MASTER'S THESIS

Modelling of a supporting system to perform a case studies as a research method

van den Camp, M. (Marc)

Award date:
2020

Modelling of a supporting system to perform a case studies as a research method

Degree programme:	Open University of the Netherlands, Faculty of Management, Science & Technology Business Process Management & IT master's programme
Course:	IM9806 Business Process Management and IT Graduation Assignment
Student:	Marc van den Camp
Date:	December 2nd, 2020
Thesis supervisor	Dr. Ir. Ella Roubtsova
Second reader	Prof. Dr. Ir. Stef Joosten
Version number:	1.01
Status:	AF-report (Final)

Abstract

The Case Study Research method is an-in depth inquiry into a topic or phenomenon within its real-life setting. It can be applied to several research fields. It can also serve in several areas, as it can be applied for exploratory and descriptive purposes. The method has gained in popularity in the recent period, but it is also been criticised for lack of scientific rigour and reliability and that it does not address the issues of generalizability. Literature review shows that the quality of Case Study Research reports leaves to be desired. In this thesis, an area where this might be improved is researched. With a literature study following a rapid prototyping approach, a support tool is modelled that could support research teams in their Case Study Research project conduct and capturing their results in a structured way. The aim is also to support in factoring in the research deliverables and results into the report. With a prototyping approach, the requirements for such a tool are evaluated. It would also add insights into the main components and functions that the tool would have. Finally, the model reveals and supports possibilities for further evaluation and research in this area.

Key terms

Knowledge Management Systems, Case Study Research, Goal-Protocol Modelling, Research Quality management.

Abbreviations

CSP	Communicating Sequential Processes
CSR	Case Study Research
CST	Case Study Research Type

Content

1. Introduction

1.1. Case Study as a research method

The Case Study Research (CSR) method is an-in depth inquiry into a topic or phenomenon within its real-life setting (Mark Saunders, Philip Lewis, 2016). It can be applied in several research fields. Examples are found in business or IT related domains (Gagnon, 2010; Runeson & Höst, 2009; Verner & Abdullah, 2012), in healthcare (Houghton, Casey, Shaw, & Murphy, 2013) and others. Also, it can address a number of objectives while the data collection can be done in a great variety of methods, as interviews, surveys, text analysis and so forth.

The key difference of a case study research from other types is its way of decomposition of real life by selection and description of a part of real life, like for example, an organization or a business process with all real settings of it. In this decomposition, a case study researcher already identifies a repeated occurrence of this part of real life and assumes a domain of possible applicability of future research results.

The CSR method is applied in several stages. These range from designing the approach, preparing and executing on it until reporting the findings (Gagnon, 2010; Host & Runeson, 2007; Tellis W. M., 1997). During these stages, decisions are made on how to complete the stage and then proceed onto the next and eventually, feed the findings into a report. These logically related stages are generally applied in sequence.
When designing a research approach the researchers need to decide on dimensioning the approach as also on the Case Study Type to be fulfilled. Saunders et al. (Mark Saunders, Philip Lewis, 2016) identified two discrete dimensions in which the researcher determines the nature of the approach: 1) Single case versus multiple cases and 2) Holistic case versus embedded case. But the Case Study Type can be also defined by the domain of interest for example, processes in organizations, enterprise models etc. The dimensions refer to the scope of the research to be applied. Where it can be one case versus a set of similar cases, or the domain of the scope of research is defined to be a whole domain, organisation or branch or is considered, on the other hand or a singular entity within those domains, on the other.

While the CSR strategy is a very popular method it also has been criticised for the lack of scientific rigour and reliability as well as the lack of addressing issues of generalizability (Easton, 2010). Nevertheless, in particular, in business this can be the only possible method of researching a complex or emerging phenomenon, as in particular there are also strengths in a CSR strategy, for example a CSR enables the researcher a holistic view of a certain phenomenon or series of events and also it can be useful in capturing the emergent and immanent properties of life in organisations. (Noor, 2008).

1.2. Quality of Case Study Research

When preparing for a research Case Study project, the researcher has to make a number of design choices. These reflect on all the stages of the research project. Goffin et al. (Goffin, Åhlström, Bianchi, & Richtnér, 2019) investigated CSR projects in innovation management and they published an assessment tool for evaluating CSR projects and their reports. For their evaluation of 818 CSR articles published in leading journals, they developed a Case Study Evaluation Template (CaSET). With this tool, they mapped the application of 10 quality practices onto a metric, ranging from 0 to 10. The set of evaluation criteria and their categories are listed in table 1.1.

Table 1.1 CASET Case Study Research evaluation criteria as published by Goffin et al.

Research Design	1. Theoretical foundation	Grounding the decisions for carrying out the research
	2. Pilot study	Conducting a small pre study to better design the approach in the final study
	3. Theoretical sampling	Basing the areas for where to sample on the emerging theoretical basis
Data collection	4. Triangulation	Using more than one source of data and method for collection to confirm the validity, credibility and authenticity of the research data
	5. Review and validation of evidence	Having the evidence formally reviewed and validated by people other than the researchers.
	6. Transparency of data collection	Demonstrating a transparent data collection process, so that others can replicate the study or, at least, have a detailed understanding of the type of data that was collected
Data analysis	7. Case presentation	Providing a comprehensive trail of evidence into the report
	8. Case Interpretation	of the results in the light of existing concepts, models and findings from the extant literature,
	9. Inter-coder agreement	Arranging mutual alignment between various researchers on their methodological and automated interpretation of researched texts
Post hoc – reflection on rigor	10. Reflecting on validity and reliability.	Meaningful reflection on the quality achieved in the research conducted, covering one or more of the dimensions: construct validity, internal validity, external validity, and reliability

Finally, they related the metrics back into the different journals (~ domains) from which the articles were taken. They found that in the field of innovation management, the quality of CSR is relatively poor, 3.05 on a scale from 0.00 to 10.00. This implies that the researchers on average addressed only 3 out of 10 quality criteria when conducting and reporting on a CSR project in the field of innovation management.

1.3. Supporting hypothesis

The hypothesis is that the CSR's quality criteria categories data collection and data analysis can be improved with a tool's support within a CSR project. The tool would be supporting these criteria by creating transparency in helping to structure measurement collection and publication.

This all then would result in a setting in which eventually the criteria have a better chance of being met. Additionally, a pilot study could be easier organized with a CSR support tool, as the results captured in that tool can later be linked to the data set captured during the final study.

In the light of the analysis noted above, the need of a supportive tool for a CSR projects seems to be emerging. After having captured the criteria for data collection and data analysis in such a tool on beforehand, the tool could act as a guidance during the research project execution itself. This then could help to improve the project quality when it comes to keeping track of the project phases and related deliverables for each of phase.

That said, the topic on the origin of the criteria, translated into project deliverables, is indicated by Goffin et al. (Goffin et al., 2019), as they also concluded that "if Case Study Research has already been conducted in the area of interest, then researchers can learn from previous Case Study designs", implying that the set of criteria and deliverables would best be the result of lessons learned from previous conducted and similar projects. Thus, a tool would support this learning from similar Case Study projects in an organized manner.

1.4. Motivation and relevance of a supportive system

The interest in support and guidance when conducting a CSR is significant. Host et al. (Runeson & Höst, 2009), in their effort to supply guidelines for conducting CSRs in Software development were cited 1200 times. Also in publications from Baxter et al. (Baxter, P., & Jack, 2008), establishing such guidelines, and work from Yin (Yin, 1999) and Gibbert et al. (Gibbert, Ruigrok, & Wicki, 2008), where not only descriptions and guidelines where published on when and how to apply this method but also checklists on conducting and reporting on it.

Baxter et al. (Baxter, P., & Jack, 2008) stated: "For the novice researcher a Case Study is an excellent opportunity to gain tremendous insight into a case. It enables the researcher to gather data from a variety of sources and to converge the data to illuminate the case.", implying that this method is applicable for a range of situations and fields. The method has attention in the scientific arena as well as its quality deliverables. In this light, researching methods to support and possibly enhance the quality of these deliverables, is worth the effort.

Definitely in CSR, the focus for quality is relevant as Gibbert et al. stipulates that "While deficiencies in any methodology are problematic, lacking rigor in case studies is particularly problematic for at least two reasons.". "First, as this methodology is an appropriate tool in the early stages of researching a new theory and deficiencies in the application of it will have a ripple effect throughout the later stages of the research.". "Second, CSR is typically carried out in close interaction with practitioners", who deal with real management situations, thus creating managerially relevant knowledge. But "without rigor, relevance in management research cannot be claimed". This more and more emphasises the necessity on process quality in the field of Case Study Research.

1.5. Research question of this research

Reflecting on the sketched problem statements, the research problem for this investigation is phrased as:

What is the collection of requirements, abstractions (like objects, events, constraints and use cases, protocols) of a system that could support a CSR project in settings that guide the routines of data collection?

1.6. Derived research questions and research methods

Addressing the former main research problem statement, some research sub-questions have been formulated.

1. What requirements, protocols and set of constraints are applicable for a CSR initiative?
   Research methods:
   - This has been addressed by conducting a literature study

2. Can a supportive executable system be designed, and how, from the requirements, protocols, and constraints for a CSR?
   Research methods:
   - This has been addressed by developing a design of an executable protocol model
   - The designed protocol model has been tested with 2 reported CSR projects.
   - The designed protocol model has been analysed on including the setting that may improve the quality of a study research project.

2. Requirements, protocols and constraints for a Case Study Research project.

In literature, various tools can be found to support CSR projects. Mostly these consist of activity- or checklists supporting the deliverables of the phases in the project.

Authors propose various sets of stages to be completed during a CSR project. Gagnon proposes 8 stages (Gagnon, 2010) while Tellis, Host et al. (Host & Runeson, 2007; Tellis W. M., 1997) proposes 5 phases in the CSR research life cycle. There is much resemblance between the various sets of phases, found in literature. As these 5 phases can be found back in most other references, they will be used in the rest of this document:

1. Case Study design: objectives are defined, and the Case Study is planned.
2. Preparation for data collection: procedures and protocols for data collection are defined.
3. Collecting evidence: execution with data collection on the studied case.
4. Analysis of collected data
5. Reporting

Next to support phasing a Case Study project, a researcher might be interested in a knowledge management – or knowledge support system to collect the data of a CSR. Using extensive search strings in search engines and tools as google (scholar) and the digital library, approaching the search from the angle of “knowledge management development” as also from areas as “research project support”, “CSR project governance” and “CSR support”, no relevant examples were found in literature review of a knowledge management system that supports a CSR.

Reflecting on the research project life cycle the researcher needs to observe a number of aspects when conducting a CSR project. These aspects stipulate steps and considerations. Being compiled in this way they imply deliverables that should document the fulfilment of these steps and considerations. The deliverables are quite generic and can vary per type of CSR conducted.

Some of those elements may be(Mark Saunders, Philip Lewis, 2016):
- Type of research question or the specific research question itself
- factoring in certain collection techniques and potential data sources
- Criteria used for the Case Study selection or data sets
- Data coding instruments

In the quest for a supportive system, its design is not restricted by these concepts (ontologies), but it should also include the modelling of related processes and support of decisions, based on these concepts. The modelling technique should enable conceptual modelling, process modelling, process composition and decision support. Such features are promised in the method that combines goal and protocol modelling(Roubtsova, 2016).

2.1. Addressing Quality criteria for Case Study Research project approach

Evaluating the results of Goffin et al. (Goffin et al., 2019), a number of criteria stand out where the analysed 818 articles have a low scoring in meeting them. In table 2.1 the percentages derived from the article highlight that the criteria “Pilot Study”, “Inter-coder Agreement” and alternatively “Transparency of Data Collection” and “Reflecting on Validity and Reliability” achieved the lowest score.

Table 2.1 Percentage of articles meeting the CaSET criteria

Criterion	(%) of articles met criterion
Theoretical Foundation	52
Pilot Study	**7**
Theoretical Sampling	46
Triangulation	70
Review and Validation of Evidence	24
Transparency of Data Collection	**21**
Inter-coder Agreement	**8**
Case Presentation	31
Case Interpretation	25
Reflecting on Validity and Reliability	**21**

One can argue that any tool, designed to support CSR conduct, might best start with focussing on these areas.

Meeting the “Pilot Study” criterion is a matter of choice by the research team. This small pre-study will result in research data before the actual CSR project has started. A tool could facilitate this in different ways:

- Guide the team to the activity by means of stating the protocol
- Support data collection in this pilot study in a way that it can be suited into the main CSR project
- Capture the protocol as followed it the pilot and upgrade it to be used in the main CSR project

The “inter-coder agreement” supports a research design where multiple researchers participate in investigating a case or set of cases. The coding involves the categorisation of different observations in documents or phenomena investigated. According to Goffin et al (Goffin et al., 2019), when involving multiple researchers is also a form of triangulation, another criterion on the list. Goffin further states that coding is a time consuming task but it can be rewarding by producing more reliable and interesting results. It might prove intricate to have a tool support the coding activity itself as it involves text and domain analysis, and it would also need to relate to the researched phenomenon. Capturing the description of the coding system and eventually capture the coded investigation results is definitely an angle where a tool could support.

A quality CSR report is required to observe “transparency of data collection” where it documents that the data is collected in a clear and transparent way, as it also needs to specify the circumstances under which the data collection has taken place. It is to support that others can replicate the study or have a detailed understanding of the data that was collected. Next to the collected data, also the

description of the data collection process is an essential part of this criterion. Describing the data collection process and parameters is subject of the research design. A tool could support this in capturing the description document. Also when capturing the collected data appropriately, the tool can aid in documenting the results and adhere to the criterion of transparency.

Capturing a document can also support to meet the criterion of "Reflecting on Validity and Reliability". This needs to reflect on measures, taken to ensure validity and reliability as Goffin et al. states (Goffin et al., 2019). This is imperative in Case Study Research. The major reason for the absence of such a reflection, according to Goffin et al., is awareness of these quality criteria involving CSR projects. Here the tool could help to create awareness by listing it in the project's protocol and capturing it.

Abstracting the options proposed above that can be elected as requirements for a supportive tool the following list emerges:

- The tool would need to support capturing a research protocol (pilot study)
- The tool would need to be capable of capturing research data, resulting from the CSR project. (inter-coder agreement, Transparency of Data Collection)
- Also the tool would needs to support capturing research process descriptive documentation (pilot study, inter-coder agreement, transparency of data collection, reflecting on validity and reliability)

As the CSR research method is used in a variety of domains, the tool would also require to be flexible in the sense that it does not state this set of deliverables in a rigid way and won't allow deviation from protocol sequence or project deliverables.

2.2. Designing a supportive system from requirements, protocols, and constraints

The researched tool might be supporting data-, information and knowledge management, as these are very goal-oriented activities. The goal in this context would be to conduct a high-quality CSR initiative, evaluate the process on its successes and flaws and learn from this in order to improve when conducting a next similar project. This might even be applicable when moving from a pilot study to the actual CRS project. The development of such a system could initially focus on supporting the evaluation and improvement cycle that will result in a next generation version of a tool that will guide the project team in their next research project. The checklists as proposed by Gagnon and Runeson et al. (Gagnon, 2010; Gibbert et al., 2008; Runeson & Höst, 2009) are instrumental tools to facilitate the evaluation of the approach and the reporting of the CSR project.

Roubtsova (Roubtsova, 2016), and McNiele (McNeile & Simons, 2006) proposed a modelling technique that combines goal modelling, where the goal of the research team is to pursue a high quality research project, and behaviour modelling, that looks into acting on the different states and stages of the research project, could help to understand the specific requirements that such a supportive system would need to cover. By developing such a model in a tool, one could not only learn from the modelling activity itself, but also from the results, the model would produce, by testing it and evaluating it with available examples from already conducted CSR projects.

The developed protocol model is a set of deterministic acceptors (protocol machines) synchronized by Communicating Sequential Processes (CSP) that are composed in parallel. The calculus of CSP defines an operator for the composition of concurrent processes. In order to support interactive simulation and its interpretation, these acceptors contain data structures for events and the state space, extended with attributes. The approach then leads to an interactive model that eventually can be used to simulate the researched processes.

Behaviour modelling focusses on how a system should behave after certain events have taken place. Those events normally are triggered by a user and are received by all intended protocol machines. However, it is only the set of protocol machines that is susceptible for the received event that will eventually act on it by a state change.

A goal model eventually is used to dissect the underlying processes in their domain. as proposed in the approach, to end up with a workable model that can be used to analyse the intended process, both modelling techniques are applied. After having completed those, the development of a simulation model in the tool can be started.

This modelling and simulation technique has the advantage that it creates an executable model of the process. By deploying this model in the various process steps, (status) data is being gathered in such a way so that it can be used to decide on next steps in the process. It is this feature of having next steps being dependent on earlier decisions and events that makes this technique a promising approach. The proposed method is announced as a rapid prototyping approach. When applying this also a repetitive cycle can be supported where the model is optimized, based on learnings from conducting tests and evaluations.

2.3. Goal model for developing a CSR support system

Looking into the actual goal and sub-goals that need to be achieved in order to fulfil the CSR project, a subsequent set of challenging questions lead to an enhanced level of abstraction on the topic. Initially this process leads to a goal model.

In applying the approach, some stages are conducted iteratively. Initially the goal is formulated. With this, better insights into a possible supportive system for conducting a successful CSR project being pursued. This goal will be split up in sub goals. Eventually every sub goal will be challenged and further refined by continuous challenging the (sub) goal in the higher level. This drill down refinement can stop when a specific and measurable criterion is formulated.

By proceeding through successive steps where the initial goal is continuously challenged, the goal model is further refined. By raising questions on the answers found, a layered abstraction model emerges.

Stating the goal that a CSR project needs to be conducted with enhanced quality, the sequence of questions and answers would be:

Q: what are the prerequisites for a proper CSR project fulfilment?
A: compliance with the central quality judgements reliability and validity
Q: How do these translate into practical criteria that can be applied to CSR Research projects?

A: Goffin et al. (Goffin et al., 2019) listed a set of practical criteria to be met. In their work to compile this list, they evaluated it against the central criteria of validation and reliability. The major categories were based on the various stages, a CSR project goes through:

- Research Design
- Data collection
- Data analysis
- (Reporting results, not mentioned explicitly as a phase, although the applied their analysis to CSR reports)
- Post hoc reflection on rigor (needs to be factored into report)

Q: what are the stages and deliverables for these phases?
A: Stages:

- Research design: the case study is designed, planned and documented. The data collection procedures and protocols are defined and documented
- Data collection: data collection is conducted in accordance with design. Measurements are captured and coded
- Data analysis: Data analysis is performed on coded data as stipulated in research design
- Reporting results: report is available in format as foreseen in design
- Post hoc reflection on rigor: in the report is factored in, a reflection on rigor, addressing the different aspects of Validity and reliability

Q: where then, can a tool help in improving the quality of CSR project conduct?
A: The objective is that the tool would be applicable in various domains where CSR projects are conducted. Having the tool analyse domain specific sources, knowledge, processes and practices was considered not feasible. The tool can support in:

- Guiding the research team through a protocol when conducting the initiative
- Collect and enable publication of the documentation required as deliverables during project execution
- Collect an enable publication of the measurements collected during the research project conduct.

Q: how can these tool's capabilities help to improve CSR conduct quality?
A: by facilitating transparency about the project's conduct and its results.

Reflecting on these Q&A's a goal model emerges that is found in Figure 2.1.

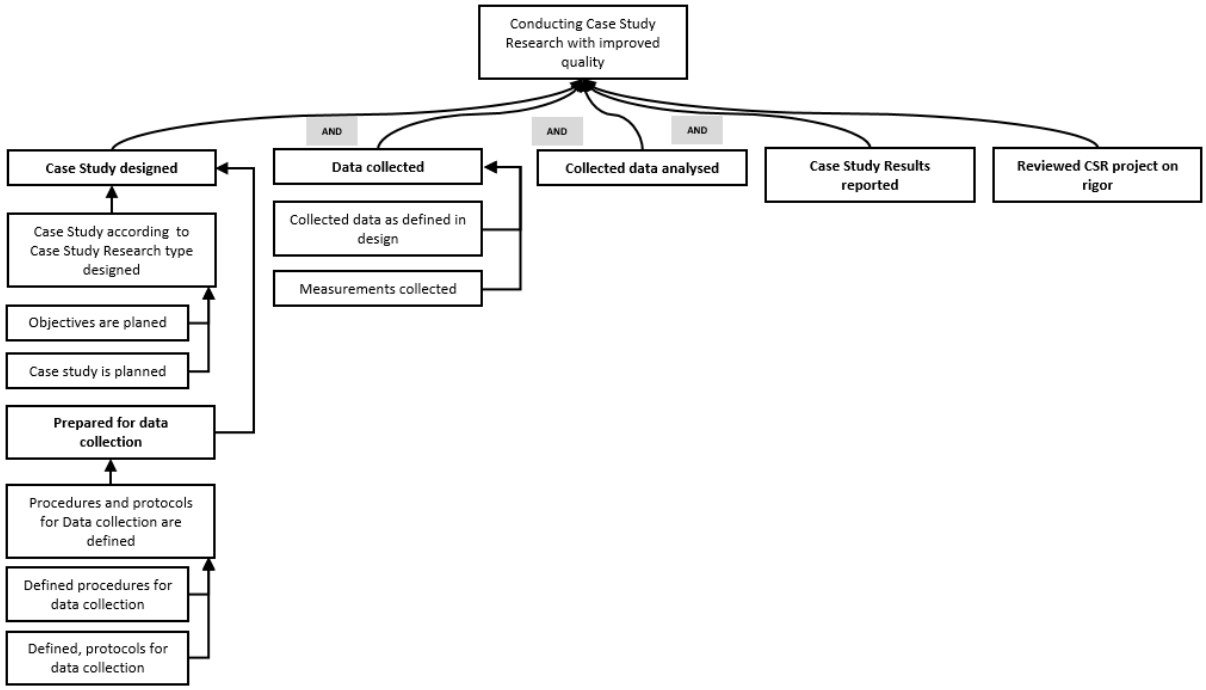


Figure 2.1 Goal model for a Case Research fullfillment support system

2.4. Identifying the requirements for the Case Study Research supportive system

The objects identified in earlier descriptions are the ones that would play a role in the design of the tool. By selecting them, every object can be scrutinized to be further implemented into the tool.

The CSR is the key abstraction that has a number of attributes:

(1) Research question about a repeated part of the world (i.e. about hospitals) are referring to a (2) **Domain of interest** (i.e. hospitals)

The major difference between CSR projects and any other research project is the initial (3) **Naming of a case type** and the (4) **Description** of a case type.

It is a decomposition of the world and a selection of a particular element of it amongst the type of elements. For example, a particular hospital (like UMC Utrecht). Repeated elements in the world would be hospitals while a specific hospital is a case.

The research question influences the choice of the research method inside a case study. It can be modelling or interviewing of document research. However it will take place within the domain of the selected case environment.

The research method within the case study defines the **(5) Protocol** of a CSR project. This protocol is a set of steps, where each step is an application of one of internal research methods to the case.

Each (**6) Step** contains data collection with one of research methods applied to the case. The data can be textual, graphical or measures.

From a (7) **CSR project type** a Case Study project is instantiated. It contains a description of the part of the studies world, the research question applied to this instantiation and the steps of data collection for this described case study. From this collected data the (8) **research report** is generated.

The Research report of the project, contains all project design descriptions, data collected in steps and the answer to the research question. If the collecting and transformation of data is transparent, then the research is regarded as reliable.

These concepts then, are eventually translated into objects with attributes defined in the supportive model. The stages of the definition process of these objects, being elements of the project can be listed as:

- Define the Case Study Type (CST)
- Define the steps in the project
- Define the data, to be collected per step (files or measurements)
- Conduct the project while observing the defined protocol and supply the requested deliverables

Refining these activities, the following sequence emerges:

- Define the generic CST
- Add the CST Steps to this generic research type
- Add measurements to the steps that are defined earlier
- Create an individual CSR project from this template CST
- Add values to the predefined measurements in the separate steps
- Add deliverable files to the individual steps where not measurements have been defined.
- Apply available reporting capabilities to extract the entered data

The hierarchy and derivation of the involved objects are graphically displayed in Figure 2.2. Here is shown that per Case Study project a new project structure is copied from the template.

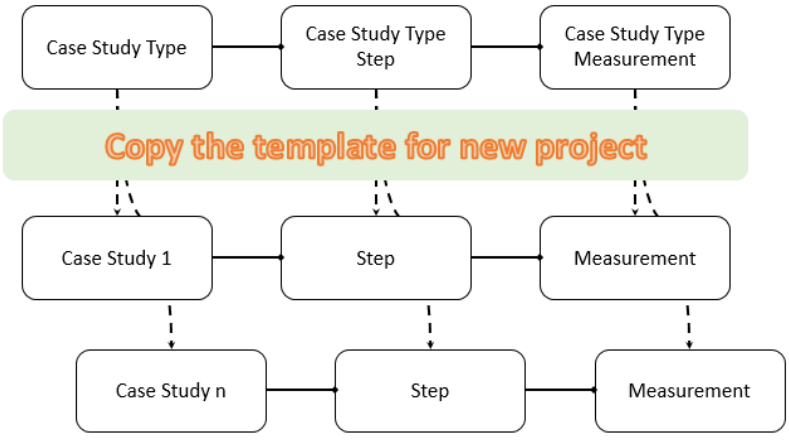


Figure 2.2 Copy the CST template into a new supportive tool of the Case Study project

Evaluating this design more and translate this into a set of requirements, a list emerges as in table 2.2. These requirements result in test cases that will be conducted on the model.

Table 2.2 Set of requrements to be applied to the support tool model

Number	Requirement	Rational
1	For the definition of a CST, the tool should support definition of the steps, as part of the CST.	The definition of the CST protocol embodies a list of steps in chronological order to be fulfilled.
2	Per Case Study Type Step, deliverables should be defined. These can be a file with the deliverables noted or a set of measurement values that the research team would be required to supply.	The research team would, in some way, need to exhibit that they have completed the step. As for every step a deliverable is defined (file or measurement), these act as a proof for completing the step. The tool should facilitate in capturing this.
3	The tool should support role-based access where there is a distinction between the user that defines the CSR (type) project and the user that carries out the research and enters the deliverables into the tool	Role based access guarantees segregation of responsibilities during the different phases of the CSR project.
4	After the definition of the CST protocol template, the definition can be saved. After saving no additions can be made to the template.	The template acts a standard for the defined CST. The option to alter it would be undesirable. If this option were to exist, side effects related to version management could arise. Here Researcher error and bias threat mitigation comes into play.
5	The tool should be able to maintain a set of predefined protocol template for fulfilling a type of CSR projects.	If the tool contains a number of templates for different kind of CSR types, the research team is able to derive a protocol from the applicable template for their specific project
6	The tool should be able to derive from the various CST definitions, an actual Case Study project support instance. This creation should be automated without further user manual labor.	Automation of this derivation process circumvents structure and typing errors.
7	The sequence in which the steps are defined in the Case Study Type should also be applied in the same order to the derived Case Study project.	A CSR project has several phases to be completed. As the sequence of these steps is chronological, this sequence should be captured and replicated in the derived Case Study project.
8	Once the created research project support tool is instantiated, the user should not be able to alter its structure.	The execution of the research project should be in line with the predefined CSR project type. The option to alter the structure would accommodate deviation from this standard.
9	The tool should support, adding deliverables and values to steps and measurements until the moment the Case Study is set to the state “completed”.	Until the Case Study is completed the research team needs to be able to amend/improve the captured data.
10	The tool should check if all deliverables are met before allowing the user, to close the case Study project.	“All deliverables” imply that for every step, one of the following conditions are met: 1. A filename is supplied of the file containing the step derivable – or – 2. All measurements related to the step contain a measurement value
11	After the project is completed alterations of the deliverable files and measurements should no longer be possible	When the project is completed, the research team is required to exhibit deliverables and measured values. Being able to change these after project completion could potentially impair integrity of the results.

Number	Requirement	Rational
12	The tool should have various reporting capabilities to support the user in establishing the progress of the project as well as the findings. These reports should aim to support all available roles.	Having an extensive set of reporting capabilities in place will increase the rate of acceptance to the user.

3. The development and testing of the executable model

The CSR support system is constructed as a model in a supportive tool called Modelscope, developed and published by McNeile and Simons (McNeile & Simons, 2012). In this tool, a model is defined in terms of a set of protocol machines. These protocol machines represent an abstraction of the studied environment.

Within the model in the tool, the protocol is generally made up of:

- Objects: a representation of the relevant entities in the case description. By adding attributes, characteristics and behaviours, the objects behave like the real live counterparts. For instance, in a banking environment one would find here objects as: account, client and loan
- Events: actions, initiated by users or maybe objects as a response to received events. They represent the verbs in the model's description. In the same banking environment, one would find events as: open account, close account, withdraw from an account or deposit on an account
- States: the situational descriptions in which an object resides after one or more events have been applied to it. In banking this might be registered (when considering a client), active (considering an account)

When modelling the Case Study support tool, in total six objects have been identified that can be divided in two sets:

- One set of objects that entail the definition of a CST template
- One set of objects intended to define an individual CSR project

Each of these sets contain elements of a Case Study project:

- A Case Study project
- The steps in this project with related deliverables
- A set of measurements to be fulfilled per step

3.1. The sets of protocol machines.

Based on the categorisation listed above, the total set of protocol machines can logically be divided in a number of subsets. A more extensive description of the developed protocol model is available in appendix A. In the next paragraphs a more concise exploration.

3.1.1. The Case Study Type template

The Case Study Type object

The protocol machine of a CST is an abstract model of a Case Study Type definition. A graphical presentation can be found in Figure 3.1. The circles in this protocol machine diagram indicate the various states that the machine can reside in while the arrows show the events that potentially trigger transitions from one state to another.

During the Case Study Type template definition phase, this protocol machine sequences through several states:

- Created: reached after a new CST is created.
- In definition: the additional template structure objects, steps and measurements, can be added to the model
- Completed: the definition phase has ended, and the template is locked for changes

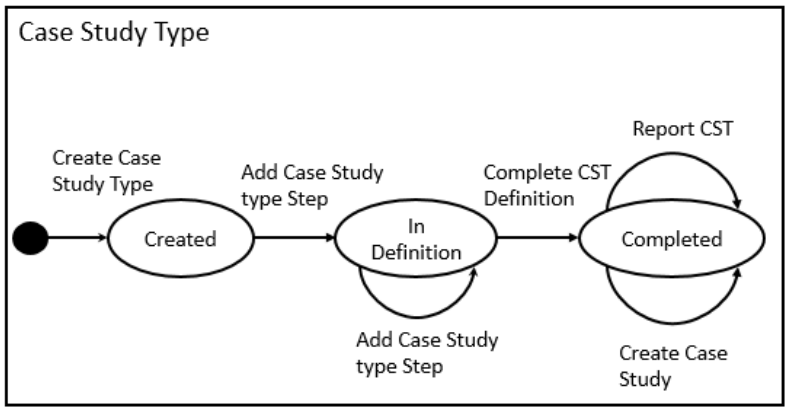


Figure 3.1 Protocol machine for Case Study Type

Listing 3.1 shows the attributes of the object. The domain of interest is defined in the attribute CaseStudyTypeName and the Case Study Type Description. The preceding exclamation mark indicates that this attribute triggers a Java call back routine that calculates the number of referenced steps in the model.

```
OBJECT CaseStudyType
    NAME CaseStudyType Name
    ATTRIBUTES      CaseStudyType Name: String,
                    CaseStudyType Description: String,
                    !Number Of Steps: String,
    STATES          created,
                    in Definition,
                    completed,
    TRANSITIONS     @new*Create CaseStudyType=created,
                    created*Add Case Study Type Step=in Definition,
                    in Definition*Add Case Study Type Step=in Definition,
                    in Definition*Complete CST Definition=completed,
                    completed*Report CaseStudyType=completed,
                    completed*Create CaseStudy=completed,
```

Listing 3.1 Modelscope Definition of the CST protocol machine

A second call back routine, in listing 3.2 is called when the Case Study Type template structure is to be set to Completed. The routine not only arranges the status for this protocol machine to Completed but it also selects all related Case Study steps and measurements and sets those to the status completed as well. In this way, the protocol machine ensures that, after the definition phase is concluded no Steps and Measures can be added any more.

```
package CSR2021;

import com.metamaxim.modelscope.callbacks.*;

/*
Complete a case Study Type and all its CST Steps
When then the whole stucture is in a "completed" state it is no longer
possible/allowed to add Cast Study Type Steps (and CST Measurements)
*/

public class CompleteCSTDefinition extends Event {

    public void handleEvent() {
        Instance thisCaseStudyType = this.getInstance("CaseStudyType");
        Instance[] setOfCSTSteps =
            thisCaseStudyType.selectByRef("CaseStudyTypeStep","CaseStudyType");
        if (setOfCSTSteps.length != 0 ){
            for (int i=0; i< setOfCSTSteps.length; i++) {
                Event completeCSTStep = this.createEvent("Complete CST Step");
                completeCSTStep.setInstance("CaseStudyTypeStep", setOfCSTSteps[i]);
                completeCSTStep.submitToModel();
            }
        }
        Event completeCST = this.createEvent("Complete CST Definition");
        completeCST.setInstance("CaseStudyType", thisCaseStudyType);
        completeCST.submitToModel();
    }
}
```

Listing 3.2 Call back routine to close the Case Study Type definition

The Case Study Type Step object

The Case Study Type Step protocol machine, as displayed in figure 3.2, is the next in the set that defines the Case Study Type template. There is a reversed reference from this step object to the higher hierarchic related Case Study Type.

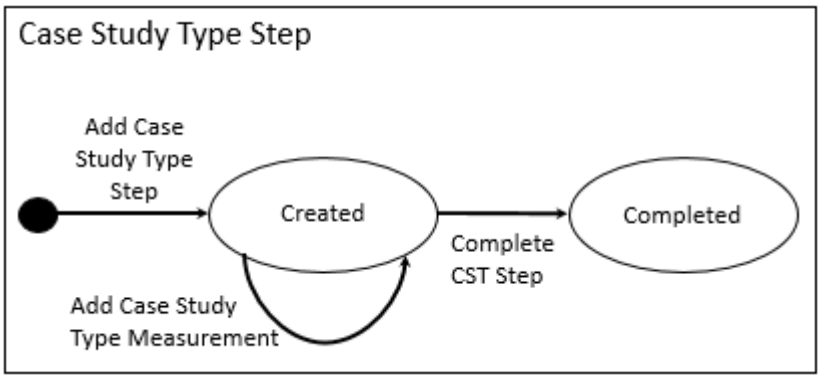


Figure 3.2 Protocol machine for a Case Study Type Step

In the Modelscope definition in listing 3.3, one of the attributes refers to a Java call back routine that calculates the number of measurements related to this Case Study Type Step.

```
OBJECT CaseStudyTypeStep
    NAME CaseStudyTypeStep Name
    ATTRIBUTES      CaseStudyTypeStep Name: String,
                    CaseStudyType: CaseStudyType,
                    !Number of CST Measurements: String,
    STATES          created,
                    completed,
    TRANSITIONS     @new*Add Case Study Type Step=created,
                    created*Add Case Study Type Measurement=created,
                    created*Complete CST Step=completed,
```

Listing 3.3 Modelscope definition of the Case Study Type Step protocol machine

The Case Study Type measurement object

One Case Study Type Step can reference multiple Case Study Type measurements. The protocol machine for this is presented in figure 3.3. and the model definition is in listing 3.4.

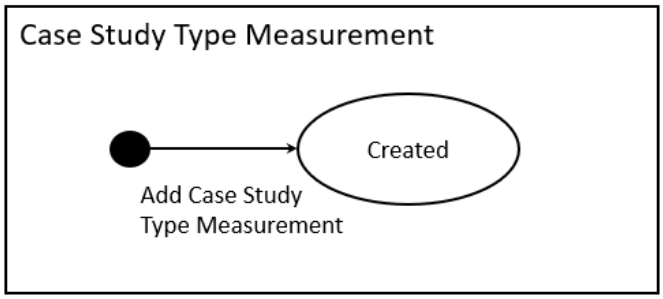


Figure 3.3 The Protocol machine model for the Case Study Type measurement protocol machine

```
OBJECT CaseStudyTypeMeasurement
    NAME CaseStudyTypeMeasurement Name
    ATTRIBUTES      CaseStudyTypeMeasurement Name: String,
                    CaseStudyTypeStep: CaseStudyTypeStep,
    STATES          created,
    TRANSITIONS     @new*Add Case Study Type Measurement=created,
```

Listing 3.4 Modelscope definition of the Case Study Type measurement protocol machine

3.1.2. The protocol machines for the Case Study project

The Case Study object

When the set of protocol machines that make up the definition of a CST is available it is possible to derive a Case Study project from it. This instantiation is done automatic by a call back routine. Before being instantiated the user has to define the Case study name, the Research Question and the Case Study description.

A graphical presentation of Case Study model is shown in figure 3.4. It effectively shows two states

- Created: deliverable data can be added to the model
- Completed: the model is closed for further adding information to it.

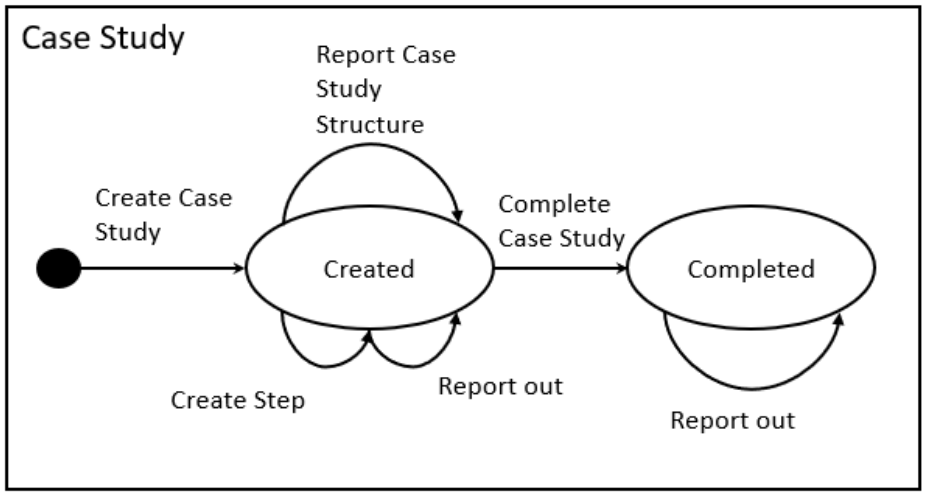


Figure 3.4 The Case Study object

Around this protocol machine there are some call back routines to make sure that the requirement is fulfilled and that no information can be amended or added after the CSR project is closed. One of these call back routines arranges that all steps and measurements are completed when the Case Study itself is set to completed. In listing 3.5., there are also several call back routines related attributes that support the user with information on the size of the project, in terms of number of steps and number of completed steps.

```
OBJECT CaseStudy
    NAME CaseStudy Name
    INCLUDES          Case Study Completion Check,
    ATTRIBUTES        CaseStudy Name: String,
                      Research Question: String,
                      CaseStudy Description: String,
                      CaseStudyType: CaseStudyType,
                      !Number Of Steps: String,
                      !Number Of Steps Completed: String,
    STATES            created,
                      completed,
    TRANSITIONS       @new*Create CaseStudy=created,
                      created*Create Step=created,
                      created*Complete Case Study=completed,
                      created*Report Case Study Structure=created,
                      created*Report Results=created,
                      completed*Report Results=completed,
```

Listing 3.5 The model representation of the Case Study project

The Case Study step

The Case Study object refers to several Case Study step protocol machines. The research team is required to fulfil the Case Study steps in chronological order. Such a step is fulfilled if one of the following conditions is met:

1) The name is provided of a file that contains the deliverable of the step. Examples of this are files that contain an inventory, a description or a diagram
2) Or, if a Step refers to measurements, all the referred measurements contain values

A Java call back routine Determines if the step can be marked as fulfilled.

In the diagram of this protocol machine in figure 3.5 two states are modelled.

- Created: adding data to the model's steps or measurements is still possible
- Completed: not additions of data or amendments of existing data is possible anymore

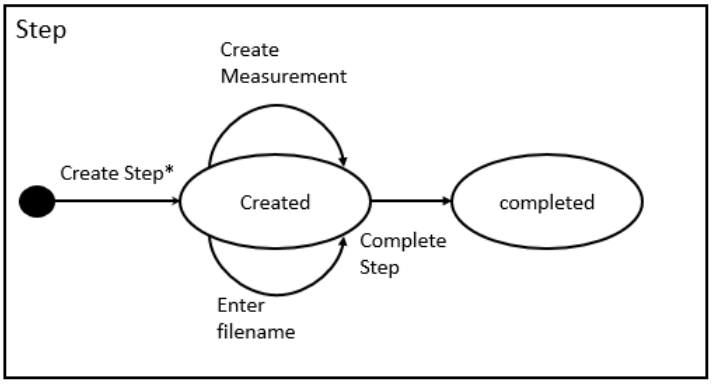


Figure 3.5 The protocol machine model for the Case Study step protocol machine

In listing 3.6 of the model definition there is room for the attribute featuring the filename of the file with data.

```
OBJECT Step
    NAME Step Name
    ATTRIBUTES      Step Name: String,
                    CaseStudy: CaseStudy,
                    File Name: String,
                    !Number of Measurements: String,
                    !Measurements Completed: String,
    INCLUDES        Step Completed Check,
    STATES          created,
                    completed,
    TRANSITIONS     @new*Create Step=created,
                    created*Create Measurement=created,
                    created*Enter filename=created,
                    created*Complete Step=completed,
```

Listing 3.6 Modelscope definition of the cast study step protocol machine

There is a behaviour associated call back routine that checks if the steps can be completed. The checks also cover the requirement if for all related measurements, values have been supplied.

The Case Study Measurement

A Case Study step can reference zero or many measurements. This is graphically displayed in figure 3.6.

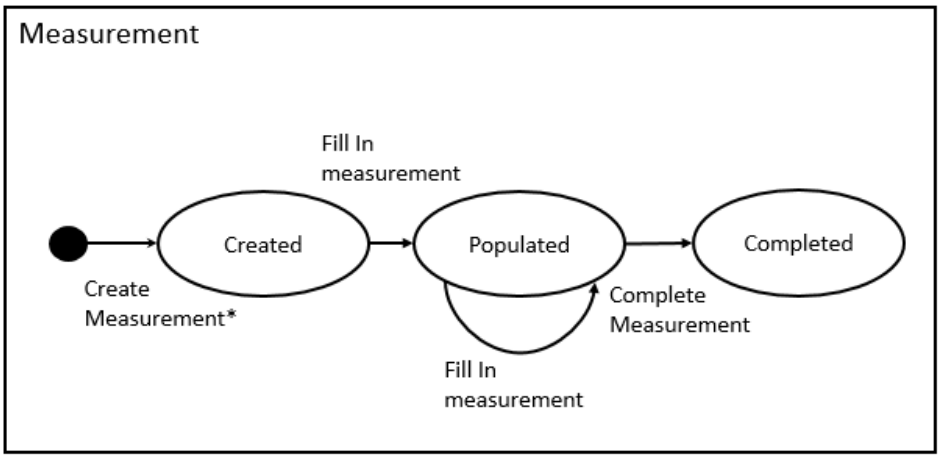


Figure 3.6 The protocol machine model for the for a Case Study measurement

The model's definition in Modelscope definition, is shown in listing 3.7.

```
OBJECT Measurement
    NAME Measurement Name
    ATTRIBUTES      CaseStudy Name: String,
                    Step: Step,
                    Measurement Name: String,
                    value: String,
    STATES          created,
                    populated,
                    completed,
    TRANSITIONS     @new*Create Measurement=created,
                    populated*FillIn Measurement=populated,
                    populated*Complete Measurement=completed,
                    created*FillIn Measurement=populated,
```

Listing 3.7 Modelscope definition of the measurement protocol machine

3.1.3. Additional routines for support

Around the model there are several provisions that arrange requirement fulfilment. These arrange proper segregation of user activities, automatic CSR project creation and reporting.

Generating the Case Study project from the Case Study type.

As stated before, the instantiation of the CSR project is done in a java call-back routine. In this routine the CST definition is instantiated into a CSR representation. To indicate the resemblance in object states and events the instantiated objects versus the source CST's objects, is displayed in figure 3.7.

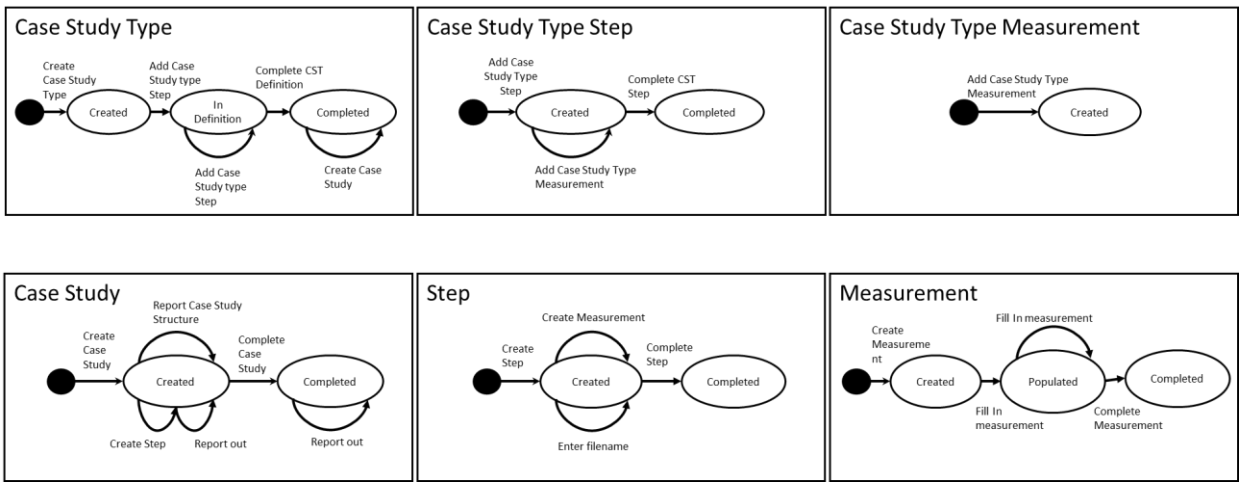


Figure 3.7 Copying the CST structure into an CSR occurrence

Definition of actors in the model

Within the setup of the model there is a segregation between the users that define the Case Study Type and the users that execute on the research project itself. In the tool, this distinction in roles is

expressed in the form of assigning different actors. The actors defined in the developed model are: Principal Scientist and Scientist.

The Principal Scientist designs the research approach and creates an instance of the Case Study Type. To do this, he or she would need to be authorized to the set of objects and events as listed in listing 3.8.

```
ACTOR Principal Scientist
    BEHAVIOURS  CaseStudyType,
                CaseStudyTypeStep,
                CaseStudyTypeMeasurement,
                CaseStudy,
    EVENTS      Create CaseStudyType,
                Add Case Study Type Step,
                Add Case Study Type Measurement,
                Complete CST Definition,
                Create CaseStudy,
                Report Out,
                Report Case Study Structure,
```

Listing 3.8 Modelscope definition of the actor Principle Scientist

As the Scientist executes the research initiative, he/she would require authorization on the objects and events, listed in listing 3.9.

```
ACTOR Scientist
    BEHAVIOURS  CaseStudy,
                Step,
                Measurement,
    EVENTS      Enter filename,
                FillIn Measurement,
                Complete Case Study,
                Report Out,
                Report Case Study Structure,
```

Listing 3.9 Modelscope definition of the actor Scientist

CSR project reporting.

The reporting capability is a key element of this model. It not only secures the adoption rate of the tool but also it supports quality criteria as Case presentation, Case Interpretation and Inter-coder agreement.

The Case Study Type Report shows the structure of the CST definition. After triggering it, it lists this definition in a file: “Case Study Type Report.txt”. An example of such a report is shown in Listing 3.10.

```
Report structure on Case Study Type: Visualizing a Gap of Changes versus analysis
    of the As-Is and the To-Be model
Case Study Type Description: Hoe houd je zicht op de veranderingen bij de vervanging
    van een spreadsheetapplicatie door een informatiesysteem.

Case Study Type Step  : Model the As-Is of the current situation
Case Study Type Step  : Determine motivation for changes
Case Study Type Step  : Model the To-Be from motivation and as is model
Case Study Type Step  : Model Gap of Changes from As-Is and To-Be
Case Study Type Step  : Determine the changes that have  been modelled and categorize them
Case Study Type Step  : Determine changes - from as-is vs. to-be - (meas.: 1 - 7)
    Case Study Type Measurements are:
    Measurement: Number of identified Changes (As-is vs. ToBe) - Oobsolete
    Measurement: Number of identified Changes (As-is vs. ToBe) - Onew
    Measurement: Number of identified Changes (As-is vs. ToBe) - Ochanged
    Measurement: Number of identified Changes (As-is vs. ToBe) - Rnew
    Measurement: Number of identified Changes (As-is vs. ToBe) - Robsolete
    Measurement: Number of identified Changes (As-is vs. ToBe) - Rborder
    Measurement: Number of identified Changes (As-is vs. ToBe) - Rchanged
Case Study Type Step  : Determine changes - changes from as-is vs. GoC - (meas.: 8 - 14)
    Case Study Type Measurements are:
    Measurement: Number of identified Changes (As-is vs. GoC) - Oobsolete
    Measurement: Number of identified Changes (As-is vs. GoC) - Onew
    Measurement: Number of identified Changes (As-is vs. GoC) - Ochanged
    Measurement: Number of identified Changes (As-is vs. GoC) - Rnew
    Measurement: Number of identified Changes (As-is vs. GoC) - Robsolete
    Measurement: Number of identified Changes (As-is vs. GoC) - Rborder
    Measurement: Number of identified Changes (As-is vs. GoC) - Rchanged
Case Study Type Step  : Interpret, categorize and conclude. Count changes per stakeholder per category
Case Study Type Step  : Analyze and interpret results
```

Listing 3.10 Example of the CST report

The structure report plainly shows the structure of the Case Study itself. It reports all found objects to a file with the standard name: “Case Study Structure.txt”.

The “report out” report, is more extensive and will not only show the model’s structure but also the values supplied in the Case Study steps as well as the measurements. An example snippet of this latter report is shown in listing 3.11.

```
Report on Case Study  : ROC - Den Bosch
Case Study Description: We are going to analyze if the changes better recognizable by visualizing a
    Gap of Changes then an analysis of the As-Is and the To-Be model with ROC - Den Bosch
Status of project is  : completed

Step: 01. Model the As-Is of the current situation
filename:   ROC - Den Bosch - As Model.vsd
            (This Step does not contain measurements.)

Step: 02. Determine motivation for changes
filename:   ROC - Den Bosch - Motivation for change.docx
            (This Step does not contain measurements.)

Step: 03. Model the To-Be from motivation and as is model
filename:   ROC - Den Bosch - To be model from motivatio-2-As Is.vsd
            (This Step does not contain measurements.)

Step: 04. Model Gap of Changes from As-Is and To-Be
filename:   ROC - Den Bosch - Gap of change from As-Is and To-Be.vsd
            (This Step does not contain measurements.)

Step: 05. Determine the changes that have  been modelled and categorize them
filename:   ROC - Den Bosch - Categorized modeled changes.docx
            (This Step does not contain measurements.)

Step: 06. Determine changes - from as-is vs. to-be - (meas.: 1 - 7)
filename:
Measurements are:
Measurement :   06-1. Number of identified Changes (As-is vs. ToBe) - Oobsolete Value:  4
Measurement :   06-2. Number of identified Changes (As-is vs. ToBe) - Onew  Value:  3
Measurement :   06-3. Number of identified Changes (As-is vs. ToBe) - Ochanged  Value:  1
Measurement :   06-4. Number of identified Changes (As-is vs. ToBe) - Rnew  Value:  5
Measurement :   06-5. Number of identified Changes (As-is vs. ToBe) - Robsolete Value:  9
Measurement :   06-6. Number of identified Changes (As-is vs. ToBe) - Rborder   Value:
Measurement :   06-7. Number of identified Changes (As-is vs. ToBe) - Rchanged  Value:  17
    The sum of these is:    39
    The number of Measurements: 7
    Average of values is: 5.571428571428571

Step: 07. Determine changes - changes from as-is vs. GoC - (meas.: 8 - 14)
```

Listing 3.11 Example report out of a CSR project

3.2. Testing the protocol model

When testing the model, the first step is to look at how to define the CST. After this a CSR project is derived that then is used to add project deliverable data into it. Eventually the available reports are run. This exercise is eventually done with test data.

3.2.1. Using the executable model

The tool platform interacts with the user over a web interface as shown in figure 3.8. The following manual helps the user in using the model:

Overview: the webpage for managing a CSR projects has three zones:

- The left section lists the model attributes: actors, objects and instances of those objects
- The middle section lists the attributes of the selected instances
- While on the right side the applicable events, and attribute user interaction fields can be found

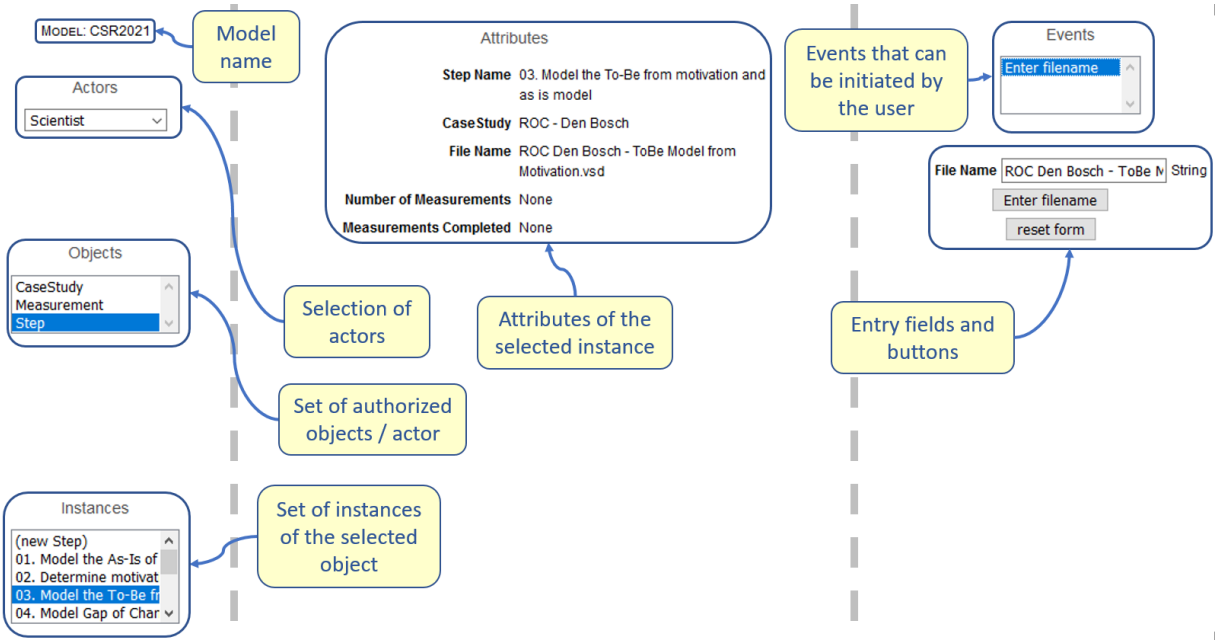
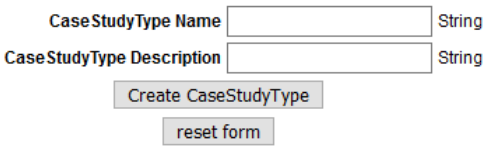
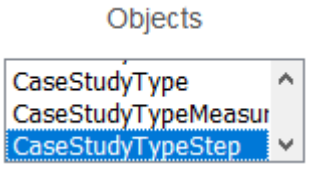
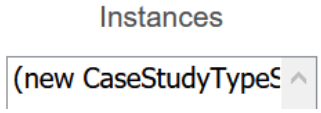
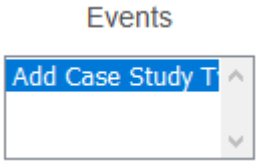
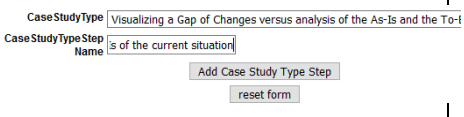
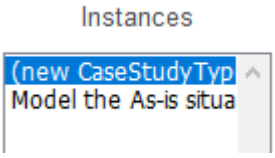


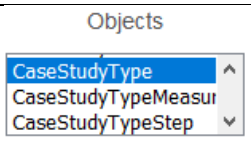
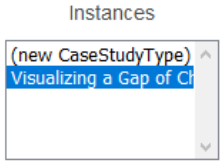
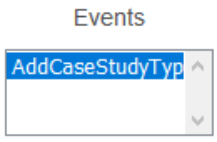
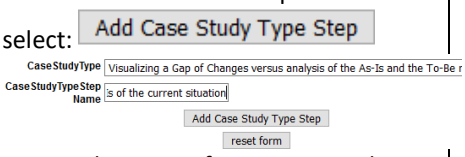
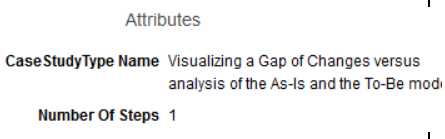
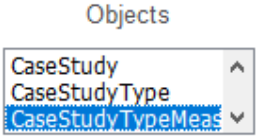
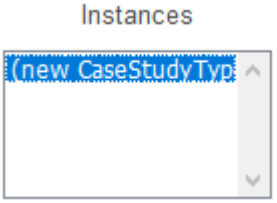
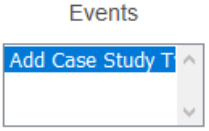
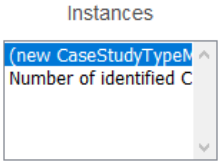
Figure 3.8 Modelscope user screen

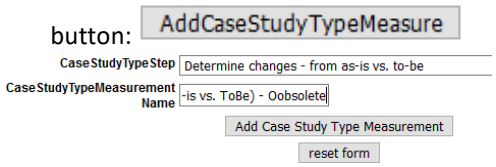
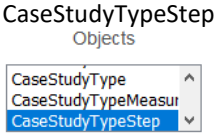
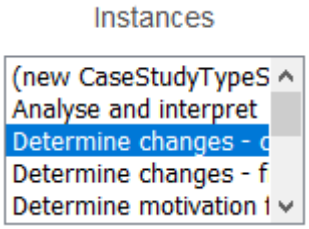
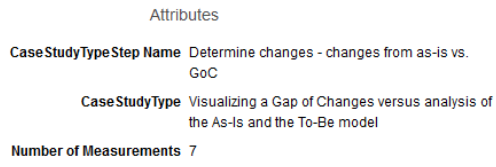
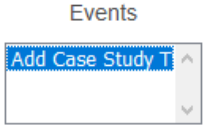
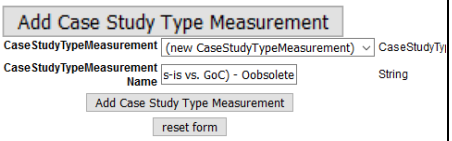
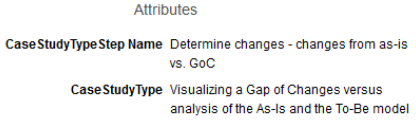
The steps required to define and manage a CSR project are listed in table 3.1.

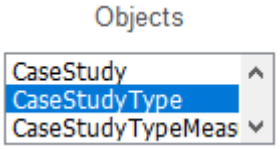
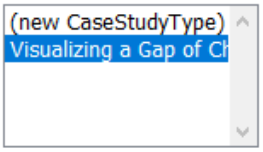
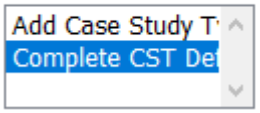
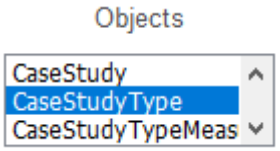
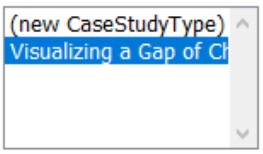
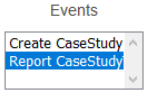
Table 3.1 Define the CST template in the tool

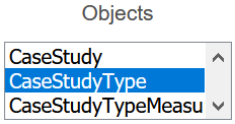
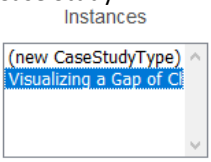
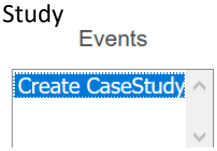
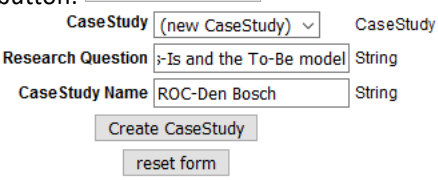
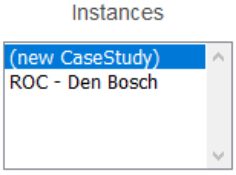
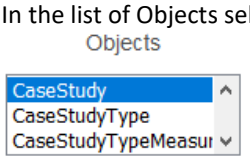
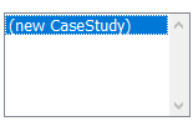
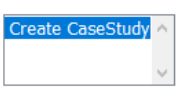
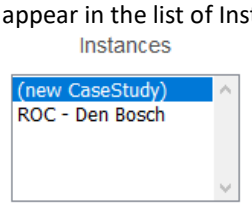
Step	Action	How to	Remarks
0	Preparation: select actor Principal Scientist	1. Select actor: Supervisor: Actors Principal Scientist	The Principal Scientist role is authorized and required to create a template protocol for carrying out the CSR. Then an individual case Study is created from this template
1	Create a Case Study Type (actor: Principal Scientist)	1. In the objects: select Case Study Type Objects CaseStudy CaseStudyType CaseStudyTypeMeasu 2. In Instances select: (news CaseStudyType) Instances (new CaseStudyType) 3. In Events select: Create CaseStudyType Events Create CaseStudy 4. A dialog box appears where you can add the name of the Case Study Type and the Case Study Type Description:	1. The created CaseStudyType will appear in Instances: Instances (new CaseStudyType) Visualizing a Gap of C

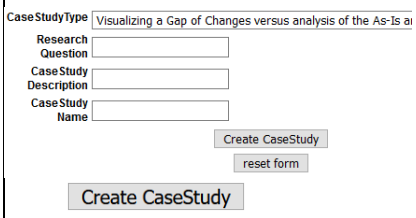
Step	Action	How to	Remarks
		 5. Enter the name and description of the Case Study Type in the box and select Create CaseStudyType	
2a	Create Case Study Type Step (Actor: Principal Scientist)	There are two scenarios how to add a Case Study Type Step to the Case Study Type: **Scenario 1:** 1. In Objects: select CaseStudyTypeStep  2. In Instances: select (new CaseStudyTypeSelect)  3. In Events: select Add Case Study Type Step  4. Select the appropriate Case Study Type. 5. Enter the Case Study Type Step name in the appropriate field and click the button: Add Case Study Type Step  6. Repeat this action for every step that needs to be added	1. The Case Study Type Steps need to be added to the Case Study Type in chronological order. It is this order that these steps will also show in the later created Case Study and reporting 2. The newly created Case Study Type Step will show in the list of instances 
2b	Create Case Study Type Step (scenario 2)	**Scenario 2:** 1. In Objects select: CaseStudyType	1. Here the added CaseStudyTypeStep is not seen as the focus is on Object: CaseStudyType. In the Attributes section of the screen one can see

Step	Action	How to	Remarks
	(Actor: Principal Scientist)	 2. In Instances select: the CaseStudyType where the Case Study Type Step needs to be added  3. In Events select: AddCaseStudyType  4. In the field CaseStudyTypeStep Name: enter the name of the step to add and select:  5. Repeat this action for every step that needs to be added	that the number Of Steps has incremented with 1 
3a	Add Case Study Type Step measurements to Case Study type Steps (Actor: Principal Scientist)	There are two scenarios how to add a Case Study Type Step to the Case Study Type: **Scenario 1:** 1. In Objects: select CaseStudyTypeMeasurement  2. In Instances: select (new CaseStudyTypeMeasurement)  3. In Events: select Add Case Study Type Measure  4. Select the CaseStudyTypeStep where you want to relate the measure to (see	1. You can link more than one measurement to a single step 2. The newly added Case Study Type Step measurement will show in the list of instances 

Step	Action	How to	Remarks
		remark), enter the measure name in the appropriate field and click the button:  5. Repeat this action for every measurement to be added.	
3b	Add Case Study Type Step measurements to Case Study type Steps (Actor: Principal Scientist)	Scenario 2: 1. In the list of objects: select: CaseStudyTypeStep  2. In the list of Instances: select the CaseStudyTypeStep where you want to add a measurement.  To verify that the correct CaseStudyTypeStep is addressed the name can be validated in the middle section of the screen:  3. In the list of Events select: Add Case Study Type Step measurement  4. Enter the name of the CaseStudyType Measurement in the appropriate field and select:  5. Repeat this action for every measurement to be added.	1. The added measurement will not be visible on the screen as this is not in focus. The Select CaseStudyTypeStep is selected and in the middle section of the screen the counter: Number of Measurements will have been incremented: 

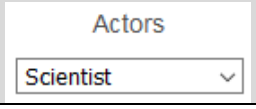
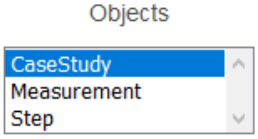
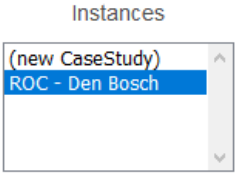
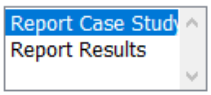
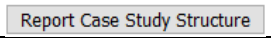
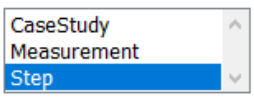
Step	Action	How to	Remarks
4	Complete Definition phase of the CaseStudyType (Actor: Principal Scientist)	To complete the definition phase of the Case Study Type you indicate this as completed: 1. In Objects: select the option CaseStudyType  2. In Instances: select the Case Study Type you want to set "completed"  3. In Events: select the event: Complete CST Definition  4. Click the button: 	1. After the Case Study Type has been "completed", it is no longer possible to change it. 2. After this the events for changing the Case Study Type are no longer enabled
5	Report out Case Study Type Structure	The Case Study Type Structure can be reported to a file: "Case Study Type Report.txt". 1. In Objects: select the option CaseStudyType  2. In Instances: select the Case Study Type you want to set "completed"  3. In Events: select the event: Report CaseStudyType  4. Click the button: 	The report can be found in the file: "Case Study Type Report.txt"

Step	Action	How to	Remarks
6a	Create Case Study (Actor: Principal Scientist)	To instantiating a Case Study Type you have two options to create a Case Study. Scenario 1: 1. In Objects: select the option CaseStudyType  2. 2. In Instances: select the Case Study Type for which you want to create a Case Study  3. In Events: select the event: Create Case Study  4. Enter the Text for the Research Question and the CaseStudy Name in the appropriate fields and click the button: Create CaseStudy 	1. This then will create a Case Study with all the steps and measures as defined in the Case Study Type. You can find the Case Study in the set of Case Study Instances of the object Case Study:  2. After this the Principal Scientist can hand over activities to the Researcher for carrying on the CSR.
6b	Create Case Study (Actor: Principal Scientist)	Scenario 2: 1. In the list of Objects select: CaseStudy  2. In the list of Instances select: (new CaseStudy)  3. In the list of Events select: Create CaseStudy  4. Enter the Text for the Research Question, the Case Study Description	1. The created Case Study will appear in the list of Instances:  2. After this the Principal Scientist can hand over activities to the Researcher for carrying on the CSR.

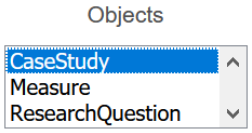
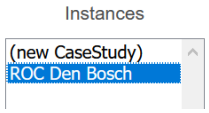
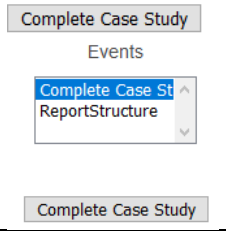
Step	Action	How to	Remarks
		and the Case Study Name in the appropriate fields and click the button: 	

When executing on the CSR project, the user enters the data into the model, using the steps listed in table 3.2.

Table 3.2 Entering data into the Case Study project support tool

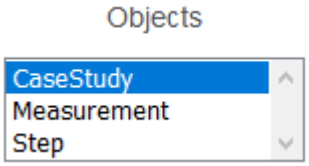
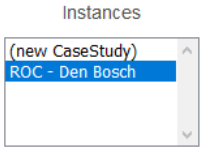
Step	Action	How to	Remarks
1	Preparation: select actor Scientist	1. Select actor: Scientist 	
2	Report structure of Case Study (Actor: Scientist)	1. In Objects: select CaseStudy  2. In Instances: select the appropriate CaseStudy that you want to conduct  3. In Events: select the event Report Case Study Structure  4. Click: 	The structure of the Case Study will be filed in the file: “Case Study Structure.txt” in the root folder of the tool
3	Add filename deliverable to steps (Actor: Scientist)	1. In objects select: Step  2. In Instances: select the appropriate step	1. In the list Instances of steps you will notice that the steps are being preceded with a number. This is to safeguard that the Step is shown in the chronological order. 2. After this the filename will be displayed in the attributes of the instance. This screen section will also show the statistics of

Step	Action	How to	Remarks
		Instances (new Step) 01. Model the As-Is of 02. Determine motivat 03. Model the To-Be fr 04. Model Gap of Char 3. In events: select Enter Filename Events Enter filename 4. Enter the file name from the deliverable in the field "File Name" and click Enter filename File Name ROC Den bosch - as is r String Enter filename reset form 5. Repeat this action for all steps where deliverables in files are required.	available measurements and completed measurements Attributes Step Name 01. Model the As-Is of the current situation CaseStudy ROC - Den Bosch File Name ROC Den Bosch - as is model.vsd Number of Measurements None Measurements Completed None 3. In case the filename needs to be corrected, this can be done with the same procedure
4	Add values to the several Measures (Actor: Scientist)	1. In Objects: select Measurement Objects CaseStudy Measurement Step 2. In Instances: select the measurement for which you want to add the value Instances (new Measurement) 06-1. Number of identi 06-2. Number of identi 06-3. Number of identi 06-4. Number of identi 3. In Events: select the event FillIn Measurement Events FillIn Measuremer value String FillIn Measurement reset form 4. Fill in the value in the appropriate field and click: FillIn Measurement 5. Repeat this action for all listed measurements	1. The value will show in the attributes list Attributes CaseStudy Name ROC - Den Bosch Step 06. Determine changes - from as-is vs. to-be Measurement Name Number of identified Changes (As-is vs. ToBe) - Robsolete value 9 2. 3. For all listed measurement within the step a value must be supplied before the step is completed 4. If a value needs to be changed, this can be done with the same procedure 5. After having supplied values for all measurements and having all steps completed the Attributes screen of the Case Study object will indicate the number of steps and steps completed as equal: Attributes CaseStudy Name ROC - Den Bosch Research Question Are the changes better recognizable by visualizing a Gap of Changes then an analysis of the As-Is and the To-Be model CaseStudyType Visualizing a Gap of Changes versus analysis of the As-Is and the To-Be model Number Of Steps 9 Number Of Steps Completed 9

Step	Action	How to	Remarks
5	Finish Case Study (Role: Scientist)	After all steps have been completed, the Case Study can be set as completed. To mark the Case Study as completed. 1. In Objects: select the object CaseStudy  2. In Instances: select the Case Study that is to be marked completed  3. In Events: select the event Accept Case Study and click: 	1. After completing this action, the Case Study can no longer be changed. 2. In the list of Events only the Report Out event is available for selection 

To complete the process the scientist can export the “report out” report. For this the steps as shown in table 3.3 apply.

Table 3.3 Reporting the Case Study project

Step	Action	How to	Remark
13	Report Results of the Case Study (Role: Principal Scientist / Scientist)	After marking a Case Study completed, a report can be generated to be used for evaluation purposes. There is one report available on the level of Case Study. This will list all available attributes per Step and per Measurement. The report can be issued by both application roles: Principal Scientist and Scientist. To issue the report: 1. In Objects: select the object ResearchQuestion  2. In Instances select the Case Study for which the reports need to be generated:  3. In the event select the event: Report Results  4. Click the button: 	1. The resulting report will appear as “Case Study report.txt” in the root folder.

3.2.2. The test cases

To validate the model several tests have been conducted. By proper translation of the requirement into a possible test, for every requirement at least one test scenario has been defined. This test scenario states the requirement, the prerequisites for the test, the actions for the tester to conduct the test and the checks to fulfil by the tester. Two basic data sets were used to validate the technical functionality of the model. These can be found in appendix B. Two more elaborate sets were used to see how the model would behave in a simulated live environment.

For two similar case studies the Case Study Type data has been configured into the model. After this, the two Case Study projects have been created and the existing case results have been entered into the tool. For this approach the Theses from Bos (Bos, 2018) and Haddouchi (Haddouchi, 2018) were used. This test data is also available in Appendix B. The two CSR theses investigated the hypotheses if stakeholders in an enterprise can recognize changes in an IT architecture better when looking at the Gap of changes then solely looking at the analysis of the as-is model versus the to-be situation. In this research several documents had to be produced, but also the researchers collected numerous changes, recognized by the stakeholders. Having this combination of documents and measurements in play made the two theses suitable as pilot material.

In appendix C an extensive description of the tests conducted, their relation to the requirements and the achieved outcome, is provided.

Eventually the tool would provide the capability to act on the level of the total initiative, describing a general protocol for CSR conduct. As an example this has been tested for the phase CSR design. The tool's output can be seen in listing 3.12

```
Report structure on Case Study Type: Case Study design
Case Study Type Description: first stage of CSR conduct: Case study design: objectives are
    defined and the case study is planned

Case Study Type Step  : The Case and its units of analysis are available
Case Study Type Step  : Clear objectives, preliminary research questions, hypotheses (if any) are defined in advance
Case Study Type Step  : The theoretical basis—relation to existing literature or other cases— is defined
Case Study Type Step  : The Author's intentions with the research are made clear
Case Study Type Step  : The case study is adequately defined (size, domain, process, subjects…)
Case Study Type Step  : if the cause-effect relation under study, it possible is to distinguish the cause from other
    factors using the proposed design
Case Study Type Step  : The design involves data from multiple sources (data triangulation), using multiple methods
    (method triangulation)
Case Study Type Step  : There is  a rationale behind the selection of subjects, roles, artifacts, viewpoints, etc.?
Case Study Type Step  : The specified case is relevant to validly address the research questions (construct validity)
Case Study Type Step  : The integrity of individuals/organizations is taken into account
```

Listing 3.12 listing of an CST definition of the CSR design phase

4. The testing results in relation to the Case Study quality requirements

The model is tested to evaluate if it meets the requirements as collected during the analysis and design phase of the initiative.

The developed model has the following characteristics:

- It is capable of capturing a Case Study Type template that can be used to generate a tool that supports conducting an individual CSR project
- It can generate this supporting tool in an automated manner without manual intervention
- The generated tool, as its templated parent, contains project steps and has the capability to capture delivered documents as well as measurements
- It also checks that these deliverables are available before enabling the researcher to proceed closing the project
- The model features a set of reports on the project structure as also on the deliverables within the project
- Additionally, the model, when it is required, supports a differentiation on actors in terms of authorizations and option visibility

The tests indicate that the model adheres to the requirements as compiled during the analysis phase. The developed tool does not include the checklist as presented by Tellis, Host et al.(Host & Runeson, 2007; Tellis W. M., 1997). However, it is capable of defining a CST specific version of this checklist. This list can be reused for project instances of this specific type.

Formalizing the tool or an equivalent in a research organisation will lead to a situation where the researchers execute on the research project in a predefined canvas.

5. Discussion, reflections and recommendations

5.1. Reflections on the results

The main contributions of this initiative are the collection of requirements for an executable protocol model of a CSR project, design of this model and its testing and analysis.

The protocol for supporting a CSR project would be one that has a set of features:

1. In order to Support CSR projects, the research team would develop a set of protocols, tailored for the specific CSR types. These protocols then can be used to support individual projects and add insights to this project by listing requirements and deliverables
   - It is this set of prepared protocols that embody the support to the research team. The protocols themselves are the result of gathering extensive knowledge on research project management in general and that of CSR project management in particular. This feature also supports a practice of continuous improvement, when after project evaluation, certain amendments to the protocol become evident and are factored into a successive version of the support tool
   - Added value also becomes evident when populating a repository of CST definitions, stipulating protocols for individual CSR project types and purposes.

2. The individual protocol would support the research team by defining and sequencing the steps, that the research team hast of fulfil in order to best complete the project
   - Having this set of stages or steps available on beforehand relieves the research team from the necessity to compile their protocol themselves. This will also mitigate the risk of project's scope creep, where the project tends to glide of the topic, initially set out, that the team intended to research. This partly also addresses the quality considerations raised by Goffin (Goffin et al., 2019), as this tool also supports in structuring the project conduct as well as the reporting

3. Capturing the deliverables and being able to report on them is an important feature of the supportive protocol
   - For multiple reasons reporting capabilities within the tool is essential. Most important, extensive reporting capabilities will enhance the adoption of the tool by research teams. The reports can be used to gain insights into the project's structure, but more important than this, the report can also be used to have research data made available to other analysis tools and platforms

The tool is also expected to support a set of quality criteria, listed by Goffin et al. (Goffin et al., 2019) where in particular the quality criteria in the categories data collection and data analysis. The criterion Triangulation is supported by the tool's capability to systematically collect data as also, to coherently report on it. This also can be stated for the quality criteria review and validation of evidence, transparency of data collection, case presentation, case Interpretation and Inter-coder agreement. This list of quality criteria representing 6 out of 10 could be regarded as a significant portion.

Regarding the scope of the tool, one can argue that as the user is able to define individual protocols for specific Case Study Types, the set of use cases in which the tool can be applied is versatile. While the initial intent was to develop the model for single CSR application, applying it to a multi CSR project can also be considered.

The Constraints of this developed model relate to the fact that it is indeed a model. Other applicable terms would be a proof of concept or a pilot setup. The tooling lacks certain functionalities, currently expected in tooling, used in business environments. The constraints in the use of this tool naturally emerge from this gap.

5.2. Reflections on validity, reliability and ethical aspects

For evaluating this research initiative on its quality design, several challenges are raised and addressed.

Internal validity

The internal validity is achieved by comparison of the two models: goal model and the protocol model. Moreover, the check lists found in literature was used to validate both models.

External validity

In the research setting a problem was presented a based on this, a possible solution was developed in the form of an executable model. The model was tested against a basic data set, to establish functional conformity, and then with a data set, taken from the “archive” that would simulate actual usage of the tool in practice. The tool itself was never used in actual practice as a CSR support tool by a research team. Actual practical use, or even adding additional archive CSR examples to the test set, might reveal additional requirements to the set, used for this development. When it comes to evaluating external validity and thus outcome generalization, no concrete states can be made.

Reliability

The method used is a well-recognized research method to explore business processes. As the tooling also is publicly available, one can safely assume that the model developed, given the fact that other researchers start with the same requirements, will be similar. The aspects to address then is, collecting the requirements. In this initiative, compiling these was part of literature review and practicing the proposed method. Here, there is the risk of subjective interpretation of the requirements. Every attempt was made to limit this risk by iterating on reasoning during the requirement collection as also during the rapid prototyping phase.

Ethical Aspects

As this initiative did not handle any sensitive data, there is no the risk for Ethical issues. Also, this initiative is aiming to support research teams in the efforts to improve on the quality of their daily work, without the aim to compromise their employment or integrity in any way.

5.3. Recommendations for future research

The tool presents some promise as it also exhibits area's to be further researched.

Measuring quality of CSR projects: In their article Goffin et al. (Goffin et al., 2019) went into their view on quality measurements and metrics of conducting and reporting Case Study projects. Basically Gibbert et al. (Gibbert et al., 2008) presented something similar in their analysis. This approach was applied after the fact, as it involved analysing the result of the project, the report, or in their case, the article. Having quality considerations being brought in, earlier in the process, might lead to a better result. This would then imply quality measures being applied "before or during the fact".

When considering quality in CSR projects a number of qualifiers can be considered (Gibbert et al., 2008; Mark Saunders, Philip Lewis, 2016):

- Reliability of the research refers to the replication and consistency of the procedure or project;
- Validity refers to the appropriateness of the measures used to conduct the method. Here a number of sub validity qualifiers are found:
    - Measurement validity: is participation of the actors in the process unbiased and independent?
    - Internal validity: are the conclusions in line with the data being collected during the project?
    - Construct validity: is the study investigating what it initially claimed it would be?
    - External validity: can the study's research findings be generalized?

Considering an automated system that would scrutinize the results of a CSR project on these qualifiers could prove challenging and was beyond the scope of this study. Looking into the supportive tools that might support a structural evaluation of a CSR project after completion, could be subject of future research.

It was also stated that the tool would support in 6 out of 10 quality criteria as listed by Goffin et al. When considering further extension of this tool or equivalent to a research environment, evaluating if the tool would also support the additional 4 criteria, theoretical foundation, pilot study, theoretical sampling and reflecting on validity and reliability, could be considered.

From the perspective that this model acts as a proof of concept, it gives insights to the characteristics that such a tool should possess. The possibility of considering the guidance that it would supply to research teams in structuring their research project efforts would be one of the major benefits of such a tool. It would also support the organisation by featuring the segregation of duties between the actor that defines the protocol and the researcher that carries out the research itself. Next, the tool can support a quality improvement practice where successively the teams go through a cycle of plan – do – study – act.

However, there are also a set of limitations to the developed tool exhibits.

- Platform: the tool runs on a java stack and when running, the tool behaves as a local webserver, it might present a challenge to factor this solution into an existing IT infrastructure standard. The tool's design also would hinder, easy sharing the application throughout the organisation. This implies that the application can only be used locally. Even running it on a network drive would impose a challenge for sharing it.

Also, the tool currently does not support interfaces to existing platforms for office authoring, mail- and content management tools. If it would have this capability, it would better support capturing the project deliverables as also it would feature sharing the research data in other analysis platforms.

- Knowledge management: while supporting a continuous improvement mechanism by implementing gained knowledge into a next version of the Case Study Type template, the tool does not support features that would facilitate in absorbing this gained knowledge easily. There is no way to edit a CSR type definition and tampering with the underlying repository is advised against. This also addresses a possible desire of having external application project repositories linked to add research project management knowledge to the tool.

- Configurability: the platform, that the tool was developed on, has limited capability of configuring the user interface. The screen is not tailorable to meet organisation publication requirements. As another example, the tool tends to show the whole set of instances of an object where the user would only like to see a filtered sub-set. Adding such a filter is likely to be cumbersome, if not impossible.

Some other areas to investigate touch on the fact that the application of the tool as presented might only be opportune in an environment that supports the specific requirements, the tool has on its platform. Also, the research team would have to be willing to cope with the limitations stated earlier.

On the other hand, learning from the model, some areas of improvement and research become obvious.

- The developed model has not been evaluated in a real live environment. Such an evaluation would give insights into the degree of appreciation of such a tool with research teams. Such teams would not be easily convinced to start using this tool, even for evaluation purposes as they primarily are concerned with conducting research within their own field. What could be done, to gain more insights into this, is to do a survey with a number of research teams to get their input on the ideas behind the model.

- The model could present an initial design for an equivalent solution, based on an industry standard document management platform as Microsoft SharePoint or equivalents. Such a solution would have a broad set of interfacing capabilities to the organisation's internal and external platforms and support further analysis, alignment and communication amongst team members and peering project teams. How this then would be integrated into knowledge management practices would be subject for further research.

- When a platform for the former noted industry standard is not available, an initiative could be started to search for a platform, equivalent to the Modelscope solution that overcomes the earlier stated limitations.

- Currently the tool leaves the activity of evaluating a completed project totally up to the user without further support. Having a feature that helps the user to step through the research steps and enabling him/her to add notes and so forth, could certainly aid to the user's convenience.

References

Baxter, P., & Jack, S. (2008). Qualitative Case Study Methodology: Study Design and Implementation for Novice Researchers. *The Qualitative Report, 13*(4), 544–559. Retrieved from https://nsuworks.nova.edu/tqr/vol13/iss4/2

Bos, M. (2018). *Evaluatie van twee op ArchiMate gebaseerde visualisatietechnieken*. Open University.

Easton, G. (2010). Critical realism in case study research. *Industrial Marketing Management*, *39*(1), 118–128. https://doi.org/10.1016/j.indmarman.2008.06.004

Gagnon, Y.-C. (2010). *The Case Study as Research Method: A Practical Handbook*. Quebec: Presses de l'Universite du Quebec.

Gibbert, M., Ruigrok, W., & Wicki, B. (2008). What Passes as a Rigorous Case Study? *Strategic Management Journal*, *29*(13), 1465–1474. Retrieved from http://www.jstor.org/stable/40060241

Goffin, K., Åhlström, P., Bianchi, M., & Richtnér, A. (2019). State-of-the-art: The quality of case study research in innovation management. *Journal of Product Innovation Management*, jpim.12492. https://doi.org/10.1111/jpim.12492

Haddouchi, E. M. (2018). *Visualisatie van wijzigingen in ArchiMate*. Open University, The Netherlands.

Host, M., & Runeson, P. (2007). Checklists for software engineering case study research. *First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007)*, 479–481. IEEE.

Houghton, C., Casey, D., Shaw, D., & Murphy, K. (2013). Rigour in qualitative case-study research. *Nurse Researcher*, *20*(4), 12–17. https://doi.org/10.7748/nr2013.03.20.4.12.e326

Mark Saunders, Philip Lewis, A. T. (2016). *Research methods for business Students* (7th ed.). Harlow.

McNeile, A., & Simons, N. (2006). Protocol modelling: A modelling approach that supports reusable behavioural abstractions. *Software & Systems Modeling*, *5*(1), 91–107. https://doi.org/10.1007/s10270-005-0100-7

McNeile, A., & Simons, N. (2012). Metamaxim. Retrieved from Website website: http://www.metamaxim.com

Noor, K. B. M. (2008). Case Study: A Strategic Research Methodology. *American Journal of Applied Sciences*, *5*(11), 1602–1604.

Roubtsova, E. (2016). *Interactive Modeling and Simulation in Business System Design*. https://doi.org/10.1007/978-3-319-15102-1

Runeson, P., & Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, *14*(2), 131–164. https://doi.org/10.1007/s10664-008-9102-8

Tellis W. M. (1997). Application of a Case Study Methodology. *The Qualitative Report*, *3*(3), 1–19. Retrieved from https://nsuworks.nova.edu/tqr/vol3/iss3/1

Verner, J. M., & Abdullah, L. M. (2012). Exploratory case study research: Outsourced project failure. *Information and Software Technology*, *54*(8), 866–886. https://doi.org/10.1016/j.infsof.2011.11.001

Yin, R. K. (1999). Enhancing the Quality of Case Studies in Health Services Research. *HEALTH SERVICES RESEARCH*, *34*(5).

List of Figures

List of Listings

List of Tables

Appendices

A. Detailed description of the model.

The CST protocol machine

The protocol machine of a CST (Figure A.1.) is an abstract model of a Case Study Type definition. The research team supervisor starts the definition of the template by creating an instance of this protocol machine.

The circles in this protocol machine diagram indicate the various states that the machine can reside in while the arrows show the events that potentially trigger transitions from one state to another.

During the Case Study Type template definition phase this protocol machine sequences through several states:

- **Created**: reached after a new CST is created.
- **In definition**: the activities are ongoing for adding steps and measurements to the model. This state has been introduced as an intermediate state. This is needed as it allows the user to add steps and measurements before confirming that the definition phase is finalized. The intermediate state, “in definition”, also is an administrative provision, required to support the requirement that the user can no longer add measurements to the definition after it is set to the state completed.
- **Completed**: the completion of the Case Study Type (template), where it is ready for copying into a new Case Study project definition. When a Case Study Type is in this state it is no longer possible to add Case Study Type Steps as also measurements to the Case Study type definition. The event complete CST definition triggers a set of events, sent to the related Steps and measurements, to also set their status to the same.

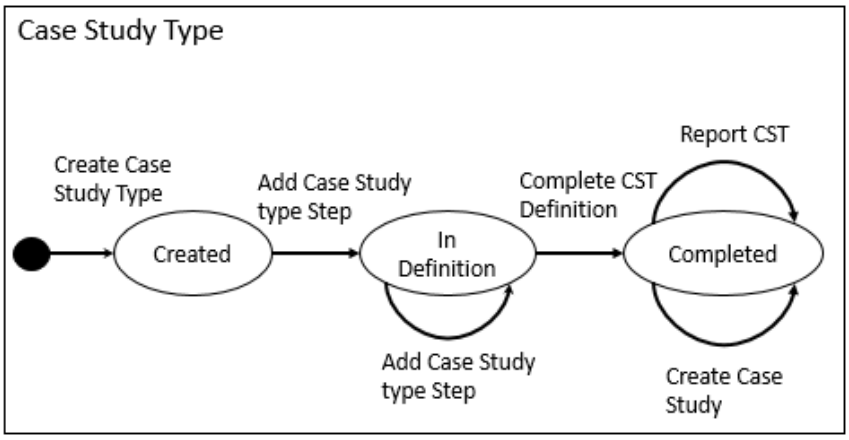


Figure A.1. Graphical representation of a CST

The Case Study Type has several attributes. The name of the Case Study Type is one, while also an attribute is defined that captures the number of steps in the Case Study Type. This is to facilitate extra information to the user on the number of steps in the definition.

In listing A.1. an exclamation mark precedes one of the attributes. This indicates that a call back routine in Java code is added to facilitate capturing the “number Of Steps” Attribute.

```
OBJECT CaseStudyType
    NAME CaseStudyType Name
    ATTRIBUTES      CaseStudyType Name: String,
                    CaseStudyType Description: String,
                    !Number Of Steps: String,
    STATES          created,
                    in Definition,
                    completed,
    TRANSITIONS     @new*Create CaseStudyType=created,
                    created*Add Case Study Type Step=in Definition,
                    in Definition*Add Case Study Type Step=in Definition,
                    in Definition*Complete CST Definition=completed,
                    completed*Report CaseStudyType=completed,
                    completed*Create CaseStudy=completed,
```

Listing A.1. Modelscope Definition of the CST protocol machine

In the Java call back routine in listing A.2., these number of steps are counted and returned into the proper attribute. The routine runs a query on related steps and determines the length of the responded array. The variable is set to string to accommodate for the value “None”.

```
package CSR2021;

import com.metamaxim.modelscope.callbacks.*;

public class CaseStudyType extends Behaviour {

    public String getNumberOfSteps () {
        String nrSteps = "";
        Instance [] setOfSteps = this.selectByRef("CaseStudyTypeStep","CaseStudyType");
        if (setOfSteps.length == 0)
            nrSteps = "None";
        else
            nrSteps = "" + setOfSteps.length;
        return nrSteps;
    }
}
```

Listing A.2. Calculating the number of steps in a CST template

A second call back routine, in listing A.3. as related to this protocol machine is triggered by the Complete CST definition event. This routine not only arranges the status for this protocol machine to Completed but it also selects all related Case Study steps and measurements and sets those to the status completed as well. In this way, the protocol machine secures that, after the definition phase is conclude no steps and measures can be added any more.

```
package CSR2021;

import com.metamaxim.modelscope.callbacks.*;

/*
Complete a case Study Type and all its CST Steps
When then the whole stucture is in a "completed" state it is no longer
possible/allowed to add Cast Study Type Steps (and CST Measurements)
*/

public class CompleteCSTDefinition extends Event {

    public void handleEvent() {
        Instance thisCaseStudyType = this.getInstance("CaseStudyType");
        Instance[] setOfCSTSteps =
            thisCaseStudyType.selectByRef("CaseStudyTypeStep","CaseStudyType");
        if (setOfCSTSteps.length != 0 ){
            for (int i=0; i< setOfCSTSteps.length; i++) {
                Event completeCSTStep = this.createEvent("Complete CST Step");
                completeCSTStep.setInstance("CaseStudyTypeStep", setOfCSTSteps[i]);
                completeCSTStep.submitToModel();
            }
        }
        Event completeCST = this.createEvent("Complete CST Definition");
        completeCST.setInstance("CaseStudyType", thisCaseStudyType);
        completeCST.submitToModel();
    }
}
```

Listing A.3. Completing the CST when definition is finished

The CST step protocol machine

The CST step protocol machine, as modelled in figure A.2., is the next in the set that defines the Case Study Type template that the research team supervisor must complete to define the CSR project of this type. The sequence, in which the steps must be completed is captured as well.

The CST Step protocol machine has two states:

- **Created**: measurements can be added to the Case Study Type definition. These measurements then are linked to the Case Study Type Step.
- **Completed**: In this status it is no longer possible to add Case Study measurements to the Case Study Type Step.

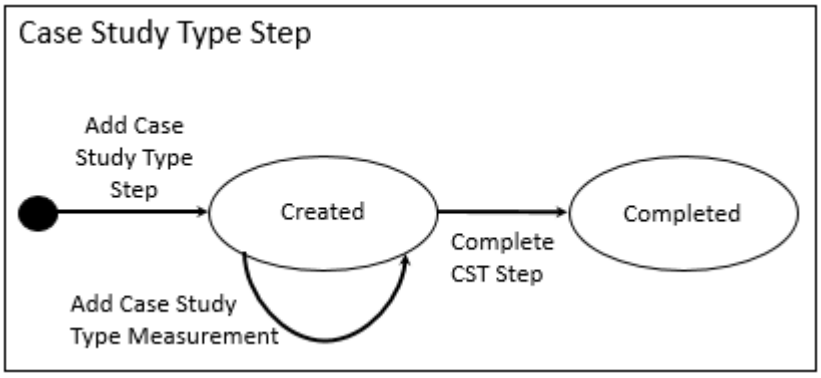


Figure A.2. Graphical representation of a CST Step

In the Modelscope definition in listing A.4., the exclamation mark before the attribute: Number of CST measurements refers to a Java call back routine that calculates the number of measurements related to this Case Study Type Step.

```
OBJECT CaseStudyTypeStep
    NAME CaseStudyTypeStep Name
    ATTRIBUTES      CaseStudyTypeStep Name: String,
                    CaseStudyType: CaseStudyType,
                    !Number of CST Measurements: String,
    STATES          created,
                    completed,
    TRANSITIONS     @new*Add Case Study Type Step=created,
                    created*Add Case Study Type Measurement=created,
                    created*Complete CST Step=completed,
```

Listing A.4. Modelscope definition of the CST step protocol machine

In the call back routine in listing A.5. the number of referenced measurements is calculated to facilitate the user with additional information on the size of the template.

```
package CSR2021;

import com.metamaxim.modelscope.callbacks.*;

public class CaseStudyTypeStep extends Behaviour {

    public String getNumberOfMeasurements () {
        String nrMeasures = "";
        Instance [] measures = this.selectByRef("CaseStudyTypeMeasurement","CaseStudyTypeStep");
        if (measures.length == 0)
            nrMeasures = "None";
        else
            nrMeasures = "" + measures.length;
        return nrMeasures;
    }
}
```

Listing A.5. Calculating the number of CST measurements

The CST measurement's protocol machine

When the user opts to add a Case Study Type Step measurement to a step an instance is created of the protocol machine as presented in figure A.3. This protocol machine only resides in the state Created. As this is the lowest level object, in what could be regarded as the Case Study – Step –

Measurement hierarchy, there is no additional logic added to this object. This is also reflected in the Modelscope definition in listing A.6.

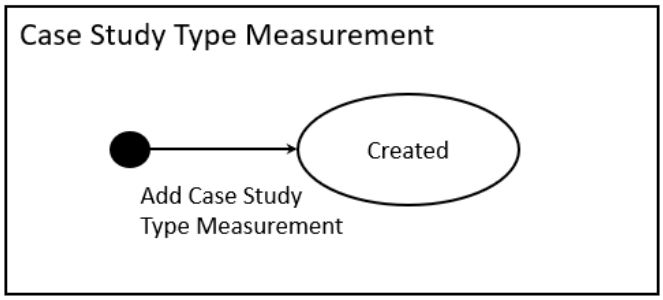


Figure A.3. The protocol machine model for the Case Study Type Step measurement protocol machine

```
OBJECT CaseStudyTypeMeasurement
    NAME CaseStudyTypeMeasurement Name
    ATTRIBUTES      CaseStudyTypeMeasurement Name: String,
                    CaseStudyTypeStep: CaseStudyTypeStep,
    STATES          created,
    TRANSITIONS     @new*Add Case Study Type Measurement=created,
```

Listing A.6. Modelscope Definition of the Case Study Type Step measurement protocol machine

The creation of a CSR project

When the set of protocol machines that make up the definition of a CST is available it is possible to derive a Case Study project from it. This instantiation is done automatic by the call back routine in listing A.7.

```
package CSR2021;

import com.metamaxim.modelscope.callbacks.*;

/*
create a case study from an existing case study type
the algorithmic description is in the procedure
*/

public class CreateCaseStudy extends Event {

    public void handleEvent() {

        String caseStudyName = this.getString("CaseStudy Name");
        // save a copy of the Case Study to later add to the measure
        Instance addedCaseStudy = this.getInstance("CaseStudy");
        // add the record to the model
        this.submitToModel();
        /*  creating the rest of the case study structure by:
            walking by the list of CaseStudyTypeSteps and per CaseStudyTypeStep:
                create a Case Study step
                walk along the list of CaseStudyTypeMeasures and per CaseStudyTypeMeasure
                    create a measure in the context of the (CaseStudy) Step
        */
        Instance[] caseStudyTypeSteps =
            this.getInstance("CaseStudyType").selectByRef("CaseStudyTypeStep","CaseStudyType");
            String stepCountString = "";
            String maxStepcountString = "" + (caseStudyTypeSteps.length + 1 );

        for (int i = 0; i < caseStudyTypeSteps.length; i++) {
            // Create a Step on every CaseStudyTypeStep
            String newStepname = caseStudyTypeSteps[i].getString("CaseStudyTypeStep Name");
            Event createStep = this.createEvent("Create Step");
            createStep.setNewInstance("Step", "Step");
            stepCountString = "" + (i + 1);
            for (int p=1; p <= (maxStepcountString.length() - stepCountString.length()); p++)
                stepCountString = "0" + stepCountString;
            createStep.setString("Step Name", stepCountString + ". " + newStepname);

            Instance addedStep = createStep.getInstance("Step"); // save added step to later put in Measurement
            createStep.submitToModel();
            // now to add a measure to the Step if needed
            Instance[] setofMeasures =
                caseStudyTypeSteps[i].selectByRef("CaseStudyTypeMeasurement", "CaseStudyTypeStep");
            String measurementCountString = "";
            String maxmeasurementcountString = "" + (setofMeasures.length + 1 );
            for ( int j = 0; j < setofMeasures.length; j++) {
                String newMeasurename = setofMeasures[j].getString("CaseStudyTypeMeasurement Name");
                Event CreateMeasure = this.createEvent("Create Measurement");
                CreateMeasure.setNewInstance("Measurement", "Measurement");
                measurementCountString = "" + (j + 1);
            for (int q=1; q <= (maxmeasurementcountString.length() - measurementCountString.length()); q++)
                measurementCountString = "0" + measurementCountString;
                CreateMeasure.setString("Measurement Name",
                    stepCountString + "-" + measurementCountString + ". "+ newMeasurename);
                CreateMeasure.setInstance("Step", addedStep);
                CreateMeasure.setString("CaseStudy Name", caseStudyName);
                CreateMeasure.submitToModel();
            }
        }
    }
}
```

Listing A.7. Automatic creation of a Case Study project.

By running several queries, the routine “travels” through the Case Study Type definition. For every found Case Study Type Step and Case Study Type Step measurement, a Case Study step and measurement is created. As the list of Steps and measurements in the query is returned in a chronological order, the routine uses a numbering scheme in naming the Steps and measurements so that these show in the lists in the tool in the same order as in the Case Study Type definition.

Effectively this logic uses an undocumented feature in the tool as in the manual it is nowhere confirmed that the query will always respond in this way. Nevertheless, no examples have been found which show otherwise. For now, it seemed safe to assume this behaviour of the tool is a structural one. Within the created Case Study all Case Study step's names are prefixed with a number in the sequence of definition. The measurements within one step are prefixed with a sequence number within the step, they are related to.

The CSR project

The model in figure A.4. is the first protocol machine of the generated CSR project definition.

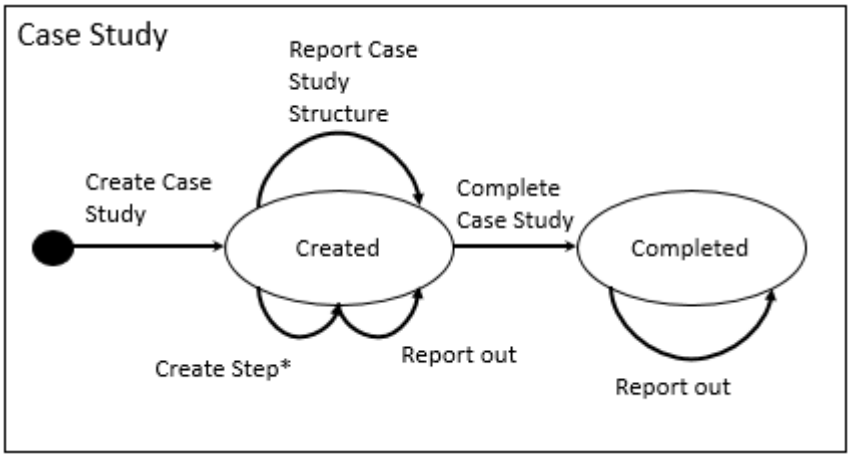


Figure A.4. The protocol machine model for a CSR Project

This protocol machine has the states:

- **Created**: The protocol machine can (automatically) be extended with additional steps from the Case Study Type definition. Also having the model in this state, the user has the option to print two reports on the structure and delivered values of the project. During the project execution, this protocol machine resides in this state allowing the user to add information to the Case Study steps.
- **Completed**: After the information is supplied for all Case Study steps and Case Study measurements, the Case Study project can be set to "Completed". In this state, the user is no longer able to add information in the Case Study steps as also modify values of Case Study.

The Modelscope listing for the CSR project is displayed in listing A.8. The requirement, for not being able to add values to the Case Study, is fulfilled by an included behaviour as in the model. Also, in this listing there are two attributes preceded with an exclamation mark.

```
OBJECT CaseStudy
    NAME CaseStudy Name
    INCLUDES          Case Study Completion Check,
    ATTRIBUTES        CaseStudy Name: String,
                      Research Question: String,
                      CaseStudy Description: String,
                      CaseStudyType: CaseStudyType,
                      !Number Of Steps: String,
                      !Number Of Steps Completed: String,
    STATES            created,
                      completed,
    TRANSITIONS       @new*Create CaseStudy=created,
                      created*Create Step=created,
                      created*Complete Case Study=completed,
                      created*Report Case Study Structure=created,
                      created*Report Results=created,
                      completed*Report Results=completed,
```

Listing A.8. Modelscope definition of the Case Study protocol machine

Both routines are part of the Java Class in listing A.9. The “Number of Steps” attribute contains the number of referenced steps in the Case Study Definition. While the “Number of Steps Completed” contains the same set steps but only takes those into account that have the status “canbecompleted” in one of the associated attributes.

```
package CSR2021;

import com.metamaxim.modelscope.callbacks.*;

/*
return the numbmer of steps in a case study type

Algorithm is to populate an array with the related steps
and then calculate the number of steps in the array
*/

public class CaseStudy extends Behaviour {

    public String getNumberOfSteps () {
        String nrSteps = "";
        Instance [] setOfSteps = this.selectByRef("Step","CaseStudy");
        if (setOfSteps.length == 0)
            nrSteps = "None";
        else
            nrSteps = "" + setOfSteps.length;
        return nrSteps;
    }



    public String getNumberOfStepsCompleted () {
        int completed = 0;
        String stepState = "";
        Instance [] setOfSteps = this.selectByRef("Step","CaseStudy");
        for (int i=0; i< setOfSteps.length; i++) {
            stepState = setOfSteps[i].getState("Step Completed Check");
            if (stepState.equals("canbecompleted"))
                completed++;
        }
        if (completed == 0)
            return "None";
        else
            return "" + completed;
    }
}
```

Listing A.9. Returning the number of steps and number of completed steps to the CSR project

An important aspect of this protocol machine is that it can only be set to “Completed” after all Case Study type Steps are fulfilled. The call back routine referenced by the “INCLUDES” tag is the Case Study Completion Check module. This is defined in the model as “BEHAVIOUR” object in listing A.10.

```
BEHAVIOUR !Case Study Completion Check
    STATES      CScanbecompleted,
                CScannotbecompleted,
    TRANSITIONS @any*Finish Case Study=CScanbecompleted,
```

Listing A.10. Behaviour related to the Case Study project to confirm allowed closure of the Case Study

The algorithm listed in listing A.11. checks for all referenced Case Study steps if they are completed.

```
package CSR2021;

import com.metamaxim.modelscope.callbacks.*;

/*
verify if the case study can be completed
the case study can be completed if all the steps in the case study
can be completed

the funtion returns:
"CScanbecompleted": if the case Study can be completed
"CScannotbecompleted": if the case study cannot be completed
*/


public class CaseStudyCompletionCheck extends Behaviour {
    public String getState() {
        // algorithm: select all steps w/i a case study and check if all are canbecompleted
        // if all are completed, the case study can be completed as well
        Boolean completionOk = true;
        Instance[] setOfSteps =
            this.selectByRef("Step","CaseStudy");
        for (int j = 0; j < setOfSteps.length; j++) {
            String stepState = setOfSteps[j].getState("Step Completed Check");
            completionOk = (completionOk && (stepState.equals("canbecompleted")));
        }
        if (completionOk)
            return "CScanbecompleted";
        else
            return "CScannotbecompleted";
    }
}
```

Listing A.11. Checking if the CSR project can be set to "completed"

The event "Complete Case Study" triggers a call back routine that sets the status of the CSR to "completed", but eventually also travels through the set of steps and measurements. These objects will then also be set to "completed" to ensure no deliverable or value can be added to the definition. In listing A.12. the reference is indicated by the exclamation mark.

```
EVENT !Complete Case Study
    ATTRIBUTES  CaseStudy: CaseStudy,
```

Listing A.12 Modelscopes definition of the complete Case Study event.

The triggered call back routine in listing A.13. travels through the structure and sequentially triggers the appropriate events to set the states of the found Steps and measurements to "completed".

```
package CSR2021;

import com.metamaxim.modelscope.callbacks.*;

/*
Complete a case Study and all its Steps and measurements
When then the whole stucture is in a "completed" state it is no longer
possible/allowed to addd Measurements and filenams, hence deliverables
*/

public class CompleteCaseStudy extends Event {

    public void handleEvent() {
        Instance thisCaseStudy = this.getInstance("CaseStudy");
        Instance[] setOfSteps =
            thisCaseStudy.selectByRef("Step","CaseStudy");
        if (setOfSteps.length != 0 ){
            for (int i=0; i< setOfSteps.length; i++) {
                Instance [] setOfMeasurements =
                    setOfSteps[i].selectByRef("Measurement","Step");
                if(setOfMeasurements.length !=0) {
                    for(int j = 0; j< setOfMeasurements.length; j++){
                        Event completeMeasurement = this.createEvent("Complete Measurement");
                        completeMeasurement.setInstance("Measurement", setOfMeasurements[j]);
                        completeMeasurement.submitToModel();
                    }
                }
                Event completeStep = this.createEvent("Complete Step");
                completeStep.setInstance("Step", setOfSteps[i]);
                completeStep.submitToModel();
            }
        }
        Event completeCaseStudy = this.createEvent("Complete Case Study");
        completeCaseStudy.setInstance("CaseStudy", thisCaseStudy);
        completeCaseStudy.submitToModel();
    }
}
```

Listing A.13 Setting the CSR project to status "Completed"

The Case Study step protocol machine

The Case Study object refers to several Case Study step protocol machines. The research team is required to fulfil the Case Study steps in chronological order. Such a step is fulfilled if one of the following conditions is met:

1) The name is provided of a file that contains the deliverable of the step. Examples of this are files that contain an inventory, a description or a diagram;
2) Or, if a Step refers to measurements, all the referred measurements contain values.

Determining, if the step can be marked as fulfilled is done by a Java call back routine.

In the diagram of this protocol machine in figure A.5. two states are modelled.

- **Created:** The Case Study project has been and can accept events for entering or changing the filename. During creation of the whole CSR structure the event, Create Measurement can be accepted as well
- **Completed:** this state indicates that the Case Study step is completed and the filename deliverable values cannot be amended.

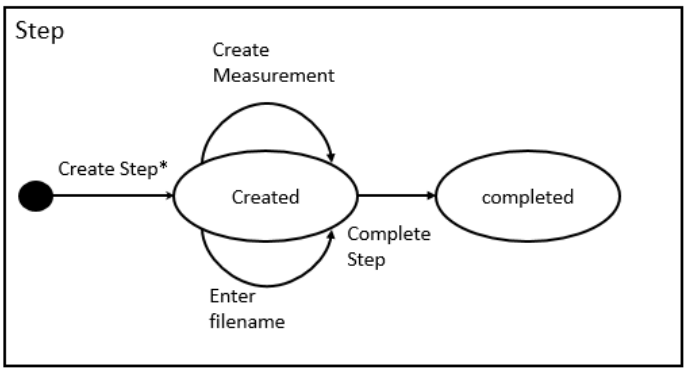


Figure A.5. The graphical model for the Case Study step protocol machine

In listing A.14. of the Modelscope definition of this protocol machine, one of the attributes contains the Filename of the Step deliverable. Several exclamation marks preceded call back references are added to supply the user with additional information on the number of related measurements as well as the number of completed measurements. In the linked call back routines in listing A.15. the algorithm counts the numbers by determining the length of the arrays returned by the query.

```
OBJECT Step
    NAME Step Name
    ATTRIBUTES      Step Name: String,
                    CaseStudy: CaseStudy,
                    File Name: String,
                    !Number of Measurements: String,
                    !Measurements Completed: String,
    INCLUDES        Step Completed Check,
    STATES          created,
                    completed,
    TRANSITIONS     @new*Create Step=created,
                    created*Create Measurement=created,
                    created*Enter filename=created,
                    created*Complete Step=completed,
```

Listing A.14. Modelscope definition of the cast study step protocol machine

```
package CSR2021;

import com.metamaxim.modelscope.callbacks.*;

/*
two function that return:
getNumberOfMeasures: the number of measures in a given step
getMeasuresCompleted: the number of completed measures in a given step

these are reported to the user for awareness
*/

public class Step extends Behaviour {
    public String getNumberOfMeasurements () {
        Instance [] measures = this.selectByRef("Measurement","Step");
        if (measures.length == 0)
            return "None";
        else
            return "" + measures.length;
    }

    public String getMeasurementsCompleted () {
        int nrCompleted = 0;
        Instance [] measures = this.selectByRef("Measurement", "Step");
        for (int j = 0; j < measures.length; j++) {
                String measureState = measures[j].getState("Measurement");
                if (measureState.equals("completed"))
                    nrCompleted++;
        }
        if (nrCompleted == 0)
            return "None";
        else
            return "" + nrCompleted;
    }
}
```

Listing A.15. Counting the number of measurements and completed measurements in a CSR project

There is an associated behaviour defined to check if the step can be completed. This routine evaluates the requirement for step completion. In listing A.16. the behaviour is linked to the outcome of the call back routine.

```
BEHAVIOUR !Step Completed Check
    STATES      canbecompleted,
                cannotbecompleted,
    TRANSITIONS @any*Mark Step Completed=canbecompleted,
```

Listing A.16. Modelscope behaviour link to the check if a step can be completed

In the referenced call back routine in listing A.17 the algorithm checks if the Step itself contains a filename or if all the related measurements contain values. When this condition is met, the routine returns the value: “canbecompleted”.

```
package CSR2021;

import com.metamaxim.modelscope.callbacks.*;

/*
verify if the step can be completed
steps can be completed if one of the following conditions is true
- a filename is provided of a deliverable (file) in the step
- a vallue is provided for all the measures defined for the step

the function will return:
"canbecompleted": if the condtions are met
"cannotbecompleted": if the conditions are not met
*/

public class StepCompletedCheck extends Behaviour {
    public String getState() {
        // the algorithm is to first check if there are meaasures,
        // check if for all measures a value is supplied
        // if there are no measures, then check if the file name is supplied
        Boolean completionOk = true;
        Instance[] stepMeasurements =
            this.selectByRef("Measurement","Step");
        if (stepMeasurements.length!=0) {
            for (int j = 0; j < stepMeasurements.length; j++) {
                String measureState = stepMeasurements[j].getState();
                completionOk = completionOk && (measureState.equals("populated") ||
                    measureState.equals("completed"));
            }
        }
        else {
            String thisFileName = this.getString("File Name");
            completionOk = !(thisFileName.equals(""));
        }
        if (completionOk)
            return "canbecompleted";
        else
            return "cannotbecompleted";
    }
}
```

Listing A.17. Checking if a Case Study step can be completed

A Case Study measurement

A Case Study step that contains measurements, then references instances of the protocol machine, Case Study measurement. One Case Study step can link a number from zero to multiple measurements. The protocol machine model in figure A.6. shows the three states in which it can reside.

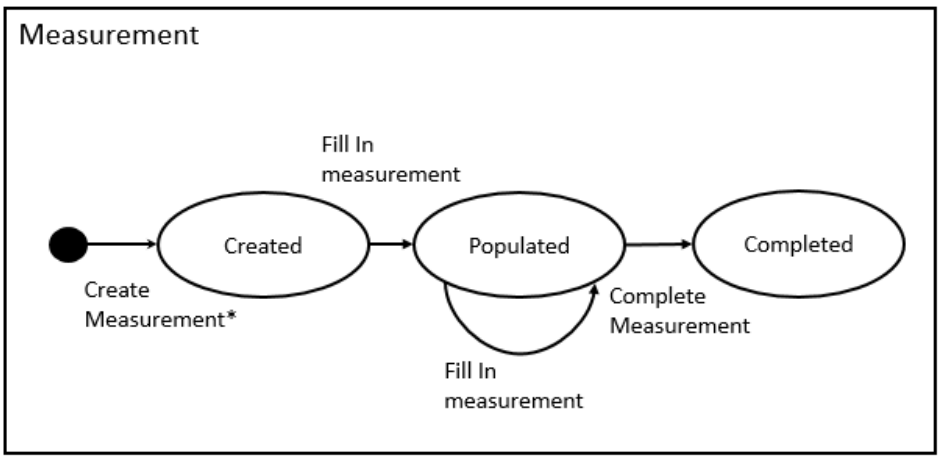


Figure A.6.The protocol machine model for the for a Case Study measurement

- **Created**: The protocol machine has been created and an actual value can be entered. Entering a value, the first time will trigger the state change to the status populated
- **Populated**: The machine already contains a measurement value, but the user is able to amend it when required.
- **Completed**: this status is accomplished by the event, triggered by a routine that sets the whole CSR project to status completed. In this state it is no longer possible to amend the measurement value in the object.

This model's definition in Modelscope definition, as in listing A.18., does not contain any references to call back routines or additional logic. The related Case Study name attribute is added to facilitate reporting.

```
OBJECT Measurement
    NAME Measurement Name
    ATTRIBUTES      CaseStudy Name: String,
                    Step: Step,
                    Measurement Name: String,
                    value: String,
    STATES          created,
                    populated,
                    completed,
    TRANSITIONS     @new*Create Measurement=created,
                    populated*FillIn Measurement=populated,
                    populated*Complete Measurement=completed,
                    created*FillIn Measurement=populated,
```

Listing A.18. Modelscope definition of the measurement protocol machine

Actors in using the model in the tool

Within the setup of the model there is a segregation between the users that define the Case Study type, hence the template, and the users that execute on the research project itself. In the tool this distinction in roles is expressed in the form of assigning different actors.

The actors defined in the developed model are: Principal Scientist and Scientist. The definition of the actors in the Modelscope model consists of listing the set of Objects and events that the Actor is authorized to. The lists reflect the roles of the two actors.

In the model definition the Principal Scientist designs the research approach and creates an instance of the Case Study Type. To do this, he or she would need to be authorized to the set of objects and events as listed in listing A.19.

```
ACTOR Principal Scientist
    BEHAVIOURS  CaseStudyType,
                CaseStudyTypeStep,
                CaseStudyTypeMeasurement,
                CaseStudy,
    EVENTS      Create CaseStudyType,
                Add Case Study Type Step,
                Add Case Study Type Measurement,
                Complete CST Definition,
                Report CaseStudyType,
                Create CaseStudy,
                Report Results,
                Report Case Study Structure,
```

Listing A.19. Modelscope definition of the actor principle scientist

In the model's language these authorized objects are reflected as Behaviours. As the Prinicipal Scientist would also require access to reporting, the related events for these are added to the portfolio.

As the Scientist executes the research initiative, he/she would require authorization on the objects and events listed in listing A.20.

```
ACTOR Scientist
    BEHAVIOURS  CaseStudyType,
                CaseStudy,
                Step,
                Measurement,
    EVENTS      Enter filename,
                FillIn Measurement,
                Complete Case Study,
                Report CaseStudyType,
                Report Results,
                Report Case Study Structure,
```

Listing A.20. Modelscope definition of the actor scientist

Also, this user would require access to reporting and for that the related events are added to the portfolio.

In a more formalized tool, the definition of these two roles would be set to the level of username during login. The Modelscope tool does not require logging in and so it allows the actor selection on the main screen itself.

Reporting the Case Study Type Structure

Reporting the Case Study Type Definition supports the understanding of the Case Study Type that eventually results in a CSR project. The report is written to file with the standard name: "Case Study Type Report.txt". The appropriate call back routing is found in listing A.21

```
package CSR2021;

import com.metamaxim.modelscope.callbacks.*;
import java.io.FileWriter;
import java.io.IOException;

/*
report the structure of a given Case Study Type
this report is intended for the researcher to act protocol on how to conduct the Case Study research assignment
the filename is defaulted to: "Case Study Type Report.txt"
*/

public class ReportCaseStudyType extends Event {
    public void handleEvent() {
        try {
            String CSTFileName = "Case Study Type Report.txt";
            FileWriter writer = new FileWriter(CSTFileName);
            // start the report
            Instance thisCaseStudyType = this.getInstance("CaseStudyType");
            writer.write("Report structure on Case Study Type: "+ thisCaseStudyType.getString("CaseStudyType Name") + "\r\n");
            writer.write("Case Study Type Description: " + thisCaseStudyType.getString("CaseStudyType Description") + "\r\n\r\n");
            Instance [] caseStudyTypeSteps =
                thisCaseStudyType.selectByRef("CaseStudyTypeStep", "CaseStudyType");
            if (caseStudyTypeSteps.length != 0) {
                for (int i=0; i < caseStudyTypeSteps.length; i++) {
                    writer.write ("Case Study Type Step  : " + caseStudyTypeSteps[i].getString("CaseStudyTypeStep Name") + "\r\n");
                    //writer.write ("Number of Measurements: " + caseStudyTypeSteps[i].getString("Number of CST Measurements") + "\r\n");
                    Instance [] CSTMeasurements = caseStudyTypeSteps[i].selectByRef("CaseStudyTypeMeasurement", "CaseStudyTypeStep");
                    if ( CSTMeasurements.length != 0) {
                        writer.write ("\tCase Study Type Measurements are: \r\n");
                        for ( int j = 0; j < CSTMeasurements.length; j++) {
                            writer.write("\tMeasurement: " + CSTMeasurements[j].getString("CaseStudyTypeMeasurement Name") + "\r\n");
                        }
                    }
                }
            }
            else {
                writer.write("This Case Study Type does not contain steps.\r\n");
            }
            writer.close();
        } catch (IOException e) {
                e.printStackTrace();
        }
    }
}
```

Listing A.21. The routine that reports the structure of a CST definition.

An example of such a repot can be found in listing A.22.

```
Report structure on Case Study Type: Visualizing a Gap of Changes versus analysis
    of the As-Is and the To-Be model
Case Study Type Description: Hoe houd je zicht op de veranderingen bij de vervanging
    van een spreadsheetapplicatie door een informatiesysteem.

Case Study Type Step  : Model the As-Is of the current situation
Case Study Type Step  : Determine motivation for changes
Case Study Type Step  : Model the To-Be from motivation and as is model
Case Study Type Step  : Model Gap of Changes from As-Is and To-Be
Case Study Type Step  : Determine the changes that have  been modelled and categorize them
Case Study Type Step  : Determine changes - from as-is vs. to-be - (meas.: 1 - 7)
    Case Study Type Measurements are:
    Measurement: Number of identified Changes (As-is vs. ToBe) - Oobsolete
    Measurement: Number of identified Changes (As-is vs. ToBe) - Onew
    Measurement: Number of identified Changes (As-is vs. ToBe) - Ochanged
    Measurement: Number of identified Changes (As-is vs. ToBe) - Rnew
    Measurement: Number of identified Changes (As-is vs. ToBe) - Robsolete
    Measurement: Number of identified Changes (As-is vs. ToBe) - Rborder
    Measurement: Number of identified Changes (As-is vs. ToBe) - Rchanged
Case Study Type Step  : Determine changes - changes from as-is vs. GoC - (meas.: 8 - 14)
    Case Study Type Measurements are:
    Measurement: Number of identified Changes (As-is vs. GoC) - Oobsolete
    Measurement: Number of identified Changes (As-is vs. GoC) - Onew
    Measurement: Number of identified Changes (As-is vs. GoC) - Ochanged
    Measurement: Number of identified Changes (As-is vs. GoC) - Rnew
    Measurement: Number of identified Changes (As-is vs. GoC) - Robsolete
    Measurement: Number of identified Changes (As-is vs. GoC) - Rborder
    Measurement: Number of identified Changes (As-is vs. GoC) - Rchanged
Case Study Type Step  : Interpret, categorize and conclude. Count changes per stakeholder per category
Case Study Type Step  : Analyze and interpret results
```

Listing A.22. Example listing of the CST report

Reporting the values and structure of the Case Study

Under various conditions the user would require a report on the structure and the content of the Case Study. For this, two reports have been added to the protocol model.

The structure report plainly shows the structure of the Case Study itself. While the call back routine in listing A.23 inventories all Case Study steps and measurements, it reports all found objects to a file with the standard name: “Case Study Structure.txt”. The user can initiate this report by manually triggering the appropriate event from the object Case Study

```
package CSR2021;

import com.metamaxim.modelscope.callbacks.*;
import java.io.FileWriter;
import java.io.IOException;

/*
report the structure of a given Case Study
this report is intended for the researcher to act protocol on how to conduct the Case Study research assignment
the filename is defaulted to: "Case Study Structure.txt"
*/

public class ReportCaseStudyStructure extends Event {
    public void handleEvent() {
        try {
            String structureFileName = "Case Study Structure.txt";
            FileWriter writer = new FileWriter(structureFileName);
            // start the report
            Instance thisCaseStudy = this.getInstance("CaseStudy");
            writer.write("Report structure on Case Study: "+ thisCaseStudy.getString("CaseStudy Name") + "\r\n");
            writer.write("Case Study Description        : "+ thisCaseStudy.getString("CaseStudy Description") + "\r\n\r\n");
            Instance [] caseStudySteps =
                thisCaseStudy.selectByRef("Step", "CaseStudy");
            if (caseStudySteps.length != 0) {
                for (int i=0; i < caseStudySteps.length; i++) {
                    writer.write ("Step: " + caseStudySteps[i].getString("Step Name") + "\r\n");
                    Instance [] measures = caseStudySteps[i].selectByRef("Measurement", "Step");
                    if ( measures.length != 0) {
                        writer.write ("Measurements are: \r\n");
                        writer.write ("(Please supply values for all listed measurements.)\r\n");
                        for ( int j = 0; j < measures.length; j++) {
                            writer.write("\tMeasurement: " + measures[j].getString("Measurement Name") + "\r\n");
                        }
                        writer.write("(Supplying additional evidence in this step is optional.)\r\n");
                    }
                    else {
                        writer.write("(Please supply filename with evidence for proper completion of this step. This is mandatory.)\r\n");
                    }
                }
            }
            else {
                writer.write("This Case Study does not contain steps.\r\n");
            }
            writer.close();
        } catch (IOException e) {
                e.printStackTrace();
        }
    }
}
```

Listing A.23. The routine that reports the structure of a CSR project.

The report, report out is more extensive and will not only show the model’s structure but also the values supplied in the Case Study steps as well as the measurements. On the measurement values, the call back routine as in listing A.24, applies some arithmetic and reports the sum and average per Step.

```
package CSR2021;

import com.metamaxim.modelscope.callbacks.*;
import java.io.FileWriter;
import java.io.IOException;

/*
report the structure of a given Case Study
this report is intended for the researcher to act protocol on how to conduct the Case Study research assignment
the filename is defaulted to: "Case Study Structure.txt"
*/

public class ReportResults extends Event {
    public void handleEvent() {
        try {
            String structureFileName = "Case Study report.txt";
            FileWriter writer = new FileWriter(structureFileName);
            // start the report
            Instance thisCaseStudy = this.getInstance("CaseStudy");
            writer.write("Report on Case Study  : "+ thisCaseStudy.getString("CaseStudy Name") + "\r\n");
            writer.write("Case Study Description: "+ thisCaseStudy.getString("CaseStudy Description") + "\r\n");
            writer.write("Status of project is  : "+ thisCaseStudy.getState("CaseStudy") + "\r\n");

            Instance [] caseStudySteps =
                thisCaseStudy.selectByRef("Step", "CaseStudy");
            if (caseStudySteps.length != 0) {
                for (int i=0; i < caseStudySteps.length; i++) {
                    writer.write ("\r\nStep: " + caseStudySteps[i].getString("Step Name") + "\r\n");
                    writer.write("filename:\t" + caseStudySteps[i].getString("File Name") + "\r\n");
                    Instance [] measures = caseStudySteps[i].selectByRef("Measurement", "Step");
                    if ( measures.length != 0) {
                        writer.write ("Measurements are: \r\n");
                        int measuresSum = 0;
                        int numberOfMeasures = 0;
                        for ( int m = 0; m < measures.length; m++) {
                            writer.write("Measurement :\t" + measures[m].getString("Measurement Name"));
                            writer.write("\tValue:\t" + measures[m].getString("value") + "\r\n");
                            if (measures[m].getString("value").trim().length() != 0)
                                measuresSum += Integer.parseInt(measures[m].getString("value"));
                                //measuresSum ++;
                            numberOfMeasures++;
                        }
                        writer.write("\tThe sum of these is:\t" + measuresSum + "\r\n");
                        writer.write("\tThe number of Measurements:\t" + numberOfMeasures + "\r\n");
                        if ( numberOfMeasures != 0) {
                            double averageOfMeasures = (measuresSum * 1.0) / numberOfMeasures;
                            writer.write("\tAverage of values is: " + averageOfMeasures + "\r\n");
                        }
                    }
                    else {
                        writer.write("\t\t\t(This Step does not contain measurements.)\r\n");
                    }
                }
            }
            else {
                writer.write("This Case Study does not contain steps.\r\n");
            }
            writer.close();
        } catch (IOException e) {
                e.printStackTrace();
        }
    }
}
```

Listing A.24. The routine that reports the structure plust values of a CSR project.

B. Data used for testing the model

The tool is tested, using a number of generic CST data sets as also some data sets to enter in derived CSR project. Then the model is tested with available CSR Thesis data.

Table B.1. CST Data set number 1 used to functionally test the model.

Parameter	Value
Case Study type name	A first Case Study type for testing the tool
Case Study type Description	This is the first Case Study Type, to use for testing
Case Study type steps	Case Study Type one – one Case Study Type one – two (measurement 1) Case Study Type one – three (measurement 2, 3) Case Study Type one – four
Case Study type measurements	1. Measurement one – one 2. Measurement one – two 3. Measurement one – three

Table B.2. CSR project Data set number 1 used to functionally test the model.

Case Study project one:	
Research question	This is the first research question for testing the tool
CaseStudy Description	This is the description of the first Case Study
Case Study project name	Case Study project one
Deliverable for step one	Filename one. Vsd
Deliverable for step two	Measurements: 1. 12
Deliverable for step three	Measurements: 2. 14 3. 16
Deliverable for step four	Filename two. Docx

Table B.3. CSR project Data set number 2, used to functionally test the model.

Case Study project two:	
Research question	This is the second research question for testing the tool
CaseStudy Description	This is the description of the second Case Study
Case Study project name	Case Study project two
Deliverable for step one	Filename three. Vsd
Deliverable for step two	(Measurement one): 120
Deliverable for step three	(Measurement two): 120 (Measurement three): 140
Deliverable for step four	Filename four. Docx

Table B.3. CSR project Data set number 3, used to functionally test the model.

Case Study project four:	
Research question	This is the fourth research question for testing the tool
CaseStudy Description	This is the description of the first Case Study
Case Study project name	Case Study project fourth
Deliverable for step one	Filename seven. Vsd
Deliverable for step two	(Measurement one): 12000
Deliverable for step three	(Measurement two): 14000 (Measurement three): 16000
Deliverable for step four	Filename eight. Docx

Table B.4. CST Data set number 2, used to functionally test the model.

Parameter	Value
Case Study type name	A second Case Study type for testing the tool
Case Study type Description	This is the second Case Study type to test the testing tool
Case Study type steps	Case Study Type two - one Case Study Type two - two(measurement 4) Case Study Type two - three (measurement 5, 6) Case Study Type two - four
Case Study type measurements	1. Measurement two - one 2. Measurement two - two 3. Measurement two - three

Table B.5. CSR project Data set number 4, used to functionally test the model.

Case Study project Three:	
Research question	This is the third research question for testing the tool
CaseStudy Description	This is the description of the third Case Study Type
Case Study project name	Case Study project three
Deliverable for step one	Filename five. Vsd
Deliverable for step two	Measurements: 1. 1200
Deliverable for step three	Measurements: 2. 1400 3. 1600
Deliverable for step four	Filename six.docx

A more elaborate set was used to see how the model would behave in a simulated live environment:

Table B.6. CST Data set, used to test the model in a simulated environment

Parameter	Value
Case Study Type name	Visualizing a Gap of Changes versus analysis of the As-Is and the To-Be model
Case Study Type Description	Hoe houd je zicht op de veranderingen bij de vervanging van een spreadsheetapplicatie door een informatiesysteem

Parameter	Value
Case Study type steps	Model the As-Is of the current situation Determine motivation for changes Model the To-Be from motivation and as is model Model Gap of Changes from As-Is and To-Be Determine the changes that have been modelled and categorize them Determine changes - from as-is vs. to-be - (meas.: 1 - 7) Determine changes - changes from as-is vs. GoC - (meas.: 8 - 14) Interpret, categorize and conclude. Count changes per stakeholder per category Analyse and interpret results
Case Study type measurements	1. Number of identified Changes (As-is vs. ToBe) - Oobsolete 2. Number of identified Changes (As-is vs. ToBe) - Onew 3. Number of identified Changes (As-is vs. ToBe) - Ochanged 4. Number of identified Changes (As-is vs. ToBe) - Rnew 5. Number of identified Changes (As-is vs. ToBe) - Robsolete 6. Number of identified Changes (As-is vs. ToBe) - Rborder 7. Number of identified Changes (As-is vs. ToBe) – Rchanged 8. Number of identified Changes (As-is vs. GoC) - Oobsolete 9. Number of identified Changes (As-is vs. GoC) - Onew 10. Number of identified Changes (As-is vs. GoC) - Ochanged 11. Number of identified Changes (As-is vs. GoC) - Rnew 12. Number of identified Changes (As-is vs. GoC) - Robsolete 13. Number of identified Changes (As-is vs. GoC) - Rborder 14. Number of identified Changes (As-is vs. GoC) - Rchanged

Using this Case Study Type definition two actual Case Study projects were derived:

Table B.7. CSR Data set number 1, used to test the model in a simulated environment

Case Study project one:	
Research question	Are the changes better recognizable by visualizing a Gap of Changes then an analysis of the As-Is and the To-Be model
CaseStudy Description	We are going to analyze if the changes better recognizable by visualizing a Gap of Changes then an analysis of the As-Is and the To-Be model with ROC – Den bosch
Case Study project name	ROC - Den Bosch
Deliverable for step 1	ROC - Den Bosch - As Model.vsd
Deliverable for step 2	ROC - Den Bosch - Motivation for change.docx
Deliverable for step 3	ROC - Den Bosch - To be model from motivatio-2-As Is.vsd
Deliverable for step 4	ROC - Den Bosch - Gap of change from As-Is and To-Be.vsd
Deliverable for step 5	ROC - Den Bosch - Categorized modeled changes.docx
Deliverable for step 6	Measurements: 1. 4 2. 3 3. 1 4. 5 5. 9 6. 7. 17
Deliverable for step 7	Measurements: 8. 6 9. 2 10. 4 11. 3 12. 8 13. 14. 21
Deliverable for step 8	ROC - Den Bosch - changes per stakeholder.xlsx
Deliverable for step 9	ROC - Den Bosch - analysis and interpretation.docx

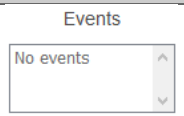
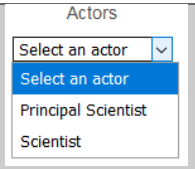
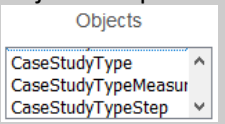
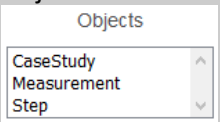
Table B.8. CSR Data set number 2, used to test the model in a simulated environment

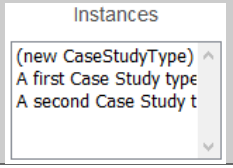
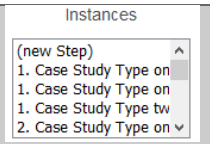
Case Study project two:	
Research question	Are the changes better recognizable by visualizing a Gap of Changes then an analysis of the As-Is and the To-Be model
CaseStudy Description	We are going to analyze if the changes better recognizable by visualizing a Gap of Changes then an analysis of the As-Is and the To-Be model with Partnerziekenhuis UMC
Case Study project name	Partnerziekenhuis UMC
Deliverable for step 1	P-UMC - As Model.vsd
Deliverable for step 2	P-UMC - Motivation for change.docx
Deliverable for step 3	P-UMC - To be model from motivatio-2-As Is.vsd
Deliverable for step 4	P-UMC - Gap of change from As-Is and To-Be.vsd
Deliverable for step 5	P-UMC - Categorized modelled changes.docx
Deliverable for step 6	Measurements: 1. 22 2. 26 3. 4. 11 5. 2 6. 7.
Deliverable for step 7	Measurements: 8. 33 9. 34 10. 11. 23 12. 11 13. 14.
Deliverable for step 8	P-UMC - changes per stakeholder.xlsx
Deliverable for step 9	P-UMC - analysis and interpretation.docx

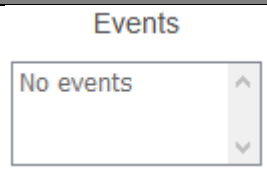
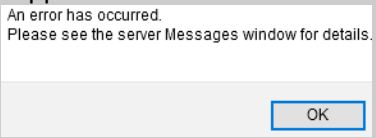
C. Results of the tests conducted on the model

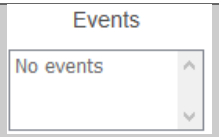
The tests conducted on the model including test results.

Table C.1 Test plan and outcome of conducted tests.

Test number	Requirement	Test set / scenario	Outcome	Remark
1	For the definition of a CST, the tool should support definition of the steps, as part of the CST.	Action: (user: prin. Scientist). use basic data set 1 to define a Case Study Type Checks: if entered steps are added to the definition	Passed	
2	After the definition of the CST protocol template, the definition can be saved. After saving no additions can be made to the template.	Action: use definition of previous test (if status: passed) and set the definition of the Case Study Type to completed Checks: check if you can add additional steps or measurements. This should not be allowed/possible	Passed	
3	The tool should support role based access where there is a distinction between the user that defines the CSR (type) project and the user that carries out the research and enters the deliverables into the tool	Action: Start the tool and check if there are different users defined. Checks: if authorization of the users and assess if one user is able to define CSR types and projects. Where the other user can only access the CSR project for entering deliverables	Passed	 Objects for prin. Scientist:  Objects for scientist: 

Test number	Requirement	Test set / scenario	Outcome	Remark
4	Per Case Study Type Step, deliverables should be defined. These can be a file with the deliverables noted or a set of measurement values that the research team would be required to supply.	Action: use definition of previous test (if status: passed). Create a Case Study from the definition and add deliverables per step Checks: check if it is possible to add measurements and if in every step there is "room" for a filename deliverable	Passed	
5	The tool should be able to maintain a set of predefined CST protocols for fulfilling a type of CSR projects.	Action: use definition of previous test (if status: passed) and add the definition of the basic dataset 2. Set the 2nd CST def. to "completed" Checks: check if both sets are in the tool.	Passed	
6	The tool should be able to derive from the various CST definitions, an actual Case Study project support instance. This creation should be automated without further user manual labor.	Action: use definitions of previous test (if status: passed) and instantiate both CSR types into case studies, using the data in the set Checks: 1) If creation of the Case Study is automatic 2) if all steps and measurements have been copied into the Case Study Action: run the "report structure" reports on one of the case studies (test 12)	1) passed 2) passed	
7	This sequence in which the steps are defined in the Case Study type should also be applied in the same order to the derived Case Study project.	Action: use definition of previous test (if status: passed) for data set 1 Checks: check if the sequence of the steps in the Case Study matches the Case Study Type, it was created from	Passed	

Test number	Requirement	Test set / scenario	Outcome	Remark
8	Once the created research Case Study project is created, the user should not be able to alter its structure.	Action: use definition of previous test (if status: passed) Checks: if the user can alter the structure of the Case Study, 1) Can you add steps to the project? 2) Can you add measurements to the project? Both changes should not be possible/allowed.	passed	
9	The tool should support, adding deliverables and values to steps and measurements until the moment the Case Study is set to the state “completed”.	Action: use definition of previous test (if status: passed) for data set 1. Add values to the steps and measurements as noted in the set. Then try and change file names in steps and/or values in measurements Checks: 1) When not all values are supplied (in various stages of completing the project) try to set the Case Study to “completed”. This should not be possible (test #10) 2) Try to alter data in the steps and measurements, this should be possible	1) Passed 2) Passed	When trying to complete the Case Study, and not all data suppled 

Test number	Requirement	Test set / scenario	Outcome	Remark
10	The tool should check if all deliverables are met before allowing the user, to close the case Study project.	Action: use definition of previous test (if status: passed) for data set 1. Now all deliverables have been supplied, set the status of the Case Study to “completed”. Checks: 1) When only filenames are supplied for all steps (no measurements) it should not be possible to complete the Case Study 2) When only measurements are supplied and no filenames, it should not be possible to complete the Case Study 3) when indeed all deliverables have been supplied the tool should allow to complete the Case Study	1) Passed 2) Passed 3) Passed	
11	After the project is completed, alterations of the deliverable files and measurements should no longer be possible	Action: use definition of previous test (if status: passed)). Try to amend filenames in steps or values in measurements Checks: amending data should not be possible/allowed	Passed	
12	The tool should have various reporting capabilities to support the user in establishing the progress of the project as well as the findings. These reports should aim to support all available roles and in all stages of the process.	Action: use definition of previous test (if status: passed)). Start the “report out” report. Checks: if the output in both reports (also test 6) matches the Case Study structure and data.	Passed	

Test number	Requirement	Test set / scenario	Outcome	Remark
13	The tool should support at least one example of a Case Study from literature	Action: populate a CST as defined in data set. Add two case studies with data from real live example Check: if tool supports this scenario	Passed	