scripts and ended with the DEMO scripts, and group B worked vice versa. The second testing day, on which the groups used the other script, took two hours less because the testers had become familiar with the way of testing. All testers finished the scripts in more or less the same time (day one: 9 hours and day two: 7 hours), irrespective of which script was used. The difference between individual testers was 15 minutes at most.

Moreover, the second testing day was four days later, so that the testers started the testing day as fresh as on the first one. Another aspect that improved the consistency of the testing results, was that the groups were energetically directed and controlled during both testing days. This paid off in the consistency of results, for instance, in that the number of neutral (cosmetic) findings per script was almost equal.

Table 23 summarizes the criteria discussed above and presents a quick overview of the comparison.

Table 23: Overview of CEPO versus DEMO

	CEPO	DEMO
# Test instructions	78	98
Development time	40 hours	54 hours
Walkthrough time	1 day (8 hours)	1 day (8 hours)
Understandability	Medium	Good
Reproduction possibilities	Medium	Good
Testing results	16 x 😊	17 x 😊
	15 x 😐	20 x 😐
	8 x 🌟	20 x 🌟

Most of the criteria in Table 23 were quantified except for *Reproduction possibilities* and *Understandability*. The corresponding scores are explained in the following paragraphs.

When the report is used in discussions with the client and the software supplier, the reproduction of errors is easier in the DEMO-way because scenarios are built up in series, and parallel in the CEPO-way (cf. section 3.8). This means that a larger part of the script has to be redone in the CEPO-way if one wants to reproduce the same error that was found. Therefore, the DEMO-way scores 'Good' and the CEPO-way 'Medium'. CEPO agreed with the author on these scores.

The testers were asked to give their opinion about the understandability of both scripts on an ordinal scale of: 'Bad', 'Medium' and 'Good'. The testing team thought that the DEMO scripts were more understandable than the CEPO scripts and they scored respectively 'Good' and 'Medium'. The instructions were described more precisely, so that navigating through the system was easier. This also explains why the DEMO-way and CEPO-way took the same amount of time to go through even though there were more DEMO instructions.

5 Conclusion

Considering the testing results and the effort put into the derivation of testing scripts, the DEMO-way seems to be more suitable for use during testing than the CEPO-way. Although it takes more time to construct DEMO models of an organization, in the end this way of modelling definitely pays off. The outcomes of the DEMO-way showed more shortcomings in the information system than the CEPO-way. This is explained by the preciseness of describing processes with additional models from the DEMO methodology. The Process Model (expressed in Transaction Process Diagrams), in combination with DSC tool, proved its added value in this case, because it obliges the scripter to use the cancellation patterns. These patterns were used to produce structured testing scripts for the information system.

The State Model, which is expressed in the Object Fact Diagram and the (Time-) Object Property List, and the Information Use Table were also of importance during the derivation of the DEMO scripts. DSC tool made it possible to connect every process step in the Process Model to fact types and one or more object classes from the Object Fact Diagram with their properties from the (Time-) Object Property List. It became clear that these models were a valuable addition when entering transactions into DSC tool. This tool saved a considerable amount of development time during the derivation of testing scripts.

The DEMO and CEPO testing scripts took the same amount of time to walk through, which shows that the testers did not sense any difference in using either the DEMO-way or CEPO-way scripts. This means that the client company does not have to spend more time or money to perform the tests in the DEMO-way.

Afterwards, the testers stated that the DEMO scripts were easier to understand due to more precise navigation. This explains the same walk-through time despite the higher number of testing instructions in the DEMO-way.

When a client report is constructed it is more easy in the DEMO-way to reproduce errors and / or weaknesses of the information system than in the CEPO-way. This is because scenarios are split up in the DEMO-way, while in the CEPO-way they are intertwined. Therefore, it will cost the software supplier less effort to reproduce the errors and / or weaknesses from the DEMO-based client report. The information system can then be corrected more cost-efficiently.

The outcome of this research presents a considerable improvement in the discovery of errors and weaknesses of information systems against a fair increase of development time. The most crucial models were the Transaction Process Diagrams, because they include cancellation patterns. For the foundation of this research it was necessary to construct them all but in practice one should lay the complete pattern template (including cancellation patterns) next to the transaction for which the scripts are written. This way, the Transaction Process Diagram is still used, but it is not such a time-consuming activity as modelling all the Transaction Process Diagrams. Using the complete pattern template causes less deviation per person in the final scripts and it reaches more boundaries of the information system.

As was stated in the introduction, many software projects fail due to a lack of communication with the stakeholders. When an information system is tested as meticulously as is presented in this study,

the communication improves considerably. The report for the client company can be used to improve communication because the business processes are modelled in detail thanks to the DEMO methodology, acceptance criteria are determined, a risk analysis is performed and the information system has been tested. These aspects all contribute to preventing the projects being cancelled or the budget being exceeded.

6 Recommendations

My advice is to start using the following steps as given in the brief overview below.

Brief overview of performing the new testing framework

1. Construct Interaction Model; organization's overview is clear and unambiguous (3.1).
2. Construct Business Process Diagram; process order is determined (3.2).
3. Construct State Model; important objects are then identified (3.3).
4. Determine transaction groups / business process; this will be used for the risk analysis (0).
5. Perform risk analysis; most important processes are identified (0).
6. Determine acceptance criteria; create a feedback mechanism for the final report (3.6).
7. Explore the information system; what is possible with the information system (3.7).
8. Insert the State Model into DSC tool; this information will be used in the next step (3.8.4).
9. Start building testing scripts based on the Business Process Diagram-order while using DSC tool. If a new transaction starts, use the Transaction Process Diagram to explore possible cancellation patterns (3.8.4).
10. Generate the scripts per transaction. Open the information system and do a pre-test to check whether the testing scripts have missed any important instructions so that the scripts can be performed one after another. If there are any gaps between these transactions, fill them up with extra instructions.
11. Check that all acceptance criteria have been tested in the instructions. If not, add them manually between the scripts.
12. Continue building testing scripts and put *Termination of Object Classes* behind the scripts (3.8.4).
13. Start testing (Appendix B Testing Plan of Action).
14. Check the testing results with the testers; remove duplicated and unclear findings.
15. Draw up the client report; final deliverable (Appendix F Client Report (in Dutch)).

For further detail, please refer to all the sections, chapters and appendices that were mentioned above.

7 Reflection

My thesis project was the final stage of my Computer Science programme. In this project I was able to put to use a lot of the knowledge that I had acquired in the last years of my study. Although I first thought that testing was not a very interesting subject, it turned out otherwise. The enthusiasm of the people I worked with and the way CEPO conducts testing pushed me in the right direction. This made the subject interesting, so that I could be completely dedicated to this study.

Thanks to the experience of my supervisor at CEPO, R. Ceelen, the quality of this project increased significantly. One of the best pieces of advice was to start building the report from the third month, in this way every text that was written about testing and / or CEPO could be placed in one single report that eventually would become part of the final report.

Another important aspect was that I was obliged to construct a plan for every next phase. I used MS Project and from now on when I will start a complex project, I will definitely use planning software to monitor finished and unfinished activities.

During the project we had weekly meetings to synchronize our ideas. I always drew up my own agenda and received feedback during these meetings. These meetings were extremely useful. When I encountered a problem outside a meeting I was always able to consult CEPO.

References

- [1] J. L. G. Dietz, *Enterprise ontology theory and methodology Online resource*. Berlin: Springer, 2006.
- [2] K. Ewusi-Mensah, "Critical Issues in ABANDONED Information Systems Development Projects," *COMMUNICATIONS OF THE ACM*, vol. 40, 1997.
- [3] B. Whittaker, "What went wrong? Unsuccessful information technology projects," *Inf. Manage. Comput. Security*, vol. 7, 1999.
- [4] IT-Cortex, *Failure Rate*, 2008, http://www.it-cortex.com/Stat_Failure_Rate.htm
- [5] M. Krigsman, *IT Project Failures*, 2008, <http://blogs.zdnet.com/projectfailures/>
- [6] M. Gijzemijter, *Peperduur ICT-Project VROM Mislukt*, <http://webwereld.nl/comments/49134/peperduur-ict-project-vrom-mislukt.html>
- [7] J. Doorn, *Overheid smijt geld over IT-balk*, <http://webwereld.nl/articles/46659/-overheid-smijt-geld-over-de-it-balk-.html>
- [8] P. Carla Marques and S. Pedro, "Enterprise architecture: business and IT alignment," in *Proceedings of the 2005 ACM symposium on Applied computing* Santa Fe, New Mexico: ACM, 2005.
- [9] G. Carter, *Avoiding IT project failure: Holding vendors' feet to the fire*, 2008, <http://wistechology.com/articles/4310/>
- [10] K. Mark, E. C. Paul, L. Kalle, and C. S. Roy, "A framework for identifying software project risks." vol. 41: ACM, 1998, pp. 76-83.
- [11] S. Mary, "Critical success factors in enterprise wide information management systems projects," in *Proceedings of the 1999 ACM SIGCPR conference on Computer personnel research* New Orleans, Louisiana, United States: ACM, 1999.
- [12] D. Remenyi, *Stop IT Project Failures: Through Risk Management*: Butterworth-Heinemann, 1999.
- [13] F. Blankena, "Lek toont zwakte open source," in *Automatiseringsgids*. vol. 22, 2008.
- [14] R. N. Charette, "Why Software Fails," *IEEE Spectrum*, 2005.
- [15] R. Ceelen, *CEPO Beyond ICT*, <http://www.cepo.nl/>
- [16] W. contributors, *Acceptance Testing*, 2008, http://en.wikipedia.org/wiki/Acceptance_testing
- [17] W. contributors, *Framework*, 2008, <http://en.wikipedia.org/wiki/Framework>
- [18] Sogeti, *Sogeti.nl*, <http://www.sogeti.nl/Home/index.jsp>
- [19] Sogeti-Nederland-B.V., *TMap next*, <http://www.tmap.net>
- [20] T. Koomen, L. v. d. Aalst, B. Broekman, and M. Vroon, *TMap Next, for result-driven testing*: UTN Publishers, 2006.
- [21] W. Contributors, *Test Management Approach*, 2008, http://nl.wikipedia.org/wiki/Test_Management_Approach
- [22] W. Contributors, *Beta Testing*, http://en.wikipedia.org/wiki/Beta_testing#Beta_testing
- [23] H. Mulder, *Rapid Enterprise Design*. Den Haag: Koninklijke Bibliotheek, 2006.
- [24] R. Ceelen, "In het Oosten / Procesbeheersing," CEPO, Gorinchem 2005.
- [25] P. Kastelijn, "Referentiemodel Thuiszorg," CEPO, Gorinchem 2007.
- [26] R. v. Oosterwijk, "Testaanpak met ACTA Kwaliteitsmeter," Gorinchem 2008.
- [27] W. Contributors, *Methodology*, 2008, <http://en.wikipedia.org/wiki/Methodology>
- [28] W. Contributors, *Scientific Control*, 2008, http://en.wikipedia.org/wiki/Scientific_control
- [29] J. Muns, "Development and Maintenance of a DEMO Modeler," TU Delft, Delft, November, 2007.
- [30] W. contributors, *Business process*, 2008, http://en.wikipedia.org/wiki/Business_Process
- [31] H. Splinter, *Wilhelmina Zorg en Welzijn*, 2008, <http://www.wilhelminazorgenwelzijn.nl/>
- [32] A. Bloemers, *Per Saldo*, <http://www.pgb.nl/showpage.php?pa=3&menu=0,0>
- [33] ChangeSource-Pty-Ltd., *Change Management Toolbox*, <http://www.change-management-toolbook.com/Default.aspx?tabid=499>

- [34] J. L. G. Dietz, "DEMO 3.0 Library," DEMO Kenniscentrum, Delft 2007.
- [35] I. As, "Strategisch Marketing Plan," Wilhelmina Senioren Diensten, Leidschendam 2007.
- [36] W. Contributors, *ISO 9126*, http://en.wikipedia.org/wiki/ISO_9126
- [37] A. Boeters, *Kwaliteit op maat* vol. 2: tenHagenStam, 2004.
- [38] CareArrangement-Producent, "Functional Specifications," Care Arrangement Producent, Nieuwegein 2008.
- [39] CareArrangement-Producent, "An Introduction Course of Care Arrangement " Care Arrangement Producent, Nieuwegein 2008.
- [40] CareArrangement-Producent, "Parameterization of WSD," Care Arrangement Producent, Nieuwegein 2008.
- [41] CareArrangement-Producent, "User Manual for Care Arrangement," Care Arrangement Producent, Nieuwegein 2007.
- [42] W. contributors, *Scenario* 2008, <http://en.wikipedia.org/wiki/Scenario>
- [43] R. Ceelen, "Functionele Fitgap ASTA," CEPO, Gorinchem 2007.

Appendix A DEMO description

Design and Engineering Methodology for Organizations

This paragraph will give a short description of the DEMO methodology. The main objectives of this methodology are highlighted throughout the text and explained by two figures.

Performance in Social Interaction

The DEMO methodology for (re)designing and (re)engineering organizations, developed under the supervision of Prof. Dr.ir. Jan L.G. Dietz, is based on the PSI-paradigm (Performance in Social Interaction). This paradigm states that the operational principle of organizations is the ability of human beings to enter into and comply with commitments towards each other, collectively called social interaction. It constitutes the essential difference between organizations and other kinds of systems, like information systems. The PSI-paradigm explains at the same time, why applying information systems oriented methods and techniques to organizations makes little sense; it can only lead to confusion. In DEMO, an organization is considered as a system of actors that perform production acts and coordination acts. By performing production acts (P-acts), actors contribute directly to the realization of the function of the organization. By performing coordination acts (C-acts), actors enter into and comply with commitments about P-acts. An actor role is the authority to perform a particular type of P-acts and the corresponding C-acts.

The Atoms, Molecules and Fibres of Organizations

P-acts and C-acts are the 'atoms' of organizations. They always and only occur in universal patterns, called (business) transactions. Transactions therefore constitute the 'molecules' of organizations. Two actor roles are involved in carrying through a transaction, one is called the customer (initiator) and the other one the producer (executor). A business process is a tree-structure of connected transactions, starting with the transaction through which a service is delivered to the environment of the organization, or with an internal self-activation. Business processes constitute the 'fibers' of organizations.

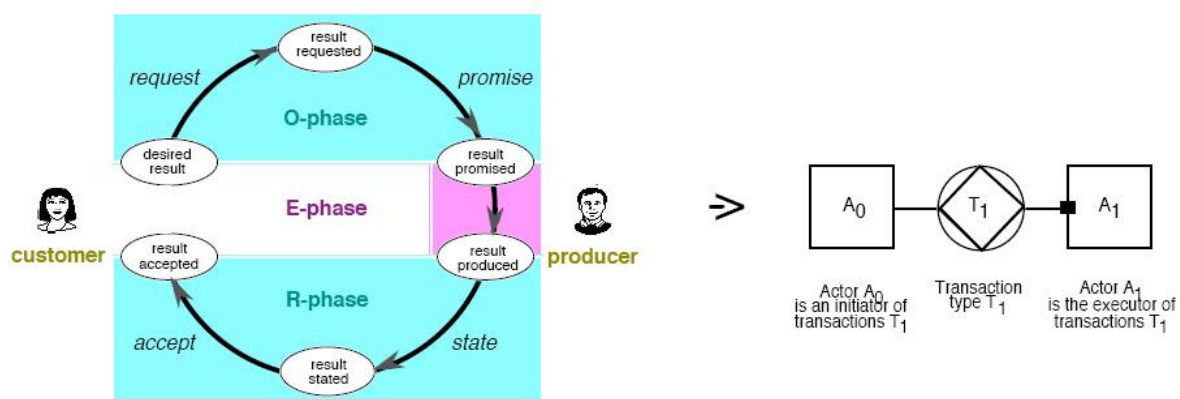


Figure 34: DEMO transaction

A transaction evolves in three phases. In the order phase or O-phase, the two actors strive to reach agreement about the result (the P-fact) to be established (e.g., the selling of a good). The basic C-acts are the request and the promise. In the execution phase or E-phase, the executor produces the result (e.g., deciding to sell). In the result phase or R-phase, the two actors strive to reach agreement about

the established result. It starts with the statement by the producer (executor) and ends with the acceptance by the customer (initiator). The figure above shows a transaction.

The three Aspect Organizations

In fulfilling actor roles, three human abilities are distinguished (see figure below). The forma ability in coordination means being able to speak, write, listen and read (formative acts); regarding production, it means being able to perform datalogical acts, like storing, transmitting, and copying data or documents. The informa ability in coordination means being able to formulate and interpret thoughts (informative acts); regarding production, it means being able to perform infological acts, like reasoning and computing. The performa ability in coordination means being able to feel committed as performer or addressee of a coordination act; regarding production, it means being able to perform ontological acts, like deciding and judging.

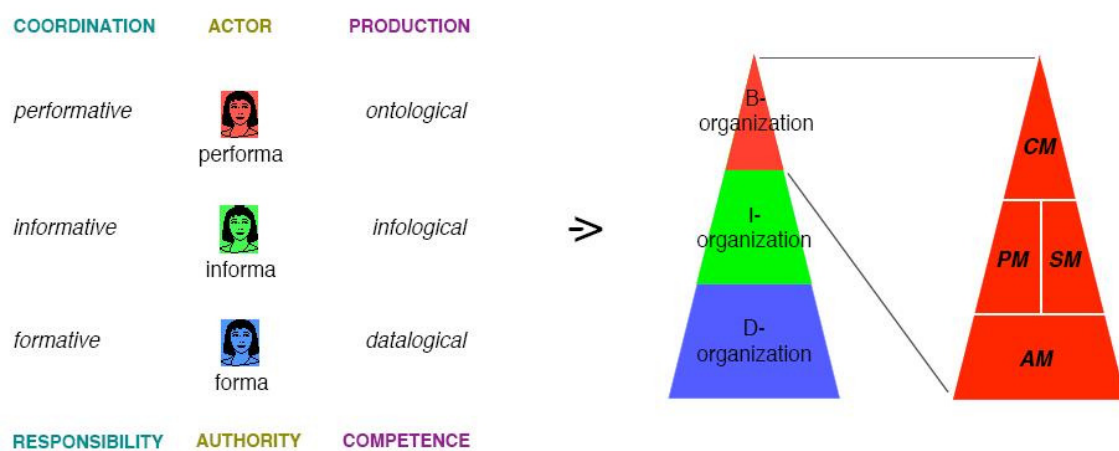


Figure 35: Ontological, Infological and Datalogical

Looking at the production side of an organization, three different aspect organizations can be distinguished: the D-organization (D of documents), the I-organization (I of information), and the B-organization (B of business). The actual members of an organization may partake in all three aspect organizations. For example, a person fulfilling a B-actor role (e.g., deciding on a sales order) may take his or her shape of I-actor for checking the current stock, and may take his or her shape of D-actor for sending a confirmation letter to the customer. Sometimes, one's 'highest' shape may be I-actor only (e.g., an accountant or an arithmetician) or D-actor only (e.g., a postman or an archivist).

Enterprise Ontology

The ontology of an enterprise is the knowledge of the construction and operation of its B-organization. The ontological model of an enterprise shows its essence, fully abstracted from realization and implementation issues.

Appendix B Testing Plan of Action (in Dutch)

Testsdraaiboek Care Arrangement

<i>Datum</i>	:	28-04-2008
<i>Klant / Project</i>	:	Thuiszorg / WSD
<i>Van</i>	:	Paul Kastelijn, Lauwerens Metz
<i>Kenmerk</i>	:	Acceptatietest

Inleiding

Op 28 april 2008 wordt een acceptatietest gedaan op het systeem Care Arrangement van de firma Dataproof IT. Dit systeem is ingericht voor WSD om het administratieve proces te ondersteunen.

Doel

Het uitvoeren van een acceptatietest voor WSD heeft als doel om op basis van het organisatieproces, het systeem te testen en een objectief oordeel te vormen over wel of niet accepteren van het systeem teneinde hiermee operationeel te kunnen gaan.

Testbasis

Voor het kunnen uitvoeren van de acceptatietest zal het volgende worden gebruikt:

- Computer met toegang tot Care Arrangement
- Acceptatietestscripts (2 verschillende versies)
- CEPO Testapplicatie voor vastleggen van testbevindingen

Testaanpak

De acceptatietest wordt gedaan in 2 testrondes en op basis van 2 zogenaamde testscripts. Een testscript is een set van instructies (teststappen) die de tester gebruikt om Conexus te doorlopen. Per teststap wordt aangegeven welke handelingen de tester moet uitvoeren in het systeem. Ook wordt na elke teststap gevraagd de bevinding vast te leggen. Zie verderop: teststappen en testbevindingen.

Testonderdelen

Het systeem wordt getest op de volgende hoofdonderdelen:

- Cliënt en dossier maken
- Zorgvraag aanmaken
- Zorgvraag roosteren
- Inplannen ZZP'ers
- Acute zorg leveren
- Aanmaken facturen

Agenda

8:00	8:30	30 min.	Instructie acceptatietest en CEPO Testtool
8:30	10:00	90 min.	Start testronde 1
10:00	10:15	15 min.	Korte pauze
10:15	12:30	75 min.	Vervolg testronde 1
12:30	13:15	45 min.	Middaglunch
13:15	15:30	135 min.	Start testronde 2
15:30	15:45	15 min.	Korte pauze
15:45	16:45	60 min.	Gezamenlijke evaluatie

Applicatie

Tijdens de acceptatietest gebruikt u 2 applicaties:

1. [Conexus](#)

Open internetexplorer en ga naar onderstaande URL:

<https://conexus.dataconcept.nl/wilhelminatest/conexus/webdb/login.php>

Log in met de volgende gegevens:

Inlognaam: lauwerens

Paswoord: xxxxxxxx

2. CEPO testtool

Voor het verwerken van de bevindingen van de acceptatietest wordt de testtool van CEPO gebruikt. In dit Excel-programma kunnen ook printscreens bijgevoegd worden.

Open de CEPO testtool op de desktop met de snelkoppeling:

Acceptatietest 1 of **Acceptatietest 2**.

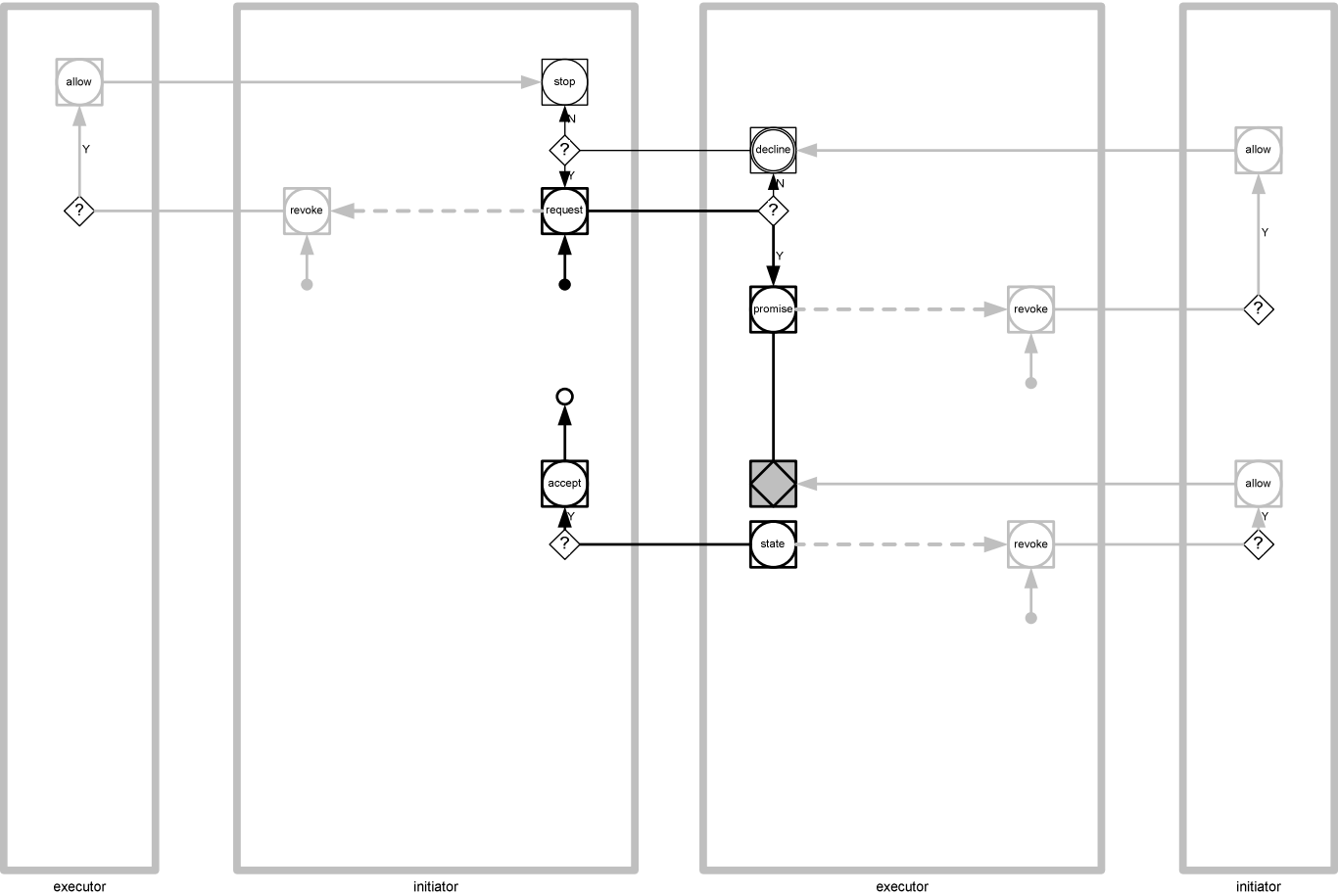
Teststappen en testbevindingen

Elke teststap heeft een testnummer (bv. 010 – 010). Per teststap zal in de CEPO testtool worden gevraagd om een bevinding vast te leggen. De volgende bevindingen zijn mogelijk.

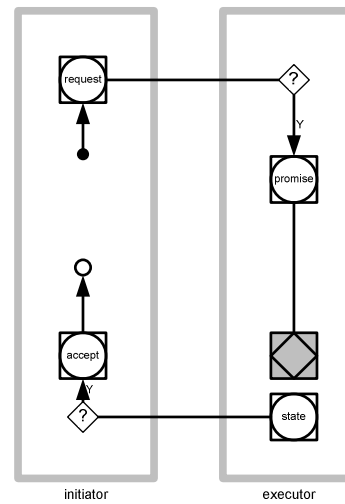
- ☺ = de teststap kan goed uitgevoerd worden
- ☹ = de teststap kan wel uitgevoerd worden, maar er dient een kleine opmerking geplaatst te worden (bv. een cosmetische opmerking)
- ☹ = de teststap kan niet uitgevoerd worden, maar er kan wel verder gegaan worden met de volgende teststap. Het proces blokkeert dus niet.
- 💣 = de teststap kan niet uitgevoerd worden omdat een blokkerende fout optreedt (bv. Conexus loopt vast of er verschijnt een foutmelding, de gegevens worden niet opgeslagen)

Appendix C Transaction Process Diagrams

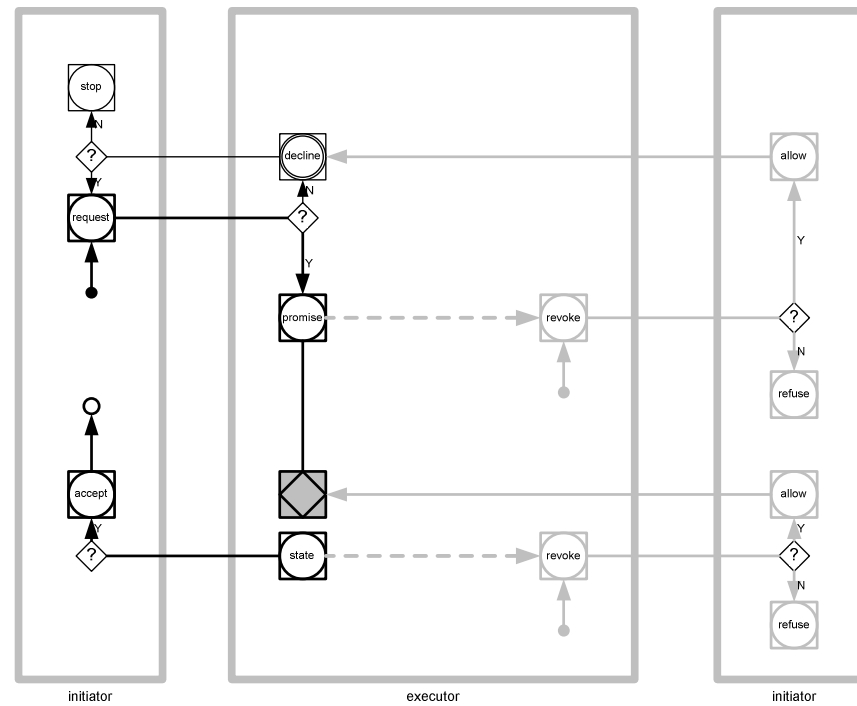
Transaction T01 Care problem determining R01 Care problem CP has been determined



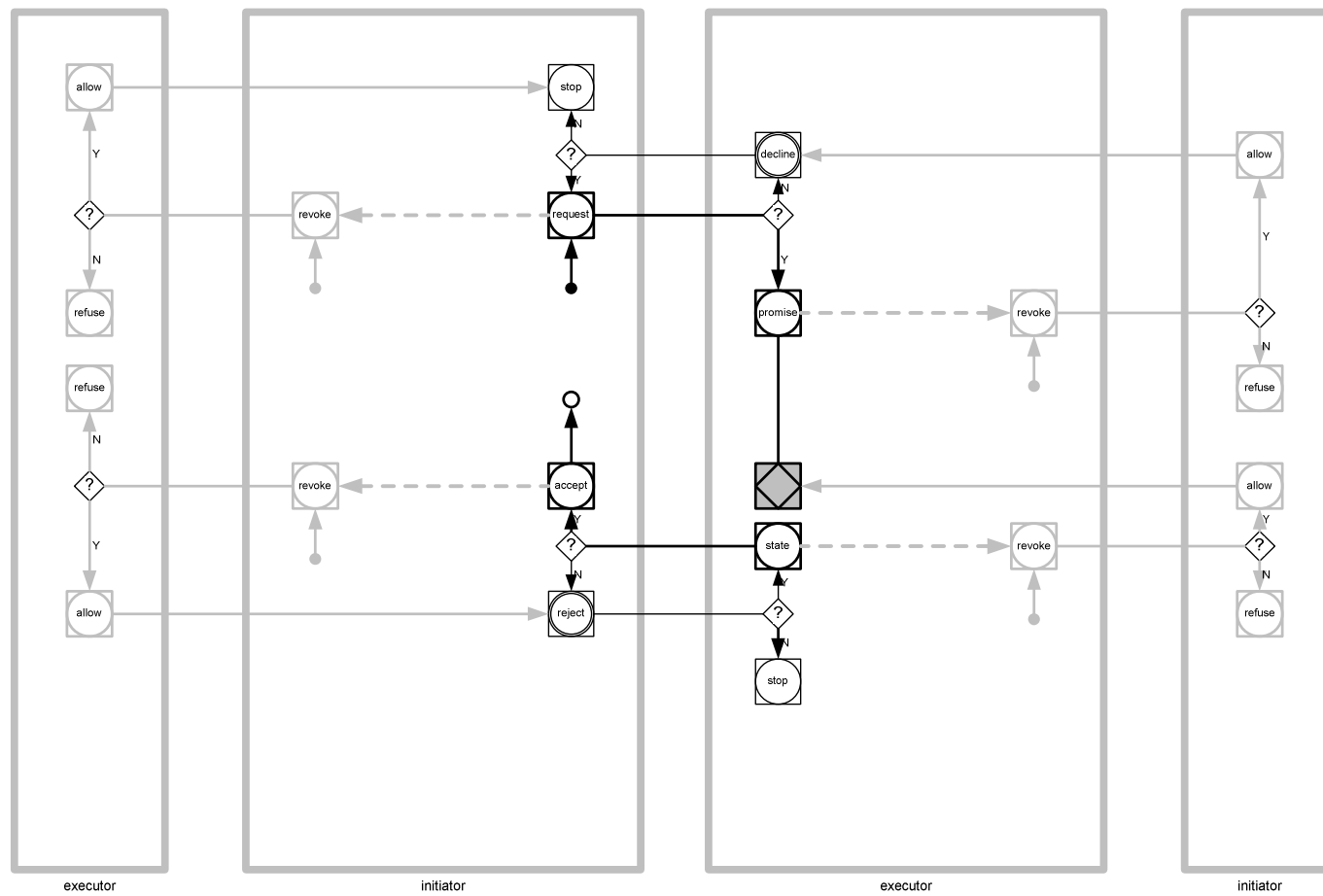
Transaction T02 Internal care query composing R02 Care query CQ has been composed



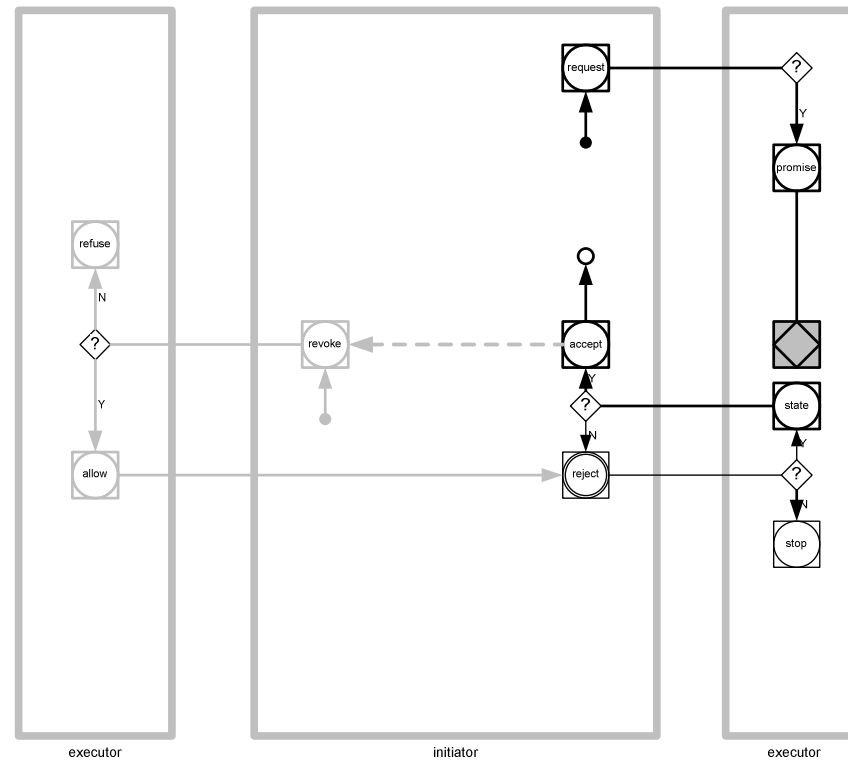
Transaction T03 Care query approval R03 Care query CQ has been approved



Transaction T04 Emergency query handling R04 ECQ has been fulfilled

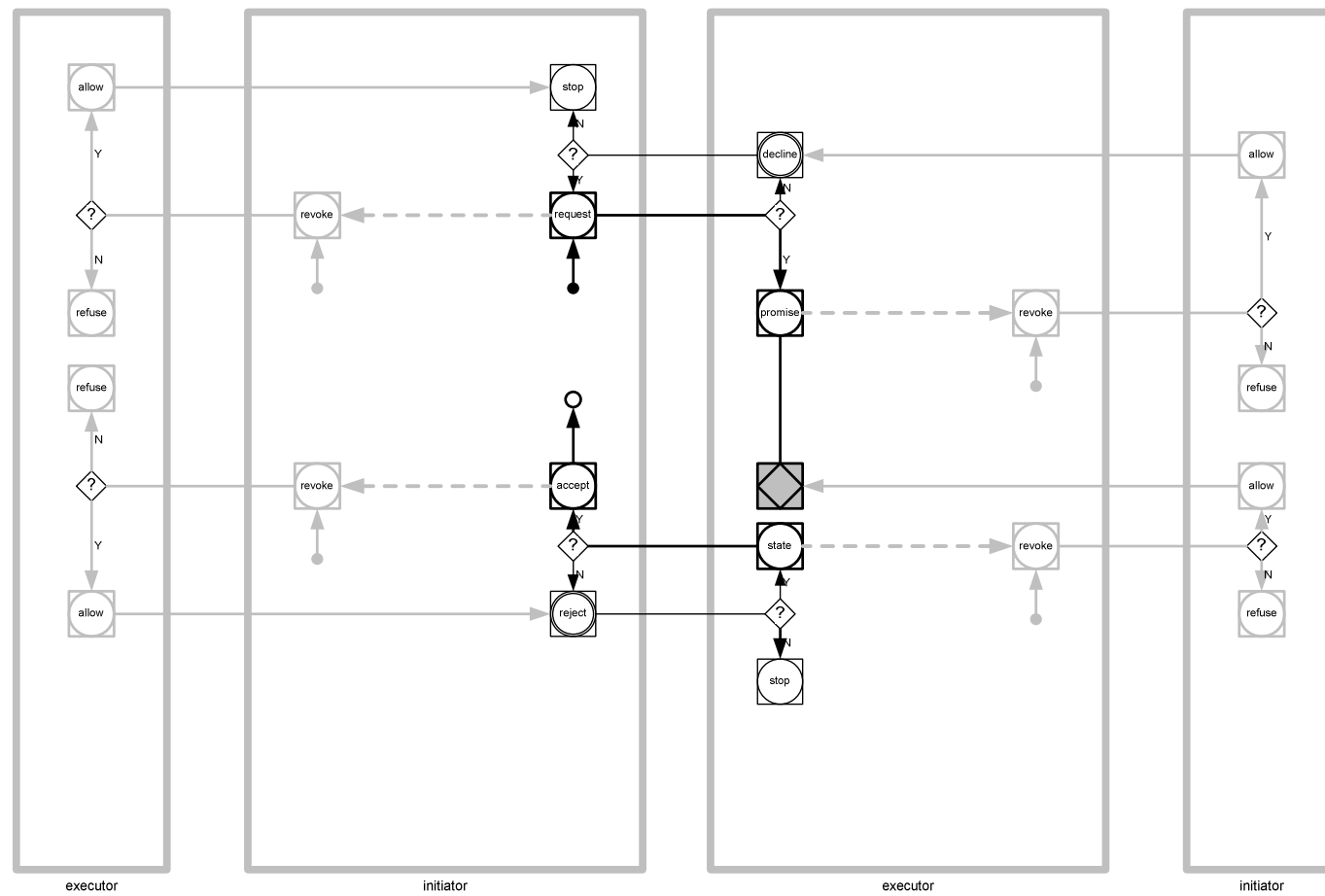


Transaction T05 Care delivery fulfillment R05 CQ has been fulfilled

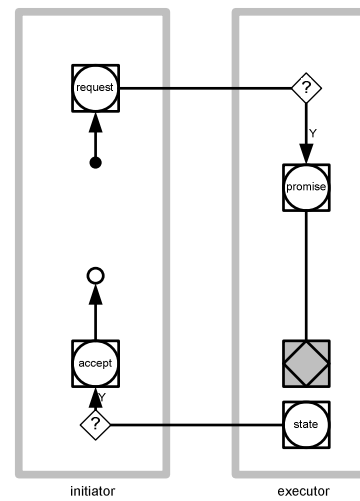


[illegible]

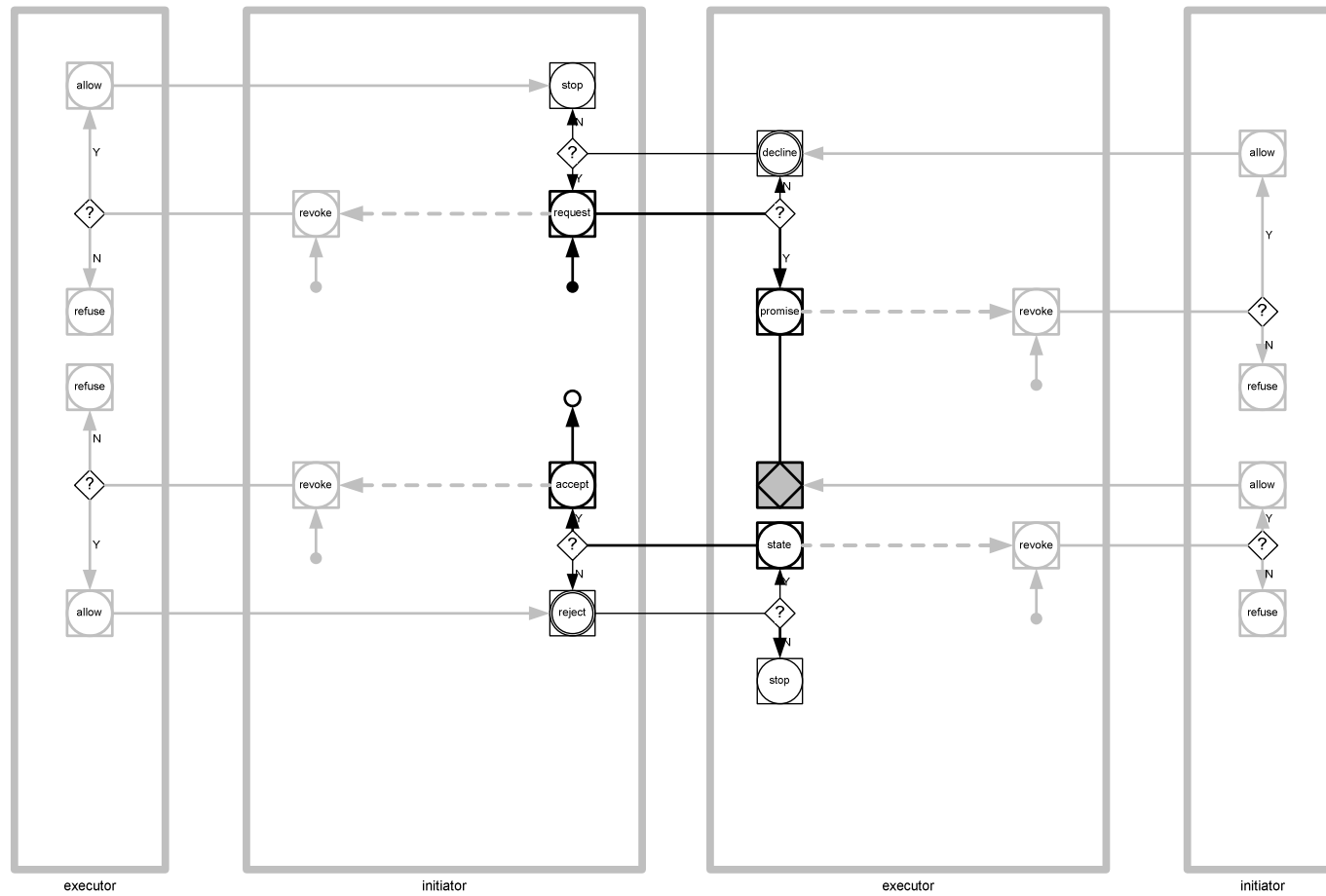
Transaction T07 Freelancer payment R07 Payment P has been paid



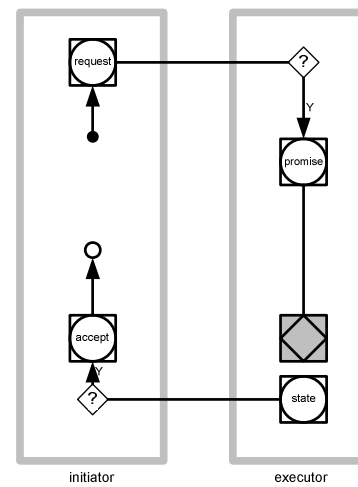
Transaction T08 Invoice managing R08 Invoice management for period P has been done



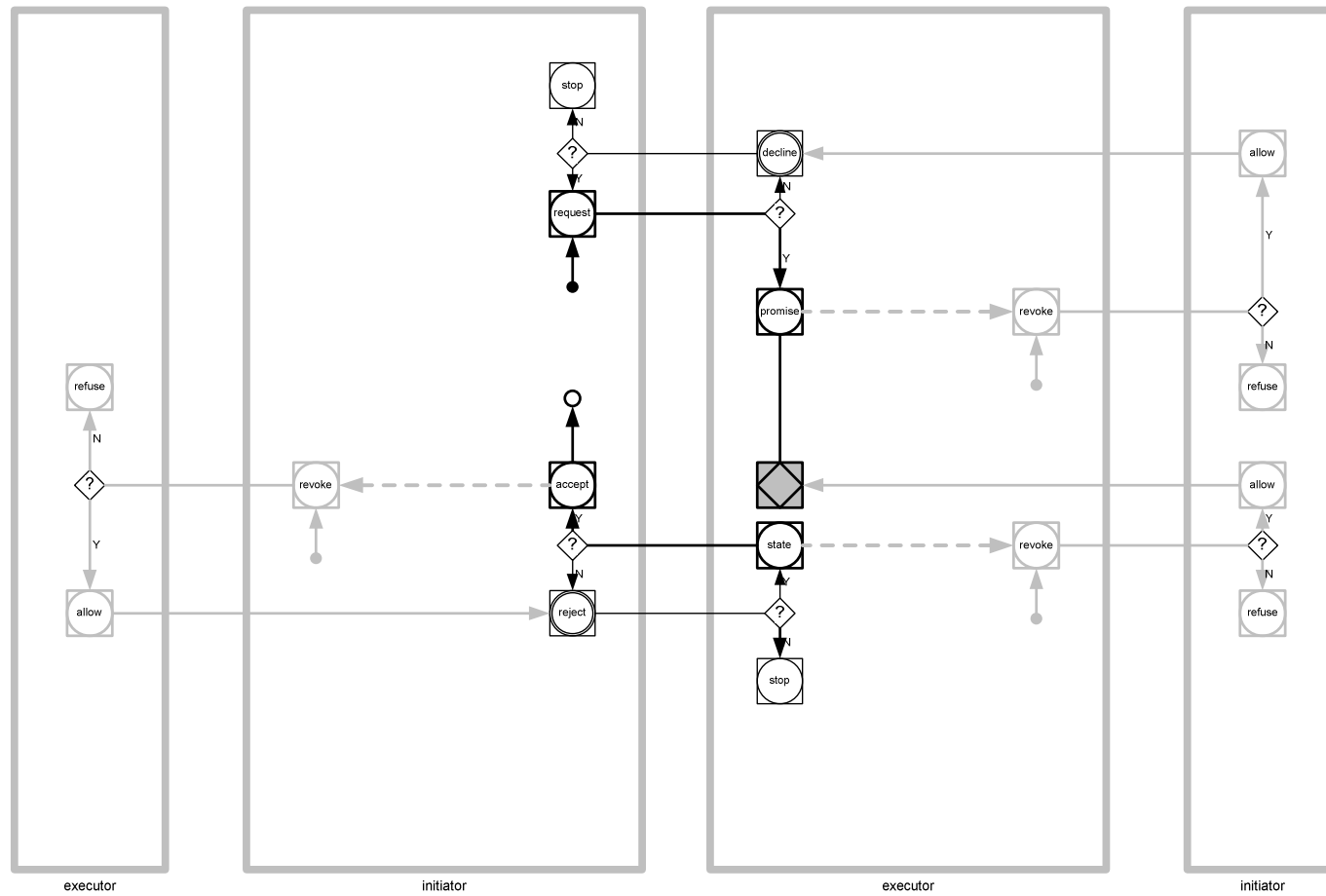
Transaction T09 Invoice payment R09 Payment P has been paid



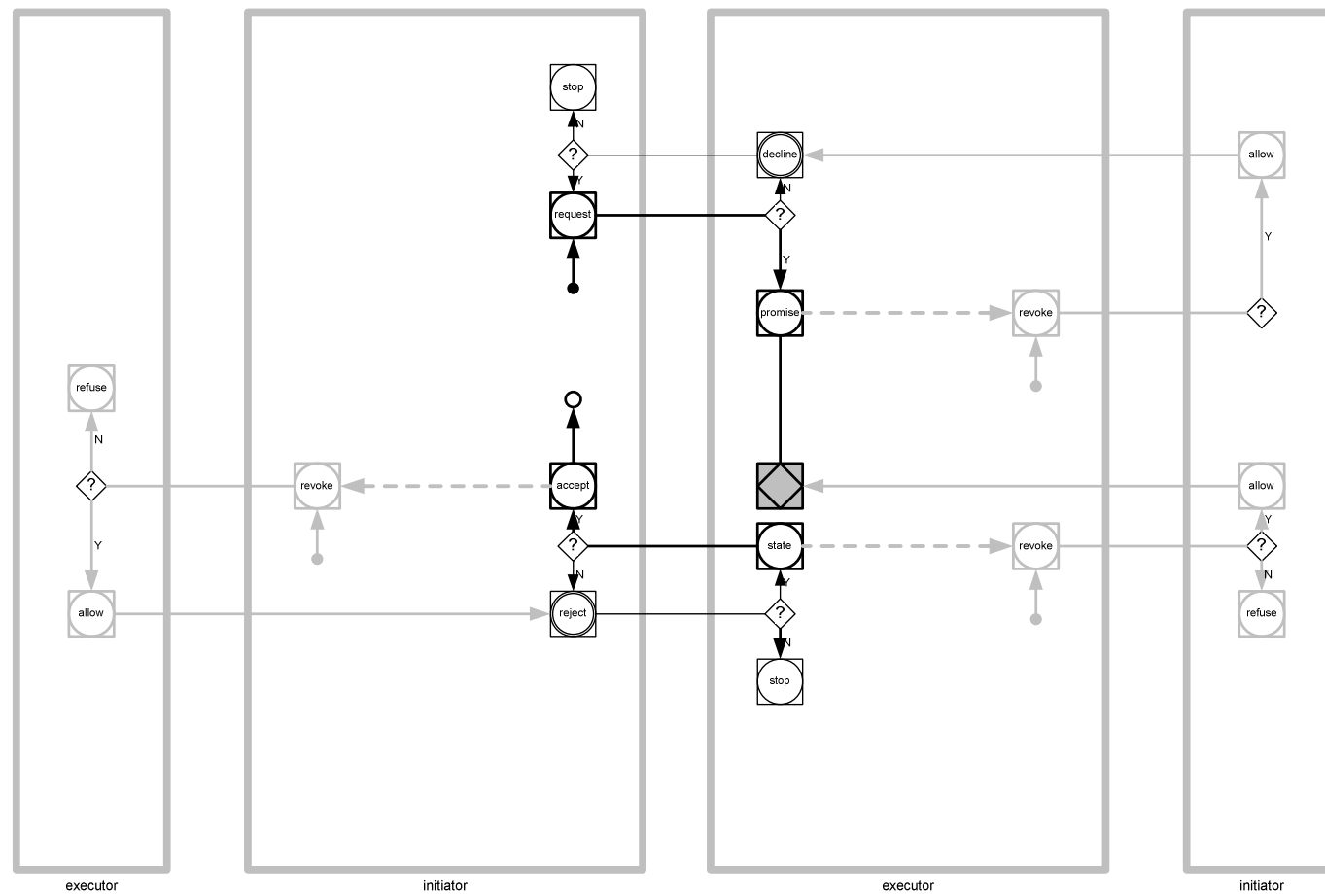
Transaction T10 Schedules management R10 Schedule management for period P has been done



Transaction T11 Care scheduling R11 Care C has been scheduled



Transaction T12 Freelancer scheduling R12 Care assignment CA has been done



Appendix D Testing Scripts CEPO-way (in Dutch)

Acceptatietest Leveren Thuiszorg					
T11 - Zorgvraag roosteren					
Onderdeel-Testitem	😊	😐	😞	💣	
Dossier en zorgvraag aanmaken					
010 - 010 - a. Dossier aanmaken - Open cliënt A. Maak één dossier (AD1) aan van type PGB. Nieuw dossier (AD1) is aangemaakt.					
010 - 020 - b. Zorgvraag aanmaken, toevoegen artikelen - Maak een zorgvraag aan van zorgtype 'Welzijn / Ondersteunende begeleiding'. Vul het aantal uren zorg voor de cliënt per week in. Sluit af met vinkje. De zorgvraag voor dossier (AD1) is aangemaakt.					
010 - 030 - c. Verdeel zorgvraag - Open zorgvraag die zojuist is aangemaakt. Verdeel de uren over de week inclusief condities (tijdstippen) en sla op met diskette.					
010 - 040 - d. Controleren zorgvraag - Controleer bij Zorgvraag – Onderdelen of de 'onderdelen' voor de komende periode zijn aangemaakt.					
Testscenario 2 - overige dossiers					
020 - 010 - e. Dossier aanmaken - Open de andere cliënt (B). Maak een dossier (BD1) aan van het type PGB met startdatum 1 maand geleden en geen einddatum. Maak een tweede dossier (BD2)aan met startdatum huidige datum + 2 dagen en geen einddatum. Twee nieuwe dossiers (BD1) en (BD2) zijn aangemaakt.					
020 - 020 - f. Voer stappen b t/m d (zorgvraag aanmaken, toevoegen artikelen t/m controleren zorgvraag) opnieuw uit voor cliënt B, voor beide dossiers.					
Testscenario 3 - overige dossiers					
030 - 010 - g. Dossiers aanmaken - Open de andere cliënt (C). Maak één dossier (CD1) aan van type PGB met startdatum 1 maand geleden en einddatum over 1 maand. Maak tweede dossier (CD2) aan met startdatum vandaag en geen einddatum. Twee nieuwe dossiers zijn aangemaakt.					
Testscenario 4 - overige dossiers					
040 - 010 - h. Zorgvraag aanmaken dossier, toevoegen artikelen - Maak voor cliënt C, binnen dossier CD1 een zorgvraag aan van zorgtype 'gespecialiseerde zorg (slaap)'. Vul voor begindatum dezelfde datum in als startdatum dossier. Vul geen einddatum in. Vul het aantal uren zorg voor de cliënt per week in. Sluit af met vinkje. De zorgvraag voor dossier CD1 is aangemaakt.					
040 - 020 - i. Voer stappen b t/m d (Verdeel zorgvraag t/m controleren zorgvraag) opnieuw uit voor cliënt C en zorgvraag binnen dossier CD1.					

040 - 030 - j. Zorgvraag aanmaken dossier 2, toevoegen artikelen - Maak voor cliënt C, binnen dossier CD2, twee zorgvraag aan. Zorgvraag 1 van het zorgtype 'Welzijn / Ondersteunende begeleiding', zorgvraag 2 van het type 'gespecialiseerde zorg'. Vul voor begindatums van beide zorgvragen dezelfde datum in als startdatum dossier. Vul geen einddatum in. Vul het aantal uren zorg voor de cliënt per week in. Sluit af met vinkje. De twee zorgvragen voor dossier CD2 zijn aangemaakt.					
040 - 040 - k. Voer stappen b t/m d (zorgvraag aanmaken, toevoegen artikelen t/m controleren zorgvraag) opnieuw uit voor cliënt C, voor dossier CD2 en beide zorgvragen.					
Testscenario 5 - overige dossiers					
050 - 010 - l. Dossier aanmaken - Open cliënt D. Maak één dossier DD1 aan van type PGB. Nieuw dossier is aangemaakt.					
050 - 020 - m. Zorgvraag aanmaken dossier 2, toevoegen artikelen - Maak voor cliënt D, binnen dossier DD1, twee zorgvraag aan. Zorgvraag Z1 van het zorgtype 'Welzijn / Ondersteunende begeleiding', zorgvraag Z2 van het type 'gespecialiseerde zorg'. Vul voor begindatums Z1 en Z2 dezelfde datum in als startdatum dossier. Vul geen einddatum in. Vul het aantal uren zorg per week in voor de cliënt. Sluit af met vinkje. De twee zorgvragen voor dossier DD1 zijn aangemaakt.					
050 - 030 - n. Voer stappen b t/m d (Verdeel zorgvraag t/m controleren zorgvraag) uit voor cliënt D, voor dossier D1 en beide zorgvragen.					

Acceptatietest Leveren Thuiszorg					
T12 - Inplannen ZZP'ers (vastleggen beschikbaarheid, zorgvraag, planning)					
Onderdeel-Testitem	😊	😐	😞	💣	
Beschikbaarheid ZZP'ers vastleggen					
010 - 010 - a. Vastleggen beschikbaarheid ZZP'ers - Open het dienstrooster beschikbaarheid van de ZZP'ers. Registreer dan voor de ZZP'er zijn / haar beschikbaarheid in de vorm van aanwezigheid en afwezigheid. Sluit af met diskette.					
010 - 020 - Herhaal instructie 010 - 010 voor 3 andere ZZP'ers.					
010 - 030 - Beschikbaarheid ZZP'ers controleren - Open het rooster Thuiszorg (wk) – V&V. Controleer of de beschikbaarheid van de ZZP'ers correct wordt weergegeven.					
010 - 040 - Zorgvraag controleren en restvraag plaatsen - Controleer in de restvraag of de zorgvragen worden weergegeven om gepland te worden.					
010 - 050 - Restvraag plaatsen - Plan alle restvragen van de cliënten. Controleer of het systeem wensconflicten juist heeft gesignaleerd.					
010 - 060 - Controleer rooster van de cliënt - Ga naar Dossier – Cliënt en open een van de zojuist ingeplande cliënten. Ga naar Dossiers, bewerk en ga dan naar Afspraken. Klik op 'Print deze maand' en 'Print vlgnd maand' en controleer de overzichten.					
010 - 070 - Controleer rooster van de ZZP'er - Ga naar P&O – Personeel en open de zojuist ingeplande ZZP'er. Ga naar Rooster. Klik op 'Print deze maand' en 'Print vlgnd maand' en controleer de overzichten.					
010 - 080 - Herhaal instructie 010 - 070 voor elke ingeplande ZZP'er.					
Testscenario 6 - Beëindigen zorgvraag					
020 - 010 - Beëindigen zorgvraag - Open Dossier BD1 van cliënt B – en ga naar de Zorgvraag. Klik op knop 'Bewerken'. Vul veld einddatum in met de datum vandaag en klik op knop opslaan.					
020 - 020 - Controleren beëindiging zorgvraag - Controleer voor dossier BD1 of de resterende zorgonderdelen juist staan ingepland. Controleer of de geplande zorgonderdelen na de einddatum zorgvraag, correct zijn verwijderd.					
Testscenario 7 - Beëindigen dossier					
030 - 010 - Beëindigen dossier - Open Dossier BD2 van cliënt B. Ga naar onderdeel Beëindigen. Vul de reden beëindigen in en bij stopdatum de datum over 1 maand. Een melding wordt gegeven om de gegevens op te slaan en geplande zorgvragen af te sluiten.					

030 - 020 - Klik op knop opslaan. Een melding wordt gegeven dat het dossier per ingevoerde einddatum niet meer actief is, en het aantal zorgvraagonderdelen dat daardoor verwijderd is.					
030 - 030 - Klik op knop 'Annuleren afspraken'. Een melding wordt gegeven dat alle afspraken vanaf de beëindigingdatum worden verwijderd.					
030 - 040 - Controleren beëindigd dossier - Controleer voor dossier BD2 of de zorgvraag een juiste einddatum heeft.					
030 - 050 - Controleer voor dossier BD2 of de resterende zorgonderdelen juist staan ingepland.					
030 - 060 - Controleer of de geplande zorgonderdelen na de einddatum zorgvraag, correct zijn verwijderd.					
Testscenario 8 - Beëindigen zorg					
040 - 010 - Beëindigen zorg - 1. Ga naar Dossier – Client – Openen en open cliënt A. Ga naar onderdeel 'Einde zorg'. Vul reden einde zorg in en bij datum einde zorg de datum over 1 maand. Het systeem maakt melding dat de client per datum over 1 maand uit zorg is gemeld.					
040 - 020 - Controleren Beëindiging zorg - Klik op knop OK. Controleer of het systeem ook de einddatum heeft doorgevoerd voor het dossier AD1, en de daaronder vallende zorgvraag. Controleer of de geplande zorgonderdelen na de einddatum zorgvraag, correct zijn verwijderd.					

Acceptatietest Leveren Thuiszorg					
T04 - Acute zorg leveren					
Onderdeel-Testitem	😊	😐	😞	💣	
Acute levering					
010 - 010 - ZZP'er selecteren - Ga naar P&O – Controleer of er een ZZP'er is die de tijd en beschikbaarheid heeft om acute zorg te leveren. Boek een ZZP'er die de juiste functie heeft met de knop 'Inboeken'.					
010 - 020 - Controleer boeking - Open de zojuist geboekte ZZP'er via Personeel – Openen – Gewerkte uren. Controleer of de acute zorg tussen de gewerkte uren staat. Klik op knop 'Print' en controleer het overzicht.					

Acceptatietest Leveren Thuiszorg					
T07 - T08 - T09 - Factureringsperiode, aanmaken facturen					
Onderdeel-Testitem	😊	😐	😞	💣	
Afhandelen zorg					
010 - 010 - Zorg afhandelen / controleren geleverde zorg - Ga naar Rooster – Afspraken – Afhandelen. Handel voor cliënt A de geleverde afspraken af over de geboekte periode.					
010 - 020 - Pas met knop 'Bewerken' de afspraakdetails start- en einddatum, tijd en memo aan. Sluit af met vinkje.					
010 - 030 - Selecteer de zojuist bewerkte afspraak en klik op knop 'Afsprk. afhandelen'. Handel de hele lijst af door op knop 'Lijst afhandelen' te klikken.					
010 - 040 - Herhaal handelingen 010 - 010 t/m 010 - 030 voor de overige cliënten (B t/m D). Alle afspraken zijn gekoppeld aan de medewerker.					
Herstellen en bewerken afspraken					
020 - 010 - Herstellen afspraken - Herstel enkele afspraken. Voer bewerkingen door onder 'afhandelen' en handel afspraak opnieuw af.					
020 - 020 - Bewerken afspraken - Voer direct een bewerking uit op een afspraak, zonder deze te herstellen.					
Aanmaken facturen (proef)					
030 - 010 - Aanmaken facturen (proef) - Open Financieel – Facturatie – Proef. Controleer op de juiste datum selectie vanaf tot en met.					
030 - 020 - Ga naar tab 'cliënt'. Selecteer cliënt A en klik op knop 'genereren'. Het deelscherm 'Te factureren leveringen (cliënt)' wordt getoond. De proef facturen zijn gegenereerd.					
030 - 030 - Voer bovenstaande stappen 030 - 010 t/m 030 - 020 opnieuw uit voor cliënten B t/m D.					
Controleren proeffacturen					
040 - 010 - Controleren proeffacturen - Controleer voor Client A of de drie (proef)facturen correct zijn aangemaakt: de verzamelfactuur, de bemiddelingsfactuur en de ZZP-factuur.					
040 - 020 - Controleer alle drie de facturen op volledigheid en correctheid.					
040 - 030 - Voer dezelfde factuurcontrole uit voor cliënten B t/m D. Voor alle cliënten ongeacht 1 of meer dossiers of 1 of meer zorgvragen per dossier, zijn de factuurgegevens correct aangemaakt.					
Aanmaken echte facturen					
050 - 010 - Aanmaken echte facturen - Herhaal de facturatie stappen voor aanmaken facturen en controleren facturen binnen onderdeel Facturatie - tab 'Echt'.					

050 - 020 - Bekijken facturen - Ga naar onderdeel Financieel - Bekijken - Facturen. Print factuurgegevens uit met de knop 'Printen'.					
050 - 030 - Controleer alle factuurgegevens op het scherm en op de printafdruk op juistheid.					
050 - 040 - Betaling inboeken - Ga naar onderdeel Financieel - Betalingen - Inboeken. Selecteer een factuur en klik op knop bewerken. Wijzig een factuurgegeven door het handmatig inboeken van een betaling (datum, tijd en bedrag) en klik op knop OK. De factuurgegevens zijn aangepast door toevoeging van een handmatige inboeking.					
Afdrukken facturen					
060 - 010 - Afdrukken facturen - Ga naar onderdeel Financieel - Afdrukken - Facturen. Vul de juiste gegevens in voor afzender, startdatum en einddatum. Klik op knop toon. Factuur worden getoond om af te drukken. Klik op knop 'Print lijst'. Controleer de facturen en factuurgegevens op juistheid.					

Appendix E Testing Scripts DEMO-way (in Dutch)

Acceptatietest Leveren Thuiszorg					
Cliënt aanmaken en zorg verdelen 1					
Onderdeel-Testitem	😊	😐	😞	💣	
Cliënt aanmaken en zorg verdelen 1					
010 - 010 - Een persoon met een zorgprobleem en een AWBZ-indicatie met startdatum begin van het jaar en einddatum eind van het jaar voor functie 'Activerende begeleiding' van klasse 2, dus 2 - 3,9 uur per week. De persoon vraagt of WSD hem / haar kan helpen met het zorgprobleem. WSD onderzoekt het zorgprobleem en geeft aan dat het in behandeling wordt genomen. Er wordt per direct zorg geleverd. Maak via module Dossier - Cliënt - Toevoegen een cliënt aan met de naw gegevens. Vink ook Signalering Memo aan. De cliënt is aangemaakt.					
010 - 020 - Binnen WSD wordt er bepaald wat het zorgprobleem is. Het CIZ indicatie bedrag, CIZ indicatie startdatum en de CIZ indicatie einddatum moet worden ingevoerd bij de cliënt. Sla dit op in Tabblad Memo.					
010 - 030 - Het zorgprobleem dat is vastgesteld moet nu in een Dossier verwerkt worden in Care Arrangement. Open de gegevens van de cliënt via Dossier - Cliënt - Openen. De cliënt gegevens worden nu getoond.					
010 - 040 - WSD gaat het zorgprobleem oplossen. Hiervoor moet een dossier worden aangemaakt. Maak een dossier van type PGB aan via module Dossier - Dossier - Toevoegen. Controleer of er een melding wordt gegeven dat er een Memo voor deze cliënt is. Sla op met diskette. Het dossier is toegevoegd.					
010 - 050 - Om het zorgprobleem op te lossen, moet er een zorgvraag worden aangemaakt in een dossier. Ga naar tabblad Zorgvraag. Maak een zorgvraag aan via knop Toevoegen met startdatum 'vandaag', einddatum leeg laten, zorgniveau 'Reguliere zorg' en vul 3 uur per week in. Sla op met vinkje. De zorgvraag is aangemaakt.					
010 - 060 - De zorgvraag wordt voorgelegd aan de cliënt (of de familie van de cliënt) door middel van een offerte en er wordt om acceptatie gevraagd. De cliënt (of de familie van de cliënt) kijkt naar de zorgvraag. De zorgvraag om het zorgprobleem op te lossen wordt goed bevonden en er wordt aangegeven dat de cliënt de zorg op de dinsdag van 1100 - 1400 wil. Open via tabblad Zorgvraag, knop Bewerken de zorgvraag. Ga naar tabblad Verdeling en vul 3 uur in bij dinsdag met conditie om 1100. Sla op met diskette. Klik op Herbereken onderdelen. De uren zijn verdeeld.					
010 - 070 - Ga naar tabblad Onderdelen en controleer of de zorg is aangemaakt en of de juiste starttijd en eindtijd erin zijn verwerkt. Sla op met vinkje.					

Acceptatietest Leveren Thuiszorg					
Personeel aanmaken en inplannen van zorg 1					
Onderdeel-Testitem	😊	😐	😞	💣	
Personeel aanmaken en inplannen van zorg 1					
010 - 010 - Ga naar module P&O - Personeel - Toevoegen, vul eerst de naw gegevens in en ga daarna naar Tabblad HR om via de knop Toevoegen de functie aan te maken met zorgniveau 'verzorgende', startdatum 'begin dit jaar' en einddatum onbekend. Herhaal dit drie keer. Er zijn dan 3 ZZPers aangemaakt.					
010 - 020 - Ga naar module P&O - Dienstrooster - Standaardweek en vul voor alle 3 de ZZPers 24 uur 7 dagen per week de beschikbaarheid. Klik 1 keer op WO en dan op elke dag van de week. Sla op met vinkje. De standaardbeschikbaarheid is ingevuld.					
010 - 030 - Ga naar module P&O - Dienstrooster - Beschikbaarheid en klik bij ZZPers X, Y en Z op S (vul standaardweek in). Sla op met vinkje. De beschikbaarheid is ingevuld.					
010 - 040 - Vul in module Dossier - Cliënt - Openen tabblad Voorkeur een voorkeur in voor twee van de zojuist aangemaakt ZZPers. Vul voor een van de ZZPers een afkeur in. Sla op met vinkje. De voorkeuren zijn nu aangemaakt.					
010 - 050 - Open de module Rooster - Thuiszorg (wk) - V&V. Zet het rooster dusdanig neer dat het de volgende maand weergeeft. Controleer via Configuratie Opties Functie of er ZZPers met de juiste functie (verzorgende) beschikbaar zijn voor de komende maand.					
010 - 060 - Controleer in de restvraag of de zorg voor van de cliënt wordt getoond.					
010 - 070 - Toon eerst alle zorgonderdelen voor de betreffende cliënt door te sorteren op de naam van de cliënt. Koppel dan vanuit de restvraag voor de komende maand de zorg van de cliënt aan de geschikte ZZPers (oftewel: plan de zorg in). Sla op met diskette. De zorg is nu gekoppeld aan een ZZPer.					
010 - 080 - Ga naar Dossier - Cliënt - Openen, tabblad Rooster en controleer of de zorg naar wens van de cliënt ingepland is.					
010 - 090 - De ZZPer heeft een rooster nodig om te weten wanneer de zorg geleverd moet worden. Open via module P&O - Personeel - Openen de ingeplande ZZPer voor de cliënt en ga naar tabblad Rooster. Klik op de knop Print deze maand en stuur het rooster via e-mail op naar de ZZPer. Nu is de ZZPer verzocht om de ingeplande zorg te leveren.					

Acceptatietest Leveren Thuiszorg					
Factureren 1					
Onderdeel-Testitem	😊	😐	😞	💣	
Factureren 1					
010 - 010 - Ervanuitgaande dat de ZZPer de zorg heeft geleverd, worden nu afspraken afgehandeld, niet degene die zojuist zijn ingeroosterd maar willekeurige die nog in het systeem zitten. Handel ingeplande afspraken af via module Rooster - Afhandelen. Vul de periode in, maar laat het veld van cliënt en medewerker leeg. Handel nu 2 willekeurige afspraken af.					
010 - 020 - Er blijkt een afspraak niet volgens rooster verlopen te zijn, ga naar module Rooster - Afspraken - Herstellen. Herstel een afspraak die gecorrigeerd moet worden.					
010 - 030 - Ga naar module Rooster - Afspraken - Afhandelen en selecteer de zojuist herstelde afspraak en klik op Bewerken. Pas de tijden aan en handel dan deze afspraak af.					
010 - 040 - De factureringsperiode is angebroken. Open via module Financieel - Facturatie - Proef en genereer de facturen voor de cliënt en de ZZPer.					
010 - 050 - De leveringen op de factuur komen niet overeen met de leveringen die zijn geweest. Ga naar module Rooster - Afspraken - Herstellen en herstel enkele leveringen. Maak de aanpassingen via de knop bewerken en handel deze afspraken opnieuw af.					
010 - 060 - De factureringsperiode is angebroken. Open via module Financieel - Facturatie - Proef en genereer de facturen voor de cliënt en de ZZPer.					
010 - 070 - Bekijk de gemaakte facturen en controleer of de uren, het totaalbedrag en de data correct zijn voor cliëntfactuur					
010 - 080 - Bekijk de gemaakte facturen en controleer of de uren, het totaalbedrag en de data correct zijn voor ZZP-factuur					
010 - 090 - Bekijk de gemaakte facturen en controleer of de uren, het totaalbedrag en de data correct zijn voor WSD-ZZP-factuur					
010 - 100 - Genereer nu opnieuw facturen via module Financieel - Facturatie - Echt. Controleer of de facturen goed zijn aangemaakt.					
010 - 110 - Druk de facturen van WSD-ZZP af. Stuur de facturen via email naar de ZZPer.					
010 - 120 - Stuur de cliëntfacturen via email op.					
010 - 130 - Stuur de ZZP-facturen via email op.					

010 - 140 - De betaling van de cliënt is nog niet binnen dus dit proces is nog niet afgesloten. Verwerk in Care Arrangement via module Financieel dat deze betaling nog niet voldaan is.					

Acceptatietest Leveren Thuiszorg					
Acute zorg 1					
Onderdeel-Testitem	😊	😐	😞	💣	
Acute zorg 1					
010 - 010 - De familie van de cliënt belt WSD op en meldt dat de cliënt uit de rolstoel is gevallen. WSD belooft dat er een ZZPer naar toe wordt gestuurd. Zoek de gegevens van een beschikbare en geschikte (geschikt op 2 manieren: zowel zorgniveau als voorkeur) ZZPer, bel deze op en stuur de ZZPer naar de cliënt. Een geschikte ZZPer is uitgezocht.					
010 - 020 - Boek de zorg in via Dossier - Direct boeken - Verleende thuiszorg vandaag voor 1 uur. De acute zorg is ingeboekt.					
010 - 030 - Ga naar module Financieel - Facturatie - Proef en controleer of de acute zorg ook is verwerkt					

Acceptatietest Leveren Thuiszorg					
Clïënt uit zorg 1					
Onderdeel-Testitem	😊	😐	😞	💣	
Clïënt uit zorg 1					
010 - 010 - De cliënt is ontevreden en wil de zorg beëindigen. Ga naar module Dossier - Dossier - Zorgvraag en verwijder de zorgvraag. De zorgvraag is verwijderd.					
010 - 020 - Ga naar module Dossier - Dossier - Zorgvraag tabblad Beëindigen en klik op Annuleer afspraken. De afspraken voor de cliënt zijn nu geannuleerd.					
010 - 030 - Controleer via module P&O - Personeel - Openen of deze planning ook niet meer bij de ZZPers te vinden is.					

Acceptatietest Leveren Thuiszorg					
Cliënt aanmaken en zorg verdelen 2					
Onderdeel-Testitem	😊	😐	😞	💣	
Cliënt aanmaken en zorg verdelen 2					
010 - 010 - Een persoon met een zorgprobleem en een AWBZ-indicatie met startdatum begin van het jaar en einddatum eind van het jaar meldt zich bij WSD, de persoon heeft voor alle functies die PGB gerelateerd zijn budget, een zogenaamde 24uurs-cliënt. WSD onderzoekt het zorgprobleem en geeft aan dat het in behandeling wordt genomen. Hetzij meteen, hetzij binnen een aantal dagen. Maak via module Dossier - Toevoegen - Cliënt een cliënt aan met de naw gegevens. Vink ook Signalering Memo aan. De cliënt is aangemaakt.					
010 - 020 - Binnen WSD wordt er bepaald wat het zorgprobleem is. Het CIZ indicatie bedrag, CIZ indicatie startdatum en de CIZ indicatie einddatum moet worden ingevoerd bij de cliënt. Sla dit op in Tabblad Memo.					
010 - 030 - Het zorgprobleem dat is vastgesteld moet nu verwerkt worden in Care Arrangement. Open de gegevens van de cliënt via Dossier - Cliënt. De cliënt gegevens worden nu verkeerd getoond.					
010 - 040 - De verkeerde naw gegevens van de cliënt zijn ingevoerd. In plaats van het aanpassen van alle gegevens, verwijder de cliënt. Voeg daarna een nieuwe cliënt toe. Vink ook Signalering Memo aan en plaats in de Memo de CIZ indicatie gegevens: startdatum en einddatum.					
010 - 050 - Overleg binnen WSD bepaalt of het zorgprobleem wel door WSD opgelost kan worden. Maak een dossier aan via module Dossier - Dossier - Toevoegen. Controleer er een melding wordt gegeven van het memo.					
010 - 060 - Om het zorgprobleem op te lossen, wordt er een zorgvraag aangemaakt in een Dossier. Ga naar tabblad Zorgvraag. Maak een zorgvraag aan met startdatum begin van het jaar, einddatum eind van het jaar, zorgniveau Reguliere zorg per etmaal en vul 10080 minuten per week in. Maak op dezelfde manier nog een zorgvraag aan met zorgniveau verpleging met startdatum begin volgend jaar en einddatum eind volgend jaar voor 60 minuten per week. De zorgvragen zijn aangemaakt.					
010 - 070 - Het was niet de bedoeling om twee zorgvragen van verschillende zorgniveaus aan te maken, dus bewerk de zorgvraag verpleging en verander naar Reguliere zorg per etmaal.					
010 - 080 - Het was eigenlijk ook niet nodig om voor twee jaar twee zorgvragen aan te maken, dus verwijder 1 van de 2 en maak 1 zorgvraag van begin dit jaar naar eind volgend jaar met zorgniveau Reguliere zorg per etmaal.					

010 - 090 - De zorgvraag wordt voorgelegd aan de cliënt (of de familie van de cliënt) en er wordt om acceptatie gevraagd. De cliënt (of de familie van de cliënt) kijkt naar de zorgvraag. De zorgvraag om het zorgprobleem op te lossen wordt goed bevonden en er wordt aangegeven dat de cliënt de 24uurs zorg wil. Open via tabblad Zorgvraag, de zorgvraag om deze te bewerken. Controleer of de gegevens van de zorgvraag correct zijn ingevuld.					
010 - 100 - Ga naar tabblad Verdeling en vul de gewenste tijd per dag in (1440 minuten) met starttijd 1200 's middags. Sla op met diskette. De uren zijn verdeeld.					
010 - 110 - Ga naar tabblad Onderdelen en controleer of de zorg is aangemaakt en of de juiste starttijd erin is verwerkt. Sla op met vinkje.					

Acceptatietest Leveren Thuiszorg					
Personeel aanmaken en plannen van zorg 2					
Onderdeel-Testitem	😊	😐	😞	💣	
Personeel aanmaken en inplannen van zorg 2					
010 - 010 - Ga naar module P&O en maak 3 ZZPers aan: ZZPer A met functie (zorgniveau) verpleegkundige, ZZPers B en C met functie (zorgniveau) verzorgende. Allemaal met startdatum vandaag en einddatum onbekend. Er zijn 3 ZZPers aangemaakt.					
010 - 020 - Ga naar module P&O - Personeel - Openen en dan naar tabblad HR. Bewerk de functie (het zorgniveau) en verander het van verpleegkundige naar verzorgende. De functie is veranderd. Sla op met vinkje.					
010 - 030 - Ga naar module P&O - Dienstrooster - Standaardweek en vul voor alle 3 de ZZPers 24 uur 7 dagen per week de beschikbaarheid. Klik 1 keer op WO en dan op elke dag van de week. Sla op met vinkje. De standaardbeschikbaarheid is ingevuld.					
010 - 040 - Ga naar module P&O - Dienstrooster - Beschikbaarheid en klik bij ZZPers X, Y en Z op S (vul standaardweek in). Sla op met vinkje. De beschikbaarheid is ingevuld.					
010 - 050 - Vul in module Dossier - Cliënt - Openen tabblad Voorkeur een voorkeur in voor twee van de zojuist aangemaakte ZZPers. Vul voor een van de ZZPers een afkeur in. Sla op met vinkje. De voorkeuren zijn nu aangemaakt.					
010 - 060 - Open de module Rooster - Thuiszorg (wk) - V&V. Zet het rooster dusdanig neer dat het de volgende maand weergeeft. Controleer via Configuratie Opties Functie of er ZZPers met de juiste functie (verzorgende) beschikbaar zijn voor de komende maand.					
010 - 070 - Controleer in de restvraag of de zorg voor van de cliënt wordt getoond.					
010 - 080 - Toon eerst alle zorgonderdelen voor de betreffende cliënt door te sorteren op de naam van de cliënt. Koppel dan vanuit de restvraag voor de vorige en komende maand de zorg van de cliënt aan de geschikte ZZPers (oftewel: plan de zorg in). Plan ook een zorgonderdeel bij de ZZPer waarvoor de cliënt een afkeur heeft. Sla op met diskette. De zorg is nu gekoppeld aan een ZZPer.					
010 - 090 - Controleer of er agendaconflicten zijn. De roostering die verkeerd is geplaatst wegens de voorkeur van de cliënt wordt hier weergegeven.					
010 - 100 - Pas de ingeplande zorg, waarop een agendaconflict ontstaan is, naar de wensen van de cliënt op basis van voorkeur, dag en tijd zoals eerder ingevuld. Sla op met diskette.					
010 - 110 - Koppel vanuit de restvraag voor de komende maand de zorg aan de geschikte ZZPers (plan de zorg in). Sla op met diskette.					

010 - 120 - Ga naar module Dossier - Cliënt - Openen, dan naar tabblad Rooster en controleer of de zorg naar wens van de cliënt ingepland is.					
010 - 130 - De ZZPer heeft een rooster nodig om te weten wanneer de zorg geleverd moet worden. Open via module P&O - Personeel de ingeplande ZZPer voor de cliënt en ga naar tabblad Rooster. Klik op de knop Print deze maand en stuur het rooster via e-mail op naar de ZZPer. Nu is de ZZPer verzocht om de ingeplande zorg te leveren.					
010 - 140 - Ervanuitgaande dat de ZZPer de zorg heeft geleverd, worden nu afspraken afgehandeld, niet degene die zojuist zijn ingeroosterd maar willekeurige die nog in het systeem zitten. Handel ingeplande afspraken af via module Rooster - Afhandelen. Vul de periode in, maar laat het veld van cliënt en medewerker leeg. Handel nu 2 willekeurige afspraken af.					
010 - 150 - Er blijkt een afspraak niet volgens rooster verlopen te zijn, omdat er een medewerker ziek werd. Ga naar module Rooster - Afspraken - Herstellen. Herstel de afspraak die gecorrigeerd moet worden.					
010 - 160 - Ga naar module Rooster - Afspraken - Afhandelen en selecteer de zojuist herstelde afspraak en klik op Bewerken. Pas niks aan, maar sla op met vinkje en herstel deze levering.					

Acceptatietest Leveren Thuiszorg					
Factureren 2					
Onderdeel-Testitem	😊	😐	😞	💣	
Factureren 2					
010 - 010 - De factureringsperiode is aangebroken. Open via module Financieel - Facturatie - Proef en genereer de facturen voor de cliënt en de ZZPer.					
010 - 020 - De leveringen op de factuur komen niet overeen met de leveringen die plaats hebben gevonden. Ga naar module Rooster - Afspraken - Herstellen en herstel enkele leveringen. Maak de aanpassingen via de knop bewerken en handel deze afspraken opnieuw af.					
010 - 030 - De factureringsperiode is aangebroken. Open via module Financieel - Facturatie - Proef en genereer de facturen voor de cliënt en de ZZPer.					
010 - 040 - Bekijk de gemaakte facturen en controleer of de uren, het totaalbedrag en de data correct zijn voor cliëntfactuur					
010 - 050 - Bekijk de gemaakte facturen en controleer of de uren, het totaalbedrag en de data correct zijn voor ZZP-factuur					
010 - 060 - Bekijk de gemaakte facturen en controleer of de uren, het totaalbedrag en de data correct zijn voor WSD-ZZP-factuur					
010 - 070 - Genereer nu opnieuw facturen via module Financieel - Facturatie - Echt. Controleer of de facturen goed zijn aangemaakt.					
010 - 080 - Druk de facturen van WSD-ZZP af. Stuur de facturen via email naar de ZZPer.					
010 - 090 - Stuur de cliëntfacturen via email op.					
010 - 100 - Stuur de ZZP-facturen via email op.					
010 - 110 - De betaling van de cliënt is nog niet binnen dus dit proces is nog niet afgesloten. Verwerk in Care Arrangement via module Financieel dat deze betaling nog niet voldaan is.					
010 - 120 - De factuur WSD-ZZP A is te hoog uitgevallen. Maak via module Financieel - Crediteren - Facturen een credit factuur van deze factuur. Een credit factuur is aangemaakt.					
010 - 130 - De familie van de cliënt belt WSD op en meldt dat de cliënt direct zorg nodig heeft. WSD belooft dat er een ZZPer naar toe wordt gestuurd. Zoek de gegevens van een beschikbare en geschikte (geschikt op 2 manieren: zowel zorgniveau als voorkeur) ZZPer, bel deze op en stuur de ZZPer naar de cliënt. Een geschikte ZZP'er is uitgezocht.					
010 - 140 - Boek de zorg in via Dossier - Direct boeken - Verleende thuiszorg vandaag voor 1 uur. De acute zorg is ingeboekt.					

010 - 150 - Ga naar module Financieel - Facturatie - Proef en controleer of de acute zorg ook is verwerkt					

Acceptatietest Leveren Thuiszorg					
Personeel uit dienst 2					
Onderdeel-Testitem	😊	😐	😞	💣	
Personeel uit dienst 2					
010 - 010 - Twee van de drie ZZPers mogen niet meer voor WSD werken, want ze hebben fraude gepleegd. Ga naar module P&O - Personeel - Openen en haal de ZZPers uit dienst bij tabblad HR. De ZZPer is niet meer in dienst.					
010 - 020 - Ga naar module Rooster - Thuiszorg (wk) - V&V en controleer of de uitstaande zorg voor de cliënt is terug geplaatst in de restvraag.					
010 - 030 - De cliënt kan niet geholpen worden wegens tekort aan personeel. Beëindig de zorg bij de cliënt via module Dossier - Cliënt - Openen tabblad Einde zorg. De cliënt is nu geen cliënt van WSD meer.					

Appendix F Client Report (in Dutch)

Functionele fitgap Care Arrangement

Wilhelmina Senioren Diensten

Datum : 16 mei 2008
Versie : 1.0
Auteur : Lauwerens Metz

Inhoudsopgave

Wilhelmina Senioren Diensten.....	111
Inhoudsopgave	112
Inleiding	113
Achtergrond.....	113
Opdrachtformulering	113
Bronnen	114
Aanpak.....	114
Verzamelen gegevens.....	115
Proces	115
Kwaliteitsfactoren en norm.....	116
Analyse	117
Conclusie	118
En nu verder... ..	121

Inleiding

Achtergrond

In eerste instantie heeft Wilhelmina Senioren Diensten intern software ontwikkeld. Door gebrek aan expertise is deze ontwikkeling echter niet op het gewenste niveau gekomen. Nadat de verantwoordelijke persoon het bedrijf heeft verlaten en er is besloten om de interne ICT-ontwikkeling te stoppen moest er voor een nieuwe ICT-oplossing worden gekozen. Dit heeft ertoe geleid dat er voor een proeftijd van zes maanden een standaardpakket is geselecteerd: Care Arrangement van Dataproof IT BV.

Opdrachtformulering

Het software pakket is momenteel nog niet in gebruik. Dit ligt met name aan het feit dat enkele bewerkingen nog niet uitgevoerd konden. De bewerkingen die volgens Dataproof IT BV wel met het pakket uitgevoerd kunnen worden, zijn uitvoerig getest en de resultaten worden in dit verslag weergegeven.

Bronnen

De volgende documenten zijn gebruikt:

- Functioneel Ontwerp
- Handleiding Care Arrangement
- Care Arrangement cursus documentatie 17-01-2008

Aanpak

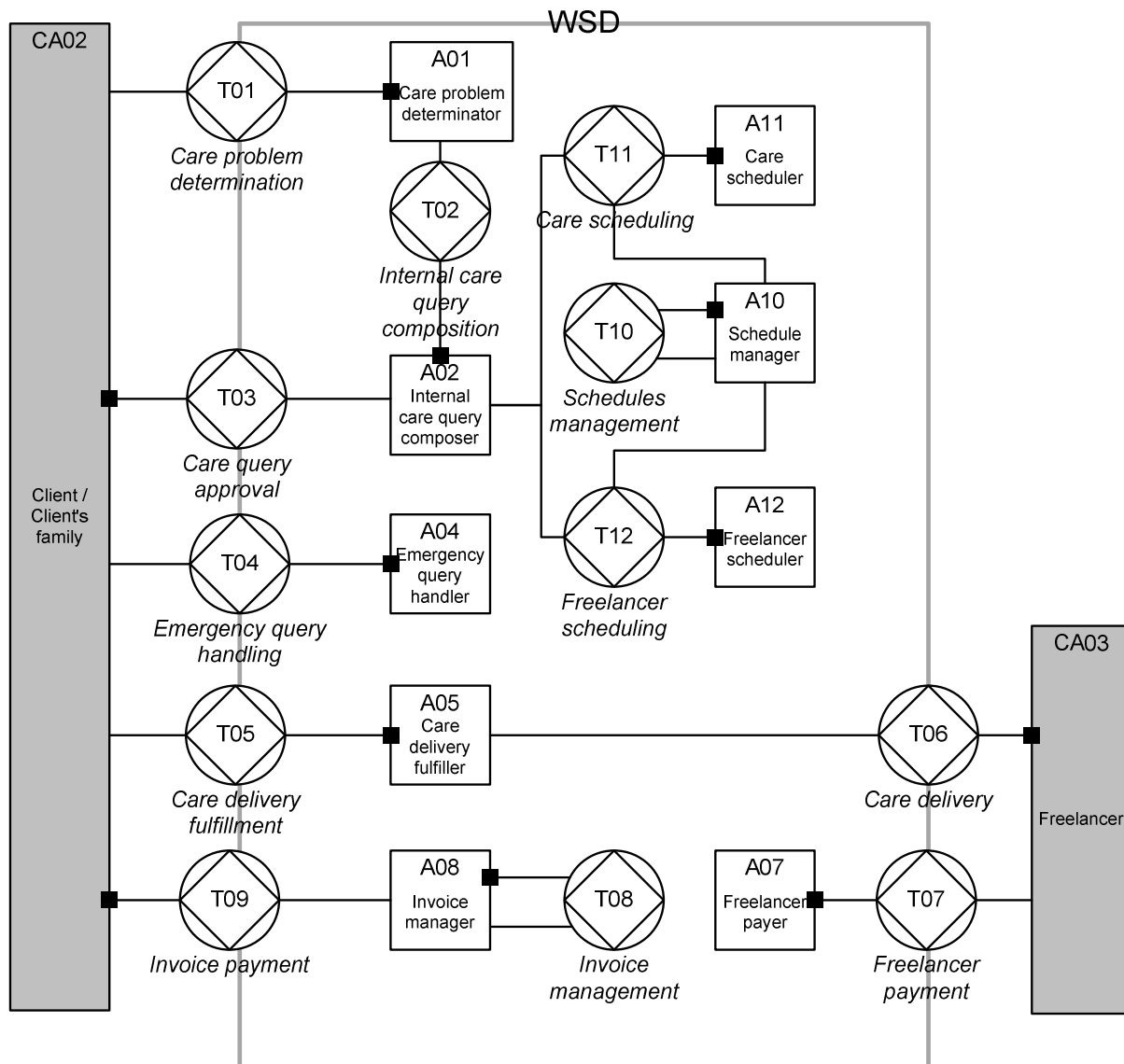
Deze analyse is tot stand gekomen met het onderstaande stappenplan:

1. Het verzamelen van de basisgegevens
 - a. Documentatie lezen
 - b. Interviews
 - c. Opstellen meetpunten
 - d. Opstellen kwaliteitsfactoren en norm
2. Het maken van de fitgap-analyse
 - a. Matchen meetpunten met kwaliteitsfactoren
 - b. Weging
 - c. Afzetten tegen norm
3. Conclusie opstellen

Verzamelen gegevens

Proces

Gebaseerd op de DEMO-methodologie is de organisatie vastgelegd, in figuur 1 is het model getoond dat vooraan deze methodologie staat.



Figuur 1 Wilhelmina Senioren Diensten in DEMO

De twee processen die als belangrijkste zijn aangewezen, zijn het plannen en de facturatie.

Kwaliteitsfactoren en norm

In de onderstaande tabel worden de kwaliteitsfactoren en acceptabele norm benoemd, welke in samenspraak met Wilhelmina Senioren Diensten tot stand zijn gekomen. Een acceptabele norm wil zeggen dat indien de kwaliteitsfactor na de analyse gelijk is aan of groter dan de norm de applicatie 'geslaagd' is.

Kwaliteitsfactor	Definitie	Norm
Betrouwbaarheid	De mate van vertrouwen van informatie van het systeem.	90%
Begrijpbaarheid	De mate van begrijpbaarheid en handelbaarheid (complexiteit) van het systeem.	80%
Gebruiksvriendelijkheid	De mate van handelingen en overzichtelijkheid van het gebruik.	80%
Operationele Functionaliteit	Toepasbaarheid systeem voor de ondersteuning van het proces (zonder facturatie).	100%
Financiële Functionaliteit	Toepasbaarheid systeem voor de ondersteuning van het facturatieproces.	100%
Traceerbaarheid / Correctiemogelijkheden	Het resultaat van het corrigeren van acties en het terugzien wat er heeft plaatsgevonden.	80%

Legenda score

Score in %	#	Beoordeling
00 – 25	☹☹	Zeer slecht
26 – 50	☹	Slecht
51 – 75	☺	Matig
76 – 100	☺☺	Goed

Analyse

De meetpunten uit de bijlagen A en B zijn gegroepeerd naar de kwaliteitsfactoren. Elk meetpunt heeft één kwaliteitsfactor, zodat elke meetpunt ook op één factor beoordeeld wordt. Het geheel is in samenspraak met Wilhelmina Senioren Diensten gewogen met behulp van onze smiley-methode.

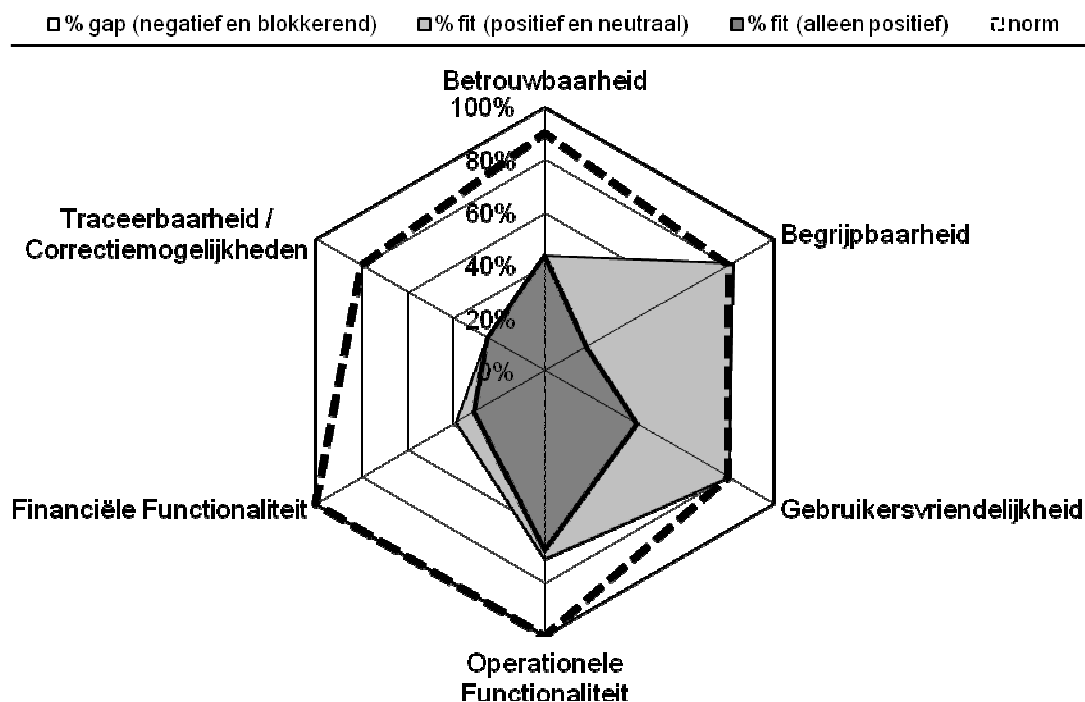
Weging	Omschrijving
😊	Goed; Passend
😐	Neutraal; Passend, maar kan beter
☹️	Fout; Niet passend
💣	Blokkerend; Niet passend en stagneert het proces

De resultaten van deze weging zijn weergegeven in de onderstaande tabellen. De getallen vertegenwoordigen het aantal meetpunten.

Kwaliteitsfactor	😊	😐	☹️	💣
Betrouwbaarheid	7	0	5	4
Begrijpbaarheid	2	7	2	0
Gebruiksvriendelijkheid	6	6	3	0
Operationele Functionaliteit	21	1	4	5
Financiële Functionaliteit	4	1	0	8
Traceerbaarheid / Correctiemogelijkheden	3	0	6	3
TOTAAL	98	15	20	20
	FIT		GAP	

Conclusie

De onderstaande grafiek in figuur 2 geeft het percentage over de onderdelen passend (fit), niet-passend (gap) en de norm aan. De percentages zijn berekend op basis van de aantallen uit de voorgaande resultaten en zijn niet gewogen ten opzichte van elkaar. Dat wil zeggen dat alle meetpunten met een factor één worden meegeteld in de onderstaande conclusie.



Figuur 2 Overzicht van de fit-gap analyse

Uit de grafiek blijkt dat Care Arrangement op dit moment op geen van de kwaliteitsfactoren boven de norm scoort. Hierbij is gekeken naar de huidige versie van het systeem. Het is dan ook niet zinvol om te kijken naar latere versies van dit systeem.

De belangrijkste bevindingen worden per kwaliteitsfactor kort toegelicht, met daarbij een korte uitleg wat de kwaliteitsfactor precies inhoudt.

- Betrouwbaarheid

Definitie: De mate van vertrouwen van informatie van het systeem.

Score: 44%

Norm: 90%

Beoordeling: Slecht

Bevindingen: Volgens de test is dit systeem is dusdanig onbetrouwbaar dat het de gebruiker vrijwel onmogelijk wordt gemaakt om belangrijke acties uit te voeren.

- Begrijpbaarheid

Definitie: De mate van begrijpbaarheid en handelbaarheid (complexiteit) van het systeem.

Score: 18%

Norm: 80%

Beoordeling: Zeer slecht

Bevinding: Het systeem is niet begrijpelijk, veel cosmetische aanpassingen zijn gewenst door de gebruikers. Tijdens het roosteren zijn de kleuren onduidelijk wegens ontbreken van een legenda. Ook de processtructuur van de software is niet duidelijk voor de gemiddelde gebruiker.

- Gebruiksvriendelijkheid

Definitie: De mate van handelingen en overzichtelijkheid van het gebruik.

Score: 40%

Norm: 80%

Beoordeling: Slecht

Bevinding: Het merendeel van de gebruikers vindt de applicatie niet gebruiksvriendelijk. Vooral tijdens het plannen van de zorg moet er onnodig veel gescrold worden. Er wordt niet ook geen duidelijke feedback door het systeem gegeven wanneer acties hebben plaatsgevonden.

- Operationele Functionaliteit

Definitie: Toepasbaarheid systeem voor de ondersteuning van het proces (zonder facturatie).

Score: 68%

Norm: 100%

Beoordeling: Matig

Bevinding: De software lijkt in het begin van het proces goed te ondersteunen. Totdat er gepland moet worden, dan blijkt bijvoorbeeld wel eens dat de gegenereerde zorgonderdelen niet in de restvraag terug komen. Dit heeft als gevolg dat er niet gepland kan worden. Aangezien plannen bij één van de twee belangrijkste functionaliteiten (andere is de facturatie)

hoort weegt dit probleem zwaar mee in de overweging om door te gaan met Care Arrangement of niet.

- Financiële Functionaliteit

Definitie: Toepasbaarheid systeem voor de ondersteuning van het facturatieproces.

Score: 31%

Norm: 100%

Beoordeling: Slecht

Bevinding: Factureren op de manier zoals Wilhelmina Senioren Diensten dit wil uitvoeren is niet mogelijk. Het stukt al bij het afhandelen van de zorg. Mocht dit werken dan hebben de facturen negatieve bedragen en wordt er op foutieve wijze BTW belast. De factuur heeft ook de verkeerde adresgegevens.

- Traceerbaarheid / Correctiemogelijkheden

Definitie: Het resultaat van het corrigeren van acties en het terugzien wat er heeft plaatsgevonden.

Score: 25%

Norm: 80%

Beoordeling: Zeer slecht

Bevinding: Als er door de gebruiker een fout is gemaakt is het in sommige gevallen onmogelijk om deze fout ongedaan te maken. Daarvoor zouden compleet nieuwe objecten aangemaakt moeten wat onnodig veel tijd kost. Een undo-knop zit wel in het systeem alleen deze werkt niet naar behoren, veel foutieve informatie blijft gewoon bestaan.

En nu verder...

Ons advies luidt als volgt:

Care Arrangement is een systeem dat niet werkt voor een bedrijf als Wilhelmina Senioren Diensten dat op basis van PGB werkt. Ook het planningssysteem is onder de maat. Wilhelmina Senioren Diensten kan nu kiezen voor twee opties:

1. Nieuwe pakketten eerst uitgebreid onderzoeken voordat een overeenkomst wordt aangegaan. Wij raden aan om een derde partij hierbij te betrekken voor deze selectie.
2. Aangezien er ICT-kennis bij Wilhelmina Senioren Diensten is, is het mogelijk om de software ook intern te ontwikkelen. Dit zou dan wel ondersteund moeten worden door een ervaren ICT-projectleider.