Hong-Vu Dang
Internship report

A FRAMEWORK FOR IMPROVING COMPONENCE'S PROJECT PROCESS (CENSORED)

Hong-Vu Dang
Internship report

A FRAMEWORK FOR IMPROVING COMPONENCE'S PROJECT PROCESS (CENSORED)

Vrije Universiteit Amsterdam
Faculty of Sciences
Master's Programme Business Mathematics and Informatics
De Boelelaan 1081a
1081 HV Amsterdam
www.few.vu.nl

Internship Company:
Componence BV
Weverstede 27B
3431 JS Nieuwegein
The Netherlands
Tel: +31306301477
Fax: +31306301480
www.componence.com

August 2006

PREFACE

This paper shows the results of my internship at Componence BV. The internship is part of the Business Mathematics and Informatics Masters program at the Vrije Universitieit in Amsterdam.

Componence BV is a fast growing software company that, among other products, offers internet portals that are built and delivered to the customer through projects. Many problems arise during these projects, and the goal of this research was to help Componence improve this process and gain more insight in it. I hope that with this research I reached this goal and that Componence managers and employees (and their partners and vendors) will discuss the recommendations and matters presented.

From the Vrije Universiteit I would like to express my gratitude to Professor Van Vliet for his guidance, advice and feedback during my internship and Professor Wojtek Kowalczyk for being the second reader of this report. Also, I would like to thank Annemieke van Goor for all the preparations that were made in advance and helping me come in contact with Componence. From Componence I would like to thank Paul Bot for his time and effort that was put in this internship, Ha Vo for providing the internship and his guidance, and all other Componence and SS (strategic partner of Componence) employees that helped me and made sure I had a great time. Thank you for your hospitality!

Hong-Vu Dang

ABSTRACT

Compared to last year, Componence has grown 100 percent, both in size and results. Many problems arise during the project process. Solving these problems takes up a great amount of time and effort, leaving little for self-analysis and self-improvement which is necessary for preventing these problems.

The focus of this report will lie mainly on an analysis of Componence's project process, gaining more insight in it and recommendations for improving it. The deliveries in this report consist of:

- A description of the current project process in flowcharts, its major problems and possible improvements;
- An analysis of what metrics could be used to measure the performance of these major problem areas;
- An analysis of the data and metrics of different Componence projects to gain more insight in different facets of the projects;
- Conclusions based on the analysis;
- Recommendations for what data to collect, how to collect the data in future projects and what metrics to track;
- Recommendations for improving effort and duration estimations.

The flowcharts of the project process give a bird's eye view of the process and may come in handy when discussing and/or identifying problems of the project process. A collection of problems and possible improvements has been made using these charts. A selection of a sub-set of these problems form the basis for the rest of the research done during this internship.

The data and metrics collected for a set of projects quantified performance levels in different aspects of the projects, and identified several possible causal relations. For instance:

- When performance testing was done during the construction phase, there was less trouble after delivering a portal to the customer;
- Implementing "requests for change" has a relation with the amount of effort overrun in projects;
- Quantitative numbers and figures for effort and time overrun are presented, showing that nearly all projects have effort overrun and many have time overrun.

A recommendation is made for tracking the most useful and valuable types of data and metrics.

One of the major causes for problems in projects is bids being used as a constraint for the amount of allowed resources and time in combination with inaccurate effort estimation and planning. There are many existing theories concerning bidding, effort estimation and planning. A recommendation of how the existing theory can be applied to Componence's bidding, effort estimation and planning processes and an example of how an effort estimation tool could be modelled is presented. The most important recommendation on this field is to simply at least record data concerning effort and costs for future projects to gain more insight and build a foundation for better bidding, effort estimation and planning.

componence
Enterprise Portal Power

TABLE OF CONTENTS

1 INTRODUCTION TO THIS REPORT

First and foremost, thank you for reading this report, hopefully you will enjoy reading it and find it useful. A quick overview of what this report is about is given below.

1.1 Problem definition

Compared to last year, Componence has grown 100 percent, both in size and results. Many problems arise during the project process. Solving them takes up a great amount of time and effort, leaving little for self-analysis and self-improvement which is necessary for preventing these problems.

1.2 The goal of this research

The focus of this report will lie mainly on an analysis of Componence's project process, gaining more insight in it and recommendations for improving it. In a nutshell, the project process comprises of all activities that are executed from a request for a project from the customer to delivering the final product(s) to the customer. Sales, support, marketing and hosting activities may be touched upon but are excluded for the most part in this report. Initially, the scope of this internship was much too big, and it included problems that were already worked on. The current deliveries consist of:

- A description of the current project process in flowcharts, its major problems and possible improvements;
- An analysis of what metrics could be used to measure the performance of these major problem areas;
- An analysis of the data of different Componence projects to gain more insight in different facets of the projects;
- Conclusions based on the analysis;
- Recommendations for what data to collect, how to collect the data in future projects and what metrics to track;
- Recommendations for improving effort and duration estimations.

Besides the results mentioned above, this document can help Componence give better insight in their portal development process for future partners and vendors.

1.3 The structure of this report

The structure and contents roughly follow the goals mentioned above:

- Chapter 2 gives more insight in what Componence does, who the stakeholders are, and how the company and its partners and vendors are structured;
- In chapter 3 the flow of a typical project process (as it was when this internship started) is extensively described. It is recommended that when reading this chapter the activity diagrams in appendix A are kept next to this report;
- The activity diagrams were used to point out problems. These problems were taken from interviews and discussions with both Dutch and offshore managers from partner SS and can be found in chapter 4;
- Possible improvements were discussed with them and listed in chapter 5. A set of "main problems" is defined for which the further research is focused on;
- To be able to measure the performance in these main problem areas, a set of metrics is defined. Data is collected for a set of projects and the metrics are calculated form this data for having a benchmark and further analysis is done. A recommendation is made of what data and metrics are useful and should be collected in the future. The results can be read in chapter 6;
- Chapter 7 gives an overview of how bidding and effort estimation is done by Componence and what experts have to say about bidding, effort estimation and planning. This chapter contains recommendations of how the processes of bidding and effort estimation can be improved using existing models and theory, and an example model is given of how effort estimation can be improved;
- All conclusions and recommendations are given in chapter 8;

- Chapter 9 contains a reflection on the internship, what I learned from it, what could have been done better, etcetera.
- All appendices round up this report. They contain the activity diagrams, issue charts, data, and glossary;

1.4 Confidentiality

If you are reading an uncensored version of this report, please keep in mind that information disclosed in this report could be confidential. The censored version can be read by anyone, any information that is included in the uncensored version but excluded from the censored version should be considered confidential and should only be disclosed to authorized persons.

2 INTRODUCTION TO COMPONENCE

This section should give people that are not familiar with Componence (for instance new vendors or readers from the Vrije Universiteit) more insight in what Componence does, what is offered to the customers, what other parties and stakeholders are involved and what their roles are. A rough sketch of Componence's organization is shown in organization charts. People who are already familiar with Componence could skip this section.

2.1 Componence products and services

Componence offers portals (portals and sometimes applications running in internet browsers) to enterprise customers. These portals can be built on two systems:

- The in-house developed ECMsuite content management system combined with the open-source application server Tomcat;
- Or the in-house developed Portletsuite combined with the closed-source application infrastructure offered by BEA. Componence has become a certified BEA strategic partner.

If needed, Componence can also provide consult to the customer before the project starts. Componence also offers hosting services, customer support in the forms of bug-fixing, adding extra functionality (RFC's), and answering general questions.

2.2 Who are the stakeholders?

The stakeholders seen from a high level are:

- Componence in the Netherlands, Nieuwegein;
- Componence offshore units;
    - Strategic partners;
    - Partners;
    - Vendors;
- Implementation partners (system integrators);
- Customers;
- Hosting partners;
- BEA (delivers application infrastructure software).

Figure 1 (Remus, 2006), shows a general portal value chain that consists of different parties:

- Supplier of portlets;
- Supplier of Portlet-Packages;
- Portal-integrator;
- Supplier of platforms;
- Customer.

The offshore units would be the suppliers of portlets, Componence the supplier of Portlet-Packages (Portletsuite), and the offshore units also take the role of portal-integrators. The supplier of platforms would be BEA. Some customer activities are also done by Componence. For instance, Componence can advise the customer what portlets to select, and Componence can handle the maintenance and support for the portal.

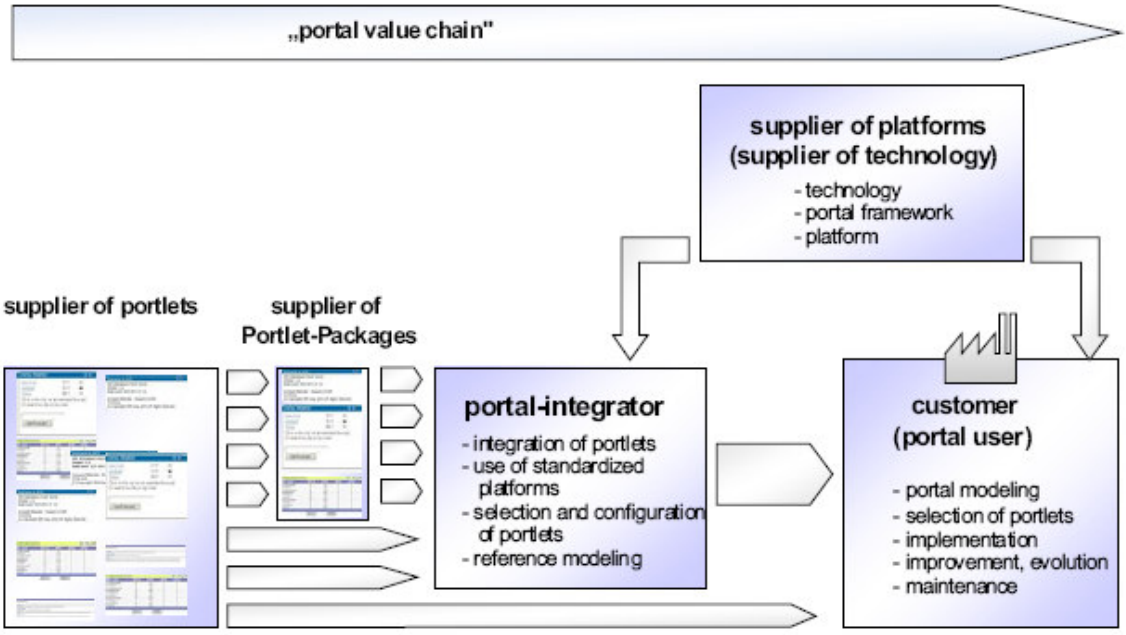


Figure 1: Portal value chain (Remus, 2006)

2.3 Componence organization charts

Figures 2 to 6 show organization charts of Componence as of March, 2006. They should, together with their descriptions, give the reader of this report a global idea of how the organization looks like and what units add what value to the project process. **Therefore, units that don't explicitly add value to the project process are left out.** Refer to the glossary if there are any problems understanding the symbols used for these charts.

A high level chart is described in the following sections, and then the project unit and operations unit are modelled in more detail. The product development unit will not be modelled in further detail because it is not explicitly involved in the project process.

2.3.1 High Level Organization Chart

The chart "Componence High Level Organization Chart", figure 2, shows the organization structure on a high abstraction level. Componence has the board of directors unit coordinating the operations, project and product development units.

The Dutch board of directors include Chris Gobel, Ha Vo and Paul Bot, who manage and coordinate all units (Operations, Project and Product Development) in the company. Staff functions include an office manager, administrator, a human resources unit and a sales unit (not included in the charts because they are not directly involved in the project process). From the board of directors Paul Bot is the most active in managing and overseeing projects (not including the sales process preceding the project process). Sometimes Ha Vo assumes the role of project leader/manager to assist Paul on executional level, but he is mostly active in expanding the offshore resources of Componence to be able to cope with the growth of the company and also is in lead of the offshoring department (not pictured). The sales unit makes sure a healthy amount of projects roll in, currently the main goal is to find bigger, long term clients that can provide long running projects.

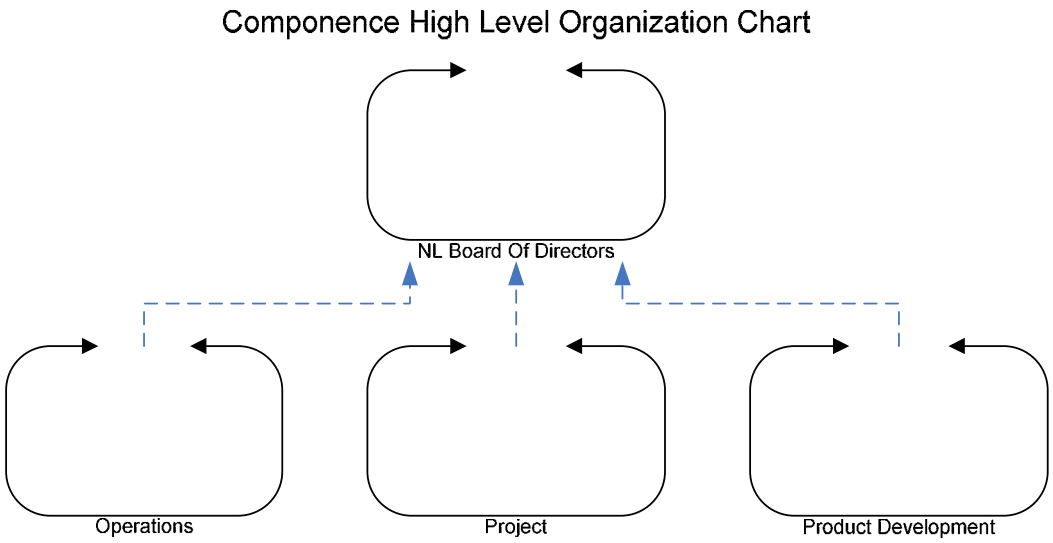


Figure 2

2.3.2 Organization chart: Project in detail

Figure 3 to 7 show the project unit in more detail. The project teams carry out the assignments and deliver the portals to the customers. The Dutch project teams communicate to the customer and offshore project teams and take part in the analysis, implementation and testing of the portals. The offshore project teams build and test the portals.

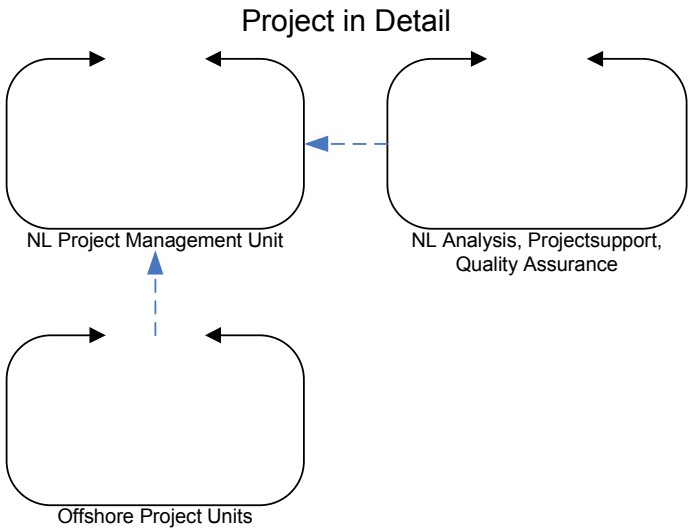


Figure 3

Figure 4 shows the Dutch Project Management Unit. The head of the Dutch project management unit is Paul Bot. This unit includes project leaders, each of which can manage one or more projects.

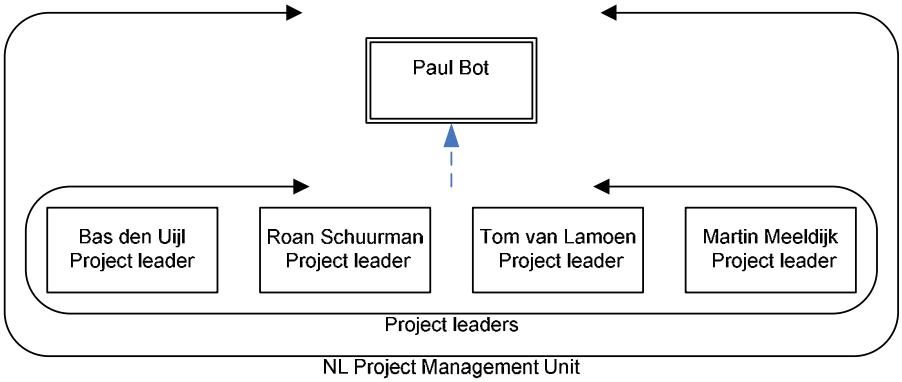


Figure 4: Dutch Project Management Unit

The Project Management unit is supported by the "Analysis, Project Support, Quality Assurance" unit, shown in figure 5. This unit mainly supports the Dutch project management unit.

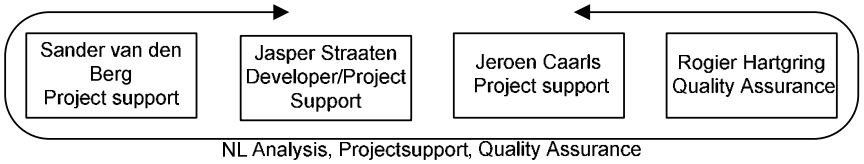


Figure 5

The Offshore Project Units (totalling about 120 people) is shown in figure 6. Componence Ukraine works mainly on developing new Portletsuite portlets, but is sometimes involved in projects too. SS is a major offshore partner (supplies 80 fte for Componence), and is managed by ML (CEO). SSt is the QA (Quality Assurance) manager. The QA unit includes a team of testers and assures the quality of the product as well as the process. AT is the CPO (Chief Project Officer), he reports directly to the Dutch project management. SS managers regularly take part in coordination and management of projects, and sometimes take part in coordinating other offshore teams when multiple offshore teams are involved in a large project. Otherwise, these teams report directly to the Dutch project management. Most current and future projects are BEA/Portletsuite based, SS is the only team that also does ECM Suite projects and provides support and maintenance for past ECM Suite projects.

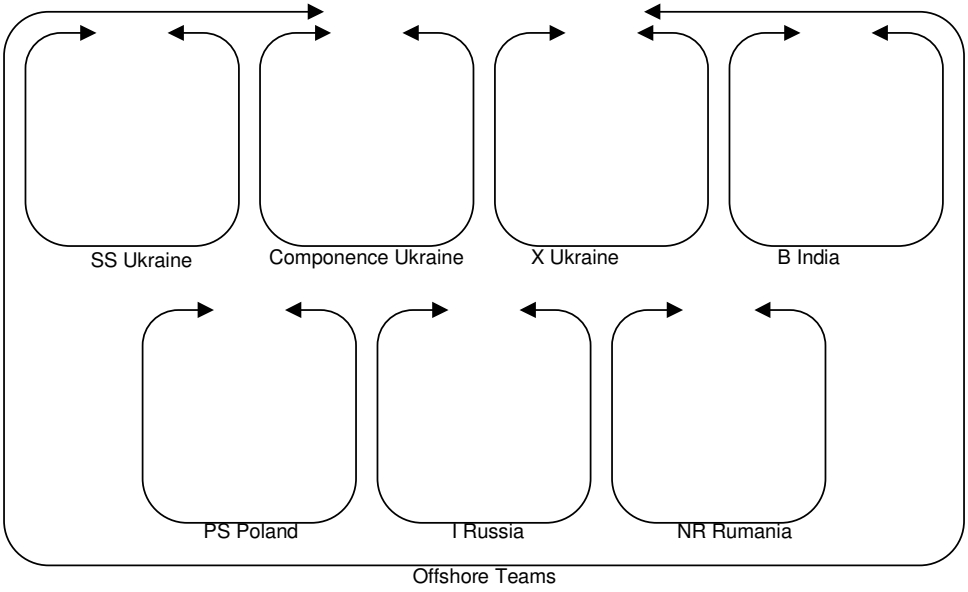


Figure 6

2.3.3 Organization chart: Operations in Detail

Figure 7 shows the operations unit in more detail. The Dutch hosting unit is managed by Tycho Luyben and Paul Bot. The Dutch support team provides customer support and if needed tasks are outsourced to the Ukraine support team. However, support is currently done mostly by the project teams in the Netherlands and Ukraine. The offshore (SS) Monitoring & Admin team handles operational issues (for instance, keep the hardware up and running, installing portals, etc). Besides reporting to Paul Bot the team also reports to the Project Management. The offshore Support team (usually this is SS) handles functional issues. Besides reporting to The Dutch Support team the offshore Support team also reports to the CPO. The Hosting team makes sure all software and documents are hosted and accessible to those who need them and make sure the portals are up and running by maintaining the hardware.

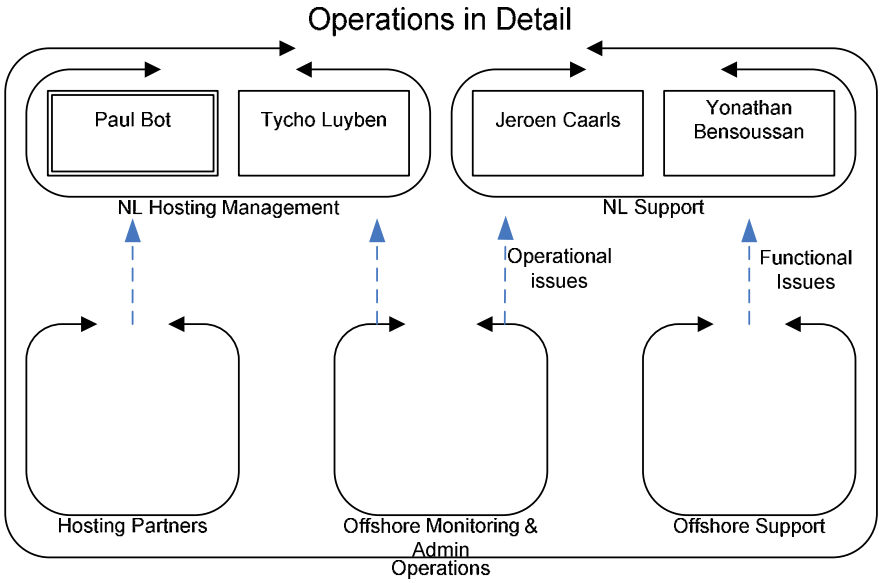


Figure 7

2.4 Offshoring benefits and drawbacks

As is evident, a large part of the production is done by offshore teams. In this section, the benefits and drawbacks are discussed. They should give a global idea of why Componence employs offshore teams and what problems this brings.

2.4.1 Benefits

The main reasons for Componence for doing offshoring are:

- Manpower is much cheaper in East-European countries and India;
- Offshore manpower is better scalable, both up and down (upscaling: manpower is easier to find and cheaper, downscaling: in East-European countries it is easier to push off manpower on short notice);
- East-European people generally have more technical and mathematical interests than Dutch people;
- The working mentality of East-European people is better suited for businesses like Componence, they are more flexible and hardworking.

2.4.2 Drawbacks

The main problems that Componence experiences with offshoring in East-European countries are:

- Communication problems because of different language, distance and time difference;
- The offshore teams have their own way of doing things that are preferably done in another way (cultural differences);
- A lot of coordination is needed between processes in Dutch and the offshore teams;
- Most of the offshore countries have a communistic legacy that might conflict with the "customer central" approach Componence has;
- There is no direct control of the work the offshore teams do. Measurements of performance can only be evaluated on deliverables;
- Quality assurance is more difficult.

3 DESCRIPTION OF THE CURRENT PROJECT PROCESS

In this section the current process inside Componence will be described to better understand the process and to form a basis for analysis and interviews. Figure 8 shows a typical project organization chart and who reports to whom. In some projects however, some units are omitted (for instance, not all customers have a customer project team or not all projects involve a project assistant). Because SS is currently the biggest partner/vendor and many projects are done with SS, the project process is described having this team in mind. Other offshore teams generally work in a similar way, but are generally smaller and some functions that are described as multiple disciplines could be done by the same person.

Typical project organization chart

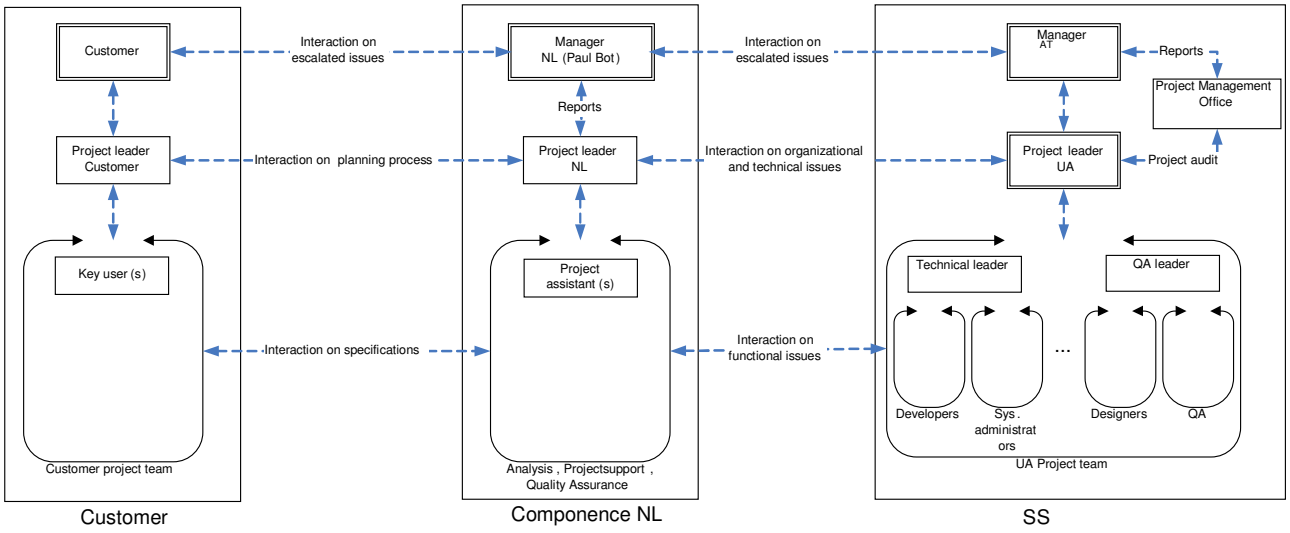


Figure 8

Currently, Componence is growing at a very high rate forcing her to restructure her business process in order to cope with the growth. The fastest growing department is the project team; this section will focus on the processes in this department.
The charts are made to gather more insight in:

- Metrics: the management would like to be able to check the status of the projects, its activities and the performance of the entire business using some high-level performance indicators, also called metrics. The models can be used to identify crucial activities. Metrics (and needed measurements) can be defined for these activities to measure performance;
- Identify what the corporate portal needs to support during the project process: existing and desired processes may need to be (better) supported by software (functionality). The models can be used to identify these processes by identifying activities that require coordination and communication;
- Efficiency of processes: existing processes could be running inefficient, the model can be used to identify these processes and provoke rearrangement of the processes;
- The people responsible for the processes: processes that are not running well might require more people. The model can provide more insight in what sub-processes are involved and how much work is needed and who is doing it for each sub-process.
- The activity diagrams can be used for identifying what processes run in parallel and what processes precede other processes.

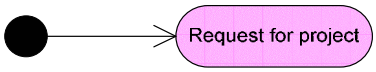
Furthermore, the models are also important in starting a discussion among people involved in the processes and have more people involved in identifying the problems and restructuring of the project process or organization. The models also help the people involved to have a better idea of what their responsibilities are and how they contribute to the entire business process.

3.1 The current project process modelled in activity diagrams

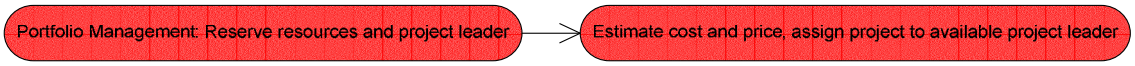
The project process, based on the industry standard Rational Unified Process (RUP) is extensively described using activity diagrams (also called flow charts) while other processes will be only touched upon or briefly mentioned. Appendix B shows the activity diagrams in their entirety, the text below describes the diagrams in more detail. It is recommended that these appendices are kept next to this text while reading. Note that "UA" (short for Ukraine) can be replaced by "offshore".

The chart "High Level Current Project Process" shows the project process on a high abstraction level. The project process can be divided in the sales and analysis, building, customer acceptation and project closing phase. Later phases can return to earlier phases. For example, when critical technical issues are discovered during the project closing phase, rework needs to be done and the project returns to the building phase. The other (sub) charts in appendix B show each of these phases in detail. Bear in mind that all projects are unique and the diagrams given represent a typical project process. Also, in the project process described below, the KlantNET issue tracking system is not included because KlantNET is not used for issue tracking during the project anymore.

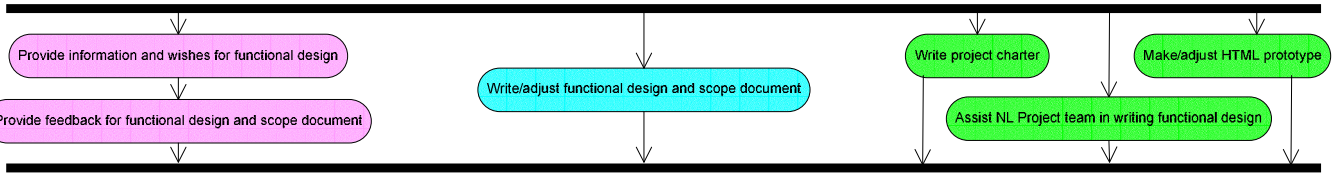
3.1.1 Activity Diagram Current Sales and Analysis Phase



Initial state: a customer requests a project and if the management decides to do the project, the project process starts. The sales process itself is not explicitly modelled in the activity diagrams as the focus lies on the project process.

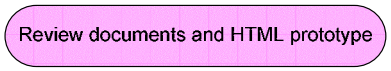


The management checks which of the project leaders is available for doing the assignment and hands it to that project leader and reserves the resources the management expects is needed (Portfolio management). An estimation of the price, cost and effort is made to determine the profitability and feasibility of the assignment.

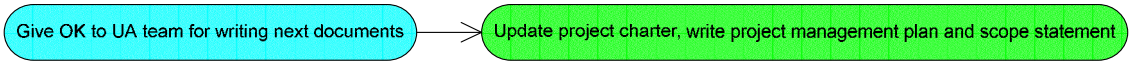


The customer is interviewed and the Dutch and offshore project teams write a functional design and scope document. These documents state what will be in the portal and what not, and describes the functionality of the portal. It is key that the customer's wishes are described well and that the customer is actively involved and provides feedback. This reduces the risk the product being too different than what the customer actually expected. The offshore Project team starts writing a project charter. This document's purpose is to give all units involved in building the product a better idea of: the context of the project, the goals, the deliverables/milestones, agreements between stakeholders, the assignment of resources, and in general produce a shared vision of the project. This document is continuously updated throughout the project. When the projectbudget permits, an HTML

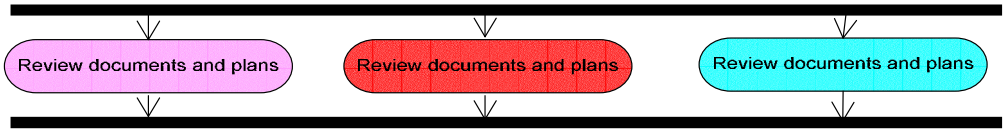
prototype is made to get a shared vision of how the portal should look like. The prototype has no functionality or portlets built in, it only gives an indication of what the portal will look like and where the portlets will be positioned.



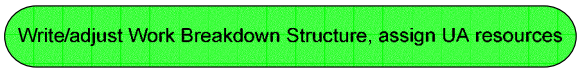
The customer then can reject or accept the documents and HTML prototype. If something is rejected, the customer indicates what needs to be changed and the Project teams make the necessary adjustments (if the adjustments are realistic). When the documents and HTML prototype are accepted by the customer, the project teams should have a clear idea of what to build.



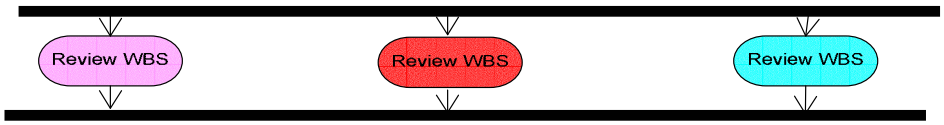
Now that the functional design and scope document are set in stone and it is clear what needs to be done the offshore Project team further elaborates the project charter, write a project management plan and a scope statement. The project management plan includes (amongst other plans): project scope management, schedule management plan, cost management plan, quality management plan, process improvement plan, staffing management plan, communications management plan, issue management plan, risk management plan, etcetera. The scope statement contains among other things: the requirements (including acceptance criteria), scope, project risks etcetera. However, the mentioned plans are not always written (yet) and are not widely used (yet).



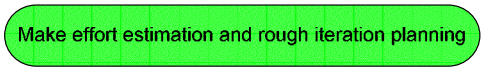
The customer, the management and the project leader give a formal OK to the offshore Project team if the project charter and scope statement are OK. If the documents and plans are rejected by someone the offshore team rewrites the rejected document.



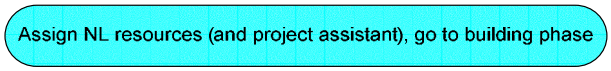
A Work Breakdown Structure (WBS) is written so everybody involved will know what activities are to be done and when. Now that it is clear what needs to be done the resources that are expected to be needed are assigned to the project.



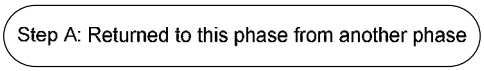
The WBS is reviewed by the customer, Dutch management and Dutch project leader and adjusted by the offshore team if needed.



The offshore team makes a detailed estimation based on the activities described in the WBS.



For larger projects the project leader can assign a project assistant (someone from the Analysis and Project Support team). Then the project enters the next phase: the building phase.



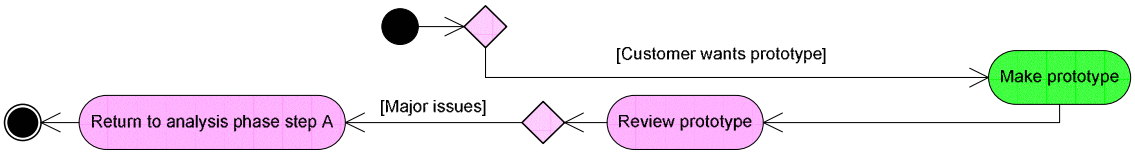
Whenever a return to this phase is necessary (for instance when a RFC (request for change) is done or the product is not what the customer expected at all) the project continues from this point.

3.1.2 Activity Diagram Current Building Phase

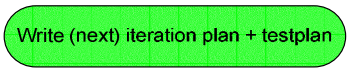
The building phase consists of different phases which can be divided in one or more iterations. This methodology is based on a combination of the PMBOK and RUP methodologies.

- Inception phase (I1): this is actually the analysis phase where the project charter, project management plan and WBS are written. Usually this phase consists of one iteration;
- Elaboration phase (E1, E2, …, E*n*): this phase can consist of more than one iteration depending on the size of the project. In these iterations the offshore team tries to get a clearer idea of the functionality and there is a lot of interaction with the Dutch team and customer. It can be seen as a preliminary phase to the construction phase, as the construction of the product has already started here. However, changes can be made in the specifications and scope in this phase, but not anymore when construction has started;
- Construction phase (C1, C2, …, C*n*): in the first construction iteration (C1) functionality is built in and the product is deployed on the test server after each iteration where testing is done by the customer and the Dutch and offshore teams. The issues resulting from these tests are posted in Jira, and the Dutch and offshore teams validate and prioritize the issues. The next construction iterations consist of a balance between building in more functionality and handling issues from previous iterations. When the building phase is nearing its end the balance shifts to mainly fixing issues as most of the functionality is implemented. It should be noted that formally there should be no changes made to the scope and specifications in these phases anymore. Unfortunately, requirements creep does occur in this phase and is one of the causes of duration and effort overrun;
- Transition phase (T1, T2, …, T*n*): the final phase of building consists mainly of fixing the final issues, all functionality has already been built in. Most of the developers have moved to other projects, only one or two developers are left for the last iterations, together with the system administrator. It is therefore key that there are no more complex issues to be fixed.

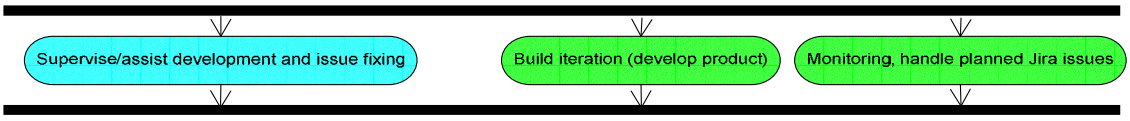
The flow of the building phase is usually as follows:



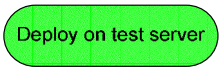
Initial state: after all documents in the analysis phase have been accepted by the customer and management the building phase starts. Usually (but not always), a browsable prototype is made so the customer has a clearer idea of what to expect. If the prototype is nothing like the customer imagined the product to be, the functional design was probably not good and a new one has to be made: the project returns to the analysis phase at step A. This rarely happens however.



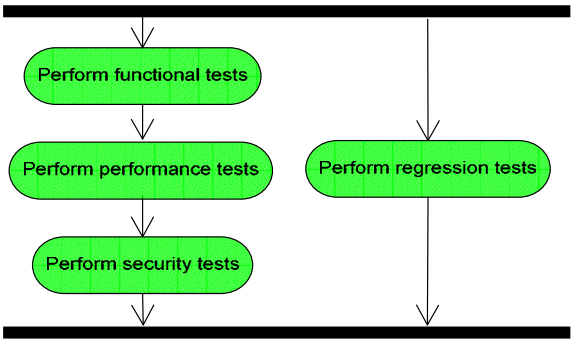
If there are no major issues (or if a prototype was not desired) an iteration and test plan is written. An iteration plan contains the scope of the work to be done or functions to be delivered in the coming iteration, the milestones of the iteration, the tasks and resources (people, financials, etc) involved, when testing starts, and the evaluation criteria (functionality, performance, capacity, etc). An iteration usually has a duration of one or (more often) two weeks, depending on the scope of the project. The iteration plan also contains a plan depicting which open issues will be fixed (issues from past iterations). The test plan states what will be tested this iteration and how.



During this parallel the iteration is executed. The offshore project team inserts or modifies existing portlets, develops new portlets and configures them. The Dutch project team supervises the iteration and assists the offshore project team during the iteration. There is a small team in The Netherlands that also develops but most development takes place in Ukraine. Any issues found during development are posted in an issue tracking system called Jira. The project management office audits the project on the offshore side at so called audit points (data determined in advance), and both project teams handle the issues posted in Jira. Depending on the nature of the issue, the issue is handled by the Dutch project team (configuration bugs) or forwarded to the offshore project team (technical issues that require changing the code).

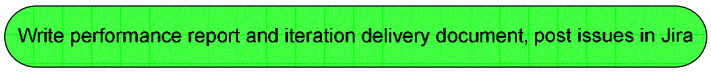


When the iteration is done the portal is deployed on the test server and testing starts.

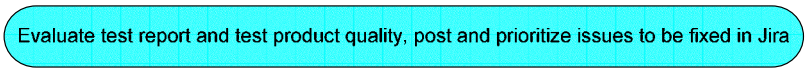


- Perform functional tests: the offshore team tests general functionality (however, because most portals are in Dutch some of the functionality testing should be done by the Dutch team). An example of a functional test: make sure the portal works as it should on different browsers;
- Perform performance tests: the offshore team tests different performance aspects of the site. An example of performance testing is testing the load the portal can handle for each page. The portal should be able to handle a certain amount of users at the same time. Performance is measured for a different number of simultaneous users;
- Perform security tests: the Dutch project teams needs to make sure the portal is sufficiently safe against attacks from the outside;
- Perform regression tests: after an issue is fixed, regression tests are performed to see if the fix affects the functionality of other components.

After the final iteration, sometimes the tests are done more extensively. This depends on the size and criticality of the project and customer.



After testing a report is written with the test results and a report is written stating what features were added/fixed during this iteration. All issues resulting from the tests are posted in Jira to be fixed in later iterations.

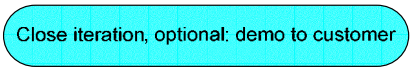


The Dutch team reviews the iteration: checking the quality of the added functionality during the iteration and reviewing the performance report. Iteration quality control consist of:

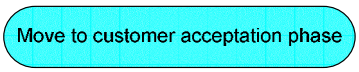
- Performance testing: make sure the portal runs acceptably smooth;
- Testing functionality: most of the customers' portals are Dutch, this makes it difficult for the offshore teams to fully understand the enduser's perception when using the needed functionality of the portal. The Dutch project team tests the portal by performing use cases and posts issues to be fixed in Jira for the offshore team to fix if the Dutch project team does not have the resources;
- Checking the quality of the code: the code should be understandable to other people than the programmers (for instance by adding comment) and the code should be efficient. This is important if the piece of software is possibly updated in the future or bug fixes need to be done. Unfortunately, this is currently not a standard process;
- Checking the quality of the documentation: because the objects developed might be reused in other and future projects, it is important the code is well documented and can be understood by other and future developers;
- Make sure the same versions of the software that is built in the portal are used: for instance, sometimes a project consists of more than one portal but is developed over more than one team. In this case it is important that the same versions are used for different components and software or the project ends up having different and/or faulty functionality for the different portals;
- Make sure the portal is secure: it should be difficult for outsiders to perform an attack on the portal of the customer.

Currently the above mentioned activities are often not done in current and past projects, mainly because of budgetary reasons, but according to the management they should be done.

The issues found after doing the tests and performing quality control are posted in Jira to be fixed in later iterations.

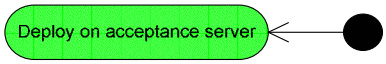


The iteration is closed and the Dutch team gives an OK to the offshore project team to start the next iteration (if more iterations are to be done). The offshore team usually starts writing the next iteration plan halfway during the current iteration. At the end of the current iteration the scope of the next iteration plan is discussed so the iteration can be started at the beginning of the next iteration. Sometimes a demo is given to the customer after an iteration.



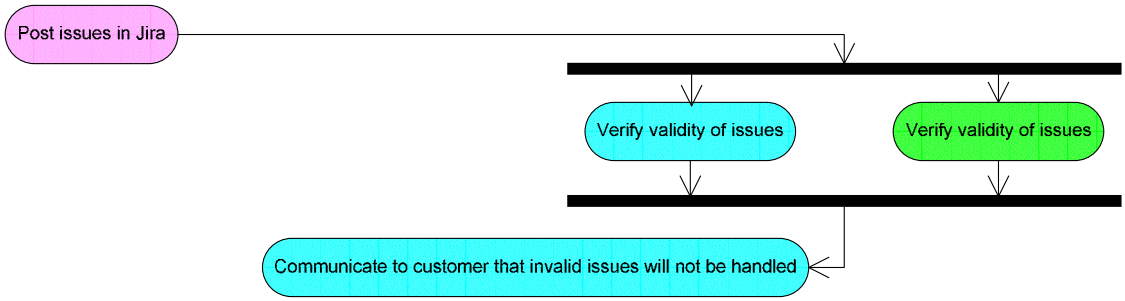
If no more iterations are left the customer acceptation phase is entered.

3.1.3 Activity Diagram Current Customer Acceptation Phase

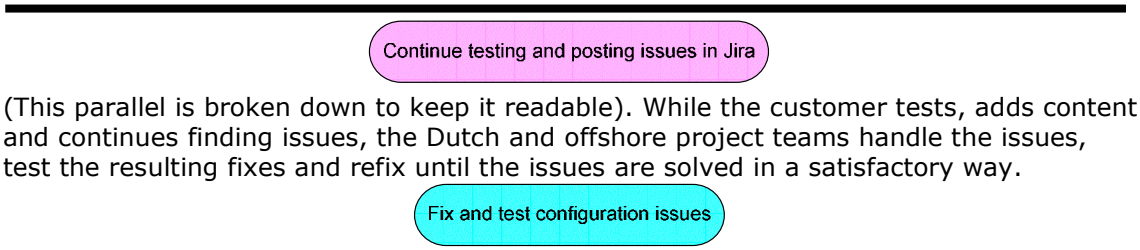


Initial state: the customer acceptation phase starts. The portal is now of sufficient quality that a customer acceptation test can be done. The portal is deployed to the acceptation server.

The customer tests if the portal performs as stated in the previously accepted functional design and tests for bugs. The Dutch team assists the customer while testing.

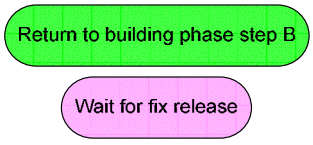


When the customer finds that the quality is unacceptable or not high enough to go live, the issues are posted in Jira. The Dutch and offshore project team check whether or not the issues are valid and decide whether or not the issues will be handled. Some issues might be converted to an RFC. This might happen when for instance a customer expected some functionality to be there but the functionality was not explicitly described in the functional design. Issues like these will not be handled and this is communicated to the customer. When an RFC is made this means that something went wrong during the analysis phase. If an RFC is made, this is treated like a side-project. For simplicity's sake this will not be described here.

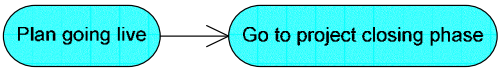


(This parallel is broken down to keep it readable). While the customer tests, adds content and continues finding issues, the Dutch and offshore project teams handle the issues, test the resulting fixes and refix until the issues are solved in a satisfactory way.

The Dutch team only fixes configuration issues. These issues don't require programming. If any issues reported are critical and require rework, the issues are being brought back to the building phase.

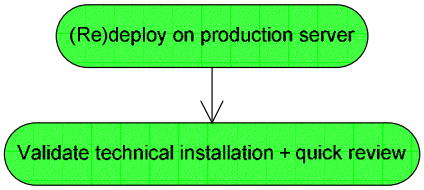


The customer then waits until the new build is installed on the acceptation server. If no critical issues are found and the configuration issues are fixed, the customer does another round of testing (the first parallel is revisited).



If the quality is sufficient (sometimes after the first time the customer is testing), the Dutch team prepares for going live and the project is moved to the project closing phase.

3.1.4 Activity Diagram Current Project Closing Phase

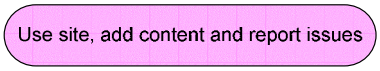


Initial state: the guarantee period starts. The portal is then deployed on the prod(uction) server. The install on the prod server is validated and a quick review is done (all links are clicked and basic functionality is tested). The portal is redeployed when there are technical issues (re-installed).

If the redeployment did not solve the problem, sometimes (if possible) a patch is installed to make the portal work for now. After the patch is installed the portal is redeployed with the patch. Eventually, to solve the real problem a visit back to the building phase has to be made.

If a patch is not possible or does not work, the project returns to the building phase again. This is very costly, because regression tests have to be done again, and there is relatively a lot of functionality installed. There is a great chance that fixing one component affects another component, and fixing that one affects another, etc. Most of all other tests have to be done too.

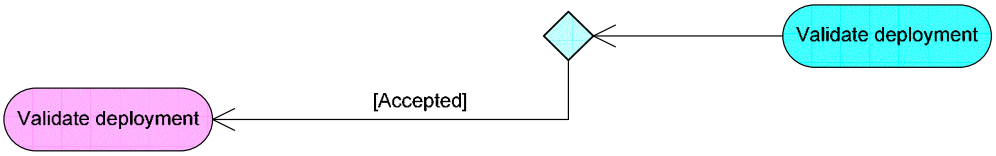
If no technical issues are found, the Dutch team validates the deployment. If the deployment is not good, the Dutch team gives feedback to the offshore team and the project returns to the redeployment step. The offshore team makes sure that the issues are fixed and redeploy the portal. If nothing is wrong with the deployment, the two-month guarantee period starts.



If the quality is sufficient the customer can start using the site, the portal goes live and visitors can visit the portal. Any issues found are reported in Jira. Basically, the same happens as in the customer acceptation phase here. The issues are validated and only valid issues will be fixed. Configuration issues are handled by the Dutch team and if any critical issues are found that require reworking the code, the project returns to the building phase again. If no critical issues are found, the configuration issues are fixed and after one guarantee month of issue fixing the portal is redeployed. This process repeats

until the portal is redeployed twice after the first and second month of the guarantee period.

After the guarantee period ends, the final version is installed and validated. If this goes well, the project closes; all necessary documentation and manuals are handed to the customer.



The Dutch team validates the deployment and if the quality is sufficient the customer can validate the deployment. If either of the validations results in a rejection, the portal may need to be redeployed or issues need to be fixed until the deployments are accepted.

The project now enters the support phase. Any issues from now on are to be handled by the support teams. Eventually the portal is stable and the number of issues posted by the customer comes to a minimum. The issues still need resolving, but should come in at a very low frequency.

4 PROBLEM ANALYSIS OF THE CURRENT PROJECT PROCESS

After interviewing different stakeholders, including Dutch managers and Ukrainian managers, it is clear that **cost and planning overruns are currently the biggest problems**. These are caused by several sub-problems, the most critical problems are discussed in this chapter. The flow charts in appendix A were used during these interviews, they allowed us to point to different activities and discuss each of them.

4.1 Problems in the sales and analysis phase

This phase is also called inception phase.

- Not enough planning is done. Often when a (simple) project starts the people just start working without a plan and/or schedule;
- For commercial reasons, the effort estimation (or time to delivery) often is unrealistic, putting a lot of pressure on the projectmanagers and the development teams. Most of the time the initial estimations are done without the consult of an offshore project team manager (most of the time SS manager) who often disagrees with the commercial estimations, but has to work with them. Development and testing then need to be rushed in order to meet deadlines. This results in poor product quality and in turn requires more rework.

4.2 Problems in the building phase

This phase consists of the RUP elaboration and construction phases.

- Unclear specifications. This is especially costly when combined with a tight schedule. The developers need to develop some functionality that they do not fully understand in a short time. There is little time to gain a better understanding of the functionality (elaboration phases) and little time to demonstrate the functionality to the customer and receive feedback. This results in dissatisfied customers and a high rework rate. The project scope is changed (read: expanded) too often (also called requirements creep). This results in extra unplanned work and causes delays that can also affect subsequent projects due to the resources that cannot be released from the project;
- All the steps described in the project flow are not always carried out or are carried out insufficiently. Performance, regression and (deep) functional testing are often skipped when building took more time than planned. This results in bugs that are discovered in later phases or discovered by the customer and in turn result in cost overruns, more rework and customer dissatisfaction;

4.3 Problems in the customer acceptation phase

- Bugs that should be discovered in the test sessions in the building phase are discovered in this phase too often. Also, bugs that should be discovered internally are too often discovered by the customer. This results in returning to the building phase -which is very costly- and customer dissatisfaction.
- Sometimes the customer has not been trained how to "behave properly". This means that the customer has received insufficient instructions in posting issues and understanding the scope and contracts. This results in too many invalid issues/non-issues (and in turn more effort validating issues), or the scope becoming bigger (and in turn extra unplanned work).

4.4 Problems in project closing phase

This is also known as the RUP transition phase.

- Problems during deployment:
  - Memory leaks;
  - Usage of wrong versions;
  - Deployment and configurations require too much manual effort;

- The first wave of issues after the deployment on the prod server is too large. This is caused by issues being discovered too late and in turn is caused by insufficient testing. Insufficient testing is in turn caused partially by cannibalizing on testing;
- Too many performance and stability problems arise during the guarantee period, i.e. the delivered sites are down too often or too slow. This is the case for almost all BEA projects. The offshore teams did not have enough time to get to know the BEA platform really well before working with it and therefore have not been able to optimize performance on this platform.

5 POSSIBLE IMPROVEMENTS

The possible improvements mentioned in this section are results from interviews and discussions with Componence managers and employees. Some implementations of improvements are already in motion. Of course, not all improvements will be implemented in this study. However, a part of the requirements of this study is to also give an advice for what can be improved; therefore this chapter contains a list of possible improvements. [QW] stands for "quick win", an improvement that can be implemented on short term. [MT] stands for "mid term", the improvement will probably not show immediate results, but may do so on mid term. [LT] stands for "long term", results may show after a longer period of time and constant effort should be put in it.

5.1 What can be improved in general?

- [MT]/[LT] Using standard documents (templates) or reusing documents for all documents written in the project. Currently, SS's Project team has started making templates, but they are not used widely yet and the templates are not made for all phases. On top of this, not all Dutch project managers are convinced of the necessity of some of these documents. The difficulty in standardizing everything is that each project is considered unique. For instance, in some projects a designer is hired by the customer who also does the functional design, but this is not always the case. Sometimes the Componence project team therefore has to write the functional design herself. Sometimes the functional design of the external designer contains functions that are not possible to implement and adjustment of the functional design is needed. Also, the customer sometimes has no clear idea of the needed functionality of the product and more effort is spent during analysis.
- Returning to the building phase from a later phase is very costly. The later the phase that is returned from, the costlier it is, as fixing bugs require another round of different tests, could result in defects in other components, require another deployment, etc. So preferably critical issues are discovered during internal testing as much as possible. To reduce returning to the building phase:
    - [QW] More effort and time should be put (and reserved) in testing and QA in the earlier phases so bugs and performance issues are found during building and not after deployment on production;
    - [QW] More reusing components and/or portlets (that are less likely to contain bugs);
    - [QW] Technical audits should be performed to discover memory leaks in an early stage, resources can be assigned for reviewing the quality of the code, tools can be used for discovering bugs;
    - [QW] There should be better communication between all the different units to prevent the usage of wrong versions and more knowledge sharing and reusing options should be present.
- [MT]/[LT] Using a set of metrics the performance of different activities in the project process can be measured so that the management can track the impact an implementation of an improvement has;
- [QW – in progress] Calculating and managing on target allowed amount of hours, that can be used by management and development resources for each project. It seems that substantial parts projects are carried out because of availability of resources and lack of profitability awareness. Resources are sometimes not re-assigned fast enough. With a clear target the Dutch and offshore managers will be able to manage resources more profitable. Especially time-material projects will be done more profitable. While writing the last versions of this report, Componence and SS have started to make this a part of the Project Charter;
- [MT] Increasing Project Management skills and efforts to manage expectations of clients better. It is commonly known that many projects are delivered satisfyingly despite of the fact that the delivered scope is far under the original scope when

the project started. Componence project managers just don't spend (have enough) time to utilize this tool to increase the satisfaction of customers.

5.2 What can be improved in the current sales and analysis phase?

- [LT] Currently there is no set routine for determining the price, cost and effort of the portal(s) to be delivered (commercial estimation). A tool will be needed to make more accurate estimations. This is to make sure the project process will run more smoothly and capacity can be better assigned and managed.
- [QW] The commercial estimation made towards the customer should be done with the consult of a project manager or someone who has sufficient experience in developing similar products;
- [QW] The commercial bid towards the customer should be separated from the "real" effort estimation. The commercial bid should not be a constraint for the development teams. The management should consider not only the direct costs versus direct profits when developing a product, but also take the long term profit into consideration when bidding and making an effort estimation;
- [LT – in progress] There should be an analysis of what portlets can be reused. This also requires the documentation of the portlets to be of sufficient quality and requires the portlets to be managed. When reusing and adapting existing portlets the code should also be well structured and self descriptive to improve reusing possibilities. Reusing has several benefits:
    - Saving time by not having to develop similar functionality and write similair specifications;
    - Less need for bug fixing;
    - Easier to provide support;
    - It is easier to estimate the effort needed for implementing existing portlets.

  Currently Componence is setting up Solution Consultants and Architect disciplines to support different procedures in the Inception and Conception phase to increase reusability of solutions and the level of reusage in projects;
- [QW]/[MT] The inception phase should be more integrated in the sales and analysis phase, rather than be in the construction phase, as analysis is typically done in the inception RUP phase. Currently, SS treat the analysis and building phases as separate projects;
- [QW] Building should not start until the portal specifications are approved to the customer. This prevents the customer ending up with some functionality that they expected to be different and in turn rework is needed. In other words, better closure of the functional design is needed.

5.3 What can be improved in the current building phase?

- [QW] Iteration plannings should be made two iterations ahead of time. Currently only a planning for the next iteration is made. This gives better steering of the building phase;
- [QW] The progress in development of a portal should be easier to keep track of. The offshore developers should be able to provide the status on different components of the product on a portal accessible to the project leaders and managers in the Netherlands. The goal is to make the progress measurable to identify schedule overruns earlier and take measures on time. Demanding this however, requires the developers to put effort in providing the statuses of all activities, but this could be easily implemented in Jira by for instance posting all tasks at the project's start and then closing the tasks when finished. Another reason this problem exists is that the tool used for planning by SS (Microsoft Project) is not widely used by Componence, this should change now that Componence has enough licenses. However, the problem of not updating the

statuses for all activities should also be resolved. Therefore, the offshore teams should also update the project status more regularly;

- [LT] An evaluation of the project is to be done on a regular basis (for example after each iteration) so management has a better overview of what went right and wrong in different projects. This can help the management plan projects better in the future;
- [QW] The software version administration should be done consistently, through a single portal to prevent usage of different versions of the same piece of software;
- [MT] Code audits should be performed. This helps keeping the quality of the code high so reusing becomes easier and cheaper, support becomes easier, and this increases the quality of Componence products in general;
- [QW] Doing so called Maven tests/unit tests, these tests automatically check if changes in software code affect other functionality. This results in more efficient regression testing during the building phase;
- [MT] Documentation of the produced code should be of higher quality so support can be done by other people than the developers themselves (i.e. the support teams). The documentation is also to be shared on a portal (for example using Sharepoint). Better documentation also leads to better reusability of the code and documentation;
- [MT]/[LT] Stimulate reusing objects/components and products as much as possible which makes it easier for the future support teams and eliminates the need for programming similar components. Not only the code can be reused but also the documentation that comes with it;
- [LT] More knowledge sharing is desired: developers should maximize reusing of (each other's) existing software and objects and interchange knowledge and ideas. To support this, a portal should be available where developers can communicate the functionality of the software and objects they created and ideas for new components or objects can be shared;
- [QW] Currently most of the tests are only done at the end of the project, or not done at all. All tests should be carried out in every iteration in the whole project in the future. This way more issues are found earlier in the construction phase and it takes pressure off the project team in the guarantee period and support team after that. The testing team should report to the management on a regular basis, to make sure testing is done. The reports does not need to be extensive, the management just wants to make sure that testing was done;
- [QW] Better standard testing checklists and tools should be available for the testing team so testing is done more consistently and efficiently;
- [QW] The project teams might consider assigning the tasks of fixing issues to different persons depending on the difficulty of the task;
- [QW] No cannibalizing on testing. When development takes longer or more resources are needed for development, the testing department is often cannibalized. This means that less testing is done and testing resources are used for development. This results in untested or insufficient tested code. This in turn results in more critical issues in later phases and revisiting the building phase which is very costly;
- [MT – already implemented] After the guarantee period the project should be handed over to the support team. But the project team should still be responsible for support in the guarantee period. This way, the management is sure the development teams do their best to make the portal as good as possible, as they have to "clean their own mess" if they were to deliver a messy product. The project team can then concentrate on new projects and let the support team handle customer support. At least performance testing and tests for memory leaks should be carried out and the resulting issues should be fixed before the project is handed to the support team. Currently, all projects on the BEA platform have performance issues and many have memory leaks even after deployment on production.

Preferably all points that require a portal are to be implemented and integrated in the same portal. SS is using Microsoft Sharepoint for current sharing purposes, and are configuring it for the Dutch team. For issue management there are two issue tracking portals, one called KlantNET and one called Jira. Eventually, Componence wants to use one issue tracking system. Preparations are now being made to move all issues to the Jira system. In July 2006 this should be completed.

5.4 What can be improved in the current customer acceptation phase?

- [QW] (A member of) the support team could be involved in this phase. The support team can learn more about the customer and the current issues might relate to issues during the customer support phase.
- [QW - already implemented] There should be one integrated issue tracking system. Because KlantNET and Jira coexist, not all issue data are synchronized and the support process runs very inefficient. The switch to use only Jira for all future projects is now in motion, as Jira is more advanced and offers more functionality. KlantNET will still be functional for current customers that are accustomed to using KlantNET, but KlantNET will not be used for issues during the development phase anymore. Some customers are using Jira already, and it is expected that all customers will use Jira in the future for issue posting. In July, everything should be migrated;
- [QW] The acceptance process should be more formal and standardized.

5.5 What can be improved in the current project closing phase?

- [MT]/[LT] The deployment on the production server should be more automated. This can prevent faulty installations. A standard checklist should be used for the quick review, so issues can be discovered internally rather than by the customer. Also, more time should be planned for deployment and the testing of the deployment for achieving a higher deployment quality;
- [LT – now being implemented] Currently support teams in Nieuwegein are beginning being parted from the project teams. A support team is currently being formed at SS to take load off of the project teams, supported with resources from other offshoring partners. Customer issues are now usually assigned to the project leader. If an issue is to be forwarded to the offshore teams it is usually assigned to the developer of the piece of software that the issue belongs to. The management prefers the project workers to focus on working on new projects after a portal has been delivered. The support department should be focusing on support only, in the Netherlands as well as in Ukraine to take over customer support from the project teams. The issues that are to be handled by the support team therefore should also be assigned to the support teams and not to the project leader. This could be done by for instance making a separate "project" in Jira when the support phase starts and assigning issues to the correct support persons.
- [LT] Invest more time in learning BEA in order to improve performance of the portals;
- [QW – now being implemented] Customer support issues are to be assigned to the project leader and the support team in the first two months (guarantee period), and after that support is provided exclusively by the support team. This way, the management makes sure the project teams deliver portals of sufficient quality as they have to fix the problems themselves if they deliver portals with many bugs and performance issues.

5.6 What problem areas will be focused on?

5.6.1 Selection of problem areas

The problems mentioned in the previous chapter and the possible improvements were discussed with the management. It should be clear that most problems can be traced back to root causes, and that most problems overlap each other and/or influence one another. The following problems are believed to be the root causes of most other problems and will be focused on in the rest of this study:

1. Commercial effort estimations and planning are too optimistic and/or unrealistic;
2. Not enough internal testing is done and/or level of testing is too low (resulting in performance issues among other things);
3. Unclear or no project specifications, and templates for these specifications are not widely used yet;
4. Too much requirements creep throughout project, the Dutch project leaders need to guard the scope better and/or stick to the plan made;
5. Too many issues arise after deployment on production server (the so called first wave of issues);
6. There should me more focus on reusing.

5.6.2 Relation and influences of the problem areas

The chart below (Figure 9) describes the relations between the problem areas (except for reusing) and how they are believed to be influencing one another. Keep in mind however, that the diagram below only gives an indication of how the selected problem areas are believed to influence one another. Of course, there are other external factors involved that are not in the diagram below.

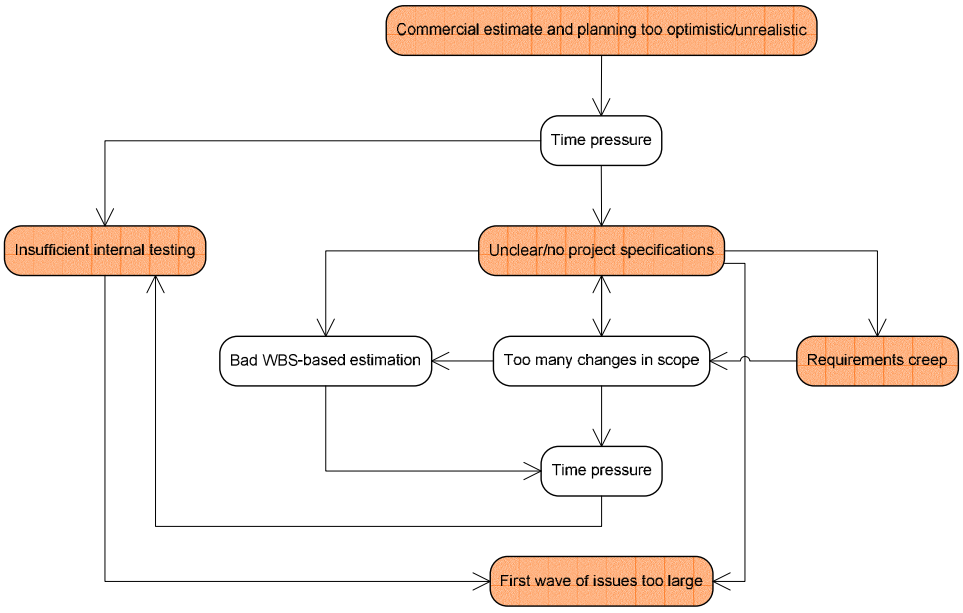


Figure 9: selected problem areas and their relationships

Additional details:

1. Unrealistic or overoptimistic commercial effort estimation leads to time pressure. Because of this time pressure there may be less or no time for product specification

and the time needed for testing may be cannibalized in order to create time for development;
2. Due to insufficient or bad testing, bugs and issues may be discovered later, for instance after deployment (first wave of issues). Keep in mind that insufficient or bad testing can also occur even when there are no time pressure issues;
3. Time pressure may lead to a poor product specification. Sometimes the time pressure is so high that there is no time to make a project specification at all. This may lead to an unclear WBS and in turn a WBS-based estimation is not possible or results in a poor estimation. A poor product specification may also lead to changes in the scope that in turn leads to changing the WBS and schedule and to more time pressure (which can result in less testing). Also, the scope is hard to guard if the specifications are either missing or unclear;
4. Too much requirements creep leads to too many changes in the scope. This in turn leads to more time pressure and changes in the WBS;
5. When the first wave of issues is too large this may lead to overtime, cost overruns, more time pressure and customer dissatisfaction. Especially the performance and stability problems lead to chaos;
6. More reusing could reduce the effort needed for writing specs (as the specs should already exist for reused portlets), construction and testing. Also, the effort needed to implement a reused portlet could be better estimated. It also should reduce the number of issues in general, as the portlet was used before and most bugs have already been found.

In the next chapter a set of metrics will be defined to measure the performance of activities in these problem areas. This gives the management the possibility to monitor the developments in the problem areas.

6 METRICS

6.1 Reasons for using metrics

Metrics (or performance indicators) can be used to understand what is happening during the projects and they help the management to control and improve the project process and its different sub-activities. Using metrics the management can track the impact of the implementation of improvement programs. Currently, at the end of the project metrics are presented in what is called a post mortem report, these metrics are presented fairly consistently after each project, but they exist only for the projects that have finished in the last half year. However, the managers feel that these metrics are mainly qualitative and that hey need more hard figures. During the projects, metrics reports are either set up in an ad hoc way, i.e. project managers extract the needed data and make a custom report which may or may not be consistent with other project manager's reports. These metrics therefore are hard to compare with each other as they lack consistency. Also, no substantial effort has been made so far actually comparing different project's metrics and data. Measuring data, setting up metrics and analyzing them however, can result in more insight in the process and product. Fortunately, the management sees the importance of metrics and this is key in a successful metrics program. However, it can be a challenge in maintaining the interest as results don't show immediately, but rather on long term (Pooley et al. 2002). Therefore, in order to successfully reap the benefits from a metrics program the management should stimulate measurement of valuable data and convince the employees of the usefulness of the data and metrics.

In this section metrics will be proposed for tracking the performance of the major problem areas using existing data. The goal is to define metrics that require no or little extra effort measuring, but are still very useful for measuring the performance in the problem areas. Some hypotheses will be discussed. An analysis and discussion of the data and metrics will result in an advice of what metrics are the most useful and interesting to keep track of. Note that the initial proposal for metrics can contain metrics that are found not to be useful or good after the analysis.

6.2 Initial proposal for metrics and motivation

In this section metrics will be defined for the selected problem areas mentioned in the previous chapter. Metrics are calculated from data, the data necessary for the metric are specified for each metric. Also, the following parameters have been assigned to each of the metrics:

- Metric occurrence in time: (IN ADVANCE, DURING PROJECT, POST MORTEM);
- Extra data measuring effort needed: (NO, LOW, MED, HI);
- Effort calculating or making metric: (LOW, MED, HI).

A metric having the parameters {IN ADVANCE, LOW, MED} means that the metric can be measured at the start of a project, requires some (but little) extra effort measuring the data needed for the metric (so the data is not available at hand), and requires some (a medium amount of) effort calculating or making (for example, some data might need to be manually prepared for input). The following metrics were calculated for a number of projects, and from the results of an analysis and discussion a smaller set will be selected for the "final" list of recommended metrics.

6.2.1 Metrics for the quality of (the commercial) effort estimation:

- Total number of people involved in the commercial estimation and their role in the project. The more people involved in the estimation, the better the quality of the estimation is in general. But also, some people are more qualified or skilled in making effort estimations than other. One of the major issues from SS was that the commercial estimation is often done without the consult of a SS project manager. Making a joint estimation should give the SS project team and its

developers some influence in the degree of feasibility of the allowed time span. The management can check if there are correlations between this metric and schedule overruns and overwork and determine how important it is to involve an offshore project manager in the commercial estimation.
DATA: number of sales, Dutch project, and offshore project managers that were involved in making the commercial estimation.
PARAMETERS: {IN ADVANCE, LOW, LOW}

- Accuracy of commercial effort estimation: realised effort/(commercial) estimated effort:
  This metric gives an indication of the quality of the commercial effort estimation. A value of smaller than 1 indicates a too pessimistic estimation, a value of greater than 1 indicates a too optimistic estimation. The allowed or estimated effort hours can be derived from the price paid for the project by the customer. One SS man-hour should cost CONFIDENTIAL Euros for the customer, and one man-hour in the Netherlands CONFIDENTIAL Euros. The ratio of offshore:Dutch resources is approximately 9:1. This leaves the number of estimated hours or effort for SS = price/(0.9* CONFIDENTIAL + 0.1*CONFIDENTIAL)/0.9 ~ price/CONFIDENTIAL.
  DATA: actual effort, project price.
  PARAMETERS: {POST MORTEM, MED, LOW}
- Accuracy of WBS effort estimation: actual effort/WBS effort estimation:
  This metric gives an indication of the quality of the WBS estimation. Again, a value of smaller than 1 indicates a too pessimistic estimation, a value of greater than 1 indicates a too optimistic estimation. Note: when a WBS was not present, the amount of hours (assigned) reserved for the project was taken as WBS estimation.
  DATA: actual effort, estimated effort as in project plan WBS.
  PARAMETERS: {POST MORTEM, MED, LOW}
- Absolute effort overrun: actual effort – (commercial) estimated effort:
  The previous effort accuracy metrics are relative and may give a false impression depending on the absolute numbers. An absolute overrun of 100 hours on a project that was planned for 100 hours is less dramatic than when there is 1000 hours of overrun on a project of 1000 hours. Both however, have the same accuracy of commercial effort estimation: 2.0. Therefore, the relative accuracy should always be considered together with the absolute accuracy.
  DATA: actual effort, estimated effort (based on price).
  PARAMETERS: {POST MORTEM, MED, LOW}
- WBS effort estimation/(commercial) effort estimation:
  This gives an indication of the amount of work that was under- or overestimated by the commercial estimation compared to the WBS estimation. This metric is useful because it immediately shows whether or not the commercial estimation was realistic according to the amount of work to be done.
  DATA: WBS effort estimation, project price.
  PARAMETERS: {IN ADVANCE, MED, LOW}
- Accuracy of (calendar) time estimation: actual time/estimated time:
  This metric gives an indication of the quality of time estimation. A value of smaller than 1 indicates a too pessimistic estimation, a value of greater than 1 indicates a too optimistic estimation. Keep in mind that this is not the same as the previous metric. The previous was total effort, this one concerns time. For instance: effort of 16 hours can be done in two days by one person or in one day by two persons. Also, the estimator should be aware of holidays in the period.
  DATA: actual start and live date, estimated start and live date.
  PARAMETERS: {POST MORTEM, NO, LOW}
- Amount of rework:
  How is rework defined? In general rework is the work that results from resolving defects. However, at componence the amount of time taken to resolve issues is not consistently measured. To still get an indication of the amount of rework, the number of bugs will be used as an indication of the amount of rework.

DATA: number of bugs.
PARAMETERS: {POST MORTEM, LOW, LOW}

- Amount of rework per construction day:
  In order to compare projects with each other, the rework (number of issues) per construction day has to be compared. A total number of 1000 bugs is more dramatic in a project that lasted one month than in a project that lasted six months. This metric can be used for future rework-effort estimations. Because most to be analyzed projects are still in their guarantee period, the total number of issues will be taken at three moments: at live date, after one month of guarantee, after two months of guarantee.
  DATA: number of issues, number of construction days.
  PARAMETERS: {POST MORTEM, LOW, LOW}
- (Planned) custom portlets/total (planned) number of portlets:
  (see below: Metrics for the degree of reusing)

6.2.2 Metrics for the quality of testing:

- Number of different tests actually done during the project before going live (0-5):
  This number indicates the completeness of testing. It is expected that more testing should result in more bugs being found before going live, and less after going live.
  DATA:
  - Functional testing done (no: +0, yes: +1)
  - Stress testing done (no: +0, yes: +1)
  - Performance testing done (no: +0, yes: +1)
  - Security testing done (no: +0, yes: +1)
  - Customer acceptation testing done (no: +0, yes: +1)

  So: no tests done gives a value of 0 and all types of tests done a value of 5.
  PARAMETERS: {POST MORTEM, NO, LOW}
- Number of bugs found after deployment on prod (live)/total number of bugs:
  This ratio gives an indication of bugs that were missed during testing and hence gives an indication of the quality of testing.
  DATA: number of bugs after deployment, total number of bugs.
  PARAMETERS {POST MORTEM. NO, LOW}
- The percentage of bugs posted by {customer, Dutch team, offshore team}:
  Ideally (but unrealistically) this distribution should be {0, 0, 100}. This means that all found bugs were found by the offshore team and testing was done very good. If many bugs were posted by the customer this means that the quality of testing was poor.
  DATA: number of bugs posted by customer, Dutch team, offshore team.
  PARAMETERS: {POST MORTEM, NO, MED}
- Load performance requirements available:
  If no load performance requirements are available, this could influence the quality of performance testing as it might not be clear what performance level is desired and therefore what level of testing is required.
  DATA: performance requirements.
  PARAMETERS: {IN ADVANCE, LOW, LOW}
- Response time requirements available:
  The same as previous metric, if these requirements are not available, it is unclear what level for response time is needed.
  DATA: response time requirements.
  PARAMETERS: {IN ADVANCE, LOW, LOW}

6.2.3 Metrics for the quality of product specification:

- Page coverage of functional specs: number of web-pages described/number of web-pages to be implemented.
  This metric gives an indication of how many pages were covered by the functional specifications.

DATA: sitemap in specs (number of pages to be implemented), number of pages described in specs.
PARAMETERS: {IN ADVANCE, NO, MED}

- Portlet coverage: number of unique portlets described/number of unique portlets to be implemented.
  This metric gives an indication of how many portlets were covered by the functional specifications. Standard portlets have existing documentation, and don't need to be specified.
  DATA: number of unique portlets to be implemented, number of portlets described in specs.
  PARAMETERS: {IN ADVANCE/POST MORTEM, LOW, MED}
- Visual design page coverage: number of pages present in VD/number of pages to be implemented
  If a visual design is present for a page it should be easier for the developers to position the portlets on the pages and make a style sheet.
  DATA: number of pages in VD, sitemap in specs.
  PARAMETERS: {IN ADVANCE/POST MORTEM, NO, LOW}
- Number of RFCs
  A large amount of RFCs means that a large amount of functionality was missed in the functional specs.
  DATA: total number of RFCs.
  PARAMETERS: {POST MORTEM, NO, LOW}

6.2.4 Metrics for the degree of requirements creep:

- Number of RFCs that ended up being built in the current project/number of RFCs requested:
  This metric gives an indication whether or not the scope has been successfully guarded (so requirements creep was kept to a minimum).
  DATA: RFCs requested (improvements/new functionality), RFCs implemented
  PARAMETERS: {POST MORTEM, NO, LOW}
- Number of pages actually built/number of pages initially planned:
  An indication of the growth of the scope.
  DATA: number of pages implemented (sitemap actual portal), number of portlets to be implemented.
  PARAMETERS: {IN ADVANCE/POST MORTEM, LOW/MED, LOW}
- Number of portlets actually built/number of portlets initially planned:
  An indication of the growth of the scope.
  DATA: number of portlets implemented, number of portlets to be implemented.
  PARAMETERS: {IN ADVANCE/POST MORTEM, LOW/MED, LOW}
- Number of unplanned custom portlets:
  An indication of the growth in scope.
  DATA: total number of custom portlets, total number of custom portlets in specifications.
  PARAMETERS: {POST MORTEM, MED, LOW}
- Actual custom portlets/actual total number of portlets
  Actual degree of customization. Also gives an indication of real degree of reuse.
  DATA: actual number of custom portlets, actual total number of portlets.
  PARAMETERS: {POST MORTEM, MED, LOW}

6.2.5 Metrics for the size of the first wave of issues:

- A chart of the accumulation of posted and open issues at different moments in time:
  An overall view of the development of issues over time. This chart shows the cumulative number of issues posted and open issues.
  DATA: cumulative number of posted and closed issues (excl. (sub)questions and (sub)tasks) every week.
  PARAMETERS: {DURING PROJECT/POST MORTEM, NO, HI}

- Percentage of issues posted before live date (date on prod server) and after live date:
  This metric gives a quantitative insight in the first wave of issues.
  DATA: total number of issues posted before live date, total number of issues posted after live date.
  PARAMETERS: {POST MORTEM, NO, LOW}

6.2.6 Metrics for the degree of reusing:

- (Planned) custom portlets/total (planned) number of portlets:
  This ratio indicates how much functionality has to be custom built and hence the reuse factor and uncertainty of the project.
  Also, if the uncertainty is high accurate estimations are harder to make.This metric should give the management a better idea of how much deviation in required effort to expect and help her manage her portfolio.
  DATA: number of custom portlets, total number of portlets.
  PARAMETERS: {IN ADVANCE, MED, LOW}

The metrics above are related to the problem areas mentioned. Besides these mentioned metrics and data, more data and metrics were collected. Refer to Appendix C for the entire set.

6.3 Data collection

The data needed for the metrics (and more) were collected for the following portals:

Type	Portal		Type	Portal
PS	Toeristiek		TITLE	Zorgvisie
PS	Agis		TITLE	SalarisNet
PS	Boerderij		TITLE	PenoActueel
PS	FEMBusiness		TITLE	Vakblad AGF
PS	Fiscaal Totaal		TITLE	Pigprogress
PS	Gemeente		TITLE	Vakwerk
PS	Personal Finance		TITLE	TTM
TITLE	ZorgWelzijn		TITLE	Nursing
TITLE	Primair Onderwijs		TITLE	Tuin en Landschap
TITLE	Voortgezet Onderwijs		TITLE	Overheidsmanagement
TITLE	Sociaal Bestek		TITLE	FZ
TITLE	Bizz.nl		TITLE	Landleven
TITLE	WorldPoultry		TITLE	Ornet
TITLE	DeBoomkwekerij		TITLE	Flexmarkt
TITLE	WeekbladGroentenEnFruit		TITLE	Ruimtelijke Ontwikkeling
TITLE	VakbladBloemisterij		TITLE	Opleidingen Ontwikkeling
TITLE	BloemEnBlad		TITLE	Marketingtribune
TITLE	MKB-E/Gribb		TITLE	Controllers Magazine
TITLE	Distrifood		TITLE	Reisrevue
TITLE	Agrarisch Dagblad		TITLE	ICTZorg
TITLE	Pluimveehouderij			

Table 1: portals for which data and metrics have been collected respectively made

Where TITLE stands for titlesite, and PS stands for Portletsuite (the platform the sites are built on, see chapter 2). The titlesites are actually one large project consisting of many sub-projects. Each sub-project results in a portal that has almost all of its components reused. Each title portal has many characteristics in common:

- Each titlesite has the same price;
- Each was planned to be built in nine weeks;

- Most were managed by the same Dutch project manager;
- Approximately, the same amount of resources were (initially) assigned to each titlesite;
- The specifications were known beforehand and there was nearly no requirements creep;
- Almost every component is reused, most sites share the same HTML templates and only standard Portletsuite portlets were used;
- The documentation for each titlesite is also reused, therefore the data obtained from these documents were mostly consistent and reliable.

Because of this nature, it can be very useful to compare them because they have many external factors in common. Also, the data for the titlesites were relatively easy to obtain. The issue data for instance, were all in Jira, unlike some other projects which have issue data in multiple issue tracking systems.

The Portletsuite projects and ECMsuite project are more regular projects. These projects were chosen because they were interesting to analyze, i.e. major requirements creep, effort overrun, or they were executed really well, etc. The biggest difference between a "normal" Portletsuite project and a titlesite is that a normal Portletsuite project can contain custom made portlets additional to the Portletsuite portlets.

6.4 Hypotheses

Before and during the collection of the data, some hypotheses existed and arose. Discussion with the management resulted in the following hypotheses:

1. There is a relationship between poor specifications (page and portlet coverage, VD coverage) and the number of issues: when the specs are poor, the developers might have to improvise which may cause undesired functionality.
2. There is a relationship between poor specifications (page and portlet coverage, VD coverage) and requirements creep (however, we know this is not true for titlesite projects, titlesite projects often have only one to four pages of VD intentionally, and requirements creep was kept to a minimum nonetheless): when specs are poor (and many specs are left blank), there is more room for changes in the scope.
3. There is a relationship between poor specifications (page and portlet coverage, VD coverage) and effort/time overrun: when the specs are poor, this could result in more issues which in turn causes more rework. Also poor specs might lead to requirements creep and in turn causes effort and time overrun.
4. The ratio WBS estimation/price estimation is relatively large when a project manager is not involved in the commercial estimation: a commercial estimation is usually very tight, because a sales manager wants to win the project by offering a low price. The project manager however prefers to have more time.
5. When a project manager is not involved in the commercial estimation, it is likely that there will be effort/time overrun.
6. The degree of customization influences the actual effort and/or time: custom functionality brings uncertainty and therefore could influence the amount of effort and time overrun.
7. The degree of customization influences the amount of rework: custom functionality brings uncertainty and therefore might result in more unexpected problems or bugs, reusing might results in fewer issues.
8. Bad testing results in a relatively large amount of issues after deployment on prod: when testing is omitted or done very poor, less bugs are found during testing. This also implies that good testing results in a relatively large amount of issues during construction.
9. Bad testing results in a relatively large amount of bugs found by customer: when bugs are not found by testers they are more likely to be found by the customer. This also means that good testing results in a relatively large amount of bugs found by the offshore and Dutch teams.

10. There is a relationship between effort/time overrun and the number of issues: effort/time overrun indicates that there were too little time and resources. This results in cannibalizing on the testing effort but also results in haste during construction.
11. There is a relationship between the size of a project and the number of issues: a big project is expected to generate more issues. If there is indeed a relationship, this information can be used to roughly predict the amount of issues for a project with a given size.
12. There is a relation between the number of issues and the time span of the project: like the previous metric, but instead of size, the duration is tested for a relation with the number of issues;
13. There is a relation between effort overrun and time overrun: what relation exists between time and effort overrun? This relation (if it exists) can be used to estimate time overrun when effort overrun is expected to occur;
14. There is a relation between the size of a project and the time span of a project: if there is a relation, this information can be used for future effort and time estimation.
15. There is a relation between the number of issues after going live and effort/time overrun: when there is effort or time overrun, this means that the developers had too little time and/or resources. This can cause them not to test, which should result in a relatively low amount of bugs before going live and therefore a large amount of bugs after going live.
16. There is a relation between requirements creep and effort overrun: it is expected that when RFCs are implemented this results in effort overrun, as RFCs are unplanned work.

However, in order to test these hypotheses scientifically, the data would have to be independent, which they are not. For instance, if two projects are done by the same development team, and one project has effort overrun it might affect the other project (for instance, cause the other project not to meet the deadlines). This means that, even though a project took longer than planned, this could be caused by another project that runs in parallel in the same development team. An example would be the titlesites "De Boomkwekerij" and "Tuin en Landschap". These titlesites are being implemented by the same team, and both have major time overrun and probably also effort overrun. Therefore, correlation diagrams can be made, but it would be wrong to make hard conclusions out of these diagrams. **All conclusions made in the following sections should be read having this in mind.** Also, when data is missing, they will not show up in the plot. Plots are sometimes made for titlesites only, because the titlesites are roughly of the same size and therefore comparisons for some data or metric attributes are more useful when other sites are left out. It is also worth mentioning that with the live date the date on which the portal is deployed on the production server and the guarantee period starts is meant. The construction phase has ended and only bug fixes and RFCs are implemented from this date on. This date is not necessarily the same date as the date that the portal is actually accessible to everybody, that is up to the customer.

6.4.1 Hypothesis 1

There is a relationship between poor specifications (page and portlet coverage, VD coverage) and the number of issues: when the specs are poor, the developers might have to improvise which may cause undesired functionality.

In Figure 10 the amount of rework (issues until live date) per construction day is plotted against the functional specification page coverage ((pages described in functional specifications)/(pages to be implemented)). At first sight, it seems as if the opposite of what was expected is going on: the higher the page coverage, the more issues per construction day. Upon further investigation, the outliers are the non-titlesites: all non-titlesites have more than five or more issues per construction day. Figure 11 shows this same figure for the titlesites (that are live as of the 8th of June, 2006) only. The highest

amount of issues per construction day (until the live date of each titlesite) is 7.44, most of the others lie below 4. There doesn't seem to be any clear correlation here. As stated earlier, it not recommended to draw hard conclusions. Furthermore, the page coverage might not be a good measure for the quality of the page specifications. Some portals have many pages that may be very similar, which means that low page coverage does not necessarily mean that the development team will have trouble with the specs. Also, some pages can be more complex than others, and simple portals, that are less likely to generate many issues (for instance a titlesite), may have low spec page coverage because it would be overkill to cover all pages for such a simple project.

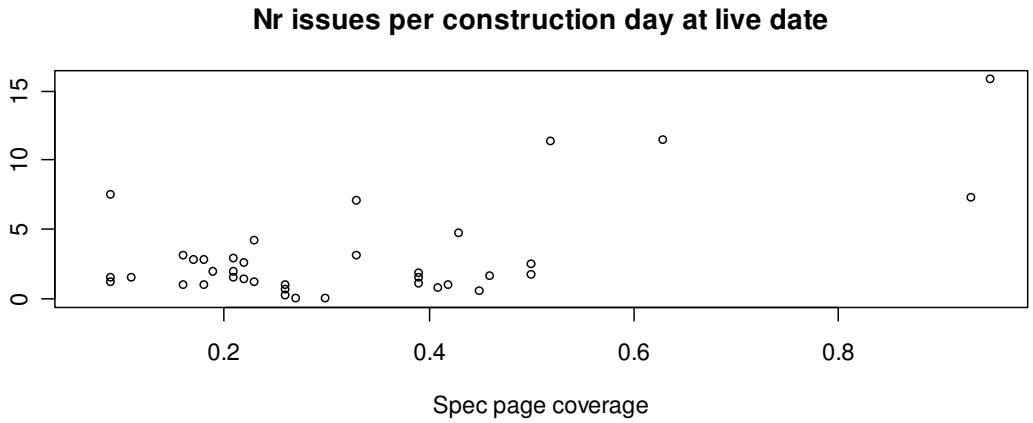


Figure 10: Number of issues (at live date) per construction day against page coverage in functional specification: all sites

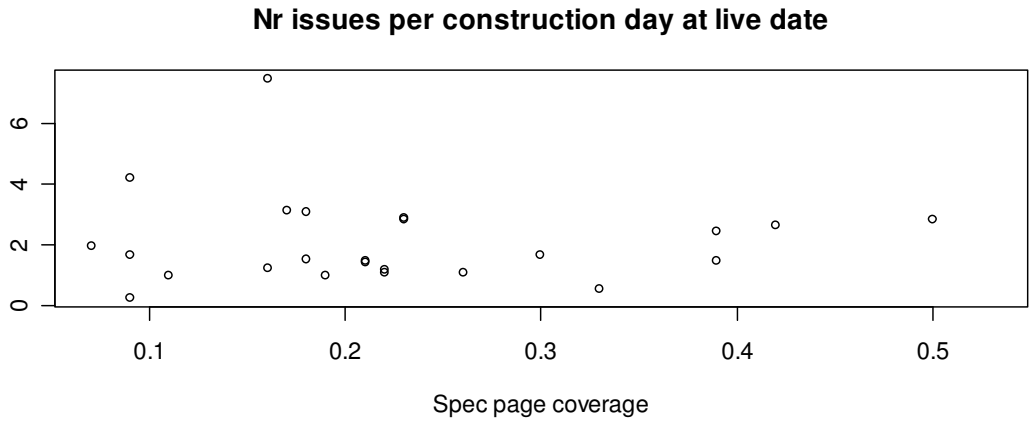


Figure 11: Number of issues (at live date) per construction day against page coverage in functional specification: titlesites

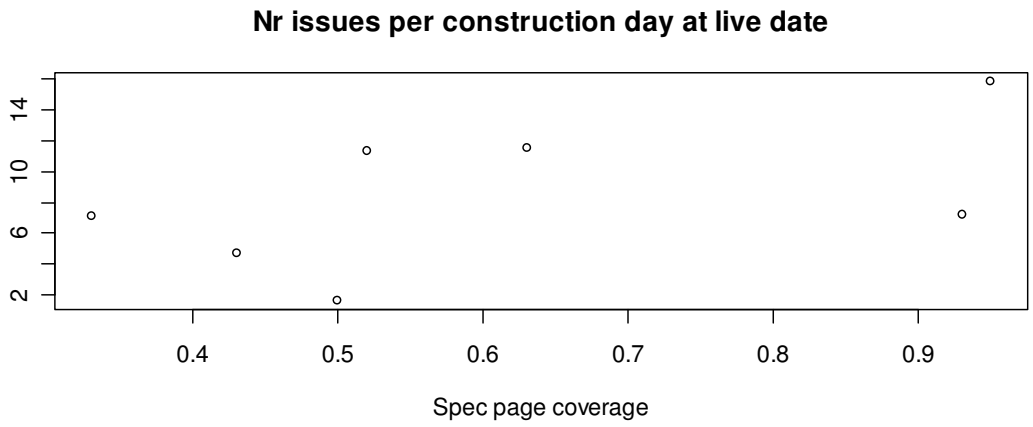


Figure 12: Number of issues (at live date) per construction day against page coverage in functional specification: all non-titlesites

Next, a similar plot is shown (for issues at live date), but instead of page coverage another metric for spec quality is given: portlet coverage. When looking at figure 13, again no hard conclusions can be really drawn, as there are only three observations that don't have full portlet coverage. It seems that most titlesites, that all have a portlet specification coverage of 1, lie below 5. Because the titlesites contain only reused portlets, and these portlets all have standard documentation, portlet coverage is 1 (or 100%) for all titlesites. It is therefore more interesting to make a boxplot of this data. The boxplot (Figure 14) shows a summary of the titlesite number of issues per construction day. The box contains 50% of all data. This could mean that in the future, when doing a project comparable with the titlesites (only reused portlets, development team and project managers with comparable skills, etc) it can be expected that with a high probability the number of issues per construction day (until live date) will likely be around 2, and will probably not exceed 4. The lonely observation is an outlier and should be considered an exception. For the interested, this was Titlesite Bizz.nl.

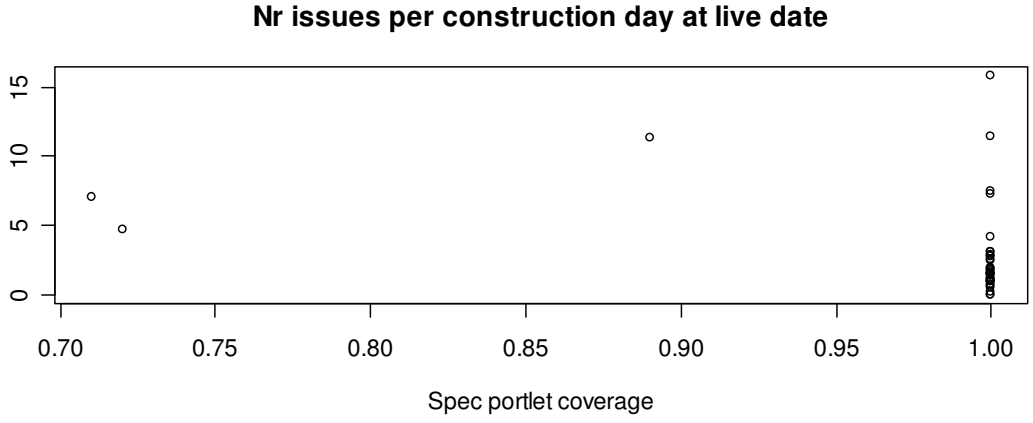


Figure 13: Total number of issues at live date against portlet coverage in functional specification: all sites

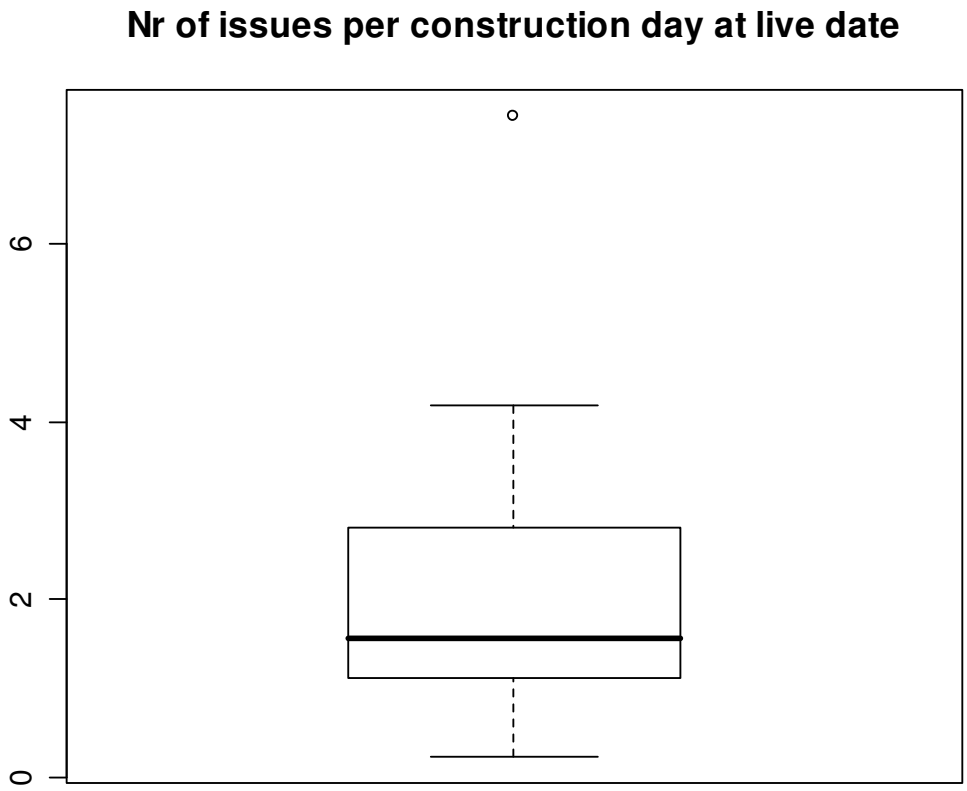


Figure 14: Boxplot of number of issues per construction day at live date: titlesites

Because the titlesites were all supposed to take nine weeks to complete and were assigned roughly the same amount of resources, it might be interesting to see the total number of issues at the live date in a boxplot for all titlesites. Figure 15 shows this. It can be seen that there is a large spread. It is hard to say however whether or not the portlet coverage is a good predictor for the total number of issues when doing another project comparable to a titlesite, there can be many other causes that would generate many issues. An interesting note however, is that the two outliers were titlesites that were done by the same offshore team and offshore project manager. These did not show up on the normalized boxplot (number of issues per construction day) because even though they have a large number of total issues, they also had major time overrun, causing the number of issues per construction day to be normal. It is therefore recommended, when the management decides to make another analysis comparable to this one, to include both types of boxplots.

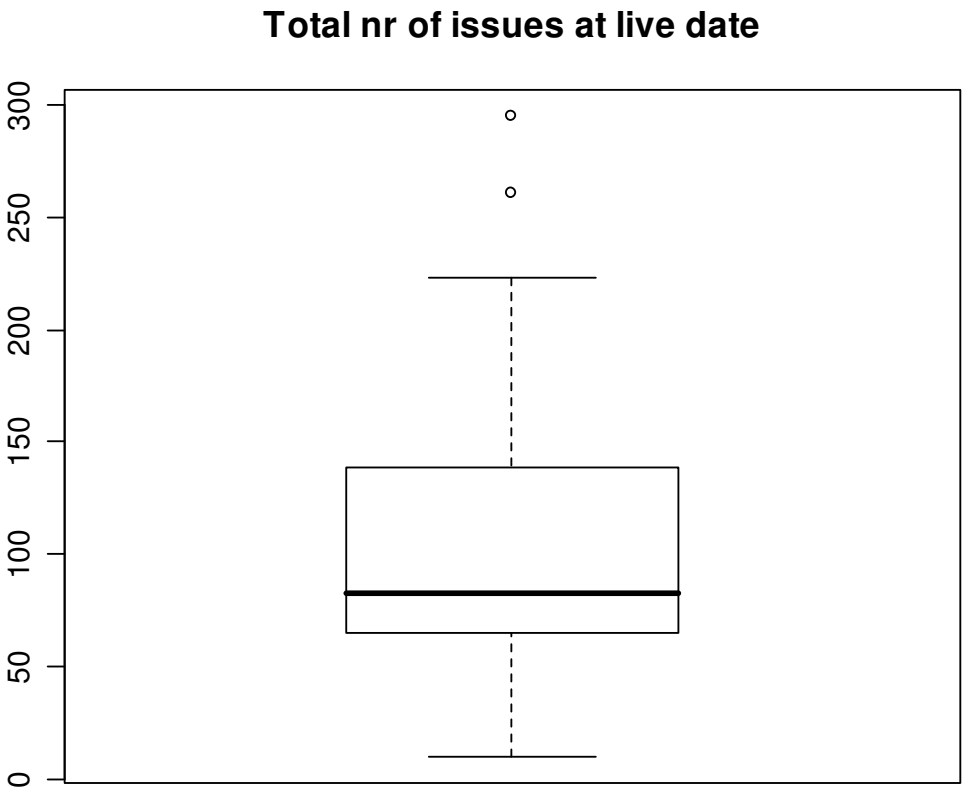


Figure 15: Boxplot of number of issues at live date: titlesites

Figure 16 shows the visual design page coverage against the number of issues per construction day (as of live date). Again, it seems like the opposite of what was expected is going on. This same chart made for the titlesites only is shown as Figure 17. Before the titlesite project started it was known that the visual design would be very meagre for each titlesite. This was intentionally done so, because many pages were reused, and only a few designs were needed for a sense of style. The concentration of observations around the lower-left corner suggests that in the titlesite project it was not a big problem to have a low page coverage in visual design. There are two titlesite portals that had a relatively high degree of page coverage: Bizz.nl (0.80) and Worldpoultry.com (0.61). It seems however, that this had no benefit in terms of issues per construction day compared to the other titlesites. A discussion with Andriy Trofimov (SS CPO) lead to a possible cause of this. Simple projects and/or portals intentionally have less effort put in design and specification, and more complex projects need more design and specifications. Therefore, the projects with high specification coverage are generally more complex portals. Of course, there are many other factors involved, but this could be the one of the major causes of the relatively large amount of issues for these projects.

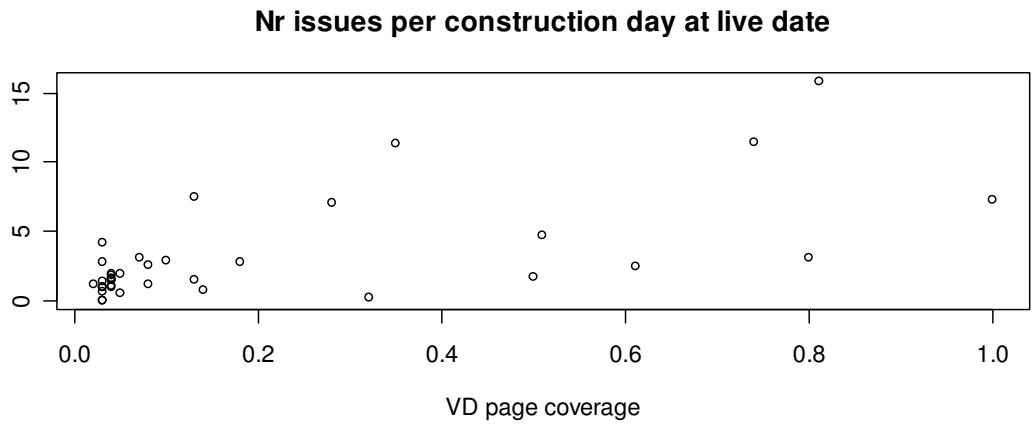


Figure 16: Number of issues (at live date) per construction day against page coverage in visual design: all sites

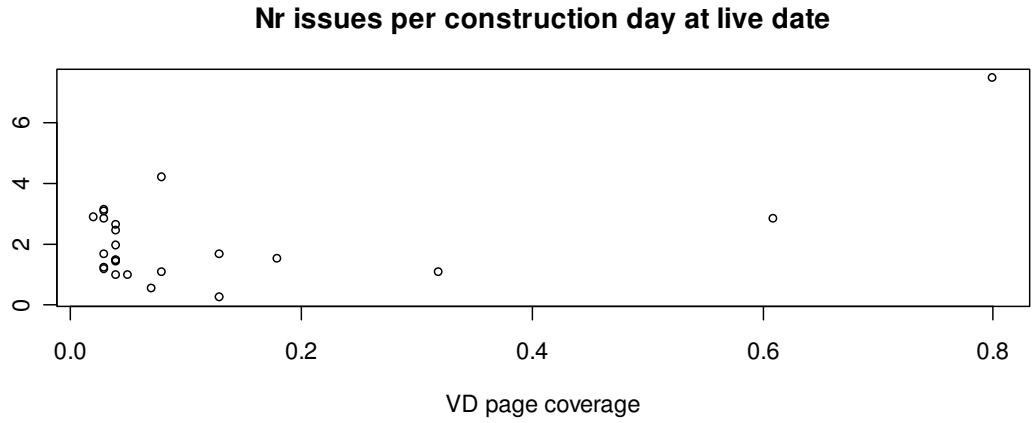


Figure 17: Number of issues per construction at live date day against page coverage in visual design: titlesites

In Figure 18 however, where total issues at live date are plotted against the visual design page coverage, there might be some relation when the outliers Bizz and Worldpoultry are disregarded. An explanation might be that, when the visual design page coverage is low, this may be caused by the number of pages being large. In other words, the visual design coverage is calculated by: (total number of pages designed)/(total number of pages to be implemented), and a low coverage could be caused by a large denominator (total number of pages to be implemented), since the numerator (number of pages designed), is generally very small. Therefore, a portal with many pages generally has a low page coverage, but is larger in size (in term of pages) and therefore might be likelier to generate more issues. The pages that have a relatively large number of issues (>100) at live date are:

- Bizz.nl: Issues: 201 # pages: 25
- Worldpoultry.com: Issues: 126 # pages: 18
- Deboomkwekerij.nl: Issues: 295 # pages: 30
- Weekbladgroentenenfruit.nl: Issues: 116 # pages: 28
- Vakbladvoordebloemisterij.nl: Issues: 261 # pages: 31
- MKB-E/gribb.nl: Issues: 151 # pages: 53
- Distrifood.nl: Issues: 112 # pages: 24
- AGF.nl: Issues: 117 # pages: 28
- Tuinenlandschap.nl: Issues: 223 # pages: 34

It can be seen that from the list above only MKB-E/gribb is an outlier. The denominator therefore is not exceptionally high for most of the observations that have a high number of issues at live date. The other outlier is Penoactueel.nl, with 75 pages to be implemented and only 3 pages designed. The number of issues at the live date is not very high though: 83. This means that a low page coverage does not necessarily mean that many issues will arise.

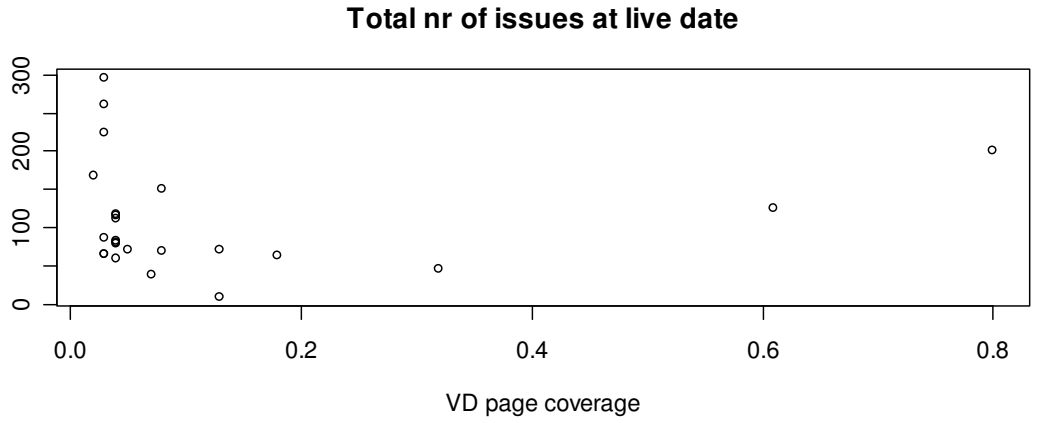


Figure 18: Number of issues (at live date) against page coverage in visual design: titlesites

Conclusions:

- There does not seem to be a relation between page coverage in the functional specifications and the number of issues per construction day. The number of covered pages might not be a good measure for specification quality due to the reusing of pages and differences in page complexity.
- There is not enough diversity in the portlet coverage to make useful charts. This is due to the titlesites which have all portlets reused, and therefore coverage is 1 for all these sites.
- The number of issues per construction day for titlesites seem to be around 2, with a maximum of around 4, except for one outlier. This might give an indication of the number of issues to expect in future, similar projects.
- There seems to be some relation between the visual design coverage and number of issues at first glance. However, most titlesites have a very low coverage intentionally because there is a lot of reusing going on. The titlesite portals that did have relatively many issues at live date (>100), generally did have a bad page coverage, but the portals that have relatively little issues at live date (<100) also generally have a bad page coverage. Furthermore, the titlesites that had relatively good page coverage don't show a low number of issues. The non-titlesites sites don't show any correlation either, but there were also too little observations to draw a conclusion.
- Due to the data not being independent and due to the likeliness that the page and portlet coverage (of functional specifications as well as visual design) are questionable as good measures for specification quality, it cannot be determined whether or not hypothesis 1 should be rejected or not.

6.4.2 Hypothesis 2

There is a relationship between poor specs (page and portlet coverage, VD coverage) and requirements creep (however, we know this is not true for titlesite projects, titlesite projects often have only one to four pages of VD intentionally, and requirements creep was kept to a minimum nonetheless): when specs are poor (and many specs are left blank), there is more room for changes in the scope.

Figures 19 and 20 show the number of requested RFCs at live date against spec page coverage. Because of the outlier at approximately 3 RFCs per day another chart was made without the outlier. There doesn't seem to be any correlation. When plotting only the titlesites (Figure 20), it is evident that there is relatively a very little amount of RFCs requested for all titlesites: it shows that a relatively high spec page coverage (>0.30) the requested number of RFCs per construction day don't exceed 0.05 RFCs per construction day, except for the outlier at 0.5 spec page coverage. When looking at Figure 21 the number of implemented RFCs per construction day (for titlesites only) also seem to have a similar relation, with portals that have a spec page coverage of >0.25 having no more than 0.05 RFCs implemented per construction day. Portlet coverage plots against RFCs requested were omitted since almost all portals have 100% portlet coverage.

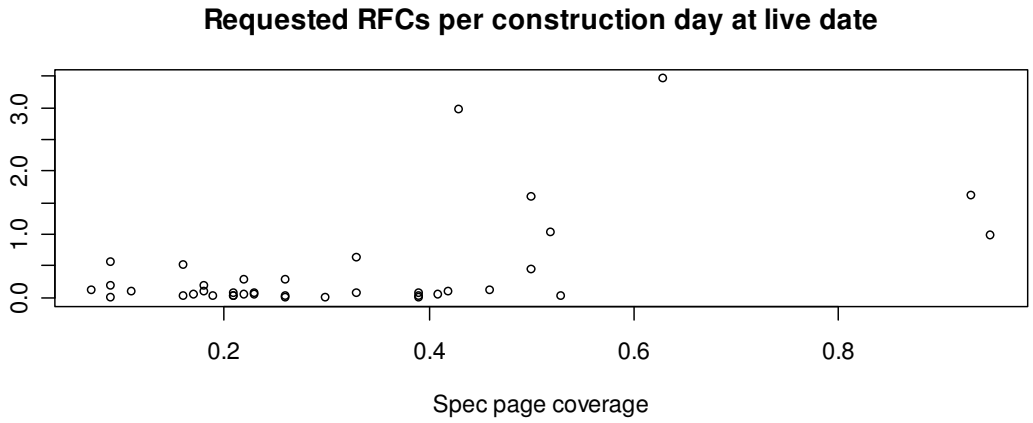


Figure 19: Number of RFCs requested against spec page coverage

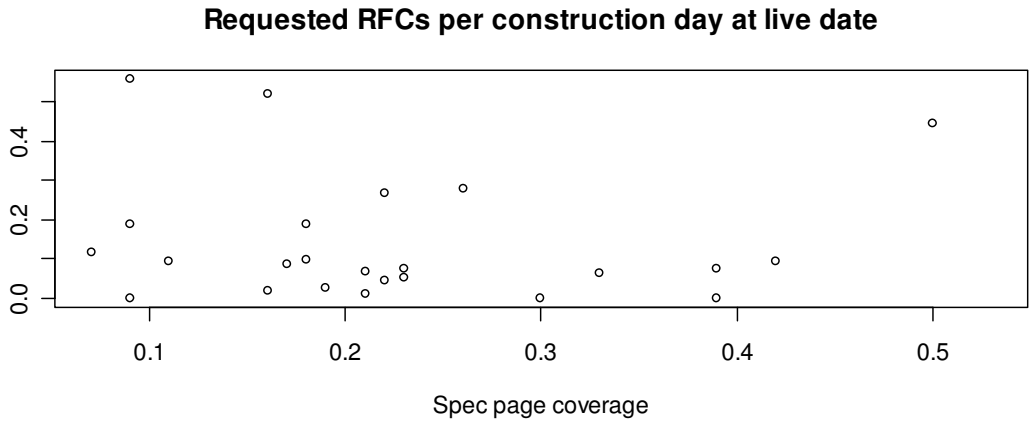


Figure 20: Number of RFCs requested against spec page coverage, titlesites only

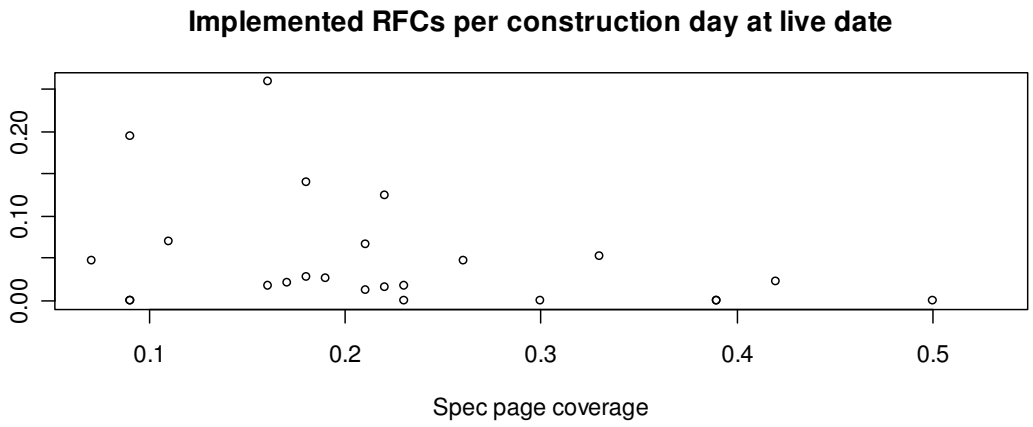


Figure 21: Number of RFCs implemented against spec page coverage, titlesites only

Figure 22 shows the relative growth of pages against the spec page coverage. There doesn't seem to be any correlation and the growth/shrinkage in the number of pages is relatively low in general. A chart was also made (but not shown here) for the absolute growth of pages, and the range was between [-3,3] and gave the same impressions as Figure 22.

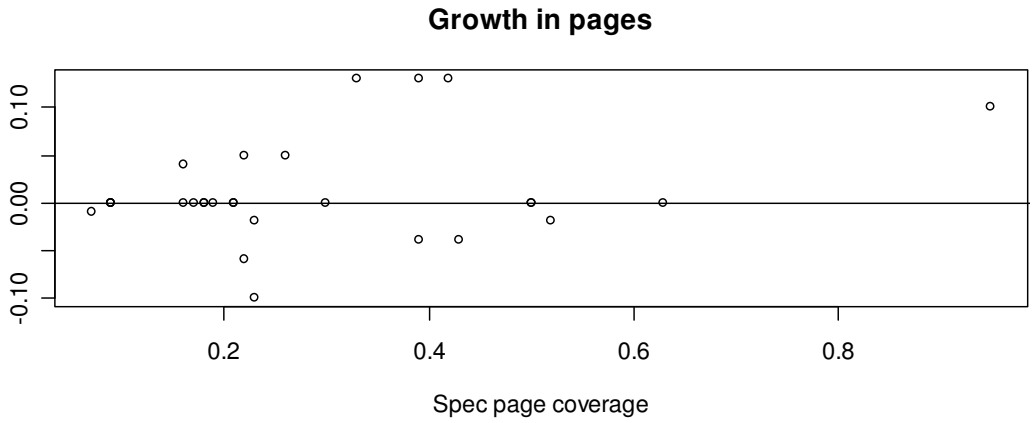


Figure 22: Relative growth of number of pages against spec page coverage

The growth in unique portlets were also close to 0 for almost all projects and no charts were made.

Conclusions:

- The hypothesis seems to be only valid for the titlesites. Figures 20 and 21 show that there might be a relation between the spec page coverage and the number of requested and implemented RFCs per construction day.
- After discussing these charts with the management, there is a strong suspicion that the growth in the number of pages and the growth in number of unique portlets are no good measures for requirements creep. There is a strong suspicion that requirements creep is within the pages and portlets, not necessarily in the growth of numbers of pages and portlets. The number of implemented RFCs seems to give a better sense of requirements creep.

6.4.3 Hypothesis 3

There is a relationship between poor specifications (page and portlet coverage, VD coverage) and effort/time overrun: when the specs are poor, this could result in more

issues which in turn causes more rework. Also poor specs might lead to requirements creep and in turn causes effort and time overrun.

Looking at the figures (23 and 24) below, It seems at first sight that there might be a relation. The observations that have major (>1.5) relative construction time overrun generally have bad spec page coverage and VD page coverage. But observations that have bad coverage don't always have time overrun. But the observations that do have overrun generally have low page coverage. The outlier is project Toeristiek for the interested.

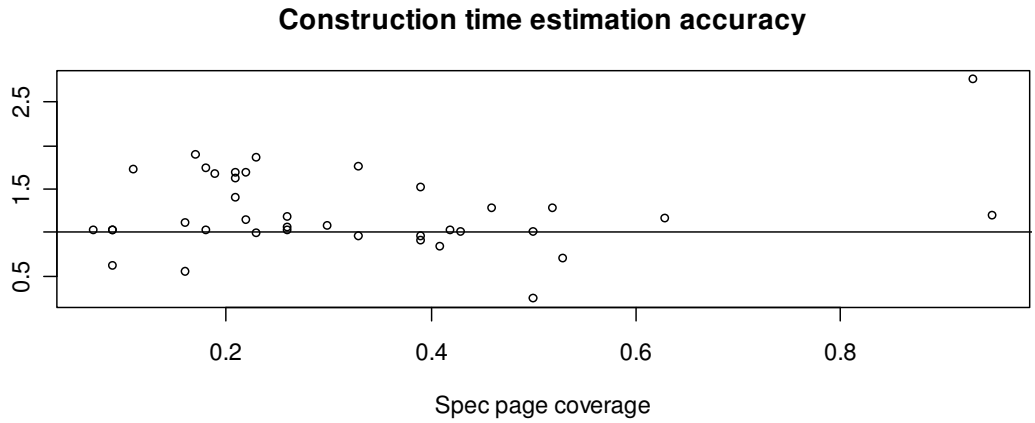


Figure 23: Spec page coverage against construction time estimation accuracy

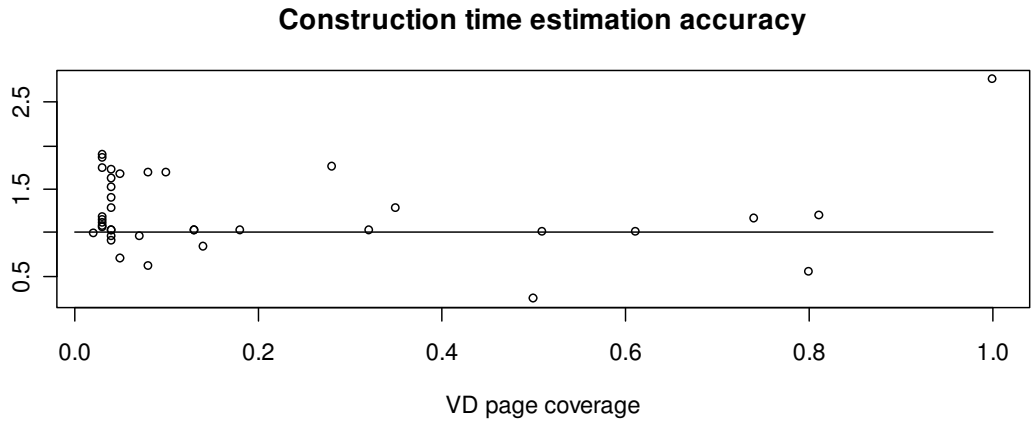


Figure 24: VD page coverage against construction time estimation accuracy

Looking at figures 25 and 26 it seems that the relation between bad spec and VD page coverage might also exist for the commercial effort estimation accuracy. Of course, it is expected that time and effort are correlated somehow. Looking at figures 25 and 26 this hypothesis may seem to hold. It is very notable that the commercial effort estimation accuracy is never lower than two and can even be twelve-fold. For the interested, this project was Agis. AT confirms that this overrun was caused by incomplete specifications.

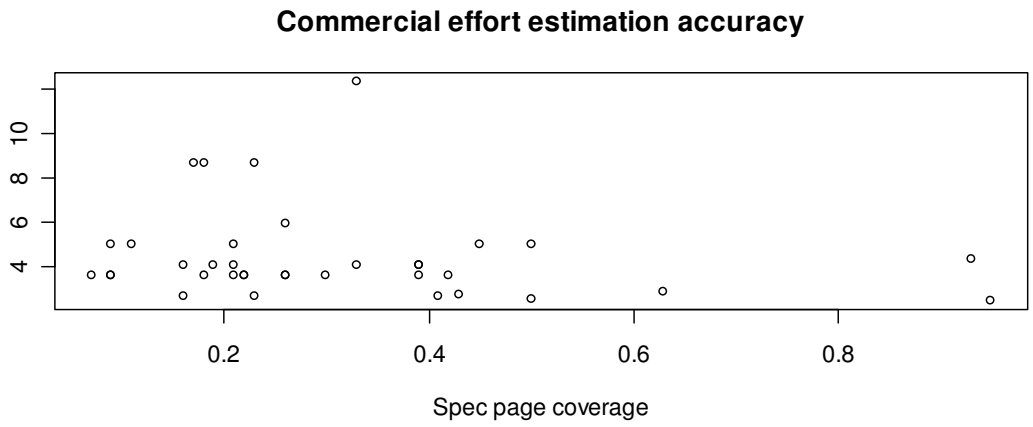


Figure 25: Spec page coverage against commercial effort estimation accuracy

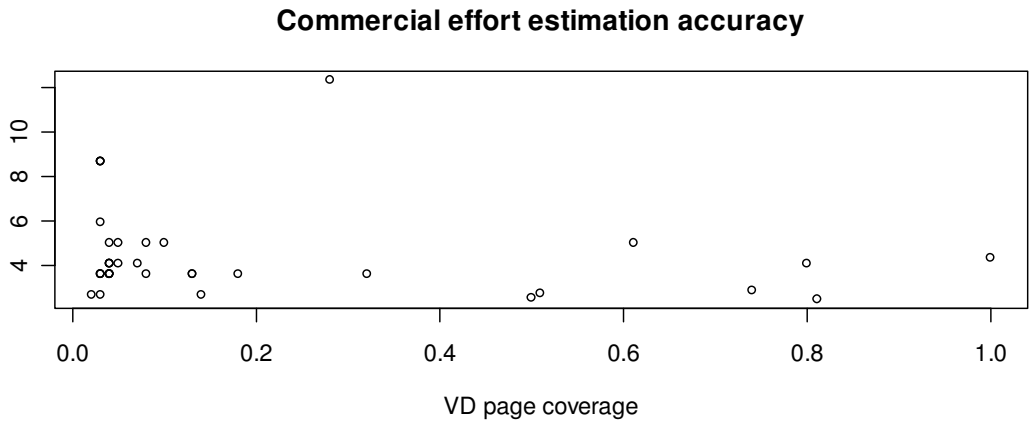


Figure 26: VD page coverage against commercial effort estimation accuracy

Figures 27 and 28 show the WBS effort estimation accuracy. The WBS effort estimations are much better than the commercial estimation, but there is still major effort overrun for many projects. Because the WBS effort estimation has some relation to the price, the plots show a comparable spread.

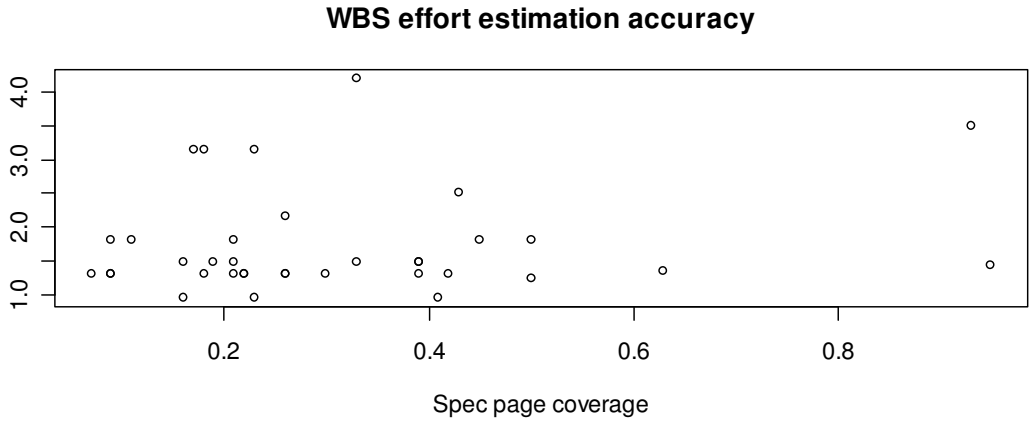


Figure 27: Spec page coverage against WBS effort estimation accuracy

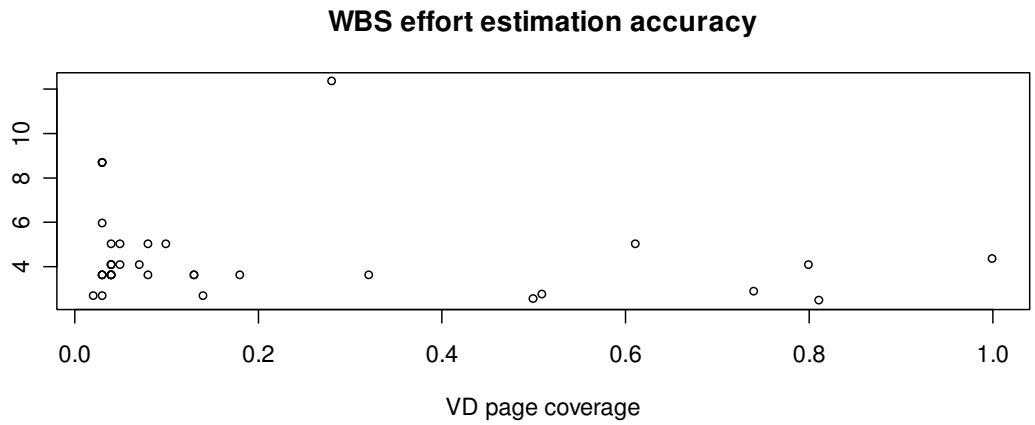


Figure 28: VD page coverage against WBS effort estimation accuracy

Conclusions:

- The charts indicate that effort and time overrun may be caused by low specifications and design coverage. Low specifications and design coverage don't necessarily mean that there will be time and/or effort overrun, but most observations with time and/or effort overrun do have low specifications and design coverage.
- The commercial effort estimations are considerably worse than the WBS estimations. The commercial effort estimation accuracy observations also show a very large spread.

6.4.4 Hypothesis 4

The ratio WBS estimation/price estimation is relatively large when a project manager is not involved in the commercial estimation: a commercial estimation is usually very tight, because a sales manager wants to win the project by offering a low price. The project manager however prefers to have more time.

For all titlesites, the price and reserved capacity were the same. One of the Dutch managers that were involved in the commercial estimation also determined the capacity for each of the titlesites. As stated earlier, the allowed effort is calculated from the price divided by 45. However, the reserved capacity was almost three times (2.77 times to be exact) the allowed effort. This means that all titlesites were either sold too cheap, or too much capacity was allocated. However, as some titlesites have time overrun (and therefore probably effort overrun too) it is likely that the reserved capacity was not too much and therefore the titlesites were sold too cheap. Of course, there are reasons for selling the titlesites for such a low price as there were competitors. This does not mean that the titlesite program was unprofitable, but a lot less profit was made.

A plot for the non-titlesites did not show anything interesting, as there are only six observations (the other non-titlesites were missing this data).

6.4.5 Hypothesis 5

When a project manager is not involved in the commercial estimation, it is likely that there will be effort/time overrun.

There is not enough data to make an interesting plot for this hypothesis. There are only two projects for which no project managers were involved and three projects where two project managers were involved. All other projects have one project manager involved in the commercial effort estimation. In charts 23 to 26 it was clear however, that the WBS

effort estimations (that are generally done by project managers) were more accurate than the commercial effort estimations (that are generally done by sales managers).

6.4.6 Hypothesis 6

The degree of customization influences the actual effort and/or time: custom functionality brings uncertainty and therefore could influence the amount of effort and time overrun.

Most of the observations in Figure 29 show that there were no custom portlets in most sites, and therefore it is hard to draw any conclusions. The same goes for effort estimation data. The remaining observations don't show any correlation. Also, it is not clear to what degree all custom portlets are custom, i.e. some portlets that were only adapted also were classified as custom. What can be seen however is that construction time overrun seems to occur even when there are no custom portlets at all. Still, there should be some classification on the difficulty of a portlet and there should be a classification of the degree of customization (and of course preferably more data points) in order to draw more reliable conclusions.

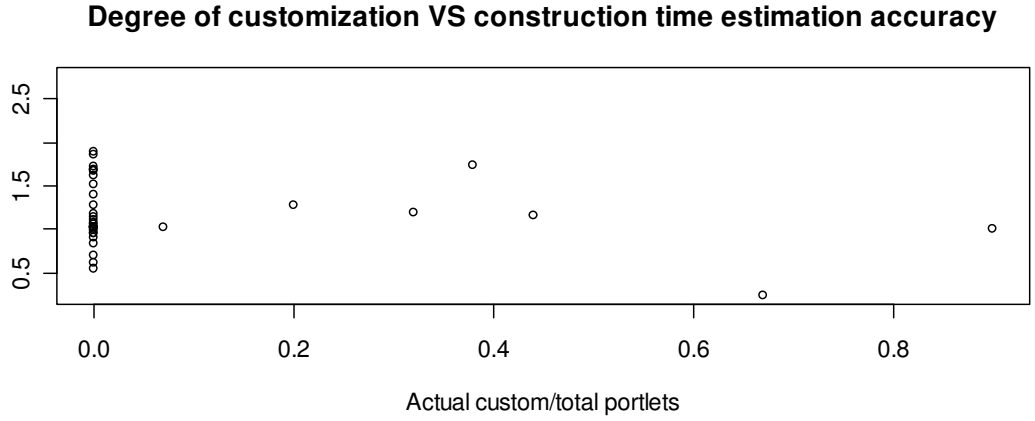


Figure 29

Conclusions:

- There should be a more clear definition of what portlets are custom, to what degree and the complexity/difficulty of the portlet.
- Besides this there seems to be no relation, many observations with 0 custom portlets still have (relatively) major time overrun.

6.4.7 Hypothesis 7

The degree of customization influences the amount of rework: custom functionality brings uncertainty and therefore might result in more unexpected problems or bugs, reusing might results in fewer issues.

Figure 30 gives the impression that most observations with 0 custom portlets have relatively a low amount of issues per construction day (<5). Most of the other points lie above this number, but again, there are not enough data points and the definition of custom is not clear enough to draw any hard conclusions.

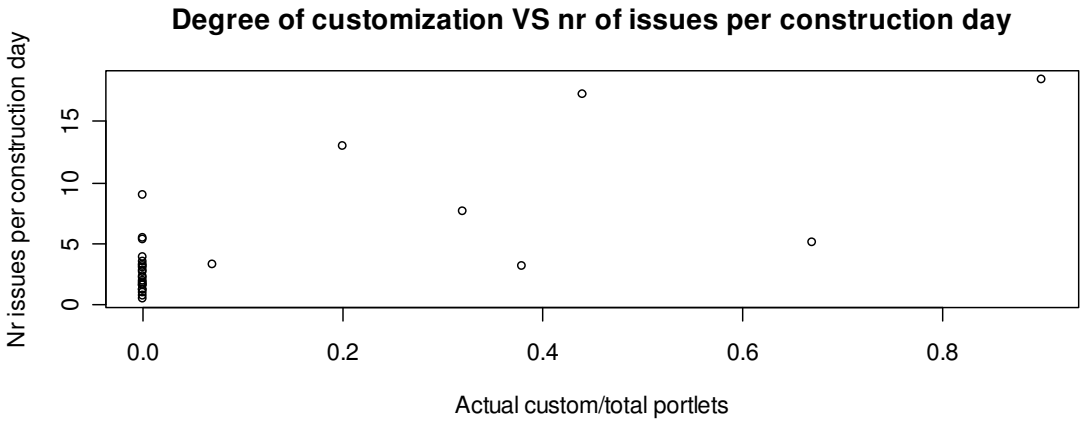


Figure 30: all sites

Conclusions:

- Most sites that have no custom functionality have a relatively low amount of issues per construction day.

6.4.8 Hypothesis 8

Bad testing results in a relatively large amount of issues after deployment on prod: when testing is omitted or done very poor, less bugs are found during testing. This also implies that good testing results in a relatively large amount of issues during construction.

In Figure 31 it can be seen that for five observations, with three projects where three different tests were done and two project where four different tests were done, testing probably helped in finding issues before going live. The tests that were done for all other projects were functionality tests and customer acceptation tests. The projects that had three types of tests had an additional performance test. The project that had four tests had an additional performance and security test. The management knew from the top of their heads that at least two projects that had performance tests done (Boerderij.nl and Fiscaal Totaal) the performance tests were done very extensive. The ratio of issues found before deployment on the production server for these projects were 0.87 and 0.92 respectively.

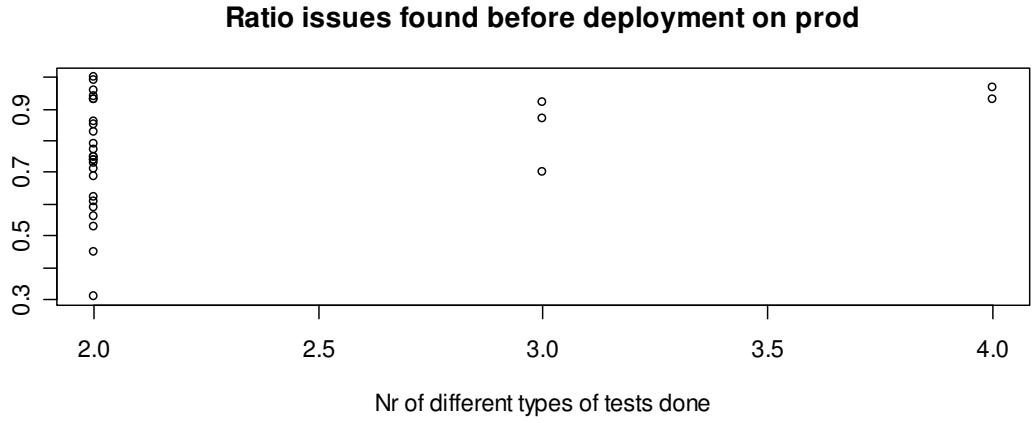


Figure 31

Conclusions:

- It is likely that a performance test helps a lot in finding issues during construction.

- It is therefore recommended that in future projects the project manager should discuss performance requirements with the client and performance testing should be standard in future projects.
- It is recommended to keep track of this metric and see if the hypothesis holds when more projects are available that have more tests done.

6.4.9 Hypothesis 9

Bad testing results in a relatively large amount of bugs found by customer: when bugs are not found by testers they are more likely to be found by the customer. This also means that good testing results in a relatively large amount of bugs found by the offshore and Dutch teams.

Figure 32 shows that for the projects that had more than two tests done, the ratio of issues posted by the customer until the live date is relatively low. Of course, again there is a relatively low amount of observations where more than two tests were done, but it would be worth it to keep tracking this in the future and see if the hypothesis holds.

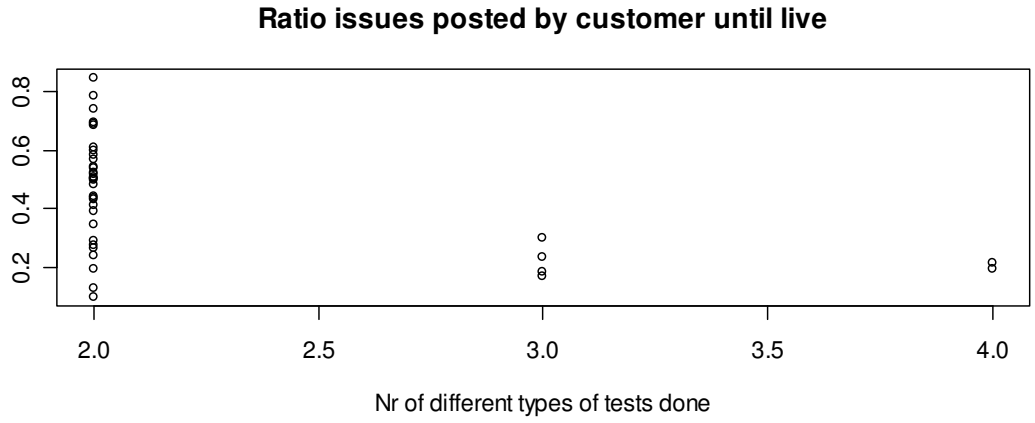


Figure 32

Conclusions:

- Doing more types of tests (at least performance tests) seems to result in the customer posting relatively less issues.

6.4.10 Hypothesis 10

There is a relationship between effort/time overrun and the number of issues: effort/time overrun indicates that there were too little time and resources. This results in cannibalizing on the testing effort but also results in haste during construction.

Figures 33 and 34 don't show a clear relation that could confirm the hypothesis. The observations with major construction time overrun don't show a lower value than the projects that had little or no construction time overrun.

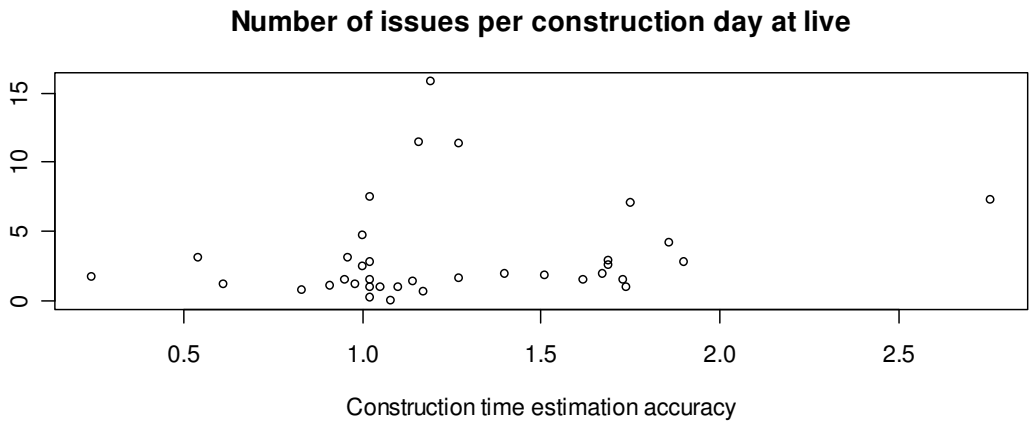


Figure 33: all sites

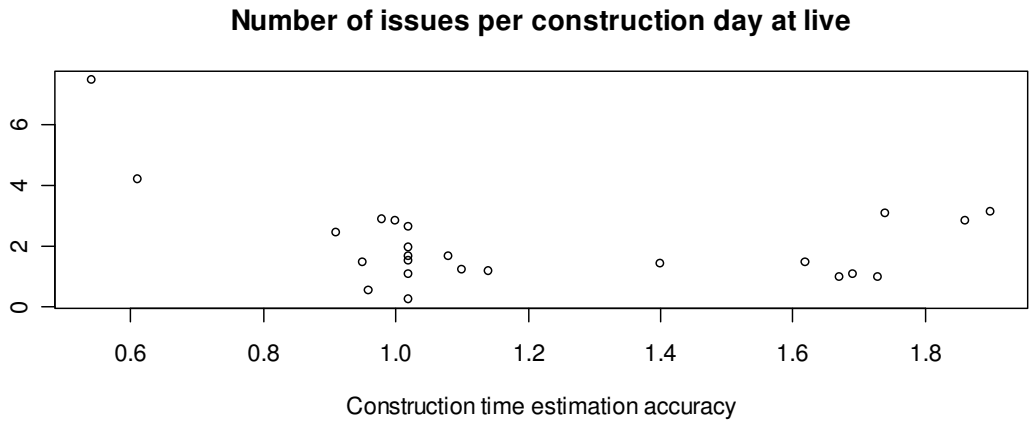


Figure 34: Titlesites only

Figures 35 and also doesn't show anything that could confirm or reject the hypothesis. Figure 36 shows a relation when the three observations with WBS effort estimation accuracy around 1.25~1.50 with more than 5 issues per construction day are neglected.

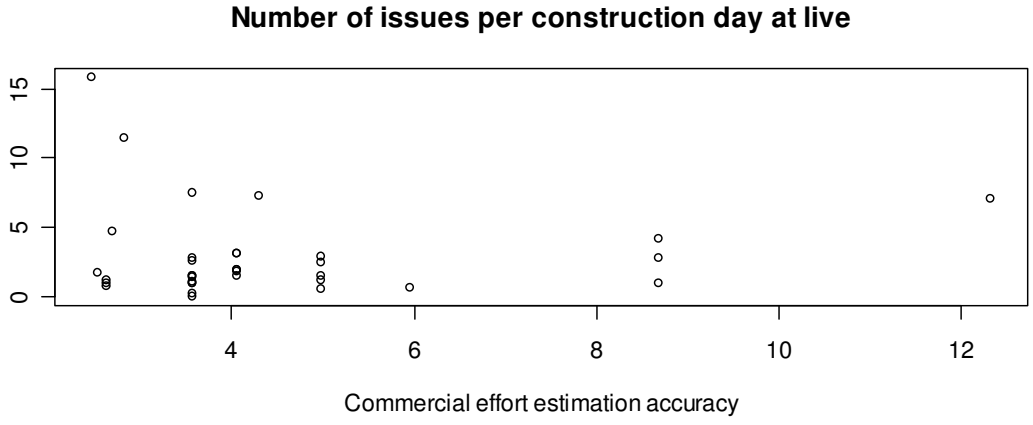


Figure 35

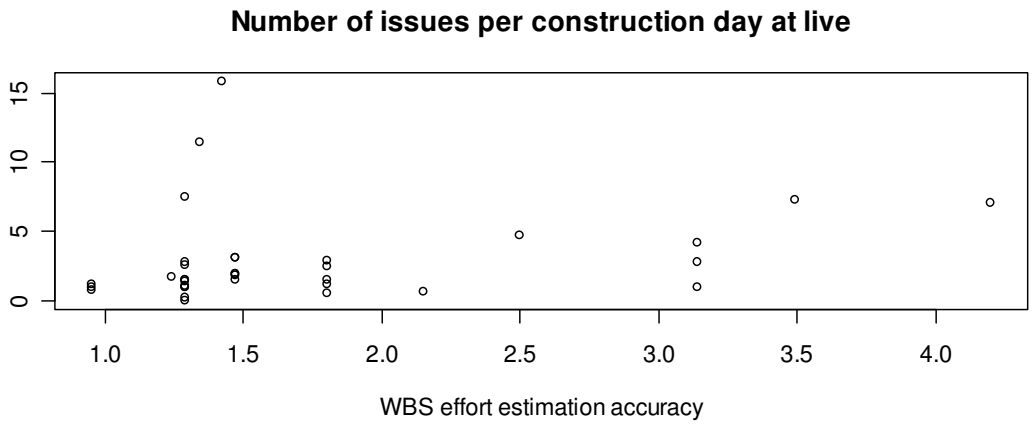


Figure 36

Conclusions:

- Only chart 36 shows that there might be a relation between WBS effort overrun and the number of issues per day. The other charts don't seem to show any relations between the effort/time overrun and number of issues per construction day.

6.4.11 Hypothesis 11

There is a relationship between the size of a project and the number of issues: a big project is expected to generate more issues. If there is indeed a relationship, this information can be used to roughly predict the amount of issues for a project with a given size.

There seems to be some correlation between the actual effort in hours and the total number of issues at live date when looking at figures 37 and 38.

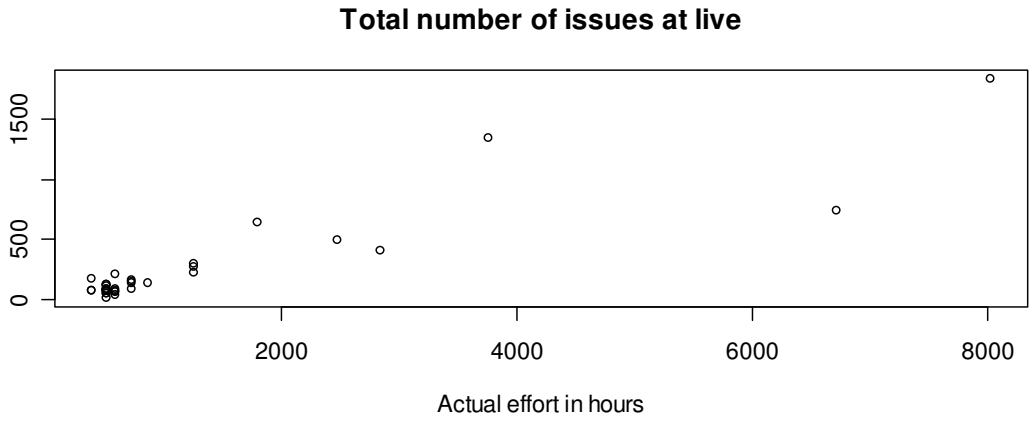


Figure 37: all sites

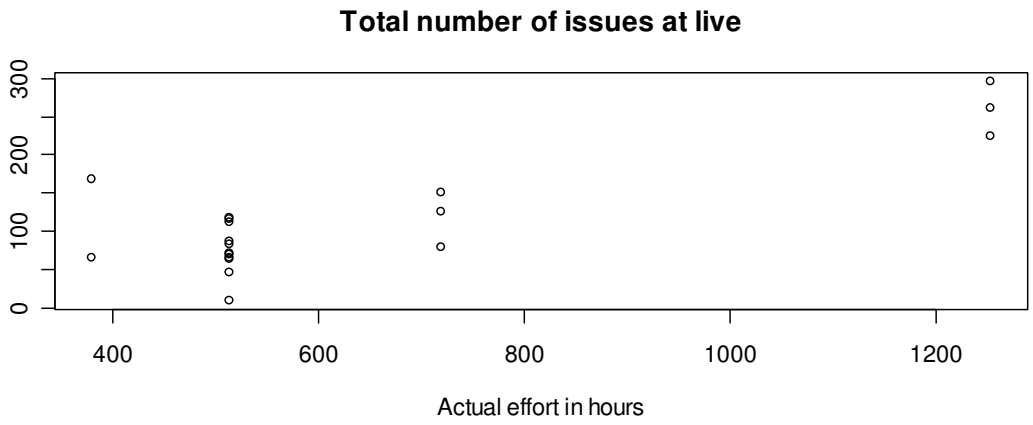


Figure 38: titlesites only

This relation also seems to hold for the total number of issues after one month of guarantee when looking at figures 39 and 40. Of course, the number of issues at live date has a very big influence on the number of issues one month later, so this was to be expected.

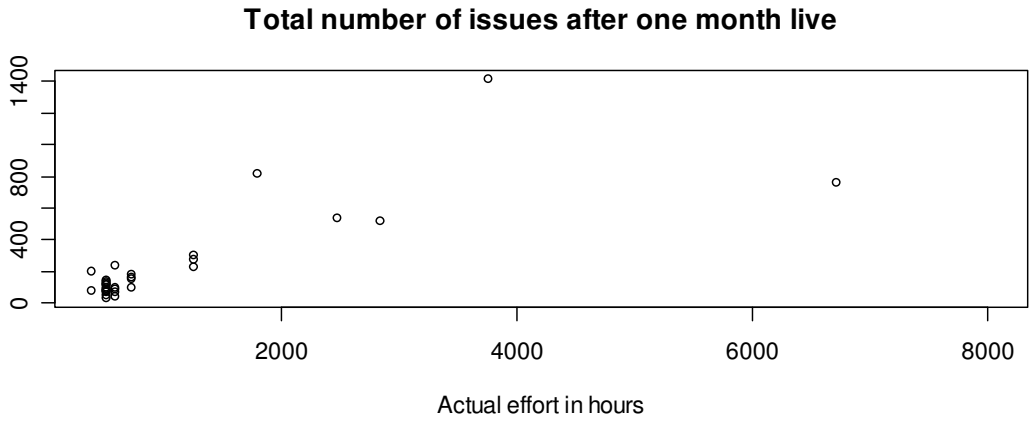


Figure 39: all sites

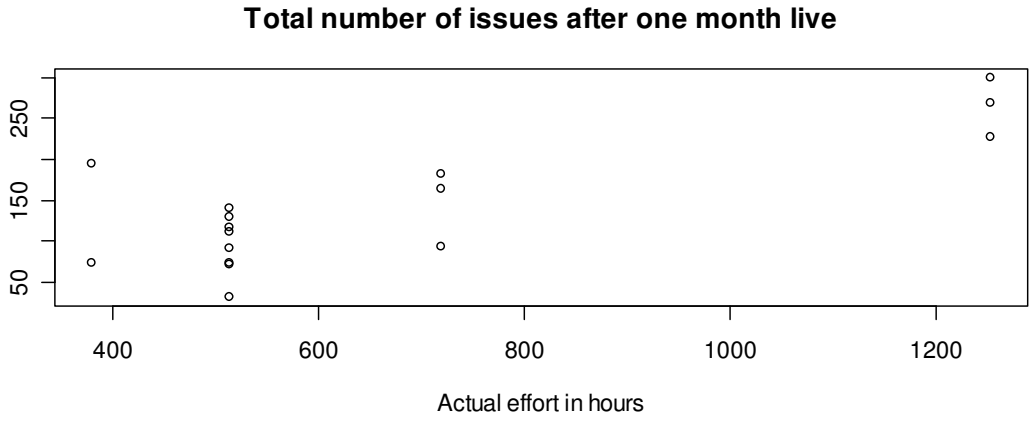


Figure 40: Titlesites only

Conclusions:

- There seems to be a relation between the effort and the number of issues at live date and after one month of guarantee.

6.4.12 Hypothesis 12

There is a relation between the number of issues and the time span of the project: like the previous metric, but instead of size, the duration is tested for a relation with the number of issues.

Figures 41 and 42 don't seem to show a clear relation between the construction duration and the total number of issues at live date. The lonely observation is project Toeristiek for the interested.

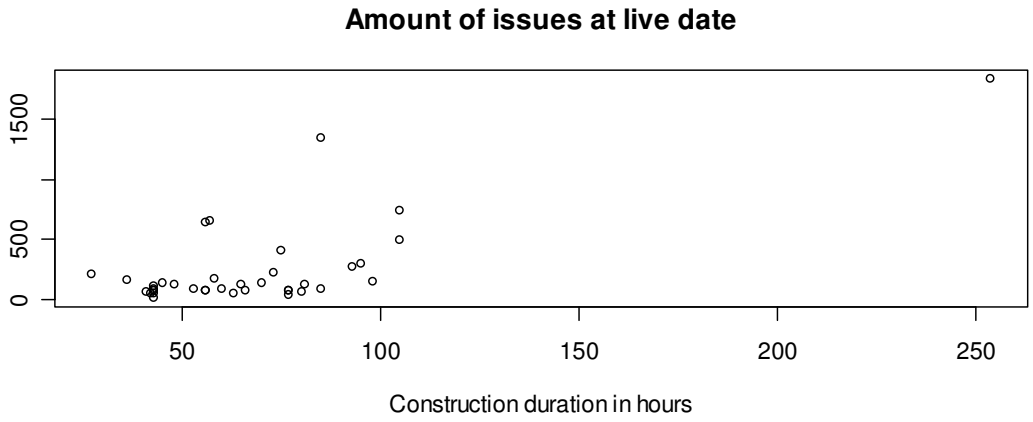


Figure 41

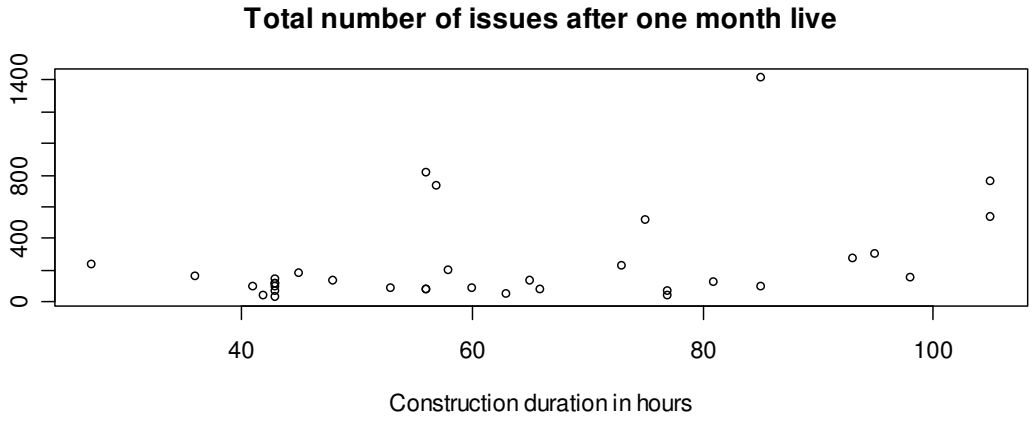


Figure 42: all sites (Toeristiek omitted because the project has just gone Live at this time)

Conclusions:

- Since effort and time are expected to have some relation, it is strange that the number of issues do have a relatively clear relation with the effort, but not with the duration of a project. The observations do not lie on a clear line.

6.4.13 Hypothesis 13

There is a relation between effort overrun and time overrun: what relation exists between time and effort overrun? This relation (if it exists) can be used to estimate time overrun when effort overrun is expected to occur.

Figure 43 shows that there seems to be no clear correlation between the construction time estimation accuracy and WBS based effort estimation accuracy. Figure 44 shows this for the total project time estimation accuracy. Again, no conclusions can be drawn. It seems that even when the WBS effort estimation is very bad (>1.5) the construction time and total time accuracy do not rise considerably compared to the observations with a WBS effort estimation accuracy of <1.5.

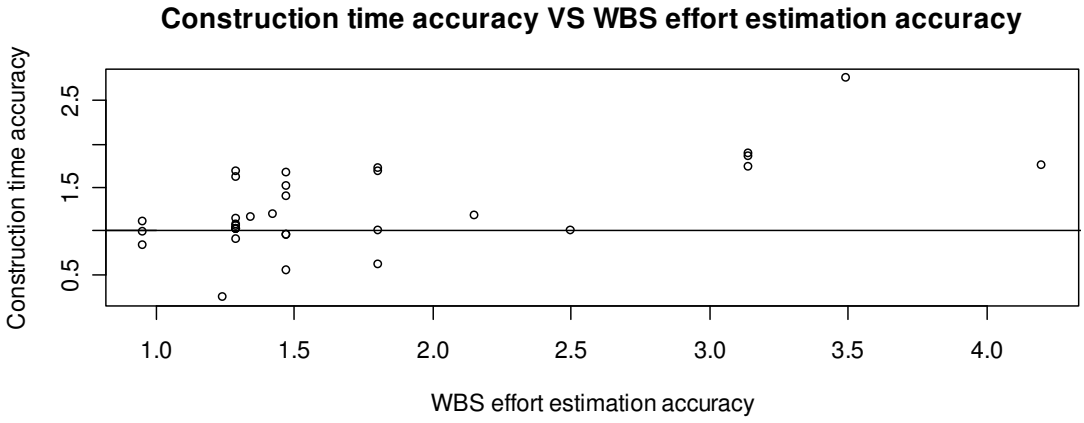


Figure 43

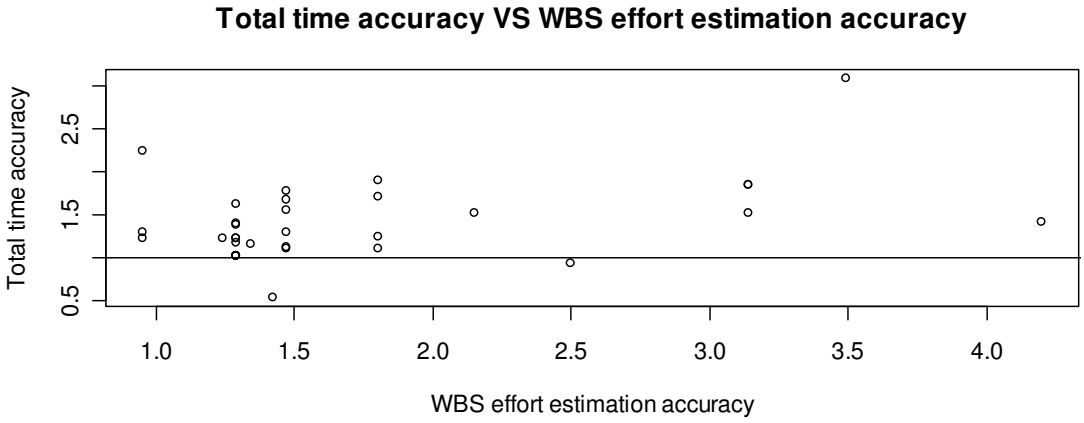


Figure 44

Conclusions:

- More effort overrun does not automatically mean that there will be more time overrun. This is probably due to people overworking when effort overrun occurs, or people might have been added to a project.

6.4.14 Hypothesis 14

There is a relation between the size of a project and the time span of a project: if there is a relation, this information can be used for future effort and time estimation.

Figures 45 and 47 show that there seems to be some relation between effort and construction duration. However, the many observations with effort <1000 show a relatively large spread, from approximately 40 days to more than 80 days for construction duration and approximately 60 to 110 for total duration.

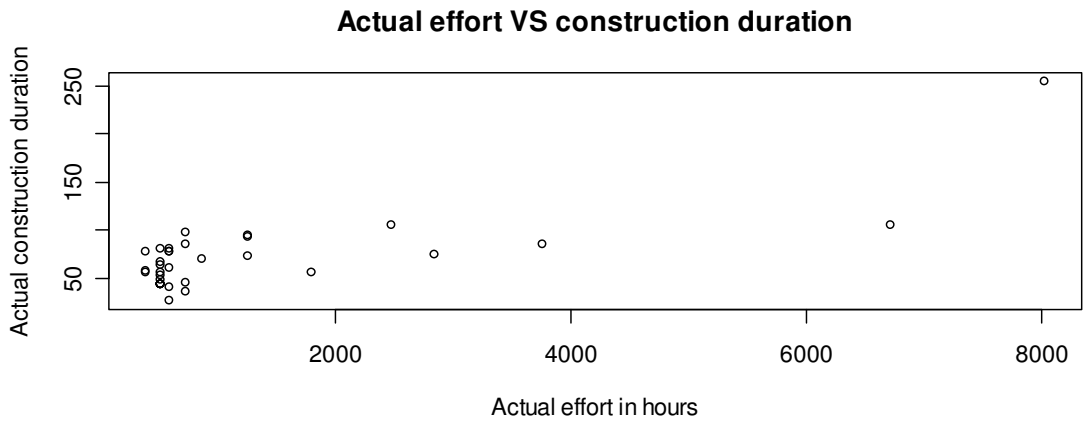


Figure 45

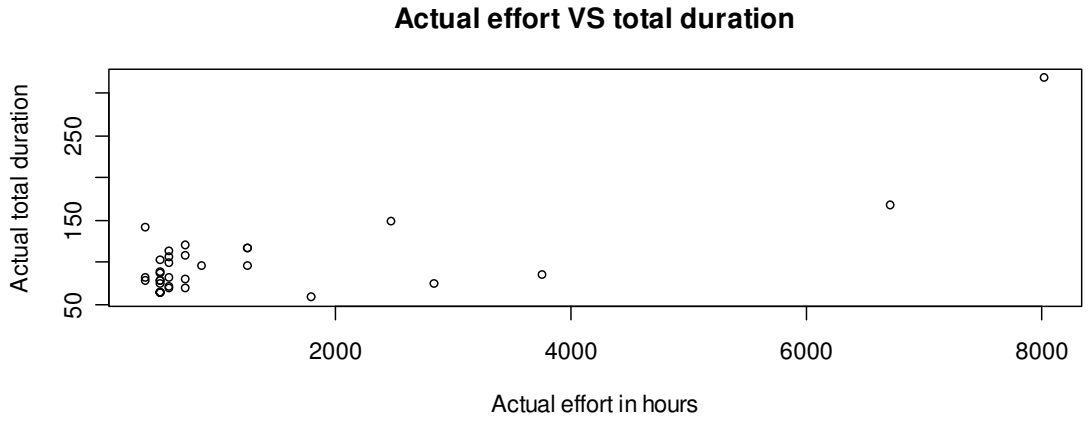


Figure 46

It is already clear that the number of pages alone, the number of unique portlets alone, or the number of custom portlets alone are no good measures for the project size. Therefore effort was taken as size for this hypothesis.

Conclusions:

- There seems to be some relation between the total effort and the construction and total duration. For future projects, when a (realistic) effort estimation has been made, these charts could help in estimating the duration of the project.

6.4.15 Hypothesis 15

There is a relation between the number of issues after going live and effort/time overrun: when there is effort or time overrun, this means that the developers had too little time and/or resources. This can cause them not to test, which should result in a relatively low amount of bugs before going live and therefore a large amount of bugs after going live.

Figure 47 does not show a reason to believe that this hypothesis holds. There are a few points that have a WBS effort estimation of higher than 3, and indeed they show a relatively low amount of issues posted in the first guarantee month. But other than these observations, there is no reason for the hypothesis to hold.

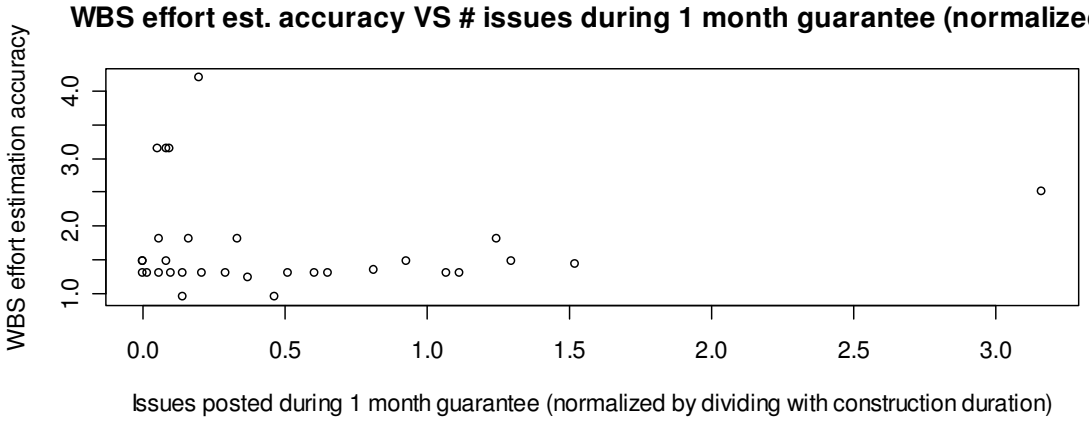


Figure 47

Figure 48 however, seems to hint at a relation between the total time estimation accuracy and the number of issues posted in the first guarantee month per day. It is strange however that the more accurate the construction time estimation was, the more this results in issues in the first guarantee month. Maybe the projects that were on time made it on time because the issues (rework) were found after they went live, and therefore rework was done after the construction phase and this relieved the rework during the construction phase.

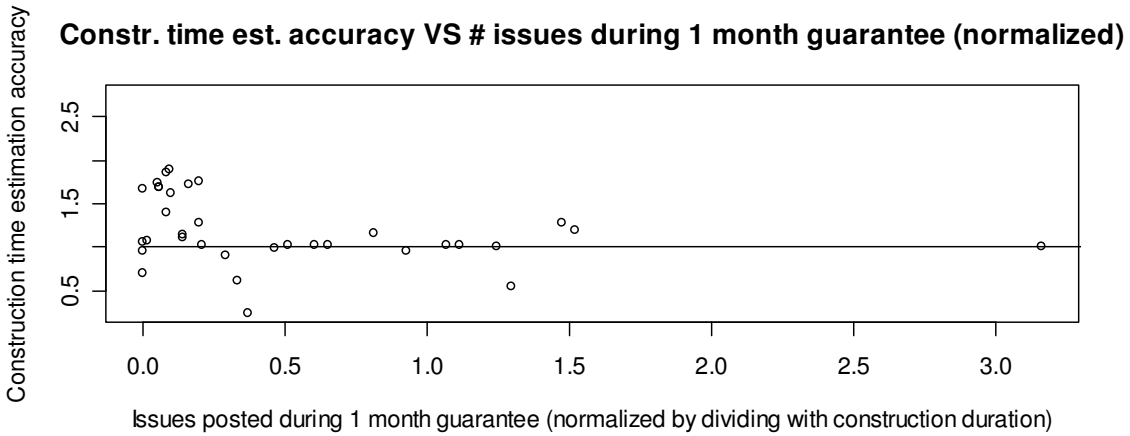


Figure 48

Conclusion:

- The observations don't show a clear relation between the number of issues after going live and the effort estimation accuracy.
- There might be a relation between the number of issues after going live and the construction time estimation accuracy. It seems that when (relatively) not many issues are posted during the construction phase, the projects are finished more on time.

6.4.16 Hypothesis 16

There is a relation between requirements creep and effort overrun: it is expected that when RFCs are implemented this results in effort overrun, as RFCs are unplanned work.

As figure 49 shows, it looks like there is indeed a relation between the number of RFCs implemented and effort overrun. It can be seen that there is still overrun when there is a

relatively small amount of implemented RFCs, but in general it seems that the more RFCs are implemented the more effort overrun occurs.

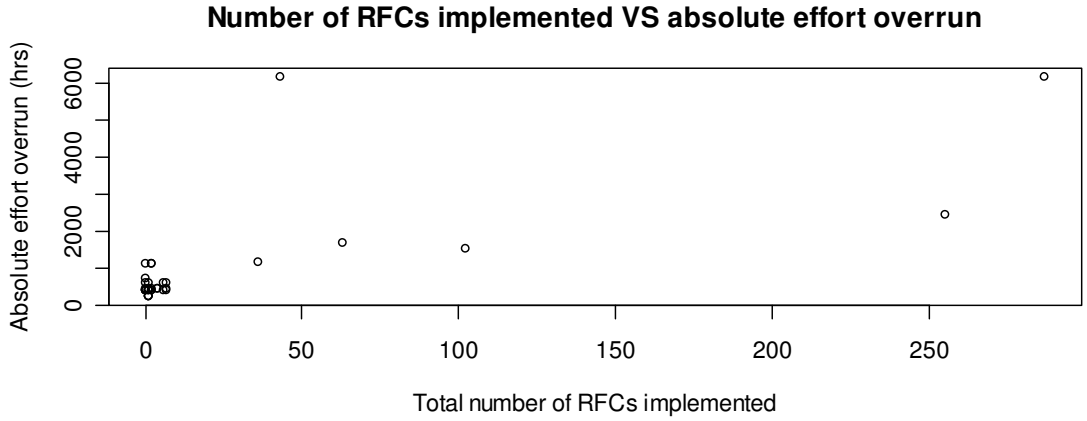


Figure 49

Conclusions:

- There seems to be a relation between the number of implemeted RFCs and absolute effort overrun as expected. Reducing the number of implemented RFCs could therefore reduce the magnitude of effort overrun.

6.5 Summary of the results and recommendations

A summary of the most important things found in the data and metrics:

- The number of pages or portlets alone is not a good measure for the size of a project because the pages or portlets differ in difficulty and may have reuse. To get a better sense of the size the number of pages and portlets should be categorized in difficulty or complexity and reuse factor;
- Without a good measure for the size, it is also hard to judge the completeness of specifications and designs. When pages are not described in the specs, this does not necessarily mean that the developers will have a hard time implementing it. However, when the description of pages are more complete, this might lead to less requirements creep as seen in chart 19. Also, when there is low spec page and visual design coverage, this does not necessarily lead to less issues. A simple portal might not need extensive specifications and design. A complex portal may therefore have more complete specifications and design, but also relatively more issues due to its complexity;
- The rise in number of pages or portlets does not seem to fully comprise the actual amount of requirements creep. First of all, this might be due to the difference in difficulty and reuse in additional pages or portlets. Second, requirements creep can also come in the form of modifications, this cannot be measured by units of additional number of pages and portlets. Therefore, it is recommended to take the number of implemented RFCs as measure for requirements creep;
- The commercial effort estimation is too optimistic for all projects;
- The WBS effort estimations were generally more realistic than the commercial effort estimation, but still show major overruns. It is therefore recommended that a project manager is involved when making the commercial estimation for a more realistic estimation;
- The management should consider distributing the amount of resources and time unevenly in future projects similar to the titlesite projects. All titlesites were assigned the same amount of resources and time, but there are many reasons to believe that it would be better to size up each titlesite and assign resources and time accordingly. Also, this analysis does not show it clearly, but relatively early

titlesites were probably more problematic than latter ones, it might have been better to assign more time and capacity to the early titlesites;
- When the reuse factor is very high (like in the titlesite project) a lower amount of issues per construction day is to be expected;
- Doing more types of tests helps a lot in finding issues during construction (and therefore the first wave of issues becomes smaller), therefore in future projects it is recommended that at least performance testing becomes mandatory besides the functional tests and customer acceptation tests. In order to stimulate this, the project managers should discuss and specify the performance and load requirements with the customer. In almost all analyzed projects these requirements were missing. Also, by finding more issues before going live, the customer finds relatively less issues and this could increase customer satisfaction;
- The total effort seems to show a relation with the number of issues posted. The time does not show a clear relation with the number of issues;
- Effort overrun does not automatically mean that there will be time overrun. This may be caused by overworking or adding more resources to a project;
- There seems to be some relation between effort and construction time. This can be used to make realistic time estimations when a (realistic) effort estimation has been made;
- Unsurprisingly, this also holds for total project time;
- There may be a relation between WBS effort overrun and the number of issues per construction day. A high number of issues per day may increase the likeliness of WBS effort overrun;
- When not many issues are posted during construction, the project is more likely to finish in time. Therefore, for future estimations it should be kept in mind that if more tests are done during construction more issues will be found and therefore more rework has to be done before going live. This again might result in time overrun, so when estimating effort and time the amount of rework plays an important role;
- Requirements creep is likely followed by effort overrun. Therefore, if in future projects it is important to reduce effort overrun the change management should be strict and the number of implemented RFCs should be kept to a minimum. After discussion with one of the offshore managers, one of the issues here is that RFCs are sometimes wrongly posted as bugs by the customer. This causes the developers to fix these bugs that are actually RFCs. Therefore, the change management should be more strict, and the customer should be trained to post RFCs as they should be posted.

However keep in mind:
- Some correlations may show in the previous section, but in time it might show that there was no correlation at all. The other way around: some correlations may not show, but in time there might be a correlation after all. Also, because of the many changes in the process, some correlations may be eliminated or even be created due to these changes;
- Relative metrics can be deceiving! Requirements creep of a few percent could give an impression of little requirements creep, but the absolute number of RFCs might be large, or the absolute number can be small but relatively big. Therefore, the management should always look at the metrics with a critical eye;
- All conclusions made are not scientifically proven; this is not possible because the data is not independent;
- A successful metrics program requires consistent, complete and frequent data measures (Fenton et al. 1997) and metrics calculations. Finding constant motivation and effort in collecting data and making metrics is hard to maintain, because most of the results show only on the long term. It is therefore recommended to assign resources that will be responsible for this task.

6.6 Issue charts

For every project a chart was made showing the development of issues. From the date that the project started, the cumulative number of posted and open issues was derived for every week until the end of the project. The charts give a global view of how many issues were found in a given period, and how many were closed and therefore might be an indicator of the productivity of testing and issue fixing. In general, sharp rises (or no rise at all) of the cumulative amount of posted issues in a particular period could say something about the testing efforts in that period. Sharp rises at the end of a project might give an indication of the size of the first wave of issues. Sharp descents in the number of open issues could give an indication of the productivity of the development team in fixing issues. Appendix B shows six of these charts.

Note: at the end of June, one of SS's employees implemented a function that can generate similar charts like these in Jira automatically. The differences between the charts made during this internship and those charts are that in this internship only issues are considered that (may) result in extra construction work, i.e. bugs and RFCs (new features and improvements). Tasks and questions are therefore excluded. The Jira charts contain all issues ((sub) questions and (sub) tasks are also included). Also, instead of posted and open issues, Jira shows posted and resolved issues. Resolved issues may give a better indication of fixed issues than closed issues, because resolved issues include issues that have been fixed but not verified by the reporter yet, closed issues include only issues that have been fixed and verified by the issue reporter. Keep in mind that resolved issues may be reopened by the reporter if the problem has not been solved in a satisfactory way, so resolved issues don't exactly represent the actual number of fixed issues, but probably comes very close to it. A drawback to closed issues is that the issue may be resolved earlier but is waiting for the reporter to verify and close it. Therefore there is a delay in the date between resolved and closed. Besides this, as of June this feature is available only on the newest Jira environment (http://jira.componence.com), generating these charts for older projects on other Jiras and KlantNET is not possible (yet). For the projects for which making these charts is possible, very little effort is needed to generate the chart, which is a big plus. The effort for making the charts in this internship was approximately 15 to 20 minutes, while the automated Jira reports cost less than one minute to generate.

6.7 Proposal for future metrics or analysis

After collecting the data and metrics for the set of projects mentioned before, some proved to be more useful than others. In this section the metrics are listed again together with a description of their usefulness. Some metrics that were not possible at this moment (for instance due to lack of data) are also included as new metrics.

6.7.1 Evaluation for metrics for the quality of (the commercial and WBS) effort estimation:

- Total number of people involved in the commercial estimation and their role in the project: Tracking this metric for each project does not require that much effort, it has to be done only once per project. It can be useful to see who was responsible for the bid, and afterwards check if the bid was good or not to get a good feel of who are good estimators and who are not. Therefore not only should the number of people be registered, but also who they are, their position and their estimation. This also helps in creating the responsibility for making good quality estimations;
- Accuracy of commercial effort estimation: realised effort/(commercial) estimated effort (=project price/CONFIDENTIAL euros): This metric also does not require that much effort, as long as the actual realised effort is recorded. It can be useful to track this metric to determine the profitability of a project. However, keep in mind that it may not be justified to divide every project price by CONFIDENTIAL euros. Some resources are more expensive than others;
- Accuracy of WBS effort estimation: actual effort/WBS effort estimation:

This metric should definitely be tracked in the future to track the quality of the WBS effort estimations;

- Absolute effort overrun: actual effort – (commercial) estimated effort: This metric (both for commercial and WBS estimation) should be tracked to complement the previous metric, as the previous metric is relative and may give false impressions;
- WBS effort estimation/(commercial) effort estimation: This metric may be useful because it immediately shows whether or not the commercial estimation was realistic according to the amount of work to be done. The management can know at the start of the project how profitable the project will probably be;
- Accuracy of (calendar) time estimation: actual time/estimated time: This metric should also definitely be tracked, it can be used to improve the estimation of project duration and takes little effort;
- Amount of rework: In the analysis the amount of rework was expressed as the number of issues. It would be better to know the actual effort needed for fixing an issue. This is done currently, but not consistently, i.e. not all issues have the effort posted. Some developers register only the effort for implementing it and not for testing it and others do, and some post the total amount of effort (fixing on local server, test server and prod server) and others only post the amount of effort that was needed for fixing the issue on the local server. It is not easy to register this effort consistently for each and every issue, and therefore it should be questioned whether or not to do this. But this data can be used for rework effort estimation, and the management should decide whether or not this is worth the trouble;
- Amount of rework per construction day: The previous metric can be used to calculate this. But instead of rework per construction day it would be better to track the amount of rework per effort hour. The reason for this is that a construction day says less about the size of the "piece of work" that was made that day. Ten coders working on one day for instance probably generates more bugs than one coder.

6.7.2 Evaluation for metrics for the quality of testing:

- Number of different tests actually done during the project before going live (0-5): As seen in the analysis, this metric helped show that performance testing was crucial for the quality of the portal after deployment. This metric can be tracked to see if it is indeed the case that performance testing is that crucial, and what the effects are of adding or excluding different types of tests. However, this metric is not very accurate because it does not give an indication of the quality of the tests. Therefore it is recommended to record for instance the total effort spent on different types of tests if a more accurate metric is desired;
- Number of bugs found after deployment on prod (live)/total number of bugs: This ratio gives an indication of bugs that were missed during testing and hence gives an indication of the quality of testing. This metric can be used together with the previous metric. It gives an indication of the quality of the portal and can for instance be used to help determine whether or not the support can be handed over to the support team;
- The percentage of bugs posted by {customer, Dutch team, offshore team}: This metric is also useful for evaluating the effectiveness of the testing done and does not take very much effort making.

6.7.3 Evaluation for metrics for the quality of product specification:

- Page coverage of functional specs: number of web-pages described/number of web-pages to be implemented: This metric proved to be not very useful. Using this definition for page coverage, a portal that had many reused pages was classified as being poorly specified, while this was not necessarily the case. However, a high value using this definition did show some correlation with less RFCs per construction day. It may therefore be useful to include a reused page in the specifications (even if it is reused almost entirely), just so that the page is

there and specifications and scope are fixed and therefore there is less room for requirements creep;
- Portlet coverage: number of unique portlets described/number of unique portlets to be implemented: Portlet coverage was generally very good for all portals. This metric is therefore not very useful;
- Visual design page coverage: number of pages present in VD/number of pages to be implemented: This metric also seems to be not very useful. The standards are not uniform, i.e. the relatively simpler portals with a high degree of reuse don't require full visual design coverage and the relatively complex portals do. Therefore, it cannot explicitly be said whether or not a low visual design page coverage is the cause of anything;
- Number of RFCs: this metric is definitely useful and can easily be extracted from Jira. As seen in hypothesis 16, it seems that implementing RFCs may be a cause for effort overrun.
- Load performance requirements available: Almost none of the portals had load performance requirements and therefore no analysis has been done involving this metric. It is recommended that load requirements are stated for all future projects in order to get the performance to a desired level. If this becomes standard, there would be no need for this metric;
- Response time requirements available: As the previous metric, it would not be necessary if these requirements became standard (which is recommended in order to get performance levels up). For all the projects mentioned, virtually none had load and/or performance requirements in the project specifications. Tracking whether or not there are load or performance requirements can stimulate the project managers to specify them. In the titlesite project (and most of all other BEA projects), where none of the portals had these requirements, server overload problems occurred as there was no analysis done beforehand concerning the expected load and required performance of the portals. When the portals went live, more visitors visited the portals than expected. The load was too much for the servers to handle and the portals went down due to this unexpected load.

6.7.4 Evaluation for metrics for the degree of requirements creep:

- Number of RFCs that ended up being built in the current project/number of RFCs requested: This metric does give an indication of the degree of requirements creep, but it is gives a qualitative indication. Therefore, this metric should be considered together with the absolute number of RFCs requested and implemented. Also, because many RFCs are requested internally (i.e. not by the customer), it might be wrong to say that there was a large amount of requirements creep when there are for instance many internal RFCs and they were all implemented. This metric should therefore be tracked for both internal (RFCs posted by Componence and the offshore team) and external RFCs (RFCs posted by the customer);
- Number of pages actually built/number of pages initially planned: In general, there was no excessive growth in pages for any of the portals. Also, this number does not give a good indication of how much extra effort was needed when there was growth in pages, i.e. an extra page could have been a near identical copy of another page (and probably required very little effort) or it could have been an entirely new page (that may require a large amount of effort);
- Number of portlets actually built/number of portlets initially planned: As the previous metric, this metric does not give a very good indication of the requirements creep as an extra implemented portlet does not say much about the effort required because the extra portlet could for instance be a simple or a complex portlet. It would be better to record the amount of effort needed for implementing functionality outside of the scope;
- Number of unplanned custom portlets: This metric also does not give a good indication of the true requirements creep as it does not give an indication of how much extra effort was needed. Another issue with this metric arose for instance in

project Gemeente. This metric did give a good indication of the requirements creep in the sense that many standard portlets (actually almost all of them: 17 out of 18) were adapted so much that they were to be regarded as custom portlets. However, this is a different case than when there were actually 17 custom portlets added, and this metric does not make a distinction between the two cases. Therefore it is recommended that, if the management wishes to track this metric, to include the extra effort that was spent for extra portlets or modifying the existing portlets. It is however not recommended to keep using this metric in its current form;

- Actual custom portlets/actual total number of portlets: Only when compared to the "planned" ratio this metric gives an indication of requirements creep, and then not even accurately as it does not say much about the effort required. A rise in this ratio does not explicitly say how much extra effort was involved.

6.7.5 Evaluation for metrics for the size of the first wave of issues:

- A chart of the accumulation of posted and open issues at different moments in time: Please refer to section 6.6 for a more extensive review of these charts;
- Percentage of issues posted before live date (date on prod server) and after live date: This metric gives a good indication of the size of the forst wave of issues. It also proved to be very useful in combination with the number of types of tests done. It does not take that much effort to track and seems to give a good indication of the quality of testing.

6.7.6 Evaluation for metrics for the degree of reusing:

- (Planned) custom portlets/total (planned) number of portlets: This metric can be useful to get a sense for the degree of planned reuse, but does not give an accurate representation of the actual planned reuse rate as the custom portlets have different grades of complexity. One complex custom portlet may take as much effort as many simple ones. It may therefore be somewhat useful, but it does not give an accurate representation of the degree of reuse and the difference between planned and actual extra effort needed.

6.7.7 New metric: root cause analysis

In the earlier sections of this chapter, it was not always clear what caused many issues. In order to truly find the causes for many issues, a root cause analysis can be done. However, in order to do this, the root cause has to be recorded for each issue when posted. This enables the managers to see what root causes have many issues attached to. For instance, if it is indeed the case that custom functionality generates many issues, and the managers want to avoid a large first wave of issues, then it might be an idea to build the custom functionalities at the start of a project and allow a larger time span for the testers to test these custom functionalities. The issues then might occur relatively earlier in the project process. Another example: if one of the major root causes was unclear specifications or bad design, the management can try to improve or demand the specifications and/or design to be of higher quality in future projects and see if the amount of issues with this root cause decreases. The root cause analysis was discussed by Pooley (2002) who implemented this for a company. It would be very easy for Componecne to implement this in the current issue tracking systems and it does not take a large amount of effort to classify the root cause for each issue. The root causes in Pooley's article are: code, design, specifications/requirements, environmental/support, documentation and other. These root causes could also be used by Componence, as Pooley's root causes are also designed for web based projects and the root causes are also applicable to Componence. Other root causes could be: bad testing (when an issue is discovered that should have been discovered by testers) or difficult custom functionality.

6.7.8 New metric: classifications of RFCs

RFCs, which are now recorded as new features and improvements, should also be classified or categorized. A change in a position of a portlet for instance is less severe than a change in functionality of a portlet. In the analysis there was no distinction in RFCs. A distinction could give more insight in the severity of an RFC and true requirements creep. Also, as RFCs seem to be causing a part of the effort overrun, a distinction could be made between the effort needed to implement RFCs and the effort needed to implement the specified/planned functionality. Doing this enables the management to exactly determine how much extra effort was spent for implementing RFCs. As RFCs are recorded as issues in Jira, it can be included in the root cause analysis so the management can see what the main causes for RFCs are.

6.7.9 New metric: actual Dutch effort

Up until now, the effort for Dutch management has never been tracked per project, let alone per portal. In order to gain more insight in the productivity and cost of the management it is recommended that the effort of the Dutch management is tracked. However, this requires extra effort from Dutch managers.

6.7.10 New metric: actual direct costs

When the actual effort of Dutch management is known, the actual direct costs can be approximated by calculating the costs for Dutch management. The outsourcing costs can be approximated using the actual effort of the offshore team. Summing this together with the infrastructure costs gives a good indication of the direct costs made for the project or portal and allows the management to better determine the profitability of a project. Also, it is better to compare the direct costs with the price (bid) than comparing only the offshore effort with the price divided by CONFIDENTIAL.

6.7.11 Summary of recommended data and metrics tracking

First of all, tracking metrics does not always show immediate results. Perseverance and continuing support from the management is needed in order to be able to reap the benefits from a metrics program. While collecting the data and metrics for the projects, a lot more effort was needed than expected. The data was often recorded inconsistently, hard to find, or simply missing. It is therefore absolutely recommended to collect these valuable data more consistently for all projects in order to gain more insight and make a foundation for improvement.

This chapter concludes with a summary of metrics to track for each portal, together with a priority and overall grade of difficulty for tracking it. Note that some metrics mentioned before may be formulated different to better represent what they mean or because they were improved. Also, metrics that were not very useful may be left out. The grade of difficulty represents the difficulty or extra effort needed for collecting the data and/or calculating the metric. A metric with a high rating and low difficulty are easy, valuable metrics. A metric with a high rating but also high difficulty is also recommended to be implemented, but require serious effort.

Estimation quality metrics	Rating	Difficulty	Data needed	Occurrence
Project price	4/5	1/5	Project price	In advance
Who was involved in the bid and made what estimation	4/5	1/5	Estimations	In advance
Absolute commercial effort overrun	5/5	2/5	Project price, actual effort	Post mortem
Accuracy of commercial effort estimation	4/5	2/5	Project price, offshore team hourly costs, actual effort	Post mortem
WBS effort estimation	5/5	1/5	WBS	In advance
Absolute effort overrun WBS	5/5	2/5	WBS estimation, actual effort	Post mortem
Accuracy of WBS effort estimation	5/5	2/5	WBS estimation, actual effort	Post mortem
WBS estimation/(commercial) effort	4/5	1/5	WBS estimation, project price	In advance

estimation				
Duration (calendar time) estimation accuracy	5/5	1/5	WBS duration estimation, actual duration	Post mortem
Amount of rework (expressed in number of bugs)	3/5	1/5	Total number of bugs	Post mortem
Amount of rework (expressed in number of hours)	5/5	5/5	Effort hours spent on fixing bugs	Post mortem
Number of bugs per effort hour	3/5	2/5	Total number of bugs, actual effort	Post mortem
Amount of rework hours per effort hour	5/5	5/5	Effort hours spent on fixing bugs, actual effort	Post mortem

Testing quality metrics	**Rating**	**Difficulty**	**Data needed**	**Occurrence**
How many types of tests were done	4/5	1/5	Test reports	Post mortem
Ratio of bugs found after deployment on prod	3/5	1/5	Date on prod, number of bugs at prod date, total number of bugs	Post mortem
Ratio of issues posted by Componence	3/5	2/5	Total number of issues, total number of issues posted by Componence	Post mortem
Ratio of issues posted by offshore team	3/5	2/5	Total number of issues, total number of issues posted by offshore team	Post mortem
Ratio of issues posted by Customer	3/5	2/5	Total number of issues, total number of issues posted by customer	Post mortem

Specs quality metrics	**Rating**	**Difficulty**	**Data needed**	**Occurrence**
Functional specs page coverage	2/5	1/5	Sitemap, number of pages described	In advance
Number of RFCs (minor)	3/5	2/5	Number of minor "New feature issues" and "Improvement issues"	Post mortem
Number of RFCs (medium)	3/5	2/5	Number of medium "New feature issues" and "Improvement issues"	Post mortem
Number of RFCs (major)	3/5	2/5	Number of major "New feature issues" and "Improvement issues"	Post mortem
Load requirements available?	2/5	1/5	Load requirements	Post mortem
Response time requirements available?	2/5	1/5	Response time requirements	In advance

Requirements creep metrics	**Rating**	**Difficulty**	**Data needed**	**Occurrence**
Number of implemented internal RFCs	3/5	1/5	Number of implemented RFCs posted by Componence and offshore team	Post mortem
Number of implemented external RFCs	3/5	1/5	Number of implemented RFCs posted by the customer	Post mortem
Effort spent on internal RFCs	5/5	4/5	Effort spent on internal RFCs	Post mortem
Effort spent on external RFCs	5/5	4/5	Effort spent on external RFCs	Post mortem
Number of implemented RFCs (minor)	3/5	2/5	Number of minor "New feature issues" and "Improvement issues" implemented	Post mortem
Number of implemented RFCs (medium)	3/5	2/5	Number of medium "New feature issues" and "Improvement issues" implemented	Post mortem
Number of implemented RFCs (major)	3/5	2/5	Number of major "New feature issues" and "Improvement issues" implemented	Post mortem

First wave issues metrics	**Rating**	**Difficulty**	**Data needed**	**Occurrence**
Issue chart	4/5	1/5	Generated in Jira	Any time
Ratio of issues found after deployment on prod	4/5	1/5	Date on prod, number of issues at prod date, total number of issues	Post mortem
Root cause analysis	5/5	2/5	New field in Jira: root cause	Post mortem

Table 2

7 EFFORT AND DURATION ESTIMATION

In this chapter different aspects of effort and duration estimation for the project process are discussed: what are the main issues of Componence's effort and duration estimations? What are possible improvements? The issues and improvements derived from discussions with the Dutch and SS Software management, Professor Van Vliet, and existing literature concerning effort estimation.

7.1 How is effort and duration estimated currently?

There are two different forms of effort estimation in Componence:

- The top-bottom commercial estimation (which is actually a bid);
- The bottom-up WBS effort estimation for the planning of resources and time.

7.1.1 The commercial bid

The commercial bid is usually made by sales (accountmanagers and pre-sales consultants) and is expressed in financial units (i.e. euros). This estimation could be influenced by competitors, who forces the sales managers to make a competitive bid, or by the customer. For instance, a new, big, customer will likely have to pay a lower price than normal in order to attract that new customer when there are competitors also bidding for the project. Currently, no tools are used when making this estimation, except for some simple calculation excel sheets. After the initial bid, an analysis is done in order to get a better sense of the expected effort required, often by defining the scope and requirements. Another bid is then made and a final price, usually fixed, will be the result of negotiations between the sales department and the customer, for instance by adapting the scope, price, or both. Lately, offshore managers are being more involved in this process, as bids have been a constraint (often an unrealistic one) on the allowed resources in past projects. This way, the offshore managers hope to have a more realistic amount of allowed resources and time.

7.1.2 The WBS estimation

The bottom-up WBS effort and duration estimation is usually made by the Dutch and offshore project managers and offshore technical leaders. Each activity is broken down into a small "piece of work" and an estimation for each of these pieces is done in units of hours. Each of these activities has some variance in needed time and effort, and when taken together, these variances should cancel each other out. This is not done for all projects, but most of the projects at least have a WBS, that sometimes is missing the estimations for each activity and therefore is missing a total estimation. One of the issues here is that these WBS estimations are done in a tool called Microsoft Project, but not all Dutch managers and employees have had a license for this tool until recently. Another issue is that when this tool is used, the progress is not always kept up to date. As mentioned before, the Dutch managers want more insight in the progress made by the offshore teams. Therefore, not only should the WBS estimation be of better quality, but the use of the tools should be uniform and the progress should be kept more up to date to allow for more timely corrective action and/or replanning when there is (a possibility for) effort and/or time overrun.

7.1.3 Project duration estimation

A date is agreed with the customer at the commercial estimation, sometimes the customer already has a deadline, and sometimes this date can be negotiated with. In the case that the customer has not set a deadline, a very rough estimate is made, as the specifications are very vague, and the duration estimate is adjusted when specifications are clearer and a WBS can be made. Otherwise, the scope of the project will be made in such a way that the project managers think that implementing the functionality in this scope is do-able before this deadline.

7.2 What is the quality of past estimations?

In this section past estimations are analyzed.

7.2.1 Bid and WBS effort estimation quality

In general, the bottom-up WBS effort estimations (or rather resource reservations in many cases) are better than the commercial top-bottom effort estimations (bids), this also shows in the data analysis in the previous chapter. There might be reasons for the top-bottom commercial bids to be low and therefore it might not be justified to compare these two types of estimations. For instance, the management could have made a low bid (and incur a loss or accept a lower profit) in order to win or retain a particular customer or to prevent the competition from making a profit from this customer.Figure 50 shows that the WBS effort estimation is also overrun often. Figure 51 shows that, when all values greater than 2.0 are removed, most portals show between 20 and 50 percent effort overrun. Therefore it is safe to say that the current effort estimations (at least safe to say for the WBS based) are too optimistic and there is plenty of room for improvement in this area. Keep in mind that a part of the effort overrun is caused by implementing RFCs. Therefore, not all effort overrun is due to a too optimistic WBS estimation. One of the offshore managers suspects that a bad WBS effort estimation accounts for roughly the same amount of overrun as requirements creep does.

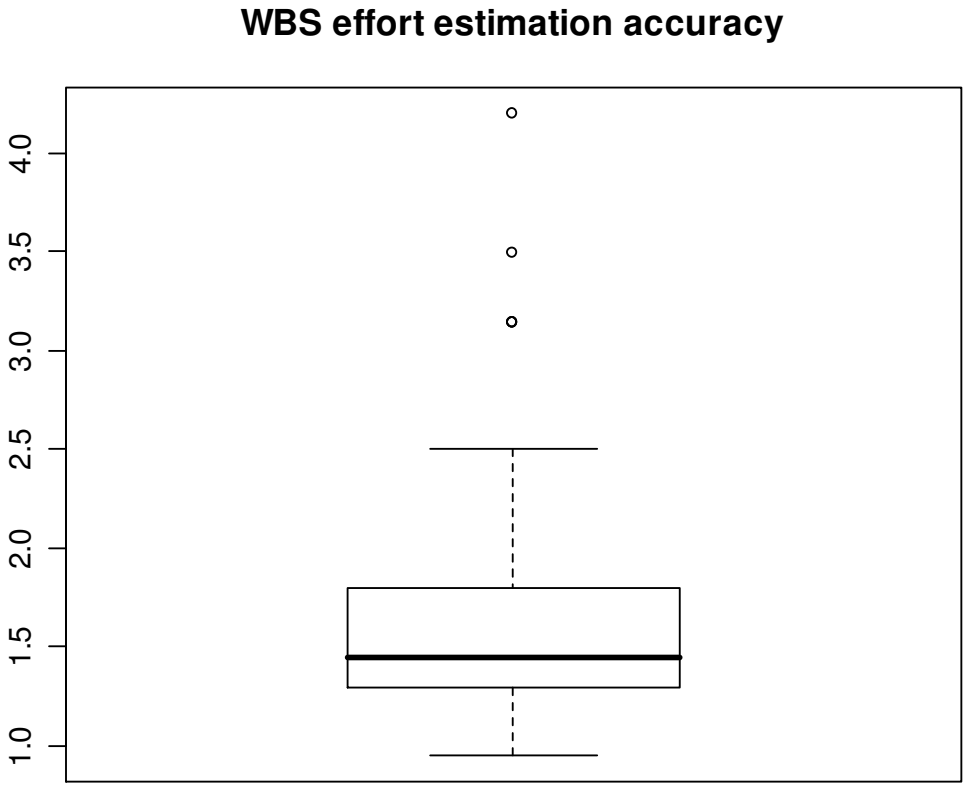


Figure 50

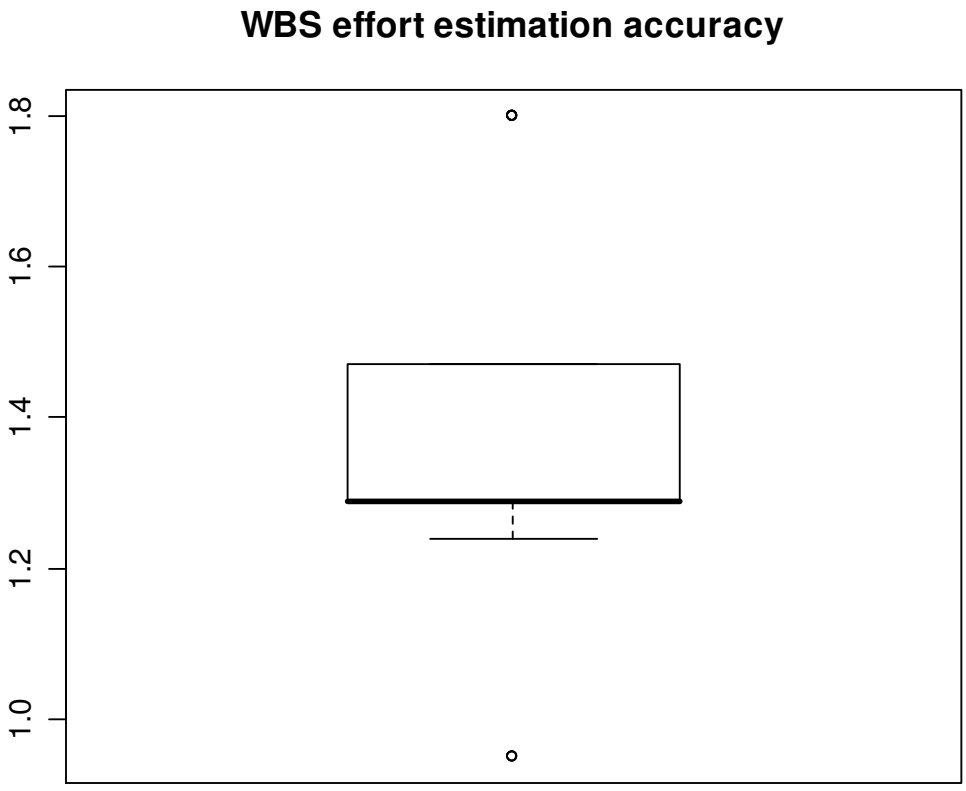


Figure 51: values >=2.0 removed

7.2.2 Duration estimation quality

Figure 52 shows a boxplot of the construction time estimation accuracy for all sites. It shows that construction time overrun occurs a lot, the goal for Componence would be to get the box centred at 1, with as little spread as possible, i.e. the height of the box should ideally be very small. The median is pretty close to 1. Figure 53 shows the same boxplot for total time estimation accuracy. It is evident that most of Componence's projects have time overrun and that there are also improvements possible for the estimation of the duration. The goal is to get the median closer to 1.0, the box more around 1.0, and less spread. It should be mentioned however that most data is taken from the titlesite project. There is not much experience for this type of project, and this could of course explain why almost all projects have time and effort overrun. It is therefore recommended that this data is recorded for all future projects, and some distinction could be made between the types of projects. It should be noted that, even though there are not many non-titlesite observations, almost all of these show overrun too.

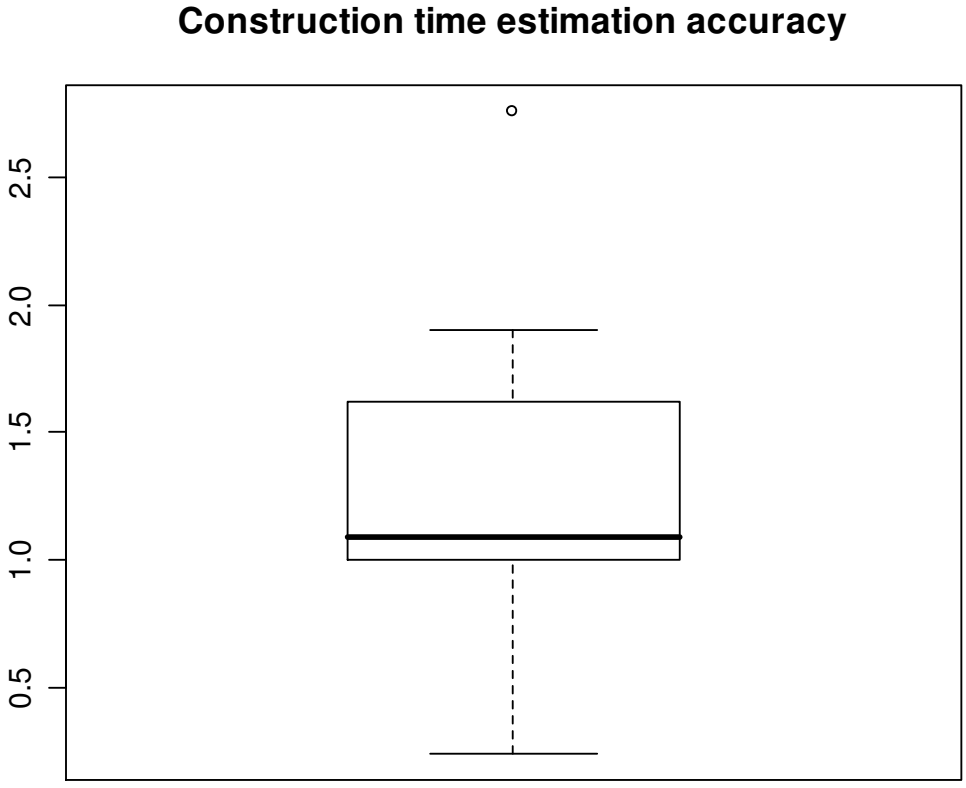


Figure 52

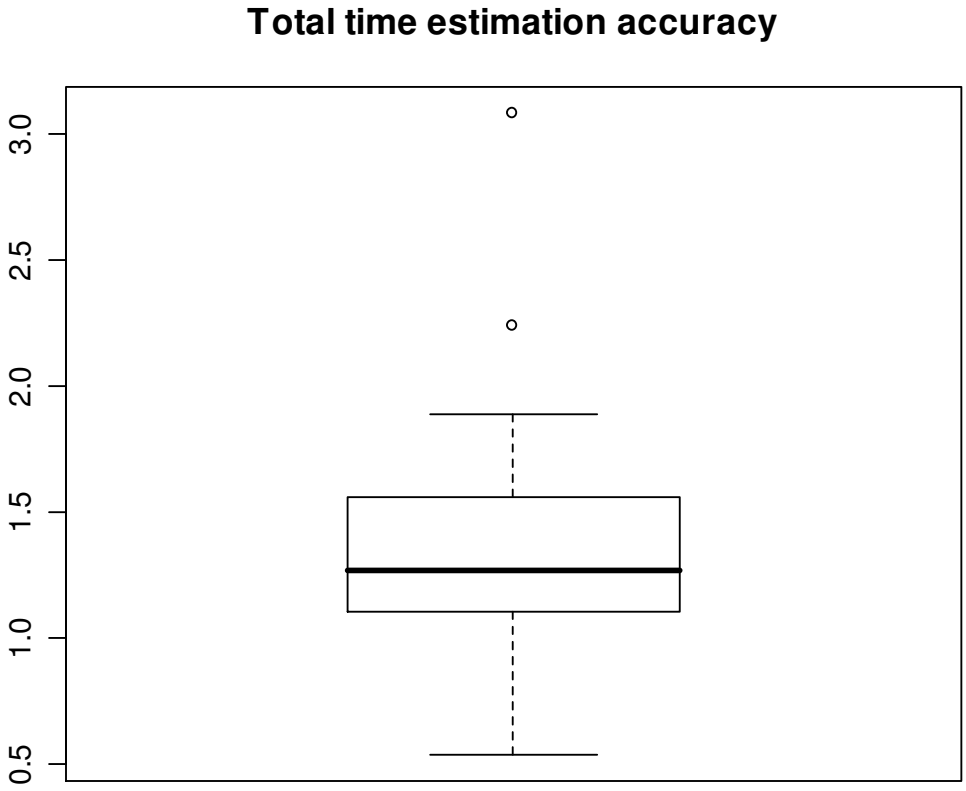


Figure 53

7.2.3 Titlesites: less problematic over time?

Figure 54 shows the total number of issues at live date for titlesites. The x-axis show the order in which each of the titlesite projects started. The first four observations have a low number of issues because these sites were almost copied from another portal (Gemeente). Due to this reuse many issues that were posted were shared and mainly posted for Gemeente. This chart is made to see whether or not the titlesites are less problematic at the end of the program. The last five observations all have relatively a small amount of issues at live date. If it is indeed the case that the later titlesites are less problematic it might be an idea for future, similar projects to adjust the time and effort estimations in such a way that early sites get more effort and time assigned to them.

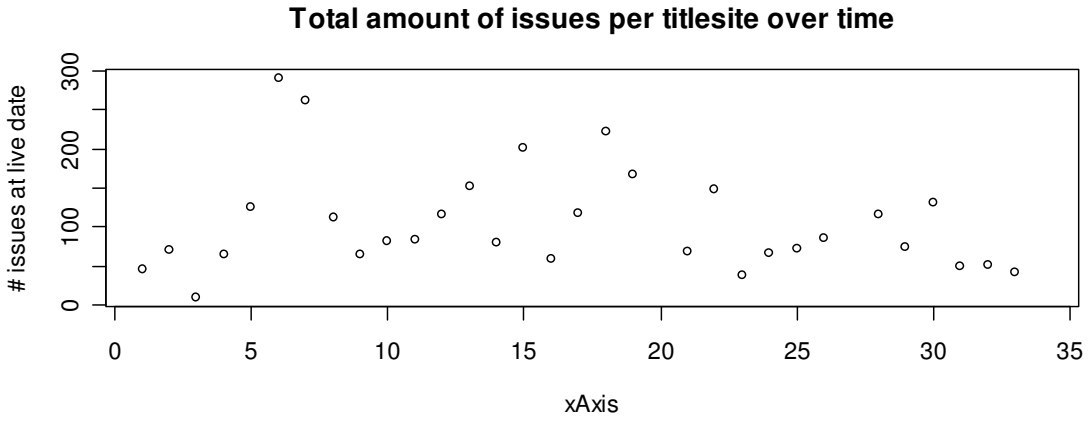


Figure 54

7.3 What improvements are possible in general?

Up until now, the bid for a project has been used as a constraint on the allowed resources and time. It has often occurred that relatively low bids in the past forced the development teams to work with too little resources and time due to this. The offshore project managers are now more involved in making the commercial price estimation in order to avoid the bid to become a unrealistic "ceiling" for the allowed amount of resources and time.

Jorgensen (2005) argues that the term "effort estimation" is often (and wrongly) used for different purposes, namely for effort estimation, planning and bidding. According to him, these are three different estimations, each with different purposes:

- When estimating the most likely effort, the goal should be accuracy alone;
- Planning should also focus on the planned effort's impact on work efficiency and the risk of overrunning the schedule;
- Bidding goals relate to winning bidding rounds and profit.

Currently, the most likely effort is now estimated by breaking down all activities into a WBS and estimating the effort and time needed for each activity. However, when these estimates were done, the bid was a constraint for the allowed resources and time in past projects. The purposes of the WBS effort estimation and bid were therefore mixed. This might have caused the WBS effort estimation not to focus only on accuracy, and could (unconsciously) reduce the realism of the WBS estimation. The boxplots for WBS effort estimation accuracy shown earlier seem to confirm this, as effort has been consistently underestimated (possibly in combination with requirements creep). Jorgensen therefore recommends the bid and effort estimation to be separated. Since this research started, commercial bid is converted in order to have a "commercial estimation" expressed in hours. This is done by dividing the price by CONFIDENTIAL euros, as is explained in section 6.2.1, as one hour of offshore effort should cost the customer approximately

CONFIDENTIAL Euros. This also indicates that the project price has formed a constraint for the allowed resources.
Would it be useful for Componence, to separate these estimations? According to Jorgensen, if accuracy is the goal of the effort estimation, then yes. Of course, one could argue that if direct profit must be made from the project, the effort estimation would have to be made with the bid in the back of the mind, as the planning and resource allocation would then fit within the constraints formed by the bid. However, according to Jorgensen, this could lead to bias, as the planner would unconsciously make a tighter or looser planning due to a relatively low respectively high bid. In the case of a planning that is too tight due to a bid that was too low (as most often is the case) effort and duration overrun is more likely to occur, resulting in all kinds of complications, including complications mentioned earlier and less profit than projected (or maybe even a loss).

Therefore, if accuracy is the goal, the management should consider separating the bid, effort estimation and planning. After all, it is better to know in advance that the needed time and resources are more than the price allows than to find out that this is the case halfway during a project.

Another recommendation is that in projects similar to the titlesite project (where a large series of similar portals are constructed), the first batches of portals could be more problematic than the later ones and therefore may require more time and effort.

7.4 What improvements are possible for the commercial bid?

What is the purpose of the bid? According to Jorgensen, bidding relates to winning bidding rounds and profit. Therefore, the bid needs to be balanced in order to win a customer from other competitors, and still result in profit. However, a distinction should be made in direct profit and indirect profit:

- Direct profit is made when the total costs for a project is less than the price paid by the customer;
- Indirect profit could for example be profit made through licenses, customer service, maintenance, or profit from future projects of the same customer.

A project could result in a direct loss, but indirect profit could compensate for this. For instance, a project could have been sold for a relatively low price (resulting in less profit than normal or even a loss) in order to win or retain a customer for future projects. Therefore, when making a bid this should be kept in mind. It can therefore be argued that the bid should not always be a constraint for the allowed resources and time. The management should therefore determine whether or not it is justified that the bid is going to be a constraint for the allowed resources and time for future projects. In any case, it is recommended to separate the bidders from the estimators as mentioned before.

In order to determine whether or not a direct profit can be made, it is important to know what the costs will be for a project. However, Componence has no tools or models for estimating the costs for a project, and data of the costs made for past projects are not systematically registered. SS has started to systematically measuring actual effort since the beginning of 2006, giving more insight in the offshore costs made for projects done by SS project teams. This can be used to determine and explain part of the direct costs made for projects done by SS. Most of the other offshore teams don't register this as detailed and only total costs are known for each team each month, but not per project. Besides this the direct costs of the Dutch employees for different projects is not known, only the total salary per month is. When collecting the effort data before, it was therefore very hard to find data for direct effort and costs for past projects. In order to determine the direct costs it is therefore recommended that effort and costs data are collected for each separate project in order to gain more insight in the costs made, and with that more insight in how much future projects should be sold for in order to break even.

Indirect profit is even more difficult to determine, as it is difficult to for instance determine how many future projects a customer will do if any, how big these projects will be, what the costs will be for these projects, whether or not the customer will sign a support contract, etcetera. If the management believes that indirect profit can be made, even when the exact amount (or even a rough estimation) cannot be determined, she can consider doing a bid that may not result in direct profit or less than normal direct profit.

One of the developments in the process of bidding now is that the offshore managers try to get more involved in the process of making the bid as the magnitude of the bids, which was often too low, has been a constraint in many of past projects. The offshore managers however, are often the ones that do most (or a great deal) of the WBS and project planning, as they have a better feeling of the effort needed when implementing functionality. The Dutch management should therefore first consider whether or not direct profit should be made, as involving offshore managers would risk the independency of the bid, effort estimation and planning.

- If a direct profit must be made from a project, the bid is a constraint. It may therefore be justified to involve an offshore project manager when making the bid in order to allow more room for a realistic planning and resource allocation. Doing this however would risk the independency of the bid, effort estimation and planning and therefore the realism of the effort estimation and planning. It would be better to separate the bid, estimation and planning and try to adjust the price or scope after the estimation and planning have been made when the management believes a risk for making a loss exists;
- If direct profit is not necessary, it is of course still important to know what the costs will be, but the bid, effort estimation and planning should be made separately. The purpose of separating would be to have a more realistic effort estimation in order to keep overrunning the planned effort to a minimum.

In any case, if the goal is to make more realistic effort estimations, it is recommended to separate the bid and estimations as Jorgensen suggests. So it can be said that in either case, whether or not the bid is a deciding factor in the amount of allowed resources and time, separating the bid from the estimations has some value, namely a more realistic effort estimation. The project price can then be raised (if possible) when too low, or the scope will have to be smaller in order to avoid budget overrun. However, the offshore managers are not convinced that they should not be involved when bidding. In the past, the bid was done only by Dutch managers, and since the offshore managers are more involved, the bids have become "better" (better as in allowed for a more realistic amount of allowed resources and time). They know that in theory separating the bid and effort estimation is better, but they have experienced that this did not work well in past projects. Therefore, if the Dutch management decides to separate the bid and effort estimation in order to obtain a more objective and realistic effort estimation and planning, the price of a project should not be a constraint. Another construction would be to involve an offshore manager when bidding but let another offshore manager, who has no information of the magnitude of the bid, make the effort estimation and planning. The persons responsible for setting the price should be held responsible for the quality of the bid (whatever goal it had), and the persons responsible for the effort estimation should be responsible for the realism and accuracy of the estimations and planning, and not overrunning the amount of estimated effort and duration. Lederer et al (1998) found that the only factor correlated with software organizations' estimation accuracy was the estimator's accountability.

Improving the bid itself would be hard to realise. After all, what is a good bid? What is a good price for a product? In general, maximum profit is of course desired. However, this all depends on many factors, for instance the amount of competitors and the price they are willing to do the project for, the complexity of the project, the knowledge of the customer about prices for similar products, the customer's monetary power, whether or not the customer is likely to do future projects, the current state of the market for

portals, etcetera. Also, Componence's products are not simple, and the whole package includes much more than the product alone. Therefore, making some kind of model for making better bids would not be realistic or at least too complex to be included in the scope of this research. In general, it is recommended that people with different backgrounds are involved (for instance sales manager, project manager, technical leader) that each make a candidate bid, and when large differences are present these are discussed, and each of them should explain what their bid was based on. However, as mentioned before, not involving the people responsible for the effort estimation and planning is recommended.

In short:

- It is at least recommended to gain more insight in the direct effort spent and the direct costs made per project in order to improve the quality of future bids, or at least make bids that will less likely result in a loss. This means at least:
  - Tracking costs and effort spent by all offshore teams per project;
  - Tracking costs and effort spent by Dutch managers and employees per project;
  - Tracking infrastructure costs per portal or project.

  As these costs make up for most of a project's costs;
- Consider whether or not direct profit is priority and whether or not it would be wise to involve an offshore manager when making a bid, at least not the offshore manager who is responsible for the effort estimation and planning. If direct profit is priority, it may be useful to involve an offshore manager (who probably has a better feeling of the magnitude of the expected costs), but if the accuracy and realism of the effort estimation and planning is important, it is at least recommended to not involve any effort estimators and planners when bidding and let the bidders, estimators and planners be responsible for the quality of their estimations;
- Consider whether or not it is justified to use the bid as a constraint for the amount of resources and time. For instance, in the program charter of the titlesite program it is nowhere stated that profit is a main priority, and therefore it is not clear whether or not the project's price should be a constraint. For each future project it is therefore recommended to state explicitly if and how much effort or cost overrun is allowed;
- Make independent candidate bids with people that have different backgrounds and functions and discuss the causes of big differences in the candidate bids.

7.5 What improvements are possible for effort estimation?

To test whether or not the WBS effort estimation (which is based on estimating the activities to be done) is reasonable, another estimation can be made based on the project size (based on the components a project comprises). These estimations from different principles and point of views can be compared, and if a large or unreasonable difference exists this can give reason to re-examine one of the estimations, or to make a new estimation. Combining different estimations into a new one, for instance simply averaging them, can also improve the quality of the estimation (Jorgensen, 2005). In this section existing estimation models are discussed and are used as a basis for making an estimation based on project size.

7.5.1 Known effort estimation models

Some famous effort estimation models are:

- COCOMO 2.0 (Boehm et al, 1995);
- SLIM (Putnam, 1978);
- Function point analysis (Albrecht, 1979);

These models use the number of function points, object points, or SLOC (source lines of code) as a basis for the size of a project. They generally focus on making an accurate estimation of relatively large projects and a relatively large amount of effort is required

making the estimation. Besides this, these models are used for traditional development, while Componence projects are web based projects. The differences between traditional development and web development are illustrated in the table below (Reifer, 2000).

Characteristics of Traditional versus Web Development Projects

Characteristics	Traditional development	Evolving Web development
Primary objective	Build quality software products at minimum cost	Bring quality products to market as quickly as possible
Typical project size	Medium to large (hundreds of team members)	Small (3–5 team members)
Typical timeline	10–18 months	3–6 months
Development approach employed*	Classical, requirements-based, phased and/or incremental delivery, use cases, documentation-driven	Rapid application development, gluing building blocks together, prototyping, Rational Unified Process,[1] MBASE[2]
Primary engineering technologies used	Object oriented methods, generators, modern programming languages (C++), CASE tools, and so forth	Component-based methods, fourth- and fifth-generation languages (HTML, Java, and so forth), visualization (motion, animation), among others
Processes employed	Capability Maturity Model-based	Ad hoc
Products developed	Code-based systems, mostly new, some reuse, many external interfaces, often complex applications	Object-based systems, many reusable components (shopping carts, etc.), few external interfaces, relatively simple
People involved	Professional software engineers typically with 5+ years of experience in at least two application domains	Graphic designers, less experienced software engineers (2+ years), new hires right out of school
Estimating technologies used	Analogy using historical data as its basis, SLOC or function point-based models, Work Breakdown Structure (WBS) approach for small projects	Analogy based upon current experience, "design-to-fit" based on available resources, WBS approach for small projects

*Often a function of best processes used in the past by the firm or industry

Table 3

The differences mentioned in table 3 suggest that the traditional models are less suitable for Componence's needs. Componence's goal is to make a fast, rough and realistic estimation. Reifer proposes to count the Web Objects in a web development project and estimate the effort using a modified COCOMO model called WebMO. After studying WebMO and discussing it with the management, we concluded that the idea of using web objects was attractive, but following WebMO's procedure for estimating would still be too effort and time consuming as it required the estimator to count query, xml and html lines, the number of internal logical files, external interface files, etcetera. These are values that are not available beforehand as the WebMO estimation focuses on projects that are specified in detail. The specifications for most of Componence's projects however are not as detailed as WebMO requires them to be at the moment the effort estimation is made and needed. Also, the portlet technology Componence is using is relatively new and existing effort estimation models based on portlet technology could not be found.

The underlying ideas of the existing models however, can be used as a basis for effort estimation for Componence's projects. When looking at the existing models from a high level, the following activities are carried out when estimating effort:

- How is size measured?
- How is this size converted to effort?
- How is the effort converted to duration?

7.5.2 Proposals for sizing a project

As of the year 2000, researchers have not agreed on what size metric is best for web projects (Reifer, 2000). Also, estimation models based on portlet technology could not be found. The most logical size metric for Componence's projects would seem to be one that should be based on the number of different pages and portlets. Before this internship started the Componence managers were already thinking about using these as sizing building blocks. However, some considerations have to be made:

- The number of pages alone is not a good sizing unit. Some portals have many pages, but they might look alike due to reusing and therefore require relatively less effort than building a page from scratch. Also, a page containing for instance mainly contact information is relatively easier than a page with many different and/or custom portlets. Therefore, a reused or simple page should be

distinguished from a "new" or complex page, or each page should be classified according to their complexity and degree of reuse. The data analysis of the previous chapter also suggests that the number of pages and portlets alone are not good measures for size;
- The different portlets also have to be classified for the same reasons: implementing a custom portlet generally requires more effort than integrating an existing portlet. Even existing portlets have different grades of difficulty when integrating them into a portlet. For instance, a "banner-" or "news-portlet" generally requires less effort to integrate than say, a "forum-portlet". Of course, custom functionality also has different degrees of difficulty depending on the complexity and degree of customization;

Therefore, the size (or a large part of it) of the project could be approximated using pages as counting-objects, but pages should be treated different depending on the portlets they contain and the effort needed to implement and configure the portlet.

7.5.3 Scaling size to effort

In the models mentioned before, when the size is known, it needs to be scaled to the amount of effort. COCOMO II has the following scaling factor:

$$Effort = A \cdot (Size)^{B}$$

Effort is measured in person-months, which, according to COCOMO is 152 hours of working time (which is corrected for holidays, vacation and absence due to sickness). (It should be noted that SS uses a different amount for a full time employee person-month, and a uniform amount of hours should be used across all offshore teams, or some conversion should be made). *A* is a constant, in COCOMO II set to default value 3.0. Of course, this value probably may not be 3.0 for Componence projects, as the units used for determining the size is not the same. *B* is dependent on the economies of scale, i.e. if *B* is taken to be <1.0, the project effort has economies of scale. This means that if the project size is doubled, the project effort is less than doubled. For small projects Boehm recommends a value of 1.0 for *B*, in other words, no (dis)economies of scale are present. Large projects are recommended to have a *B*>1.0, due to diseconomies of scale caused by larger overhead. This larger overhead is in turn caused by more interpersonell communication and larger projects require more integration, testing and maintaining of smaller components and relatively more effort for the management of the project is needed.

WebMO has the following scaling factor, note that the *B* here is not the same as Boehm's *B* (taken from Reifer, 2000):

$$Effort = A\prod_{i=1}^{9} cd_i (Size)^{P1} \qquad Duration = B(Effort)^{P2}$$

Where: A and B are constants
P1 and P2 are power laws
cd_i are cost drivers
Size is the number of Web Objects

Table 4 shows that the values are dependent on the type of portal in WebMO.

Web Development Model Parameter Values

	A	B	P1	P2
Web-based electronic commerce	2.3	2.0	1.05	*
Financial/trading applications	2.7	2.2	1.05	*
Business-to-business applications	2.0	1.5	1.00	*
Web-based information utilities	2.1	2.0	1.00	*

* Either 0.5 or 0.33 depending on the scaling (>40 Web Objects)

Table 4 (Reifer, 2000)

These numbers were obtained by a regression analysis on 46 web projects, cd_i stands for cost driver *i* and is a product of numbers depending on different factors like product complexity, platform difficulty, personell skills, etcetera. Of course, it is not recommended that Componence managers use any of these values mindlessly, as the size based on pages and portlets is of course very different from the web-objects used by Reifer and object points used by Boehm. These numbers are presented just to give an impression of what the numbers are for, i.e. that the size of Componence projects could be multiplied and raised to the power of some number to obtain the effort. However, the value for the factor for economies of scale (*B* in COCOMO II and *P1* in WebMO) might be used and set to (or around) 1.0 for small projects because:

- Most Componence projects are relatively small, and Boehm recommends this value for small projects;
- Reifer seems to have also found this number to be around the same value 1.00 – 1.05 (see table 4);

In other words, there are no or very little economies of scale for small projects. However, in larger projects with lots of reusing (for instance the titlesite project where only standard portlets were used), this value could be smaller than 1.0, as there are economies of scale (but also more overhead for coordinating many teams simultaneously). Reifer includes a reuse cost driver in the WebMO model, but has not obtained values for it yet. It would also take too much effort to determine all other cost drivers for Componence, if at all possible. To keep things simple, it might be best to not use any cost drivers at all (at least for now), as these are numbers that are difficult to measure and quantify.

Keeping in mind that in future projects Componence wants to reuse as many portlets and pages as possible (as done in the titlesite project), it is recommended that at least the effort of the following is measured in order to improve effort estimation quality of the construction phase:

- Setting up a skeleton or reusing a skeleton;
- Writing and testing new/unique pages with an x amount of portlets;
- Modifying and testing reused pages with an x amount of portlets;
- Inserting, configuring and testing the standard portlets (classified by difficulty);
- Implementing, configuring and testing custom portlets (classified by difficulty).

This data can be used to estimate and calibrate the size and effort for implementing future, similar pages and portlets. This is called analogy based estimation, as previous experience is used to make the estimation. While doing this, the constant *A* can be calculated to transform the size into effort, and new observations (actual effort it took to implement a certain type of page or portlet) can be used to calibrate this constant. Then, the effort needed for resolving issues and regression testing can be added. The amount of rework can be estimated, as the data analysis showed that it is likely that the number of issues has a relation with the total effort. However, when changes are being made in the process, for instance making performance tests mandatory in every project, this may influence the relative amount of issues that occurs during the construction phase and therefore influence the required amount of resources and time during this phase. It is therefore important to calibrate this data from time to time with new observations.
An example of how the expected construction effort could be calculated:

Expected portal construction effort ~ (([SK_E] + [T_E] + [D_E] + [1 + RR]([CP01_I_E] + [CP02_I_E] + … + [CPm_I_E] + [PG01_E] + [PG02_E] + … + [PGn_E]))/152)^{EOSF}*

- The skeleton effort (*SK_E*) is the effort needed for setting up the skeleton of a portal. The effort needed for this can be easily measured and classified in different difficulty classes depending on the size and reuse. An example:

Skeleton difficulty:	Simple
Based on:	Project x
Custom ratio:	0.60
Expected effort:	SK_E

  This skeleton is simple, based on the skeleton of project x, but should be adapted for approximately 60 percent. The expected effort is SK_E (in hours);
- For each portal, some tests are always done, and some are left out. For simplicity, every time a test is carried out, record the effort needed and use this data to estimate the expected effort for testing (*T_E*) in future projects. For more accurate estimations, functional testing could for instance be recorded per page, load, performance, stress and security testing could be dependent on the requirements, etcetera;
- The effort for (re)deployments (*D_E*) of the portals can also easily be measured. This data can be then used for effort estimations in future projects. Keep in mind that the deployments are being automated more and the number of redeployments and the effort for a deployment will likely decrease in time;
- The expected rework rate (*RR*) can be computed by measuring the total construction effort and the total bug-fixing effort (= rework effort) and divide the rework effort by construction effort for all projects. It is therefore recommended to update this data regularly, and to track the effort needed for fixing bugs. This is already done in Jira, but not consistently and not all vendors do it. Not consistently means here that it is not done for all projects, and when it is done, it is not always clear what activities were included. For instance, sometimes only the effort for fixing a bug on the local environment was included and sometimes the testing effort on both local and production server is also included. Keep in mind that for portals with more custom functionality rework is probably needed more than for portals with only reused components. Another possibility is to accumulate all construction effort for all projects and all rework effort for all projects and use that to compute the rework rate;
- *CPm_I_E* is the effort needed to implement custom portlet *m*. When the custom portlet is entirely new, this number includes the effort for implementing it, but excludes the effort for inserting and configuring it. When the custom portlet is a modified version of an existing portlet, this is the effort for modifying an existing portlet, excluding the effort for inserting and configuring it. These make up for the largest part of the effort needed for non-reused components and will probably be one of the bigger causes of uncertainty in projects, especially portlets that are built entirely new. For instance custom portlet "Marktprijzen" caused most of the problems in project Boerderij. Of course, for custom portlets it is more difficult to classify their difficulty and use analogy to determine the most likely needed effort than for reused portlets. This depends on the degree of customization (whether it is an entirely new portlet or an adapted version of an existing one), to some extent the quality of the specification of this portlet and the understanding of the desired functionality. Therefore, it can be said that the custom functionality brings some uncertainty to the accuracy of the estimation. To still make an analogy based objective estimation it is recommended to compare the difficulty and size of the custom portlet to similar custom portlets. For example, effort data from the past for custom portlets can be ranked, and when a new custom portlet is to be built, it may be possible to say that it will probably take more effort than custom

portlet X but less than custom portlet Y and make an estimation based on that. This will ensure that the estimation will be more justified and reproducible, rather than that it would be based on gut feelings. An example:

Custom portlet 01	
Based on:	News1
Custom ratio	0.30
Expected effort:	CP01_I_E

Custom portlet 02	
Based on:	N/A
Custom ratio	1.00
Expected effort:	CP02_I_E

…

Custom portlet *k*	
Based on:	N/A
Custom ratio	1.00
Expected effort:	CP*m*_I_E

Custom portlet 01 is based on News1 portlet (a standard PortletSuite portlet). Approximately 30 percent needs to be adapted. The expected effort for implementing this portlet is CP01_I_E. Custom portlet 02 is entirely custom made, the expected effort is CP02_I_E, this estimation is based on effort spent for similar portlets in the past, etcetera;

- For reused portlets, only effort is needed for inserting and configuring. The effort for page *n* (*PGn_E*) can be approximated by:
  PGn_E ~ [HTML_E] + [CP01_IC_E][CP01_F] + [CP02_IC_E]*[CP02_F] + … + [CPk_IC_E]*[CPk_F] + [RP01_IC_E]*[RP01_F] + [RP02_IC_E]*[RP02_F] + … + [RPj_IC_E]*[RPj_F] – [PGr_RF]*[PGr_E]*
  Where:
  - *HTML_E* = effort for writing HTML in hours;
  - *CPk_IC_E* = effort for inserting and configuring custom portlet *k*;
  - *CPk_F* = frequency of custom portlet *k* (the number of times it occurs in the page);
  - *RPj_IC_E* = effort for inserting and configuring reused portlet *j*;
  - *RPj_F* = frequency of reused portlet *j*;
  - *PGr_RF* = the reuse factor of page *n* regarding page *r*. For instance, page *n* is a near identical copy of page *r*, only small adaptations are needed, and all portlets in page *r* are used in page *n* with the same configurations. The reuse factor is then for instance 0.95, as there is no need for inserting and configuring these portlets another time. A unique page has a reuse factor of 0.

  This part of the estimation is probably more reliable than the custom part, and past observations should be more reliable for reused components. An example:

Page 01								
Based on:	N/A							
Reuse factor	0.00							
HTML effort	2.50							
Portlets	*CP01*	*CP02*	…	*CPk*	*RP01*	*RP02*	…	*RPj*
Frequency	0	1		0	3	2		0
Expected effort:	PG01_E							

Page 02								
Based on:	Page 01							
Reuse factor	0.95							
HTML effort	2.50							
Portlets	*CP01*	*CP02*	…	*CPk*	*RP01*	*RP02*	…	*RPj*
Frequency	0	1		0	3	2		0
Expected effort:	PG02_E							

…

Page *n*								
Based on:	N/A							
Reuse factor	0.00							
HTML effort	3.50							
Portlets	*CP01*	*CP02*	…	*CPk*	*RP01*	*RP02*	…	*RPj*
Frequency	1	2		1	1	4		0
Expected effort:	PGn_E							

In this example, page 02 is almost entirely based on page 01, and the reuse factor is approximately 0.95 percent of the effort needed for page 01. They both don't have custom portlet 01, one instance of custom portlet 02, … , three instances of reused portlet 01 (this could for instance be PortletSuite portlet News1), two instances of reused portlet 02, … , 0 instances of reused portlet *j*;

- All the effort in hours is divided by 152, to acquire a person-month as used in COCOMO. According to Boehm, this number seems to be a good measure for the number of hours in a person month and takes holidays and illness of the employee into account amongst other things. Of course, if the Componence management is convinced that this is not the case in their environment, another number can be used. Person months is rather used than hours because the EOSF (see next bullet) has more impact when effort is expressed in hours than person months;
- The economies of scale factor (EOSF) depends on the size of the project and/or team and the structure of the project. As stated before, a large team requires more overhead for coordination and therefore the management could consider setting the EOSF to for instance 1.1 or 1.05. For a project like the titlesite project, there are economies of scale as most sites are alike. In this case the management could consider setting the EOSF to for instance 0.9 or 0.8.

Some considerations with this example and effort estimation in general:

- It might be useful to make so called pX estimations. For instance, a p50 effort estimate is an estimate of such quality that the estimator believes there is a 50 percent chance of effort overrun, and therefore is also called the most likely effort. Because effort overrun is more likely than effort underrun, a confidence interval can be constructed (as proposed by Jorgensen) by for instance setting the maximum effort to 150 percent and the minimum to 90 percent of the p50 estimation;
- The size in this example is based on hours, this removes the necessity of converting size to effort (except for converting it to person months);
- The project process is constantly changing, improvements and/or changes in the process might influence the effort needed for activities, might cause activities to be dropped or added, etc. Therefore, the measured effort for these activities need to be examined and calibrated carefully from time to time;
- When specifications are incomplete and/or project managers easily allow for requirements creep, it might be useful to take this into account when estimating the expected effort. The analysis in chapter 6 suggests that incomplete

specifications lead to requirements creep (hypothesis 2) and this in turn leads to effort overrun (hypothesis 16);
- An experienced team or project manager might have less trouble working with Componence's portlets than an inexperienced team or manager, therefore some variable might be need to be tracked for each team and/or project manager, depending on the desired accuracy of the estimation. Since the management currently is mainly concerned about getting a rough but reasonable and more realistic estimation, tracking parameters like this might be best delayed for now, but this could be kept in mind when assigning projects to teams. Other factors like this are also called cost factors, for a more complete list, refer to Reifer (2000);
- Some teams are more costly per person-month than others, when this effort estimation is used for determining the costs this should be kept in mind;
- The scope of this report comprises the project process only, and this estimation comprises mostly the construction phase. Even though the construction phase is often the most costly and therefore it is important to have a good estimation for the construction phase, the other phases still require some effort. The effort for the phases other than the construction phase:
    - The effort for writing specifications, designs and project plans and other analysis (or inception and elaboration) related documents: the effort and time for writing specifications, designs and plans is not where the major problems lie and can be estimated as before. Of course, for large projects with not much reuse the effort for writing specifications and design is likely to be higher than for a simple project with many reuse. This effort could therefore be tracked for different types or sizes of projects, or the effort could be scaled for different sizes. One of the changes being discussed right now is having the specifications finished and fixed before starting a project, therefore the effort for writing the specifications will be less dispersed and more concentrated during the earlier phases when this change is taken into effect. Also, templates are now available for different documents, this could also reduce the amount of time needed to write but also read and understand them;
    - The effort needed in the guarantee period: currently each portal is assigned a guarantee period of two months, and the management should consider that for a relatively large and complex portal it is likely that more issues will be posted than for a small and simple portal in these two months. As seen in the previous chapter, when relatively many types of tests have been done it is less likely that the relative amount of issues that occurs after the live date is large. Therefore the extensiveness of testing and issue fixing during the construction phase influences the amount of resources needed in the guarantee period and the management should take this into account;
    - The management should also consider the effort and costs for hosting and support, the effort of Dutch management, etcetera.

In summary:
- Analogy based estimations have a foundation in actual observations and are therefore more reliable than gut-feeling based estimations and reproducable;
- For the inception phase: record the effort for writing specifications, designs and planning for different types/sizes of projects;
- For the elaboration phase: record the effort for setting up (or reusing/adapting) a skeleton;
- For the construction phase:
    - record the effort for implementing (or adapting), configuring, inserting and testing different types of pages and portlets;
    - Record the effort for performing tests (customer acceptation tests, functionality tests, performance tests, stress/load tests, etcetera);

  - Record the effort for resolving (different types of) issues, especially fixing bugs as this can be used to determine the rework rate;
  - Record the effort for the different (re)deployments;
- Use this data to calculate most likely effort in a model similar to the example;
- Keep this data up to date and carefully examine and calibrate the data from time to time;
- For the transition phase: record the effort needed during the guarantee period, hosting effort and support effort;
- When using this data for determining the costs, keep in mind that the costs are different for different types of resources.

7.5.4 Scaling effort to duration

According to Boehm's COCOMO model there is a square-root relationship between effort and duration, and according to Reifer's research there is either a square-root or cube-root relation, depending on the size. As seen in table 4 the value of *P2* in WebMO is 0.5 (square root) or 0.333 (cube root) depending on the size of the project. In another article of Boehm et al (2000), the classic schedule estimation rule of thumb also shows a cube root relation: calendar months = 3*(cube root(person months)). For example, when effort is estimated to be 27 person-months, the number of calendar months = 3*(cube root(27)) = 3*3 = 9 months. The average size of the team should of course be approximately effort/duration, in this example the average size of the team should be 27/9 = 3 persons.

In hypothesis 14 (chapter 6) there seems to be a similar (but not very clear) relationship between effort and duration for Componence's projects. However, because of the dominance of titlesite projects which are not representative for all Componence projects due to their small size, it is recommended to record effort and duration data for all future normal projects and determine the relation when more data is available for normal projects. This relation can then be used to estimate the time when effort is estimated. Figure 55 shows the square (higher line) and cube root (lower line) in relation with Componence's data. The constant *B* (in WebMO) was set to 1.5 (the lowest number out of the suggested ones from table 4). It is hard to say whether or not the projects under the lowest line were done in a timespan that was too short. It is recommended to add more non-titlesite observations in order to choose a constant *B* that fits better with Componence projects. Also, the observations in figure 55 can be used for analogy based duration estimation. When a project is expected to require 2000 effort hours, the duration estimate can be made with the aid of observations lying around 2000 effort hours.

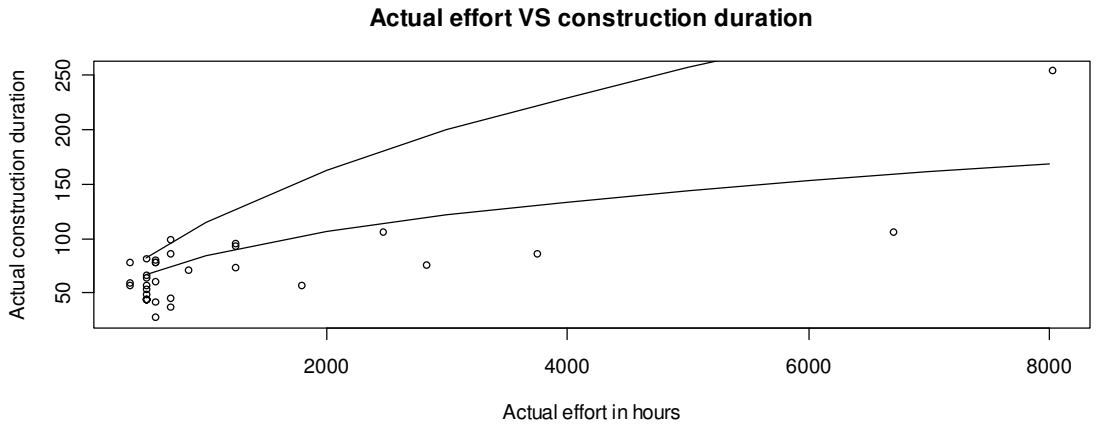


Figure 55

What else should be considered when scaling effort to duration?

- The project has been broken down in the WBS in different activities. What activities run in parallel? What activities precede other activities? The more activities can run in parallel the shorter the time could be in which those activities can be done, depending on the available resources of course. The activities that cannot run in parallel precede one another. Some activities might precede more than one activity. A delay in this preceding activity means that the following activities must wait. It is therefore recommended that these kinds of activities get more priority in order to minimize the risk of time overrun. When many of these kinds of activities are present, it might be better to make a looser planning in case one of these activities goes wrong and causes delay. An example is for instance waiting for the customer to accept the plans, specs and design. Time overrun due to this phenomenon leads to waiting time for Componence and vendor employees and managers, and this leads to time and cost overrun. Therefore, it might be useful to address this problem in future projects when estimating the duration;
- Of course, in a situation in which many activities can run in parallel, doing all those activities in parallel may not always be the best option. It has to be considered that doing many activities in parallel requires more resources, i.e. a larger team of people working together. This in turn requires more coordination effort. Another drawback is that the components worked on have to be fairly independent of each other;
- Cost drivers affect the duration: this means that team A could do a 100 person-month project in X months with a total amount of Q resources, but team B would need more time. Cost drivers could be for example: experience, how well the team works together, what tools they have, etcetera. As mentioned in the previous section, cost drivers could be best left out of scope for now, but could be kept in mind when estimating effort and duration;
- A relatively small amount of allowed resources and time can force the project team to work harder (with everything that goes with that, for instance less time for testing, more overwork, etcetera). Vice versa: a relatively large amount of resources and time can cause the project team to work more relaxed (and therefore a risk for sub-optimal productivity can exist). Therefore, in some cases it might be useful to make the planning a little tighter or looser, and replan during a project if necessary.

In summary, this method for determining the duration can be used for Componence projects. However, it is important that the relation is derived from more data from normal Componence projects, and not just use the square or cube root function from the estimated effort. Also, when using this relation for determining duration, the above mentioned points should be considered. Just as the effort estimation, this duration estimation is made from another perspective than the WBS estimation, and should be used to improve current estimations. Of course, as all other recorded data, keeping the data up to date and calibrating it should make it more reliable to use this method.

8 RESULTS

In this chapter an evaluation of the goals and results is given.

8.1 Goals

A quick overview of the goals of this report stated in chapter 1:

- A description of the current project process in flowcharts, its major problems and possible improvements;
- An analysis of what metrics could be used to measure the performance of these major problem areas;
- An analysis of the data of different Componence projects to gain more insight in different facets of the projects;
- Conclusions based on the analysis;
- Recommendations for what data to collect, how to collect the data in future projects and what metrics to track;
- Recommendations for improving effort and duration estimations.

8.2 Results

The description of the current project process in chapter and a model of the project process in appendix A helped in gaining insight in the project, and were used to point out problems. They can also be used for future partners and vendors for gaining more insight in Componence's project process. Lists of problems and improvements were made for all phases of the project. These lists can be found in chapter 4 and 5 and should give an overview of all major problems that currently exist in Componence's projects.

Chapter 6 shows a list for possible metrics. These metrics were collected for a set of projects and an analysis is done. Unfortunately, unleashing sophisticated statistical models on this data would not be useful because of the dependency of the data. The hypotheses in chapter 6 could therefore not be scientifically tested, but scatter- and boxplots were made and conclusions were carefully drawn from them. A summary of these conclusions can be found in section 6.5. The metrics were then evaluated and finally a recommendation was made for what metrics and data to track in the future for the different problem areas in section 6.7.

In chapter 7 the current quality and process for different estimations (commercial effort estimation (or bid), WBS effort estimation, and duration estimation) is analyzed. Existing models (COCOMO, SLIM, Function Point Analysis, WebMO) and theory on estimating effort and duration were discussed, including how these models and theory could help or be applied to Componence's estimating processes. An example of how an analogy based estimating model could be implemented is given.

9 CONCLUSIONS AND RECOMMENDATIONS

This report shows that many problems exist during Componence's projects. An analysis of the project process is done from an outside view by someone who was completely devoted to it, however, the time for the analysis and for writing this report was only six months. Therefore the first recommendation is to not blindly implement all suggested improvements in this report (summarized in this chapter) but to discuss the problems and improvements mentioned.

9.1 Most important conclusions

- The biggest problems currently in the project process are cost, effort and duration overrun together with performance issues of delivered portals. The causes for cost, effort and duration overrun lie mainly in the analysis phase (or RUP inception phase), namely too optimistic estimations and low quality or unclear specifications leading to requirements creep (among other things). It is likely that performance issues arise after delivery on production due to lack of (time and resources for) testing;
- It is likely that requirements creep is a major cause for effort overrun;
- The project price is almost always used as a constraint for the amount of allowed resources and time which may not always be justified;
- Almost all analyzed projects show effort overrun. In most projects the actual spent effort is between 20 to 40 percent higher than the WBS estimation;
- The actual durations for the analyzed projects show a wide spread;
- Projects are more likely to finish in time when not many issues are posted during construction;
- Data for monitoring and evaluating projects were difficult to find or derive, and were often inconsistent. Especially data for spent effort are scarce. Effort data collection has improved but is done only by SS. Componence and other offshore teams do not track the spent effort. The issue tracking systems provided a good deal of data, but for past projects often multiple issue tracking systems were used making it difficult or at least inconvenient. This is improved now, but in general project data and metrics were missing and not used for better control of and improving the process;
- One of the problems for the offshore teams is that the first wave of issues is often too large. The analysis in chapter 6 has shown that it is likely that when more types of tests have been done (especially when performance testing has been done) the first wave of issues will be relatively smaller;
- The number of pages and portlets alone are not a good measure for a project's size. The complexity and amount of reuse determine the actual size or required effort of a page or portlet. It is also hard to judge the quality of the specifications and designs based on number of covered pages and portlets. Also, requirements creep does not only show as increases in number of pages and/or portlets, but also as improvements.

9.2 Most important recommendations

- More effort and time should be reserved for testing during construction and better standard testing checklists should be made so that any performance issues and most bugs occur during construction and not after deployment on prod. At least performance testing and testing for memory leaks should become standard in future projects and performance benchmarks should be defined. In order to stimulate this, requirements for these tests should be stated before the project starts. However, more bugs during construction also increases the risk for effort overrun and therefore more effort should be assigned during the construction phase and/or more time should be reserved for the construction phase if this improvement is implemented. Also, performance testing may be expensive

because it should be done in the production environment as local or test environments may yield different performance levels;
- Reuse of both portal components and documents should decrease the amount of bugs and needed effort for providing support, planning and documenting. Also, estimating the effort for inserting reused components can be done based on analogy;
- Now that Componence has enough licenses for MS Project, this tool can be used for planning and tracking the progress. However, the offshore teams need to put more effort in updating these files for better progress tracking;
- More effort is needed for tracking data and metrics per project in order to gain more insight in projects and allow for better steering of projects and improvements in general. A recommendation for what data and metrics to track is given at the end of chapter 6. At least spent effort and actual direct costs should be tracked better. However, it has to be kept in mind that implementing a successful metrics program requires a serious amount of effort and perseverance. Also results may not show immediately making it even harder to maintain collecting data and metrics;
- In order to decrease the amount of effort and duration overrun, a decrease in requirements creep together with less optimistic effort estimations is needed. A decrease of effort overrun can be accomplished by for instance stricter change management and/or construction should start only when product specifications are clear to all involved parties and the scope is fixed. Better closure of the functional design and better product specification is needed in order to reduce requirements creep;
- Estimations (especially the bid) are currently done based on gut feelings. Future estimations should be reproducible and the estimator(s) should be accountable for the estimation made in order to increase the quality of the estimations. Bidding could be improved by for instance involving more people and stating whether or not profitability is a priority. Effort estimation could be improved by using for instance analogy based estimations. Duration estimation can be improved recording effort and duration data and deriving a relation from these data;
- Bidding and effort estimation should not be done by the same persons. At least the persons doing the effort estimation should know nothing about the project price;
- Making another effort estimation besides the WBS estimation based on analogy and portal size can improve the final estimation by for instance averaging these two estimations. An example of how this can be done is given in section 7.5;

9.3 Limitations of this research

- In order to keep the scope in line, this research is limited to the project process, which is only a part of Componence. Therefore, this report does not represent all problems and possible improvements by a long shot and might not even represent a complete list of problems for the project process;
- The organization and flowcharts represent the current situation which will likely change in the near future. Therefore, when reading these charts this should be kept in mind;
- The majority of the collected data and metrics originated from the titlesite project. The reason for doing this is that data for other projects were scarce and titlesite data was well documented. This however leaded to a higher degree of data dependency and bias towards very small projects, and therefore conclusions drawn from the analysis done in chapter 6 should not be accepted blindly. Correlations that show may actually not exist for future projects and there might be correlation in future projects where seemingly is no correlation right now;
- The effort data and specifications available are not detailed enough to actually put the proposed analogy based estimation model into practice.

9.4 Future research

- It is recommended that the current set of metrics will be evaluated and metrics will be added or even removed if they prove not to be useful after all. Also, if the management decides to start a metrics program, more data will become available and the hypotheses can be retested or new hypotheses can be tested. Especially when more accurate and detailed effort data is available and the root cause analysis is implemented interesting research can be done for further improvements;
- A model such as the proposed analogy based effort estimation model can be implemented and put to use when more effort data is available;
- Research for testing methods and tools;
- Besides research in the project process:
    - Research could be done on how to improve performance of portals since this has become a big issue. For instance, research can be done for optimal data transfer, what order components should be loaded, more efficient code, etcetera;
    - Search engine optimization (research on how search engines like Google search portals).

10 REFLECTION

What have I learned from this internship?

- I got to know and study a software company leading in its field. After this research I feel that I developed a good sense of project management and operational problems during software engineering;
- What it is like to work with offshore developers, I got to know and experience both the benefits and problems;
- I got to know and the software engineering methodology called "Rational Unified Process" or RUP;
- I got to know several models for and the theory behind effort and time estimation and used this knowledge to recommend improvements for Componence's estimations;
- I got to know what metrics are used during software engineering and introduced a set of metrics for Componence. I experienced that a great deal of effort is needed for implementing a metrics program;
- The scope of the research was much larger initially. I found that it is better to do a few things right than doing everything superficially.

What good was this research for the company?

- This report helped in initiating changes in the project process;
- The flow charts give a birds-eye view of the process which may give more insight to Componence managers and employees and (future) offshore teams;
- This report initiated discussions among managers;
- This report provides a collection of the problems encountered in projects and possible improvements;
- Through the metrics and data analysis the management gained more (quantitative) insight;
- Componence got to know what theory and models estimation experts developed for effort and duration estimation.

What could have been done better or different?

- The hypotheses, why are so many seem to be rejected?
    - Some of the metrics and data used for testing the hypotheses were not as good as they were thought to be. For instance, specification page coverage, portlet coverage and visual design coverage were found to be not very useful for quantifying the quality of the specifications and design. Better metrics for specifications and design quality may show correlation with other metrics and data where the metrics used in this report didn't. Therefore some hypotheses in this report could still hold even when the charts do not support them;
    - Because of the dependency of the data, it is not useful to apply statistical models on them. The hypotheses could therefore not be scientifically tested;
    - Because the project process is constantly changing, data from older projects may be obsolete and charts may therefore not be reliable for drawing conclusions;
    - Most of the data used in this report were taken from titlesites project. Therefore, it is not entirely justified to draw conclusions from this data and use these conclusions to "judge" the performance of the company. Also, the data may be biased towards small projects and this could also influence the charts made;
- Initially, a tool for effort estimation was to be made, but effort data was sparse and not detailed enough. Therefore only an example is given so that a tool can be made when more (detailed) data is available on spent effort;

- Because the project process is constantly changing and not all projects are the same, the flowcharts do not represent all projects but only the most common projects;
- It is often said that the quality of the specifications are low. An attempt at quantifying the quality of the specifications was made by introducing metrics like page and portlet coverage. However, after reconsideration and analysis, these metrics seemed not very good for this purpose. I have not succeeded in finding another metric that does represent the specification quality better.

11 APPENDIX A: ACTIVITY DIAGRAMS CURRENT PROJECT PROCESS

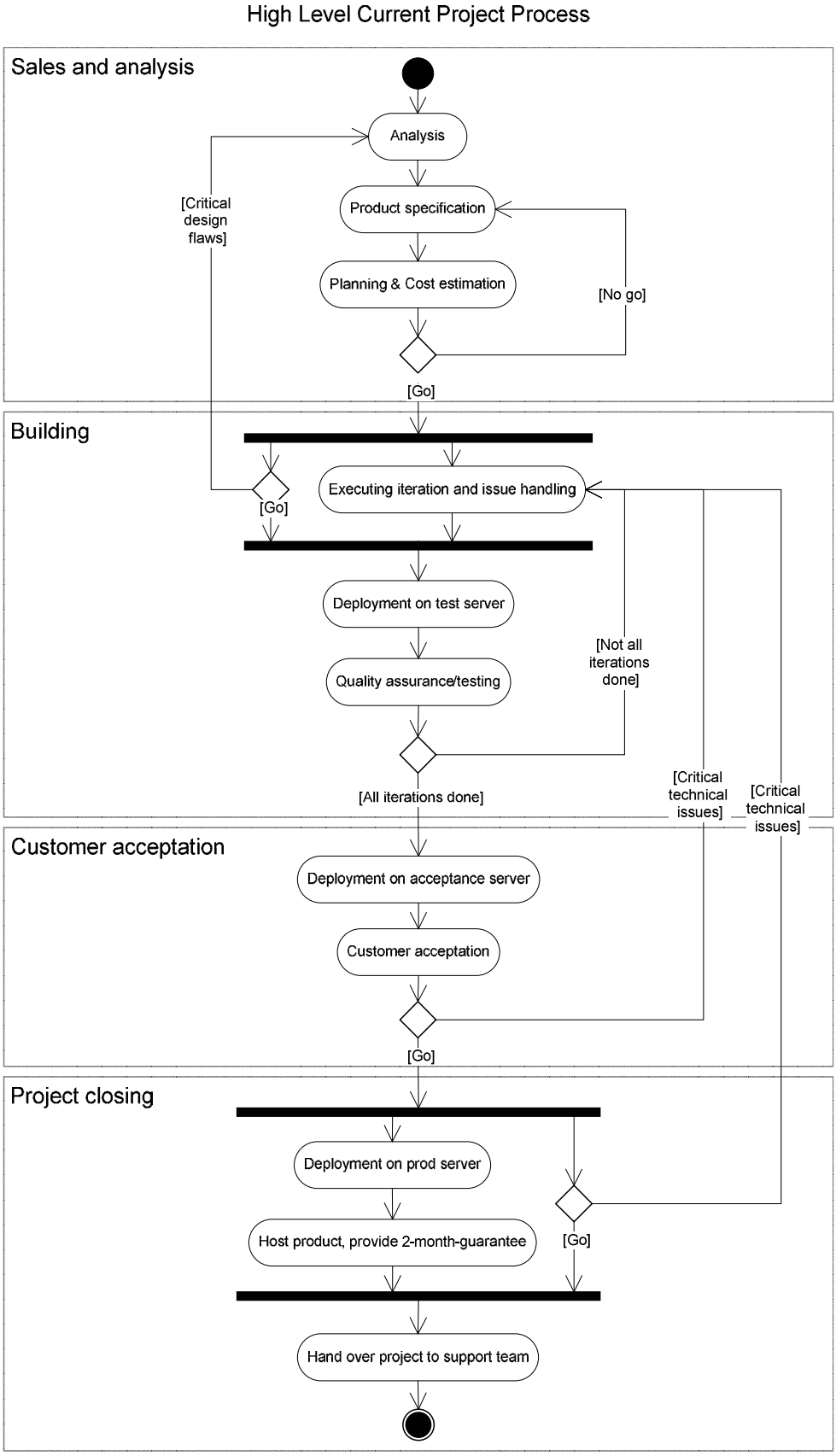

High Level Current Project Process
Sales and analysis
Analysis
Product specification
Planning & Cost estimation
[Critical design flaws]
[No go]
[Go]
Building
[Go]
Executing iteration and issue handling
Deployment on test server
Quality assurance/testing
[Not all iterations done]
[All iterations done]
[Critical technical issues]
[Critical technical issues]
Customer acceptation
Deployment on acceptance server
Customer acceptation
[Go]
Project closing
Deployment on prod server
Host product, provide 2-month-guarantee
[Go]
Hand over project to support team

Figure 56

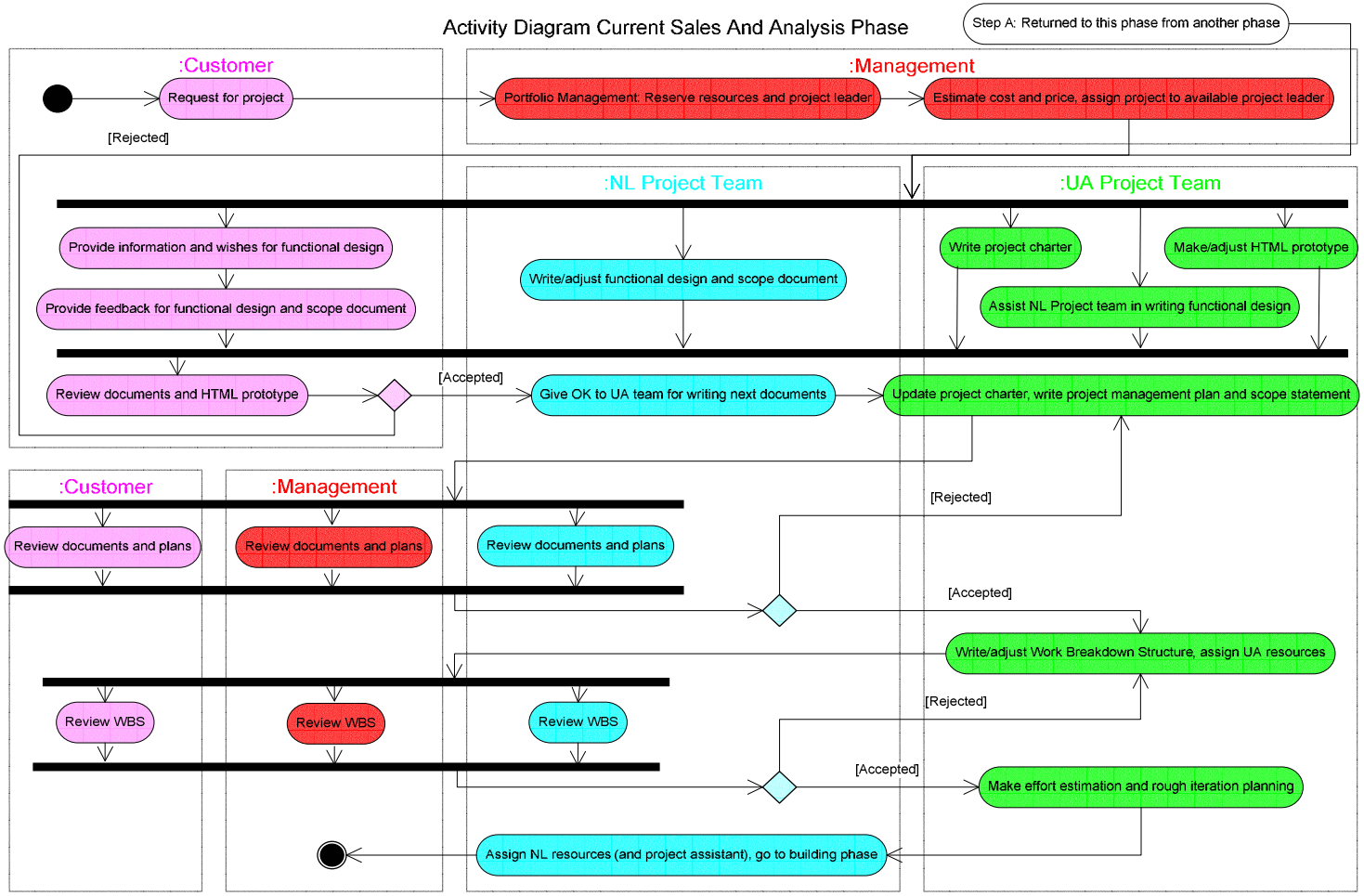

Activity Diagram Current Sales And Analysis Phase
Step A: Returned to this phase from another phase
:Customer
:Management
Request for project
Portfolio Management: Reserve resources and project leader
Estimate cost and price, assign project to available project leader
[Rejected]
:NL Project Team
:UA Project Team
Provide information and wishes for functional design
Write/adjust functional design and scope document
Write project charter
Make/adjust HTML prototype
Provide feedback for functional design and scope document
Assist NL Project team in writing functional design
Review documents and HTML prototype
[Accepted]
Give OK to UA team for writing next documents
Update project charter, write project management plan and scope statement
:Customer
:Management
[Rejected]
Review documents and plans
Review documents and plans
Review documents and plans
[Accepted]
Write/adjust Work Breakdown Structure, assign UA resources
[Rejected]
Review WBS
Review WBS
Review WBS
[Accepted]
Make effort estimation and rough iteration planning
Assign NL resources (and project assistant), go to building phase

Figure 57

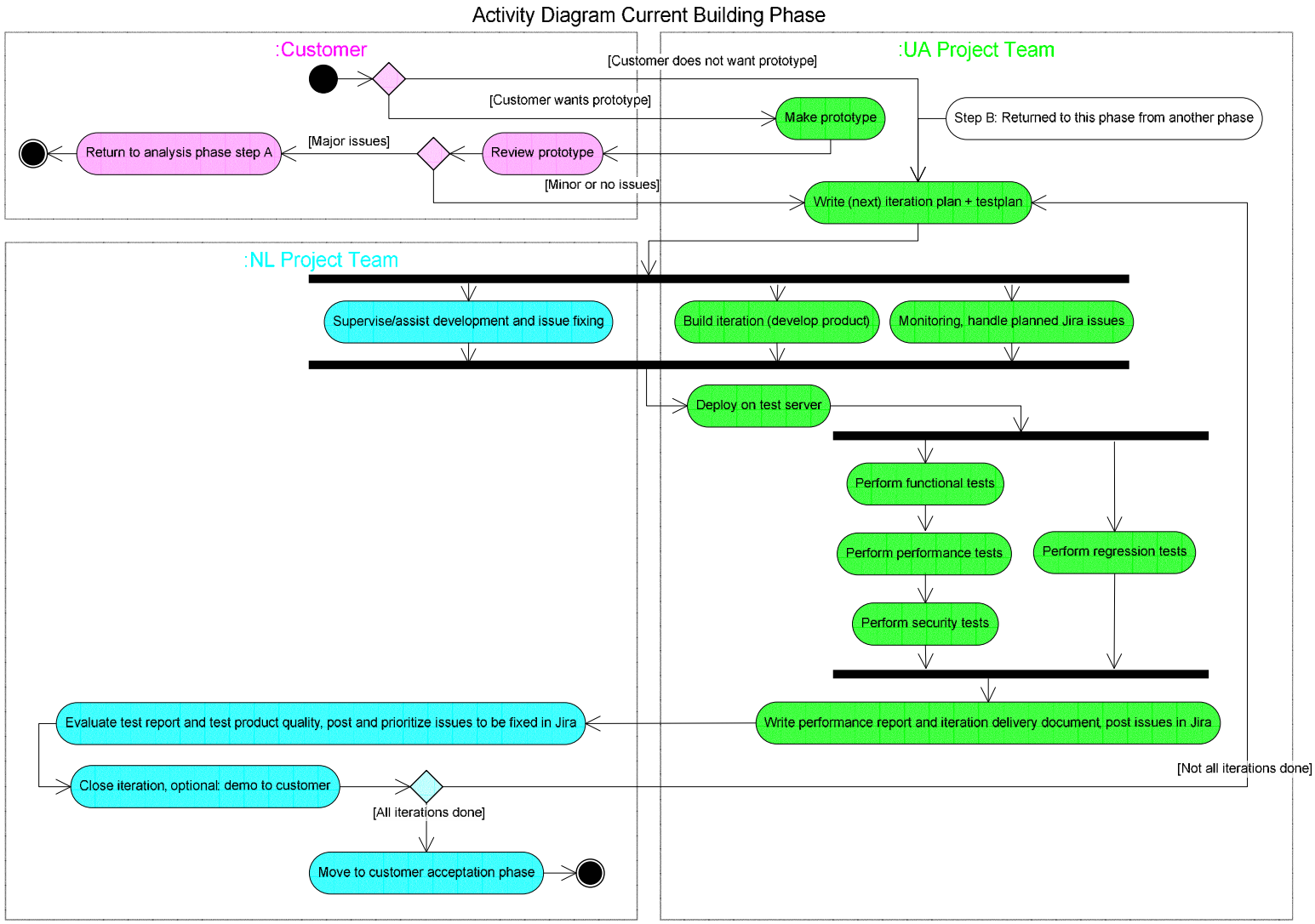

Activity Diagram Current Building Phase
:Customer
:UA Project Team
:NL Project Team
[Customer does not want prototype]
[Customer wants prototype]
Make prototype
Step B: Returned to this phase from another phase
Return to analysis phase step A
[Major issues]
Review prototype
[Minor or no issues]
Write (next) iteration plan + testplan
Supervise/assist development and issue fixing
Build iteration (develop product)
Monitoring, handle planned Jira issues
Deploy on test server
Perform functional tests
Perform performance tests
Perform regression tests
Perform security tests
Write performance report and iteration delivery document, post issues in Jira
Evaluate test report and test product quality, post and prioritize issues to be fixed in Jira
Close iteration, optional: demo to customer
[All iterations done]
[Not all iterations done]
Move to customer acceptation phase

Figure 58

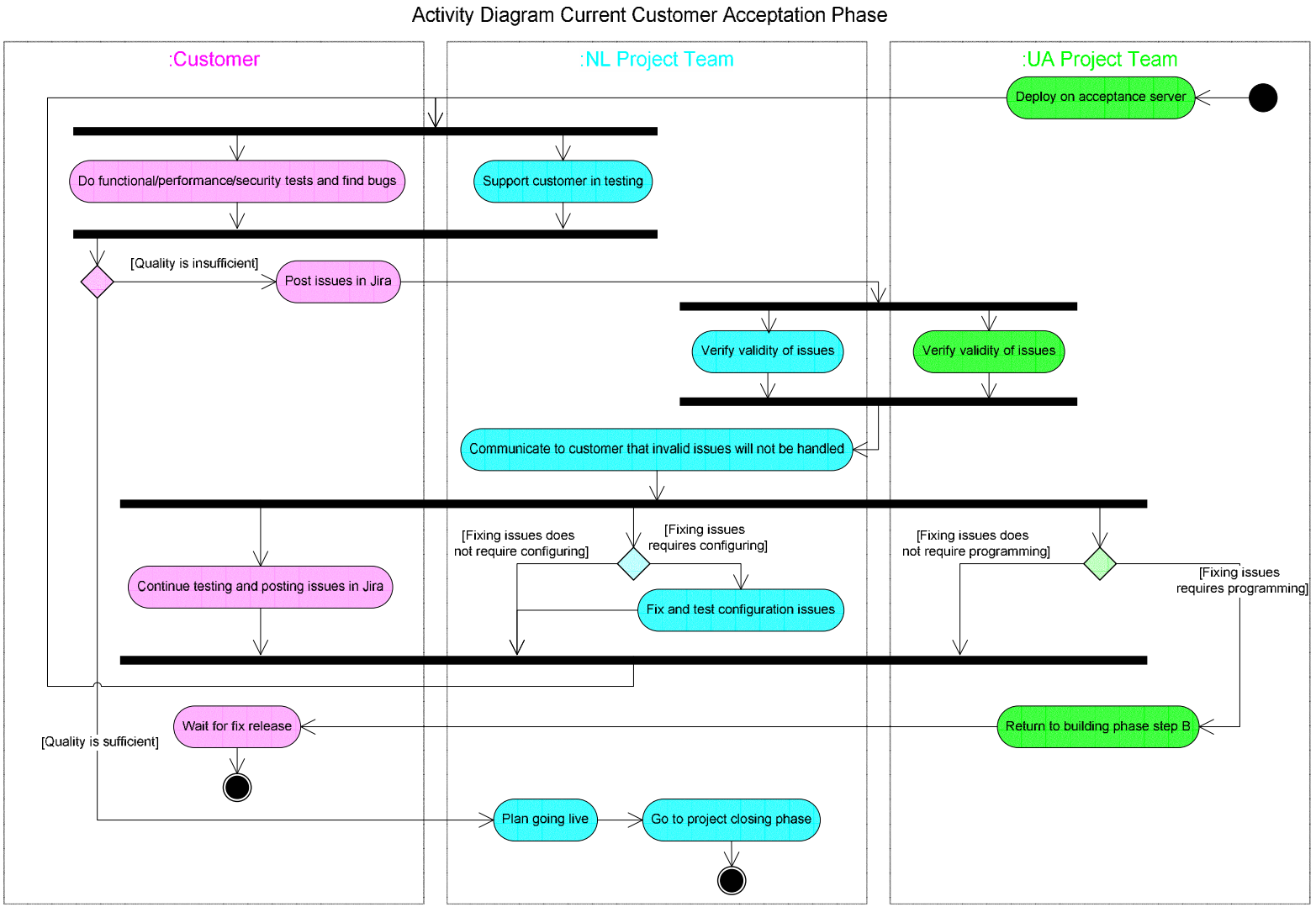

Activity Diagram Current Customer Acceptation Phase
:Customer
:NL Project Team
:UA Project Team
Deploy on acceptance server
Do functional/performance/security tests and find bugs
Support customer in testing
[Quality is insufficient]
Post issues in Jira
Verify validity of issues
Verify validity of issues
Communicate to customer that invalid issues will not be handled
[Fixing issues does not require configuring]
[Fixing issues requires configuring]
[Fixing issues does not require programming]
[Fixing issues requires programming]
Continue testing and posting issues in Jira
Fix and test configuration issues
[Quality is sufficient]
Wait for fix release
Return to building phase step B
Plan going live
Go to project closing phase

Figure 59

Activity Diagram Current Project Closing Phase

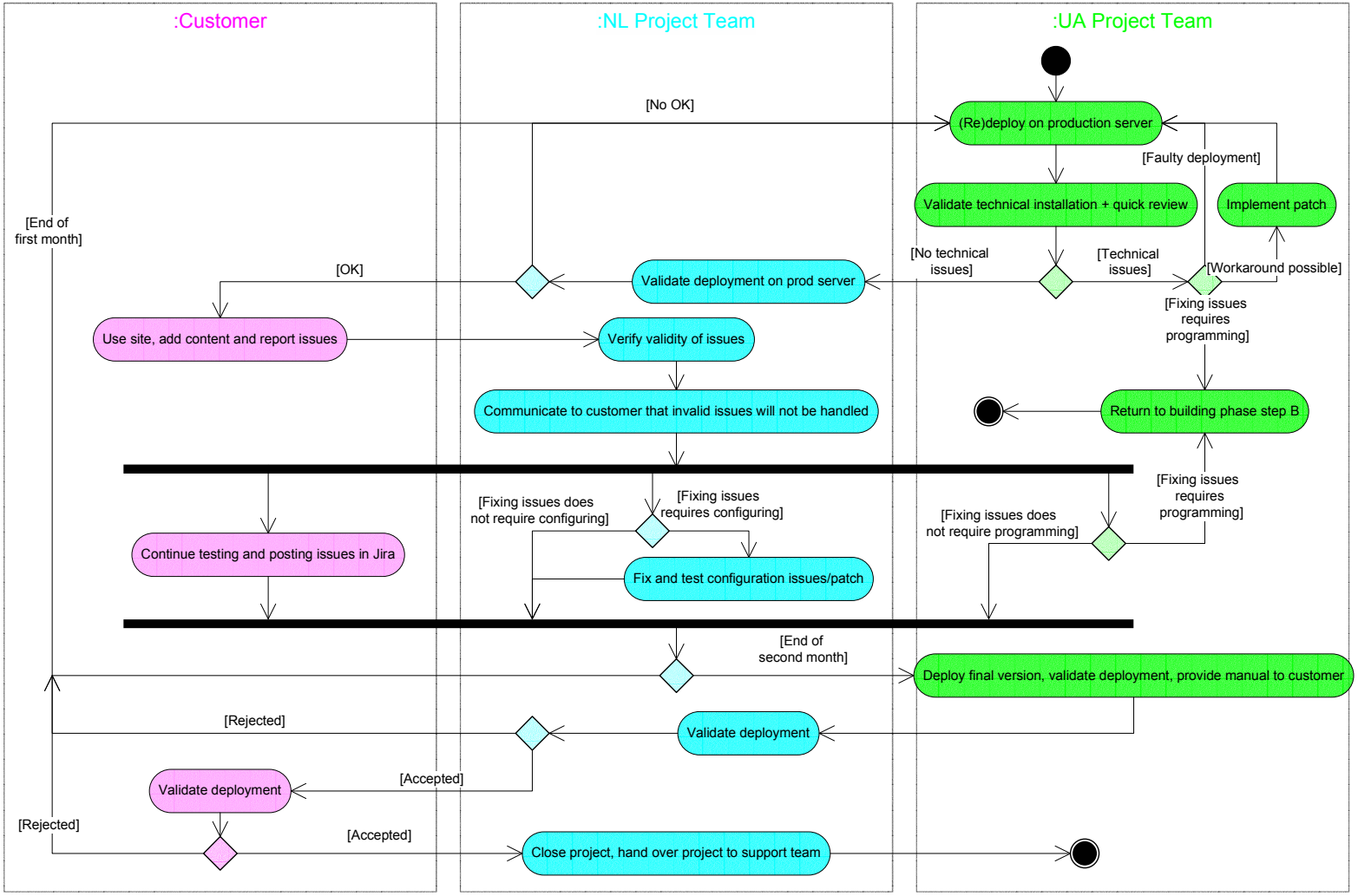


Figure 60

12 APPENDIX B: ISSUE CHARTS

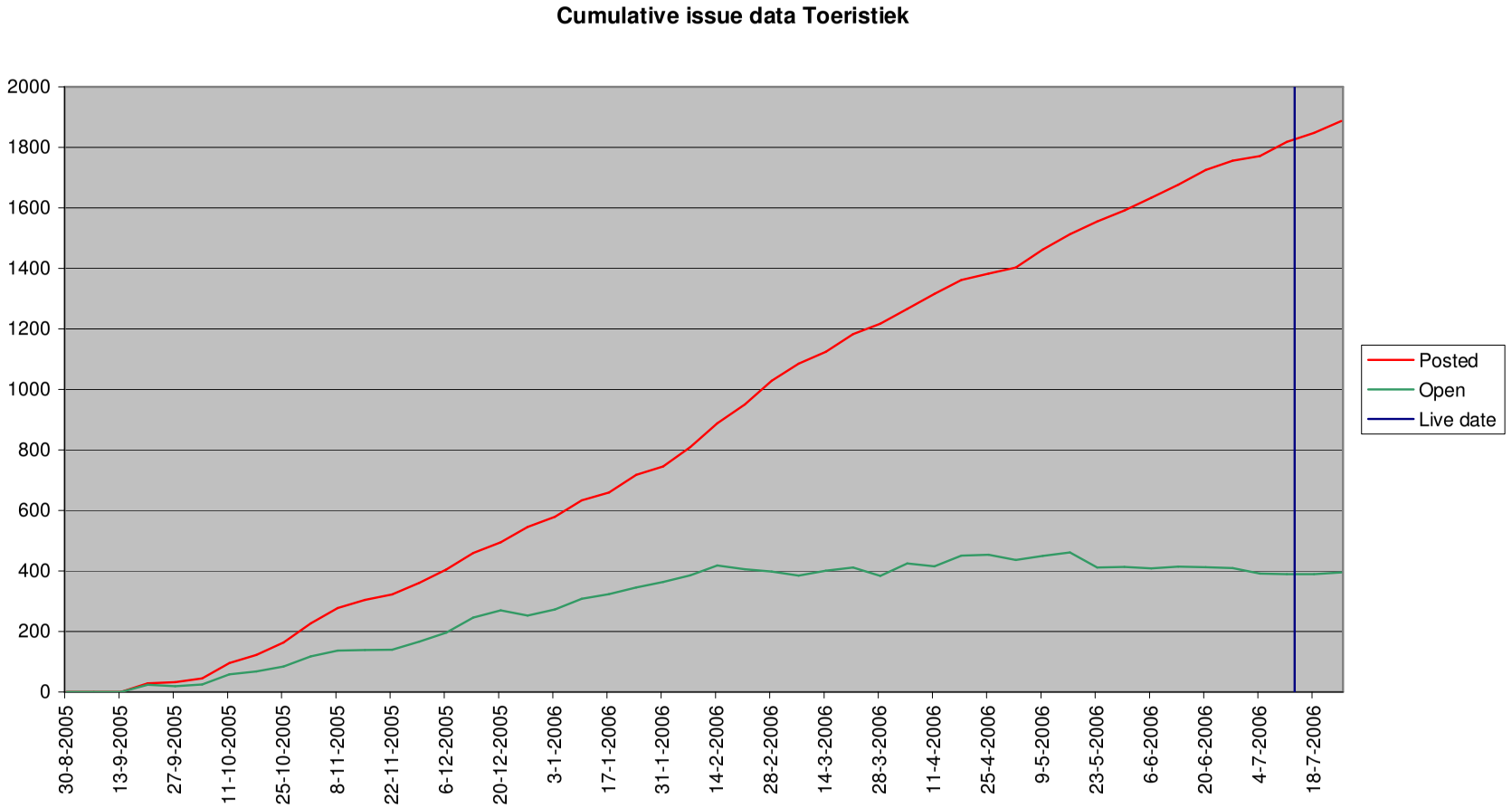

Cumulative issue data Toeristiek
2000
1800
1600
1400
1200
1000
800
600
400
200
0
30-8-2005
13-9-2005
27-9-2005
11-10-2005
25-10-2005
8-11-2005
22-11-2005
6-12-2005
20-12-2005
3-1-2006
17-1-2006
31-1-2006
14-2-2006
28-2-2006
14-3-2006
28-3-2006
11-4-2006
25-4-2006
9-5-2006
23-5-2006
6-6-2006
20-6-2006
4-7-2006
18-7-2006
Posted
Open
Live date

Figure 61

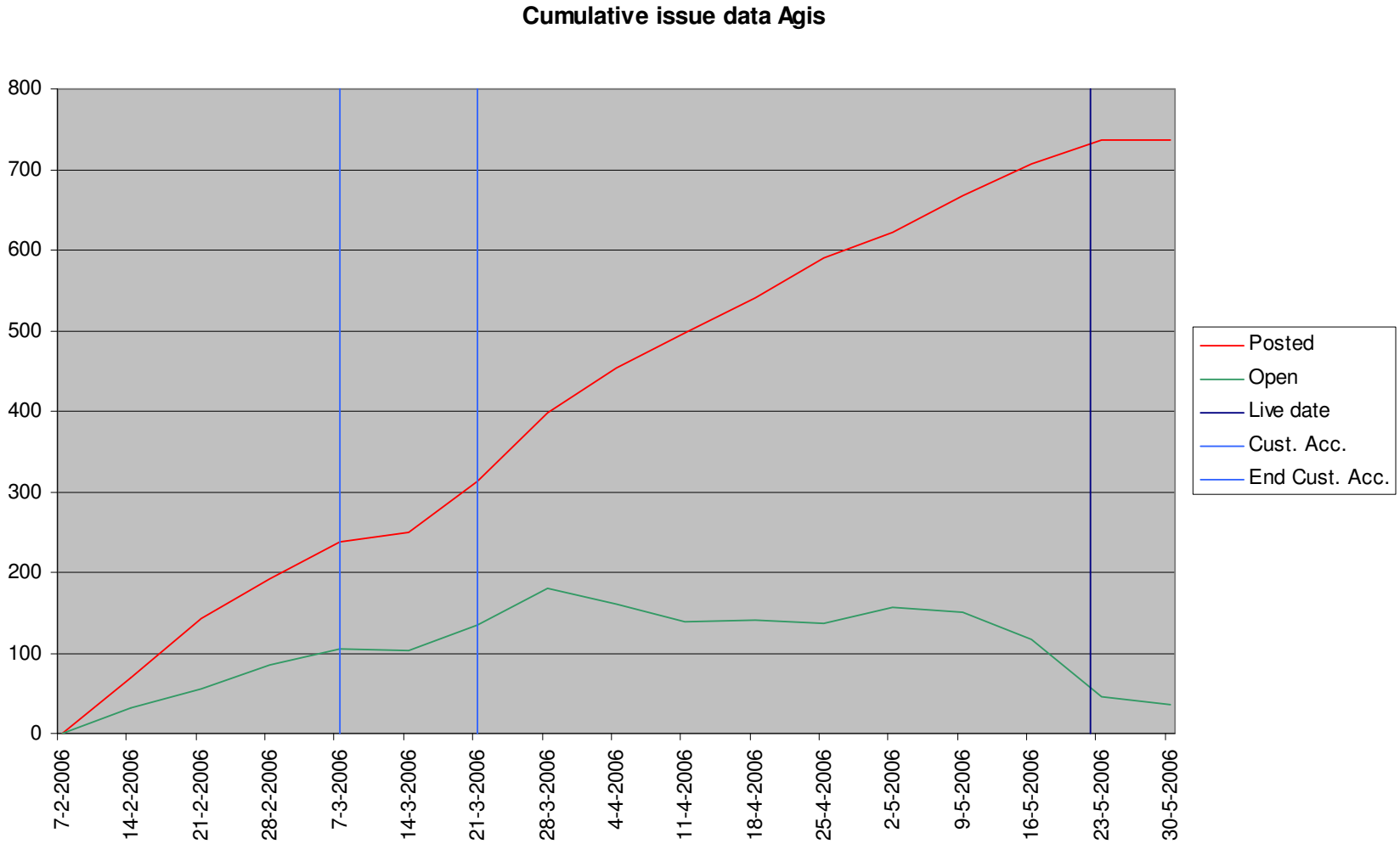

Cumulative issue data Agis
800
700
600
500
400
300
200
100
0
7-2-2006
14-2-2006
21-2-2006
28-2-2006
7-3-2006
14-3-2006
21-3-2006
28-3-2006
4-4-2006
11-4-2006
18-4-2006
25-4-2006
2-5-2006
9-5-2006
16-5-2006
23-5-2006
30-5-2006
Posted
Open
Live date
Cust. Acc.
End Cust. Acc.

Figure 62

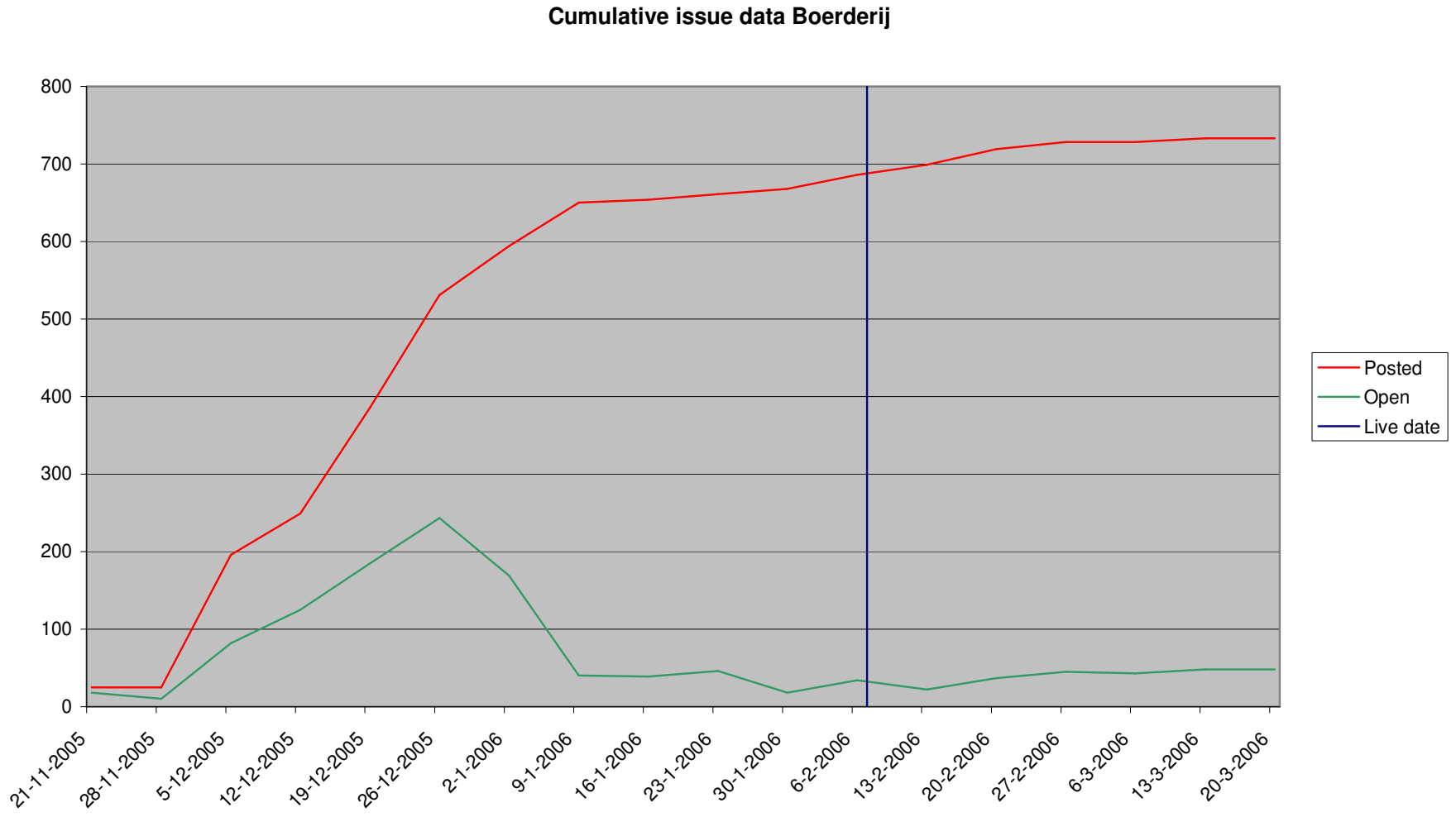

Cumulative issue data Boerderij
800
700
600
500
400
300
200
100
0
21-11-2005
28-11-2005
5-12-2005
12-12-2005
19-12-2005
26-12-2005
2-1-2006
9-1-2006
16-1-2006
23-1-2006
30-1-2006
6-2-2006
13-2-2006
20-2-2006
27-2-2006
6-3-2006
13-3-2006
20-3-2006
Posted
Open
Live date

Figure 63

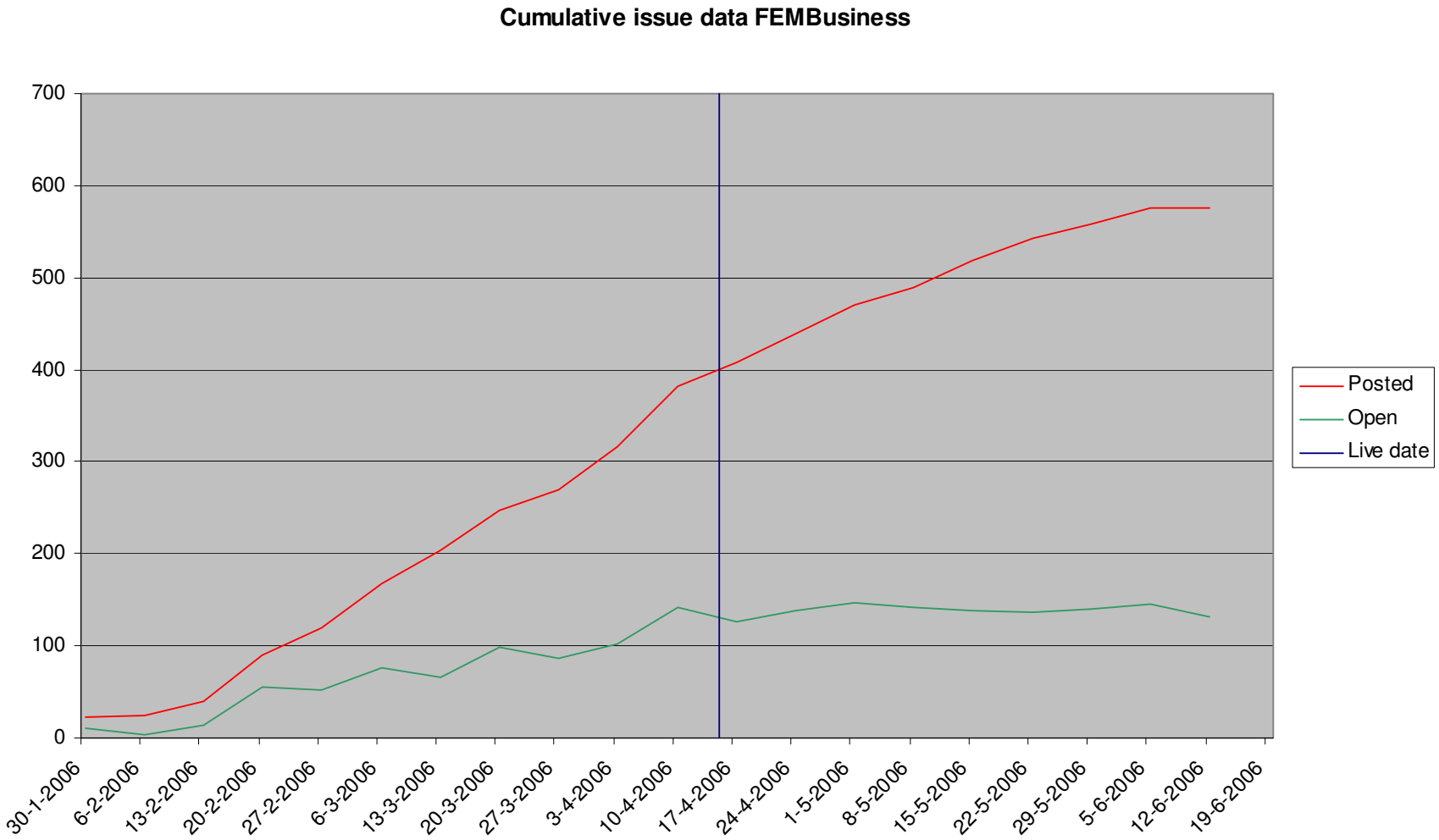

Cumulative issue data FEMBusiness
700
600
500
400
300
200
100
0
30-1-2006
6-2-2006
13-2-2006
20-2-2006
27-2-2006
6-3-2006
13-3-2006
20-3-2006
27-3-2006
3-4-2006
10-4-2006
17-4-2006
24-4-2006
1-5-2006
8-5-2006
15-5-2006
22-5-2006
29-5-2006
5-6-2006
12-6-2006
19-6-2006
Posted
Open
Live date

Figure 64

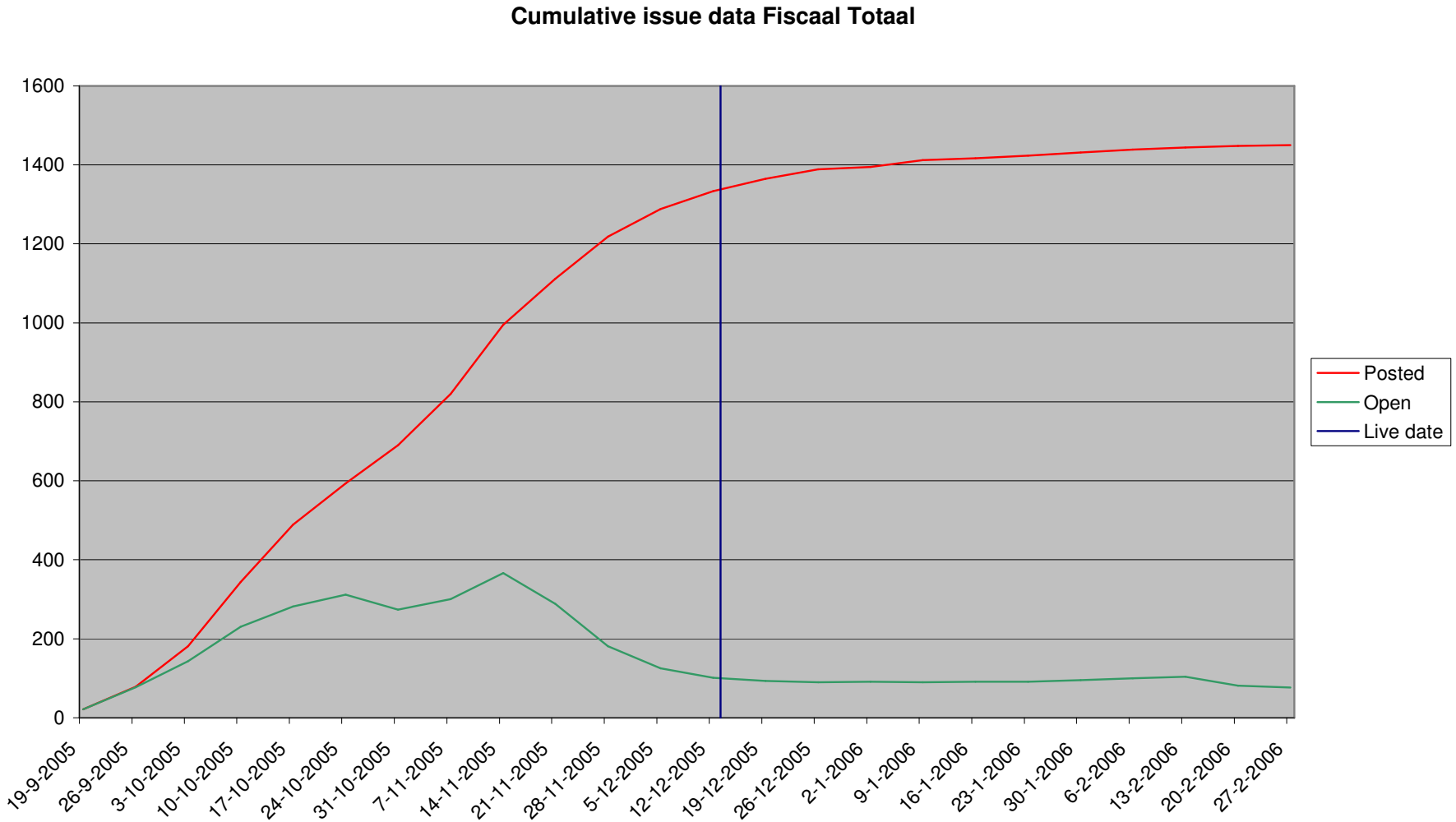

Cumulative issue data Fiscaal Totaal
1600
1400
1200
1000
800
600
400
200
0
19-9-2005
26-9-2005
3-10-2005
10-10-2005
17-10-2005
24-10-2005
31-10-2005
7-11-2005
14-11-2005
21-11-2005
28-11-2005
5-12-2005
12-12-2005
19-12-2005
26-12-2005
2-1-2006
9-1-2006
16-1-2006
23-1-2006
30-1-2006
6-2-2006
13-2-2006
20-2-2006
27-2-2006
Posted
Open
Live date

Figure 65

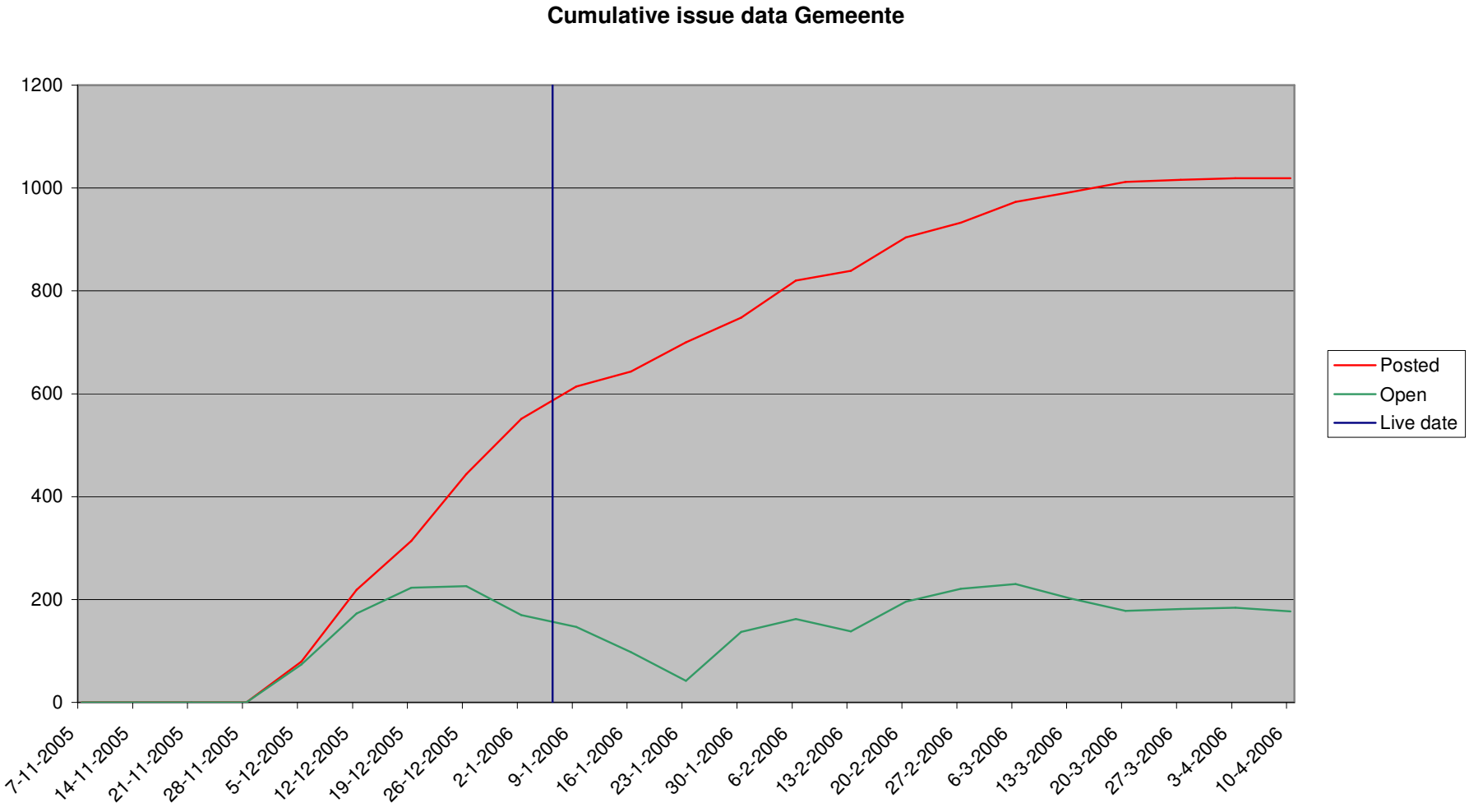

Cumulative issue data Gemeente
1200
1000
800
600
400
200
0
7-11-2005
14-11-2005
21-11-2005
28-11-2005
5-12-2005
12-12-2005
19-12-2005
26-12-2005
2-1-2006
9-1-2006
16-1-2006
23-1-2006
30-1-2006
6-2-2006
13-2-2006
20-2-2006
27-2-2006
6-3-2006
13-3-2006
20-3-2006
27-3-2006
3-4-2006
10-4-2006
Posted
Open
Live date

Figure 66

13 APPENDIX C: DATA AND R COMMANDS

The scatterplots and boxplots were made using the free statistical software tool R (http://www.r-project.org/). The commands for reproducing the plots in R are:

```
===READ IN DATA:===

r01<-
c("Toe","Agi","Boe","FEM","FT","Gem","PF","ZW","PO","VO","SB","Biz","WP","DBK","WGF","VVB","BeB","MKB","
Dis","AD","PVH","ZV","Sal","PeO","AGF","PP","VW","TTM","Nur","TeL","OM","FZ","LL","Orn","FM","RO","OO","MT
","CM","RR","ICT")
r02<-
c("PS","PS","PS","PS","PS","PS","PS","TS","TS","TS","TS","TS","TS","TS","TS","TS","TS","TS","TS","TS","TS","TS","T
S","TS","TS","TS","TS","TS","TS","TS","TS","TS","TS","TS","TS","TS","TS","TS","TS","TS","TS","TS","TS","TS")
r03<-
c("Bas","Roan","Tom","Roan","Roan","Ha","Roan","Ha","Ha","Ha","Ha","Tom","Tom","Tom","Tom","Tom","Tom"
,"Tom","Tom","Tom","Tom","Tom","Tom","Tom","Tom","Tom","Tom","Tom","Tom","Tom","Tom","Tom","Tom","Tom","
Tom","Tom","Tom","Tom","Tom","Tom","Tom","Tom")
r04<-c("SS","SS","SS","Lviv","SS","Kiev + SS","Lviv","Kiev + SS","Kiev + SS","Kiev + SS","Kiev +
SS","PL1","Kiev1","Lviv","RM1","Lviv","RM1","Kiev1","RM1","RM1","Kiev1","PL2","PL2","RM2","RM2","Kiev1","Ki
ev1","PL1","RU","Lviv","RU","PL2","RU","Lviv","RM1","PL2","RM1","Kiev2","RM","RM","Kiev2")
r05<-
c("AL","R","R","AL","ATa","R/S","AL","NA","NA","NA","NA","MM","S","V","NA","V","NA","S","CC","CC","S","NA","
NA","NA","NA","S","S","S","NA","V","NA","NA","NA","NA","NA","NA","NA","NA","NA","CC","VK")
r06<-c(CONFIDENTIAL)
r07<-c(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1)
r08<-c(1,0,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1)
r09<-c(1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
r10<-c(CONFIDENTIAL)
r11<-
c(30,36,46,21,43,51,12,19,23,23,17,25,18,30,28,31,37,53,24,32,28,28,23,75,28,20,29,15,47,34,38,21,22,34,
34,23,19,24,28,30,33)
r12<-c(28,12,24,20,27,22,6,5,2,2,3,4,9,5,11,7,8,5,10,7,3,6,9,5,6,9,6,5,11,6,6,4,9,9,9,9,10,11,11,9,9)
r13<-
c(61,34,19,20,34,18,5,14,12,11,11,11,12,11,15,14,17,13,14,18,15,14,21,20,11,13,11,8,14,8,12,8,14,9,11,15,
15,14,15,12,11)
r14<-
c(61,24,17,20,34,13,5,14,12,11,11,11,12,11,15,14,17,13,14,18,15,14,21,20,11,13,11,8,14,8,12,8,14,9,11,15,
15,14,14,12,11)
r15<-c(61,5,5,7,15,2,3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0)
r16<-c(1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
r17<-c(1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
r18<-c(30,10,16,17,32,26,6,6,3,3,3,20,11,1,1,1,3,4,1,1,1,1,1,3,1,1,3,1,1,1,1,1,3,1,1,1,1,1,1,1,1)
r19<-
c(2300,1600,790,2000,2800,720,2000,400,400,400,400,400,400,400,400,400,400,400,400,400,400,400,400,400,
400,400,400,400,400,400,400,400,400,400,400,400,400,400,400,400,400,400,400,400,400,400,400,400,400)
r20<-
c(63,56,21,0,0,3,46,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,
21,21,21,21,21,21)
r21<-
c(43,19,35,77,0,14,46,21,21,21,21,18,25,21,21,21,21,18,31,26,26,33,27,27,18,18,32,20,19,18,24,20,13,20,2
0,18,21,19,18,19,18)
r22<-
c(92,60,45,63,73,56,74,42,42,42,42,50,45,50,53,50,39,59,42,49,49,43,43,42,50,42,58,80,59,42,51,46,93,60,
60,53,60,51,NA,49,51)
r23<-c(0,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
r24<-
c(8028,6712,NA,2841,3760,1800,2478,515,515,515,515,587,720,1254,515,1254,515,720,515,515,720,587,5
87,515,515,720,720,587,380,1254,380,587,380,860,515,587,NA,NA,NA,515,NA)
r25<-
c(254,105,57,75,85,56,105,43,43,43,43,27,45,95,48,93,66,36,43,56,85,60,41,43,81,NA,98,77,58,73,56,77,77
,70,63,80,42,65,NA,53,NA)
r26<-
c(1832,740,644,404,1343,643,492,46,71,10,64,201,126,290,116,261,69,151,112,65,79,82,59,83,117,NA,148
,38,167,223,66,72,75,131,49,51,41,116,NA,86,NA)
r27<-
c(1422,673,585,330,1049,476,324,34,63,10,56,187,106,287,116,254,66,131,108,50,71,78,56,78,116,NA,146
,33,164,216,65,70,72,129,49,50,40,109,NA,86,NA)
```

```
r28<-
c(427,143,194,69,246,84,107,36,37,1,32,100,87,125,86,114,35,40,44,37,43,49,36,24,63,NA,71,26,141,99,16
,25,31,36,34,10,24,60,NA,43,NA)
r29<-
c(17,56,72,0,38,23,3,6,24,7,31,22,20,27,12,11,34,14,3,25,29,16,8,31,11,NA,19,10,3,1,26,15,44,6,1,20,17,48
,NA,43,NA)
r30<-
c(1405,541,389,335,1059,536,382,4,9,2,1,76,19,138,18,136,0,97,65,3,7,17,15,28,43,NA,58,2,23,123,24,32,1
,89,14,21,0,8,NA,0,NA)
r31<-c(410,67,59,74,294,167,168,12,8,0,8,14,20,3,0,7,3,20,4,15,8,4,3,5,1,0,2,5,3,7,1,2,3,2,0,1,1,7,0,0,0)
r32<-c(287,43,39,63,255,36,102,2,0,0,6,7,0,2,0,0,1,7,1,7,6,4,0,2,1,NA,1,4,1,2,1,2,1,0,0,1,0,7,NA,0,NA)
r33<-
c(NA,761,728,518,1412,820,531,72,117,32,112,236,182,299,130,269,73,163,140,73,93,87,97,92,125,NA,154
,38,194,227,74,72,NA,NA,49,NA,41,129,NA,87,NA)
r34<-
c(NA,NA,45,23,43,49,12,20,23,23,17,26,18,30,27,28,39,53,27,30,NA,28,26,74,28,NA,NA,17,46,34,38,21,NA,N
A,NA,NA,NA,NA,NA,30,NA)
r35<-
c(NA,NA,25,19,34,21,6,14,12,11,11,11,11,11,14,14,17,13,14,18,15,14,21,20,11,13,11,8,14,8,12,8,14,9,11,15
,15,14,15,12,11)
r36<-c(NA,NA,5,6,15,19,4,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,3,0,0,0,0,0,0,0,0,0,0,0)
r37<-
c(NA,25,97,172,120,389,39,32,46,22,50,41,113,17,52,20,12,45,30,27,26,31,71,14,9,NA,NA,0,56,10,23,1,NA,
NA,NA,NA,NA,NA,NA,30,NA)
r38<-
c(NA,76,97,92,319,171,103,21,15,3,20,28,29,8,3,8,3,24,7,24,8,9,4,6,2,NA,NA,5,11,12,2,2,NA,NA,NA,NA,NA,N
A,NA,19,NA)
r39<-
c(NA,59,46,79,265,64,103,10,7,0,16,23,19,2,1,3,1,16,4,14,7,7,2,2,1,NA,NA,4,10,5,1,2,NA,NA,NA,NA,NA,NA,N
A,8,NA)
r40<-
c(NA,158,244,161,283,130,83,52,62,3,57,129,172,137,132,130,45,72,67,59,59,73,86,36,70,NA,NA,26,183,11
0,28,26,NA,NA,NA,NA,NA,NA,NA,71,NA)
r41<-
c(NA,59,91,1,44,91,3,21,43,26,54,25,33,29,14,11,34,21,6,28,30,19,19,31,11,NA,NA,10,5,1,30,15,NA,NA,NA,
NA,NA,NA,NA,44,NA)
r42<-
c(NA,548,406,414,1136,811,445,5,12,3,3,85,34,141,22,140,2,103,69,5,16,21,25,30,45,NA,NA,2,35,122,31,32
,NA,NA,NA,NA,NA,NA,NA,1,NA)
r43<-c(1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
r44<-c(1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
r45<-
c(NA,765,741,576,1463,1032,531,78,117,32,114,242,239,307,168,281,81,196,142,92,105,113,130,97,126,N
A,NA,38,223,233,89,73,NA,NA,NA,NA,NA,NA,NA,116,NA)
r46<-
c(NA,689,644,484,1144,861,428,57,102,29,94,214,210,299,165,273,78,172,135,68,97,104,126,91,124,NA,NA
,33,212,221,87,71,NA,NA,NA,NA,NA,NA,NA,97,NA)
r47<-c(3,1,1,3,1,2,3,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2)
r48<-c(7,1,1,7,1,4,7,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4)
r49<-
c(1.23,2.94,1.32,1.73,2.10,1.08,2.05,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,
2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77,2.77)
r50<-
c(6161,6168,NA,1685,2427,1133,1500,371,371,371,371,443,576,1110,371,1110,371,576,371,371,576,443,4
43,371,371,576,576,443,236,1110,236,443,236,716,371,443,NA,NA,NA,371,NA)
r51<-
c(4.30,12.33,NA,2.46,2.82,2.70,2.53,3.57,3.57,3.57,3.57,4.06,4.98,8.68,3.57,8.68,3.57,4.98,3.57,3.57,4.98,
4.06,4.06,3.57,3.57,4.98,4.98,4.06,2.63,8.68,2.63,4.06,2.63,5.95,3.57,4.06,NA,NA,NA,3.57,NA)
r52<-
c(3.49,4.20,NA,1.42,1.34,2.50,1.24,1.29,1.29,1.29,1.29,1.47,1.80,3.14,1.29,3.14,1.29,1.80,1.29,1.29,1.80,1.
47,1.47,1.29,1.29,1.80,1.80,1.47,0.95,3.14,0.95,1.47,0.95,2.15,1.29,1.47,NA,NA,NA,1.29,NA)
r53<-
c(103,118,63,140,73,63,120,63,63,63,63,63,63,63,63,63,63,63,63,63,63,63,63,63,63,63,63,63,63,63,63,63,6
3,63,63,63,63,63,63,63,63)
r54<-
c(317,167,78,75,85,59,147,64,64,64,64,71,70,116,77,116,87,79,64,77,107,81,70,64,102,NA,119,112,81,95,7
7,98,141,95,88,105,88,86,NA,74,NA)
r55<-
c(20,27,21,77,7,21,42,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,21,2
1,21,21,21,21,21,21)
```

```
r56<-
c(43,25,35,77,0,14,42,21,21,21,21,41,29,21,29,23,21,40,31,26,27,33,35,27,18,22,32,34,21,19,24,20,56,24,2
4,22,46,19,98,19,22)
r57<-
c(83,91,42,63,66,42,78,42,42,42,42,42,42,42,42,42,42,42,42,42,42,42,42,42,42,42,42,42,42,42,42,42,42,
42,42,42,42,42,42,42)
r58<-
c(0.19,0.23,0.33,0.55,0.10,0.33,0.35,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,
0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33,0.33)
r59<-
c(2.76,1.75,1.27,1.19,1.16,1.00,1.42,1.02,1.02,1.02,1.02,0.54,1.00,1.90,0.91,1.86,1.69,0.61,1.02,1.14,1.73,
1.40,0.95,1.02,1.62,NA,1.69,0.96,0.98,1.74,1.10,1.67,0.83,1.17,1.05,1.51,0.70,1.27,NA,1.08,NA)
r60<-
c(3.08,1.42,1.24,0.54,1.16,0.94,1.23,1.02,1.02,1.02,1.02,1.13,1.11,1.84,1.22,1.84,1.38,1.25,1.02,1.22,1.70,
1.29,1.11,1.02,1.62,NA,1.89,1.78,1.29,1.51,1.22,1.56,2.24,1.51,1.40,1.67,1.40,1.37,NA,1.17,NA)
r61<-
c(NA,765,741,576,1463,1032,531,78,117,32,114,242,239,307,168,281,81,196,142,92,105,113,130,97,126,N
A,NA,38,223,233,89,73,NA,NA,NA,NA,NA,NA,NA,116,NA)
r62<-
c(7.21,7.05,11.30,15.80,11.48,4.69,1.65,0.23,1.49,7.44,2.80,3.05,2.42,2.81,1.05,4.19,2.60,1.16,0.93,1.37,1.
44,1.93,1.44,NA,1.51,0.49,2.88,3.05,1.18,0.94,0.97,1.87,0.78,0.64,0.98,1.78,NA,1.62,NA,0.00,0.00)
r63<-
c(NA,7.25,12.77,16.61,14.64,5.06,2.72,0.74,2.60,8.74,4.04,3.15,2.71,2.89,1.11,4.53,3.26,1.30,1.09,1.45,2.3
7,2.14,1.54,NA,1.57,0.49,3.34,3.11,1.32,0.94,NA,NA,0.78,NA,0.98,1.98,NA,1.64,NA,0.00,0.00)
r64<-
c(NA,7.29,13.00,7.68,17.21,18.43,5.06,1.81,2.72,0.74,2.65,8.96,5.31,3.23,3.50,3.02,1.23,5.44,3.30,1.64,1.2
4,1.88,3.17,2.26,1.56,NA,NA,0.49,3.84,3.19,1.59,0.95,NA,NA,NA,NA,NA,NA,NA,2.19,NA)
r65<-
c(1.00,0.15,0.26,0.35,0.44,0.11,0.60,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,
0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.07,0.00,0.00)
r66<-
c(0.93,0.33,0.52,0.95,0.63,0.43,0.50,0.26,0.09,0.09,0.18,0.16,0.50,0.17,0.39,0.23,0.22,0.09,0.42,0.22,0.11,
0.21,0.39,0.07,0.21,0.45,0.21,0.33,0.23,0.18,0.16,0.19,0.41,0.26,0.26,0.39,0.53,0.46,0.39,0.30,0.27)
r67<-
c(1.00,0.71,0.89,1.00,1.00,0.72,1.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00,
1.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00,0.93,1.00,1.00)
r68<-c(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1)
r69<-
c(1.00,0.28,0.35,0.81,0.74,0.51,0.50,0.32,0.13,0.13,0.18,0.80,0.61,0.03,0.04,0.03,0.08,0.08,0.04,0.03,0.04,
0.04,0.04,0.04,0.04,0.05,0.10,0.07,0.02,0.03,0.03,0.05,0.14,0.03,0.03,0.04,0.05,0.04,0.04,0.03,0.03)
r70<-c(1,1,1,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
r71<-
c(0.32,0.24,0.44,0.55,0.00,0.20,0.38,0.33,0.33,0.33,0.33,0.26,0.36,0.30,0.28,0.30,0.35,0.23,0.42,0.35,0.35,
0.43,0.39,0.39,0.26,0.30,0.36,0.20,0.24,0.30,0.32,0.30,0.12,0.25,0.25,0.25,0.26,0.27,NA,0.28,0.26)
r72<-c(3,4,3,3,3,2,4,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2)
r73<-
c(NA,0.03,0.11,0.20,0.05,0.17,0.07,0.33,0.39,0.69,0.42,0.14,0.23,0.03,0.08,0.03,0.05,0.06,0.20,0.09,0.13,0.
04,0.29,0.09,0.06,NA,0.04,0.00,0.12,0.02,0.09,0.00,NA,NA,0.00,NA,0.00,0.10,NA,0.01,NA)
r74<-
c(NA,0.03,0.13,0.30,0.08,0.38,0.07,0.41,0.39,0.69,0.44,0.17,0.47,0.06,0.31,0.07,0.15,0.23,0.21,0.29,0.25,0.
27,0.55,0.14,0.07,NA,NA,0.00,0.25,0.04,0.26,0.01,NA,NA,NA,NA,NA,NA,NA,0.26,NA)
r75<-
c(NA,0.02,0.09,0.32,0.08,0.45,0.24,0.40,0.38,0.66,0.40,0.13,0.50,0.04,0.30,0.07,0.15,0.24,0.20,0.26,0.27,0.
25,0.56,0.14,0.06,NA,NA,0.00,0.23,0.02,0.25,0.01,NA,NA,NA,NA,NA,NA,NA,0.11,NA)
r76<-
c(0.23,0.19,0.30,0.17,0.18,0.13,0.22,0.78,0.53,0.10,0.50,0.51,0.69,0.43,0.74,0.44,0.51,0.26,0.39,0.57,0.54,
0.60,0.61,0.29,0.54,NA,0.48,0.68,0.84,0.44,0.24,0.35,0.41,0.27,0.69,0.20,0.59,0.52,NA,0.50,NA)
r77<-
c(0.01,0.08,0.11,0.00,0.03,0.04,0.01,0.13,0.34,0.70,0.48,0.11,0.16,0.09,0.10,0.04,0.49,0.09,0.03,0.38,0.37,
0.20,0.14,0.37,0.09,NA,0.13,0.26,0.02,0.00,0.39,0.21,0.58,0.05,0.02,0.39,0.41,0.41,NA,0.50,NA)
r78<-
c(0.76,0.73,0.59,0.83,0.79,0.83,0.78,0.09,0.13,0.20,0.02,0.38,0.15,0.48,0.16,0.52,0.00,0.64,0.58,0.05,0.09,
0.21,0.25,0.34,0.37,NA,0.39,0.05,0.14,0.55,0.36,0.44,0.01,0.68,0.29,0.41,0.00,0.07,NA,0.00,NA)
r79<-c(NA,0.60,0.58,0.53,0.31,0.12,-
0.62,0.50,0.53,0.09,0.50,0.71,0.75,0.71,0.88,0.80,0.83,0.71,0.77,0.81,0.62,0.77,0.70,0.86,0.78,NA,NA,NA,0.
75,1.10,0.52,1.00,NA,NA,NA,NA,NA,NA,NA,0.93,NA)
r80<-
c(NA,0.12,0.22,0.01,0.05,0.17,0.00,0.47,0.40,0.86,0.46,0.07,0.12,0.12,0.04,0.00,0.00,0.16,0.10,0.11,0.04,0.
10,0.15,0.00,0.00,NA,NA,NA,0.04,0.00,0.17,0.00,NA,NA,NA,NA,NA,NA,NA,0.03,NA)
r81<-
c(NA,0.28,0.20,0.46,0.64,0.71,1.62,0.03,0.06,0.05,0.04,0.22,0.13,0.18,0.08,0.20,0.17,0.13,0.13,0.07,0.35,0.
13,0.14,0.14,0.22,NA,NA,NA,0.21,-0.10,0.30,0.00,NA,NA,NA,NA,NA,NA,NA,0.03,NA)
```

```
r82<-
c(NA,0.21,0.33,0.28,0.19,0.13,0.16,0.67,0.53,0.09,0.50,0.54,0.72,0.45,0.79,0.46,0.56,0.37,0.47,0.64,0.56,0.
65,0.66,0.37,0.56,NA,NA,0.68,0.82,0.47,0.31,0.36,NA,NA,NA,NA,NA,NA,NA,0.61,NA)
r83<-
c(NA,0.08,0.12,0.00,0.03,0.09,0.01,0.27,0.37,0.81,0.47,0.10,0.14,0.09,0.08,0.04,0.42,0.11,0.04,0.30,0.29,0.
17,0.15,0.32,0.09,NA,NA,0.26,0.02,0.00,0.34,0.21,NA,NA,NA,NA,NA,NA,NA,0.38,NA)
r84<-
c(NA,0.72,0.55,0.72,0.78,0.79,0.84,0.06,0.10,0.09,0.03,0.36,0.14,0.46,0.13,0.50,0.02,0.53,0.49,0.05,0.15,0.
19,0.19,0.31,0.36,NA,NA,0.05,0.16,0.52,0.35,0.44,NA,NA,NA,NA,NA,NA,NA,0.01,NA)
r85<-
c(NA,0.78,0.47,0.86,0.23,0.37,1.00,0.48,0.47,0.00,0.80,0.82,0.66,0.25,0.33,0.38,0.33,0.67,0.57,0.58,0.88,0.
78,0.50,0.33,0.50,NA,NA,0.80,0.91,0.42,0.50,1.00,NA,NA,NA,NA,NA,NA,NA,0.42,NA)
r86<-c(NA,NA,-0.02,0.10,0.00,-0.04,0.00,0.05,0.00,0.00,0.00,0.04,0.00,0.00,-0.04,-0.10,0.05,0.00,0.13,-
0.06,NA,0.00,0.13,-0.01,0.00,NA,NA,0.13,-0.02,0.00,0.00,0.00,NA,NA,NA,NA,NA,NA,NA,0.00,NA)
r87<-c(NA,NA,0.32,-0.05,0.00,0.17,0.20,0.00,0.00,0.00,0.00,0.00,-0.08,0.00,-
0.07,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.
00,0.00,0.00,0.00,0.00,0.00)
r88<-c(NA,NA,0,-1,0,17,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,3,0,0,0,0,0,0,0,0,-1,0,0)
r89<-
c(NA,NA,0.20,0.32,0.44,0.90,0.67,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.07,0.00,0.00,0.0
0,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.38,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00,0.00)
r90<-
c(NA,0.97,0.87,0.70,0.92,0.62,0.93,0.59,0.61,0.31,0.56,0.83,0.53,0.94,0.69,0.93,0.85,0.77,0.79,0.71,0.75,0.
73,0.45,0.86,0.93,NA,NA,1.00,0.75,0.96,0.74,0.99,NA,NA,NA,NA,NA,NA,NA,0.74,NA)
```

===READ DATA TITLESITES ONLY:===

```
rt01<-r01[8:41]
rt02<-r02[8:41]
rt03<-r03[8:41]
rt04<-r04[8:41]
rt05<-r05[8:41]
rt06<-r06[8:41]
rt07<-r07[8:41]
rt08<-r08[8:41]
rt09<-r09[8:41]
rt10<-r10[8:41]
rt11<-r11[8:41]
rt12<-r12[8:41]
rt13<-r13[8:41]
rt14<-r14[8:41]
rt15<-r15[8:41]
rt16<-r16[8:41]
rt17<-r17[8:41]
rt18<-r18[8:41]
rt19<-r19[8:41]
rt20<-r20[8:41]
rt21<-r21[8:41]
rt22<-r22[8:41]
rt23<-r23[8:41]
rt24<-r24[8:41]
rt25<-r25[8:41]
rt26<-r26[8:41]
rt27<-r27[8:41]
rt28<-r28[8:41]
rt29<-r29[8:41]
rt30<-r30[8:41]
rt31<-r31[8:41]
rt32<-r32[8:41]
rt33<-r33[8:41]
rt34<-r34[8:41]
rt35<-r35[8:41]
rt36<-r36[8:41]
rt37<-r37[8:41]
rt38<-r38[8:41]
rt39<-r39[8:41]
rt40<-r40[8:41]
rt41<-r41[8:41]
rt42<-r42[8:41]
rt43<-r43[8:41]
rt44<-r44[8:41]
```

```
rt45<-r45[8:41]
rt46<-r46[8:41]
rt47<-r47[8:41]
rt48<-r48[8:41]
rt49<-r49[8:41]
rt50<-r50[8:41]
rt51<-r51[8:41]
rt52<-r52[8:41]
rt53<-r53[8:41]
rt54<-r54[8:41]
rt55<-r55[8:41]
rt56<-r56[8:41]
rt57<-r57[8:41]
rt58<-r58[8:41]
rt59<-r59[8:41]
rt60<-r60[8:41]
rt61<-r61[8:41]
rt62<-r62[8:41]
rt63<-r63[8:41]
rt64<-r64[8:41]
rt65<-r65[8:41]
rt66<-r66[8:41]
rt67<-r67[8:41]
rt68<-r68[8:41]
rt69<-r69[8:41]
rt70<-r70[8:41]
rt71<-r71[8:41]
rt72<-r72[8:41]
rt73<-r73[8:41]
rt74<-r74[8:41]
rt75<-r75[8:41]
rt76<-r76[8:41]
rt77<-r77[8:41]
rt78<-r78[8:41]
rt79<-r79[8:41]
rt80<-r80[8:41]
rt81<-r81[8:41]
rt82<-r82[8:41]
rt83<-r83[8:41]
rt84<-r84[8:41]
rt85<-r85[8:41]
rt86<-r86[8:41]
rt87<-r87[8:41]
rt88<-r88[8:41]
rt89<-r89[8:41]
rt90<-r90[8:41]

===FIGURE  10:===
plot(r66,r62, xlab = "Spec page coverage", ylab="", main = "Nr issues per construction day at live date")

===FIGURE 11:===
plot(rt66,rt62, xlab = "Spec page coverage", ylab ="", main =  "Nr issues per construction day at live date")

===FIGURE 12:===
plot(r66[1:7],r62[1:7], xlab = "Spec page coverage", ylab="", main = "Nr issues per construction day at live
date")

===FIGURE 13:===
plot(r67,r62, xlab = "Spec portlet coverage", ylab ="", main =  "Nr issues per construction day at live date")

===FIGURE 14:===
boxplot(rt62, ylab ="", main =  "Nr of issues per construction day at live date")

===FIGURE 15:===
boxplot(rt26, ylab ="", main =  "Total nr of issues at live date")

===FIGURE 16:===
plot(r69,r62, xlab = "VD page coverage", ylab="", main = "Nr issues per construction day at live date")

===FIGURE 17:===
plot(rt69,rt62, xlab = "VD page coverage", ylab="", main = "Nr issues per construction day at live date")
```

```
===FIGURE 18:===
plot(rt69,rt26, xlab = "VD page coverage", ylab="", main = "Total nr of issues at live date")

===FIGURE 19:===
plot(r66,r31/r25, xlab = "Spec page coverage", ylab="", main = "Requested RFCs per construction day at live
date")

===FIGURE 20:===
plot(rt66,rt31/rt25, xlab = "Spec page coverage", ylab="", main = "Requested RFCs per construction day at
live date")

===FIGURE 21:===
plot(rt66,rt32/rt25, xlab = "Spec page coverage", ylab="", main = "Implemented RFCs per construction day at
live date")

===FIGURE 22:===
plot(r66,r86, xlab = "Spec page coverage", ylab="", main = "Growth in pages")
x<-c(0,1)
y<-c(0,0)
lines(x,y)

===FIGURE 23:===
plot(r66,r59, xlab = "Spec page coverage", ylab="", main = "Construction time estimation accuracy")
x<-c(0,1)
y<-c(0,1)
lines(x,y)

===FIGURE 24:===
plot(r69,r59, xlab = "VD page coverage", ylab="", main = "Construction time estimation accuracy")
x<-c(0,1)
y<-c(0,1)
lines(x,y)

===FIGURE 25:===
plot(r66,r51, xlab = "Spec page coverage", ylab="", main = "Commercial effort estimation accuracy")

===FIGURE 26:===
plot(r69,r51, xlab = "VD page coverage", ylab="", main = "Commercial effort estimation accuracy")

===FIGURE 27:===
plot(r66,r52, xlab = "Spec page coverage", ylab="", main = "WBS effort estimation accuracy")

===FIGURE 28:===
plot(r69,r51, xlab = "VD page coverage", ylab="", main = "WBS effort estimation accuracy")

===FIGURE 29:===
plot(r89,r59, main = "Degree of customization VS construction time estimation accuracy", xlab="Actual
custom/total portlets", ylab="")

===FIGURE 30:===
plot(r89,r64, main = "Degree of customization VS nr of issues per construction day", xlab="Actual custom/total
portlets", ylab="Nr issues per construction day")

===FIGURE 31:===
plot(r72,r90, xlab="Nr of different types of tests done", main="Ratio issues found before deployment on prod",
ylab = "")

===FIGURE 32:===
issRateCustAtLive = r28/r26
plot(r72,issRateCustAtLive, xlab="Nr of different types of tests done", main="Ratio issues posted by customer
until live", ylab = "")

===FIGURE 33:===
plot(r59,r62, xlab="Construction time estimation accuracy", ylab="", main = "Number of issues per
construction day at live")

===FIGURE 34:===
plot(rt59,rt62, xlab="Construction time estimation accuracy", ylab="", main = "Number of issues per
construction day at live")

===FIGURE 35:===
```

```
plot(r51,r62, xlab="Commercial effort estimation accuracy", ylab="", main = "Number of issues per construction day at live")

===FIGURE 36:===
plot(r52,r62, xlab="WBS effort estimation accuracy", ylab="", main = "Number of issues per construction day at live")

===FIGURE 37:===
plot(r24,r26, xlab="Actual effort in hours", ylab="", main = "Total number of issues at live")

===FIGURE 38:===
plot(rt24,rt26, xlab="Actual effort in hours", ylab="", main = "Total number of issues at live")

===FIGURE 39:===
plot(r24,r33, xlab="Actual effort in hours", ylab="", main = "Total number of issues after one month live")

===FIGURE 40:===
plot(rt24,rt33, xlab="Actual effort in hours", ylab="", main = "Total number of issues after one month live")

===FIGURE 41:===
plot(r25,r26, xlab="Construction duration in hours", ylab="", main="Amount of issues at live date")

===FIGURE 42:===
plot(r25,r33, xlab="Construction duration in hours", ylab="", main = "Total number of issues after one month live")

===FIGURE 43:===
plot(r52,r59, xlab="WBS effort estimation accuracy", ylab="Construction time accuracy", main="Construction time accuracy VS WBS effort estimation accuracy")
x<-c(0,5)
y<-c(1,1)
lines(x,y)

===FIGURE 44:===
plot(r52,r60, xlab="WBS effort estimation accuracy", ylab="Total time accuracy", main="Total time accuracy VS WBS effort estimation accuracy")
x<-c(0,5)
y<-c(1,1)
lines(x,y)

===FIGURE 45:===
plot(r24,r25, xlab="Actual effort in hours", ylab="Actual construction duration", main="Actual effort VS construction duration")

===FIGURE 46:===
plot(r24,r54, xlab="Actual effort in hours", ylab="Actual total duration", main="Actual effort VS total duration")

===FIGURE 47:===
totalIssDuring1MonthGuaranteePerConstrDay = (r33-r26)/r25
plot(totalIssDuring1MonthGuaranteePerConstrDay, r52, xlab="Issues posted during 1 month guarantee (normalized by dividing with construction duration)", ylab="WBS effort estimation accuracy", main="WBS effort est. accuracy VS # issues during 1 month guarantee (normalized)")

===FIGURE 48:===
plot(totalIssDuring1MonthGuaranteePerConstrDay, r59, xlab="Issues posted during 1 month guarantee (normalized by dividing with construction duration)", ylab="Construction time estimation accuracy", main="Constr. time est. accuracy VS # issues during 1 month guarantee (normalized)")
lines(x,y)

===FIGURE 49:===
plot(r32, r50, xlab="Total number of RFCs implemented", ylab="Absolute effort overrun (hrs)", main="Number of RFCs implemented VS absolute effort overrun")

===FIGURE 50:===
boxplot(r52, ylab ="", main =  "WBS effort estimation accuracy")

===FIGURE 51:===
r52b <-c(r52[4:5],r52[7:13],r52[15],r52[17:29],r52[31:33],r52[35:41])
boxplot(r52b, ylab ="", main =  "WBS effort estimation accuracy")

===FIGURE 52:===
boxplot(r59, ylab ="", main =  "Construction time estimation accuracy")
```

```
===FIGURE 53:===
boxplot(r60, ylab ="", main =  "Total time estimation accuracy")

===FIGURE 54:===
issuesTitlesiteSorted <-c(r26[8], r26[9], r26[10], r26[11], r26[13], r26[14], r26[16], r26[19], r26[20],
r26[22], r26[24], r26[15], r26[18], r26[21], r26[12], r26[23], r26[25], r26[30], r26[29], r26[26], r26[17],
r26[27], r26[28],  r26[31], r26[32], r26[40], r26[41], r26[38], r26[33], r26[34], r26[35], r26[36], r26[37],
r26[39])
xAxis <-c(1:34)
plot(xAxis, issuesTitlesiteSorted, ylab = "# issues at live date", main="Total amount of issues per titlesite over
time")

===FIGURE 55:===
x01<-c(500, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000)
y01<-c(67, 84, 106, 122, 134, 144, 153, 161, 169)
x02=x01
y02<-c(82,115,163,200,230,258,282,305,326)
plot(r24,r25, xlab="Actual effort in hours", ylab="Actual construction duration", main="Actual effort VS
construction duration")
lines(x01,y01)
lines(x02,y02)
```

The r numbers refer to:

r01	Project name
r02	Project Type
r03	Componence PM
r04	Vendor (offshore team)
r05	Vendor PM
r06	Project price (commercial estimation)
r07	Number of Sales Managers NL that made an estimation
r08	Number of Project Managers NL that made an estimation
r09	Number of Project Managers UA that made an estimation
r10	Planned effort construction (UA ==> calulated from price / CONFIDENTIAL)
r11	SCOPE: Total nr of pages to be implemented (Sitemap FO)
r12	DESIGNED: Total nr of pages described in functional specification
r13	SCOPE: Total nr of unique portlets to be implemented (planned)
r14	DESIGNED: Total nr of portlets described in functional specification
r15	DESIGNED: Number of custom portlets in functional specification
r16	Load performance requirements avaiable?
r17	Response time requirements available?
r18	Total number of pages described in visual design
r19	Expected effort in hours (UA hours)
r20	Planned Analysis duration (based on original analysis start date)
r21	Planned Analysis duration revised (based on actual start of analysis)
r22	Planned Construction duration PP
r23	Project Charter avaialble during construction phase?
r24	Actual Effort of construction phase (UA hours)
r25	Actual Construction duration
r26	Total number of issues at live date (excluding questions and tasks)
r27	Number of bugs
r28	Issues posted by Customer until Live
r29	Issues posted by Componence NL until Live
r30	Issues posted by Componence UA until Live
r31	Number of RFCs posted until Live
r32	Number of RFCs implemented until Live
r33	Total number of issues after one month of guarantee (excluding questions and tasks)
r34	ACTUAL: Total number of pages implemented (Sitemap site)

r35	ACTUAL: Total number of unique portlets implemented
r36	ACTUAL: number of custom portlets
r37	Total number of issues posted in guarantee period
r38	Number of RFCs posted
r39	Number of RFCs implemented
r40	Issues posted by Customer
r41	Issues posted by Componence NL
r42	Issues posted by Componence UA
r43	Project charter available during construction?
r44	Project plan available?
r45	Total number of issues
r46	Total number of bugs
r47	Total number of people that made a commercial estimation
r48	Joint commercial estimation quality (the bigger the better)
r49	Expected effort from project plan/expected effort based on price
r50	Absolute effort overrun
r51	Commercial effort estimation accuracy
r52	Actual effort/WBS effort estimation
r53	Estimated project duration
r54	Actual project duration
r55	Planned analysis duration
r56	Actual Analysis duration
r57	Estimated construction duration
r58	Planned analysis/planned project duration(=analysis+construction)
r59	Calender time estimation accuracy (construction)
r60	Calender time estimation accuracy (total project)
r61	Amount of rework (number of issues excl. questions and tasks)
r62	Amount of rework per construction day (until live issues)
r63	Amount of rework per construction day (until 1 month guarantee issues)
r64	Amount of rework per construction day (total issues)
r65	Planned amount of custom/total portlets
r66	SPEC Page coverage (described/to be implemented)
r67	Portlet coverage (described/to be implemented)
r68	Visual design present?
r69	VD Page coverage (described/to be implemented)
r70	Clickable HTML prototype present?
r71	Time spent analyzing/total planned project time (excl guarantee period)
r72	Number of tests actually done (of 7)
r73	Ratio issues found during 1st month guarantee
r74	Ratio issues found after deployment on prod/total issues
r75	Ratio bugs found after deployment on prod/total bugs
r76	Posted by customer up to live date
r77	Posted by NL team up to live date
r78	Posted by offshore team up to live date
r79	Posted by customer during guarantee
r80	Posted by NL team during guarantee
r81	Posted by offshore team during guarantee
r82	Posted by customer total project
r83	Posted by NL team total project
r84	Posted by offshore team total project
r85	RFCs implemented/RFCs requested

r86	Growth of product in pages
r87	Growth of product in portlets
r88	Number of unplanned custom portlets
r89	Actual custom/total portlets
r90	Issues during building/total issues

Table 5

14 GLOSSARY AND CHART EXPLANATION

14.1 Glossary

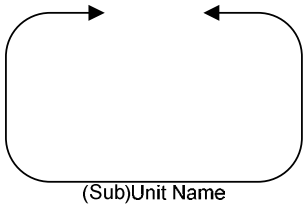
- Analogy based estimation: effort estimation made based on past experience on implementing similar functionality;
- Bug: a fault in the code which may need fixing depending on the severity;
- (Web) Portal: a web page that acts as an entrance to other web pages. The content of a portal can be edited by administrators whom do not need to learn programming languages in order to do so;
- Deployment: installation of a portal on a server;
- HTML: Hyper Text Markup Language, a language for writing web pages;
- Issue: an issue can be a (sub) bug, (sub) improvement or (sub) new feature, (sub) question, or (sub) task. Note that (sub) questions and (sub) tasks are not counted as issues in this report as they don't contribute to the amount of rework or extra work;Offshore team: a team that operates outside of the Netherlands (in the context of this report);
- Iteration: a part (wit a duration of usually one or more weeks) of the RUP methodology;
- Jira: a commercial issue tracking system for posting and following the status of issues (among other things);Portlet: pluggable component of a portal. A portal can contain multiple portlets, for instance a weather portlet for displaying the weather, a news portlet (or multiple news portlets) for displaying news, etcetera;
- KlantNET: an in-house developed issue tracking system for posting and following the status of issues (among other things);Production (server/environment) or prod: the server where the portal is installed when construction is finished;
- Memory leak: when a component of a portal fails to release memory that is no longer needed;
- Metric: a parameter that is used to quantitatively and periodically assess a process that is to be measured;
- Outsourcing: delegation of work to offshore teams;
- Performance: how well a portal can handle load and how fast a portal loads;
- Rework: work that results from resolving bugs;
- RFC: Request For Change, a change request that can result in new or improved functionality (or requirements creep);
- RUP: Rational Unified Process, a methodology used in software engineering;
- Vendor: an offshore team that implements portlets or portals for Componence.

14.2 How to read the organization charts in chapter 2

The dashed blue arrow shows what unit reports to another unit.

Reports to

The (sub-) unit box represents a unit (team) that may consist of (partial) sub units. The size and/or the orientation of the units do not necessarily represent the actual "real life" size of the units.

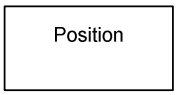


(Sub)Unit

The double framed box represents a director position.

Director position

The single framed box represents a position.

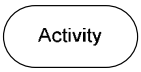


Position

14.3 How to read the flow charts in Appendix A

The solid black circle is the initial state, the flow starts here:

A balloon represents an activity:



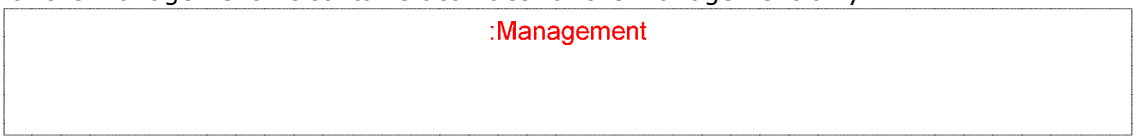
An arrow shows what activity is next. Text on an arrow means there is a condition to be met before following the arrow:

A diamond represents a decision. Multiple arrows represent multiple possible decisions:

A black bar represents the beginning or the end of a parallel. A parallel may contain multiple activities or decisions:

A circle with a solid black circle in it represents the final state. The flow chart ends here:

Large boxes containing any of the above are so called swim lanes. Below is a swim lane for the management. It contains activities for the management only.



14.4 How to read boxplots

The boxplot should be read as follows:

- The maximum (that is not an outlier) is shown by the highest bar;
- The upper quartile or third quartile cuts off the highest 25% of the data;
- The median shows the middle value (dark line);
- The lower quartile or first quartile cuts off the lowest 25% of the data;
- The minimum (that is not an outlier) is the lowest bar;
- The box contains 50% of all data.

15 REFERENCES

A. J. Albrecht. "Function point analysis", Encyclopedia of Software Engineering 1, 1994.

B.W. Boehm, B. Clark, E. Horowitz, C. Westland, R. Madachy and R. Selby. "Cost models for future life cycle processes: COCOMO 2.0", Annals of Software Engineering 1, 1995.

B.W. Boehm and K.J. Sullivan. "Software Economics", The Future Of Software Engineering, 2000.

M. Jorgensen. "Practical Guidelines for Expert-Judgment-Based Software Effort Estimation", IEEE Software May/June 2005.

A.L. Lederer and J. Prasad. "A Causal Model for Software Cost Estimating Error", IEEE Transactions on Software Engineering February, 1998.

R. Pooley, D. Senior and D. Christie. "Collecting and Analyzing Web-Based Project Metrics", IEEE Software January/February 2002.

L. Putnam. "Software Life Cycle Model (SLIM)," QSM: http://www.qsm.com, 2001.

D. J. Reifer. "Web Development: Estimating Quick-to-Market Software", IEEE Software November/December, 2000.

U. Remus. "Critical Success Factors of Implementing Enterprise Portals", Conference Proceedings of the 39th Hawaii International Conference on System Sciences (HICSS), 2006.