UNIVERSITY OF GOTHENBURG

Automated analysis of machine-generated test-results

An exploration of the efficiency gains from automated testing

Bachelor of Science Thesis in Software Engineering and Management

MARTIN BÄCKMAN CARLSSON

RICHARD BRODIE

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
Göteborg, Sweden, May 2011

Automated analysis of machine-generated test-results

An exploration of the efficiency gains from automated testing

MARTIN BÄCKMAN CARLSSON

RICHARD BRODIE

Examiner: HELENA HOLMSTRÖM OLSSON

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone: + 46 (0)31-772 1000

Cover:

Department of Computer Science and Engineering
Göteborg, Sweden, May 2011

IT University Software Engineering & Management Bachelor Thesis

Automated analysis of machine-generated test-results

An exploration of the efficiency gains from automated testing

Martin Bäckman Carlsson
martin@def-x.net

Richard Brodie
richard@samsari.org

31st May 2011

UNIVERSITY OF GOTHENBURG
IT FACULTY

Abstract

Testing and measuring has become increasingly prevalent and important in industrial settings as products and processes move towards greater complexity. To be able to more fully judge quality and progress in a developing system, companies strive to test widely and test often. How, then, can these results be best taken advantage of? Many organisations favour using automated frameworks generating data based on either random permutations of configurations or constant running of pre-defined tests. However, it becomes difficult and infeasible for humans to manually trawl through the amounts of data produced when running such automated tests. Clearly, a tool or system to assist is required. This study will examine the potential improvements to efficiency made with the introduction of a bespoke test analysis system to one such company.

---

1 Introduction

To paraphrase a rather well-known maxim:

> *It is a truth universally acknowledged, that a software developer in possession of a new product, must be in want of some testing* (J. Austen, 1813).

This whimsical expression nicely illustrates the current situation regarding testing during development. Continual and rigorous testing is being seen more and more as a necessary and unavoidable aspect of developing software or embedded systems. Anyone developing a large system now must test widely and test often. As these systems become more complex, such sophisticated - and often automated - testing regimes become absolutely necessary in order to maintain development momentum and precision.

With software products becoming more and more complicated and organisations growing larger and more sophisticated in their abilities, the need for testing also grows. The challenge facing organisations that wish to expand or larger ones that wish to produce more mature software is; how to implement massive amounts of testing in the most cost-effective and salient manner. Given this general problem, we investigate a more specific version of this issue in collaboration with Ericsson Mobile Broadband Modules:

> *How can the procedure of data analysis be automated in order to improve the efficiency of a verification department within a large software company by introducing a central analysis tool?*

Measuring in software engineering is a recognised procedure and has been codified into several recognised standards by various international bodies, including

ISO/IEC 15939 and 25021. It is of importance to ensure reliability and to be able to gather input to the processes of decision making (Staron et al., 2011). Software Process Improvement techniques have been standardised and codified in methods such as the Capability Maturity Model Integration (CMMI). By grouping these best practices into recognised and widely understood standards, they are made much easier to implement in an organisation, allowing it to effectively improve its software development processes. As these standards assist process improvement, so too do the IEEE standards assist software development more directly.

Staron et al. (2011) conduct a study on an organisation which has reached a level of sophistication where they are able to show that the need for gathering and analysing measurements becomes integral to an organisation's future development if it is to continue producing software of such complexity. In our study we have similarly undertaken to improve the efficiency of Ericsson MBM's approach to collecting and analysing measurements. Transforming the process by introducing automation and availability by an easily accessible web interface we intend to lower the workload of the integration and verification teams significantly. Our definition of efficiency for the purposes of this study is linked to the amount of workload on each team member and the time it takes them, as a team, to collectively perform the tasks needed to complete a phase of verification. A strong parallel can be made between Staron et al.'s (2011) study and ours given our organisation's matching need to move toward a more automated system in order to quickly and effectively analyse quality measurement data.

Situated within the context of a thorough examination of the current state of the field regarding testing processes and organisation complexity, our research approach was to follow the practice of design research as closely as possible and so develop the prototype iteratively, recording the feedback during the validation meetings as data collection, supplemented with more targeted interviews with a select group of users and stakeholders. This, however, was adapted to be more linear with less importance placed on the iterative aspect. Instead, emphasis was placed on a more significant data collection stage at the end of nominal development which used the repertory grid technique to gather the validation data in a more academic and rigorous fashion.

2 Related Research

Testing is now considered an integral part of software development, bringing greater efficiency and allowing an organisation to develop better and faster (Kerry and Delgado, 2009; Staron and Meding, 2009; Tang, 2010). It follows that as an organisation and the projects it wishes to develop grow in scope, automated testing becomes more necessary.

Software Process Improvement is conducted by organisations who wish to improve the quality and sophistication of their development processes and infrastructure - there is often a close relationship between the *"quality of a development process and the quality of the products developed using that process"* (Sommerville, 1996, p. 666). The very same may be said for an organisation's *testing approach*. As the quality and effectiveness of such an approach improves, the quality of the software developed as a result follows suit. Given that one of the benefits of a company following an SPI standard, such as CMMI, is that it and by extension its customers can know exactly what level of development it enjoys, so too can the same be said for following a more rigorous and standardised testing methodology. An organisation can know that it produces better software as a result.

Improving development practices through systematic testing methodologies leads directly to improved quality and reduced development time and costs (da Mota Silveira Neto et al., 2011). The same paper asserts, however, that compared to other process improvement practices that may be applied to development, testing and measuring are not as established. While this is a situation which is changing rapidly, the benefits of systematic and ubiquitous testing are certainly less accepted than more 'traditional' SPI in organisations at present.

The IEEE defines *testability* as:

> *the degree to which a system or component facilitates the establishment of test criteria and the performance of tests to determine whether those criteria have been met.* (IEEE, 1990)

This definition means that a system or product which allows or *facilitates* test criteria to be designed for it and for tests to be run against these criteria is testable. The more easily this is to do the more testable a system or product is (Voas and Miller, 1995). In modern software and embedded development, projects are massive affairs; they take many months of work by sometimes hundreds of people to produce. This requires a very detailed and thorough set or requirements to be defined beforehand to be able to gauge the completeness of a product.

Testing in a software or embedded development project becomes a huge part of any new development work, often comprising half of all work performed. The first implication of this is that there is a great need to ensure that this effort is conducted as efficiently as possible. As a project grows, the coincident testing will grow in parallel, so costs expand at a similar rate. But perhaps more importantly than mere cost accounting, as more tests are performed the likelihood of human-error creeping in rises. Furthermore, it is possible that any changes to the software will have unexpected consequences, so a test regime must be repeated regularly whenever changes are made. As the codebase grows, so do the number and frequency of tests required, which adds significantly to the cost in terms of time and resources (Koopman et al., 2003).

Staron et al. (2011) demonstrate during their study that the need for gathering and analysing measurements with which to provide the most accurate basis for decision making grows continuously as an organisation matures. In the course of their case study they observed the organisation developing and refining their process from one which primarily collected measurements manually to one following a markedly more automated approach as they matured. The new system obtained measurements on a daily basis which were made available via web sites, thus allowing the status of the project to be accessible by everyone.

It is certainly the case that, even if the need for widespread and general testing is recognised and effected in an organisation, if not implemented in an appropriate manner extensive testing can be as much a hindrance as a help. It has the potential to be labour-intensive, expensive and monopolise developers' time. As suggested earlier, this can lead to human-errors or delays or even replicated work because of miscommunication (Koopman et al., 2003). The obvious solution to this problem is to automate more of the testing duties. This would have the effect to both allow more testing to be performed more efficiently without requiring - or at least minimising - human participation or oversight, but to also help to minimise human-error and maximise the standardisation of test methodology and the ability of the results to be replicated.

Once an organisation begins to pursue the route of more extensive automation, one of the great concerns becomes the question of reliability - can a system which by definition runs without human supervision be reliable and trustworthy.

There is a great importance in having a bespoke process for gathering and analysing test results. That is to say, such a process ought to be custom-made for the particular organisation that employs it. The suitability of this process should be judged from the point of view of the *business management* instead of the *software management*. This means that measurement systems are more important for the furthering of business goals rather than the furthering of software reliability. David Wade (2001) suggests that this issue can be addressed by tailoring the testing processes so as to minimise the *learning curve* and the number and complexity of steps involved in gathering measurements for the business interests.

Koopman et al. (2003) defines a four-step procedure for what they term *functional testing*. The four together make up a complete testing process. These are defined as:

1. *formulation of a property* - roughly, what is to be tested
2. *generation of test data* - the data that is used to test a component with
3. *test execution* - feeding the generated data into the test
4. *test result analysis* - interpreting the results of the test

While Ericsson MBM's situation and requirements are somewhat different from those addressed by Koopman, we can draw some very relevant parallels. In the case of Ericsson MBM, the first three steps are analogous to the processes already being performed by existing systems which are quite adequately automated and integrated for the purposes of this study. However, the final step, analysing the gathered test result data, is slightly less easy to compare. In Koopman et al.'s (2003) example, this fourth step does not completely cover the requirements of Ericsson MBM. In short, Ericsson MBM require what could be considered a fifth step in Koopman et al.'s (2003) model. According to this model, the fourth step involves analysing the result and validating the success of the test; what we require goes further in that it calls for ongoing and specialised analysis of the same test according to different input data – performance over time, in essence. This is currently being performed mostly by hand or by a collection of poorly integrated tools. Koopman et al. develop an automated tool to perform the last three steps which they present as a self-contained solution. Naturally, we are primarily interested in the final step as our scope is limited purely to the analysis of pre-collected results and data.

There are several extant measurement systems in use already in the software engineering industry; Staron et al. (2011) compare the system they developed specifically for Ericsson to a more established and generic system – TychoMetric – and conclude that theirs is less comprehensive in terms of features but demonstrate that this can be considered a strength. As a result of this their system is, therefore, less complicated and more applicable to a particular use. Given their specific focus on ease of use for the users, they point out that TychoMetric might be too advanced to fit in certain situations, showing the necessity of having a tailored system that was very simple to modify and extend. Where sophisticated and comprehensive analysis is not a priority, a custom software tailored to the organisation becomes more and more attractive a solution (Staron et al., 2011). Wisell et al. agree that the software should be designed with the problem in mind and further that the measurement should be designed to be flexible and modular (Wisell et al., 2007). It should be noted that, although Wisell et al are aiming to describe a system closely connected to hardware, some lessons can still be learned.

Automated testing and measuring improves the efficiency of an organisation (Staron and Meding, 2009). It allows the managers in the organisation to optimise their decision making processes as they do not have to spend time and resources on ensuring testing is controlled adequately enough to be useful and trusted. Instead they can proceed along other channels, secure in the knowledge that the testing will keep up.

3 Method

3.1 Research setting

Ericsson MBM – like other, similar, organisations – routinely performs myriad tests on their software throughout the development phases. This, they say, is absolutely necessary and unavoidable; as testing and verification is a massive undertaking which takes up roughly a third to a half of the entire development effort (for confidentiality reasons, we are unfortunately unable to be more precise than this). The particular testing that they perform, routine verification against a set of very complicated requirements specifications, necessitates an elaborate and intensive process which can take several days to complete in full. This must be done every time there is a new internal release of a new build of the software (approximately every 2 weeks). As of now, the test results are deposited in a large database in the form of either detailed measurement figures or simple "[test has] passed/failed" notations. Currently, analysis of the results in the database is carried out manually and therefore takes a great deal of unnecessary effort. Another identified problem is that since the analyses are carried out manually they are not shared effectively, consequently resulting in redundant analysis efforts.

Given the nature of the industry which Ericsson MBM is participating in, they are subject to changes in requirements which occur both substantially and frequently; to that end, the implication is that flexibility and modularity is a must in their approach towards testing and measurement. Specifically, Mobile Broadband Modules – the department at Ericsson we are collaborating with – is interested in testing and verifying the performance of firmware running on laptop modules during ongoing development - a software problem. This makes the need for a flexible and reliable testing and analysis solution of great concern. Their operating environment involves an automated testing system that runs tests in random permutations from a battery of possible tests with variable input data. This produces a massive amount of result data which needs to be analysed for significant or unexpected performance deviations.

Additionally, some results of automated tests are interesting only to individual verification teams responsible for a particular domain, but some results are interesting to several stakeholders. Since there is no central analysis system, the same analysis is thus often carried out multiple times and the analyses interpreted in different ways. For this reason, Ericsson MBM required a system that could analyse the information in the database and present it in a way that is easy to understand. A centralised system would also limit the need to carry out multiple analyses on the same data. Furthermore, it would also work as a common platform to view and share analyses.

Given the general groundwork laid by Staron et al. (2011) and the specific problem facing Ericsson MBM, the research objective we aimed to tackle is a specific case of the general problem discussed in the previous section:

> *How can the procedure of data analysis be automated in order to improve the efficiency of a verification team within a large software company by introducing a central analysis tool?*

The desired solution to this problem was identified in collaboration with Ericsson MBM as a way to implement their requirements of a centralised, flexible and automated analysis tool, which would relieve the verification teams and the line managers within Ericsson MBM from manually performing analyses. In order to ensure the possibility to share results and analyses, the Graphical User Interface was to be web-based allowing employees to view the same analysis and interactively choose which kind of analysis they are interested in. This tool is named STRATA - [**S**tatistical **T**est **R**esult **A**nalysis of the **T**EXAS **A**rray]. TEXAS is the name for the existing system which enables automatic tests to be performed and which essentially generates the data which STRATA must analyse.

While Staron et al. (2011) focused on a system that was designed to be very limited in scope and function in order to be easy to use and not confusing for the users, it was designed largely with managers who were not domain experts in mind. Their principle argument was that it is difficult and time-consuming to implement the ISO standards in a new environment and so their focus was to define and standardise a framework to allow rapid design and implementation of a new measurement system by making it easier, especially for unfamiliar users, to identify which measures are important to focus on and tailor and which can be treated more generically. With Ericsson MBM, however, the situation was similar but brought some notable differences which made the work by Staron et al. unsuitable for direct and unmodified application. In short, Ericsson MBM already had a fairly sophisticated measurement system but virtually no corresponding analysis framework. Furthermore, Ericsson MBM required much more comprehensive and elaborate analyses to be performed than the team Staron et al. focused on. Together, these differences meant that, while we could use the work of Staron et al. as a very solid base, we would need to somewhat refine Staron et al.'s (2011) model to be able to apply it to this situation.

3.2 Design research

Design research focuses on the designing, implementation and application of things as a method of gathering knowledge (Hevner and Chatterjee, 2010). In design research the researcher creates and evaluates a prototype for the purpose of improving an existing solution or offering a solution to an organisational problem not yet addressed. Given that our research problem was essentially concerned with low efficiency in data analysis, design research was the logical method to choose to solve such a problem considering the formative and reflective nature of it.(Hevner et al., 2004).

In studies on design research (Hevner and Chatterjee, 2010; Hevner et al., 2004), the importance of properly and thoroughly evaluating the artefact at hand is discussed; therefore, and to ensure as close as possible a fulfilment of the stakeholders' requirements, we decided to implement the artefact in an incremental fashion. This would allow us to regularly receive feedback from the practitioners and to more precisely understand the problem in order to offer a more effective and substantial solution with quicker response times. Because of the limited means and scope of our situation, we recognised that measuring and quantifying efficiency is problematic. A quantitative study of this nature would require a very different disposition of resources than is available and feasible to us. In response, then, we instead approached the problem from a largely qualitative direction, as we expected to satisfy the requirements of our problem by conducting semi-structured interviews paired with document analysis. Primarily the documents being analysed were those provided us by Ericsson MBM describing the existing systems in place. Secondary documents were example reports linked to the analyses of the tests. The interviews were carried out during face-to-face meetings and notes were taken. Although Creswell (2008) suggests audio recording and transcription of interviews, it was understood that due to the confidentiality concerns of Ericsson MBM, we were not permitted to record the interviews and so we decided to use a different technique. Hevner and Chatterjee (2010) relies on Vaishnavi and Kuechler (2007) when defining a model of the design research methodology where the research is carried out iteratively with five major steps. The steps are:

- Awareness of Problem
- Suggestion
- Development
- Evaluation
- Conclusion.

In our case, the *awareness* of the problem was introduced and discussed during an introductory meeting at the organisation. During a followup meeting and further discussion with an organisation-side supervisor, we began to get a clearer idea of exactly the scope of the problem facing Ericsson MBM. Building on this initial meeting, our main information- and requirements-gathering efforts were given towards conducting a series of interviews amongst the expected users of the system.

The first goal of these interviews was to gather the information which would form the basis of our *suggestion*. We decided it would be important to get a thorough understanding of the current systems and processes for which the primary task should be to replicate. Secondarily, we aimed to gather information about what is deficient and what needs to be changed or improved. We predicted that some day-to-day tasks of the current users would become deprecated or obsolete with the introduction of a more uniform and integrated automated analysis system while others would show themselves to be integral and valuable and necessary to preserve and translate to the new system. These user-interviews were complemented with interviews with the stakeholders and management to try to gather any other relevant information which the users were not well-placed to impart. The results of the different interviews were then compiled and collated in order to find common points raised.

The *development* was carried out on location at Ericsson MBM in order to make use of the domain knowledge within the organisation. The development was carried out in two major steps: alpha and final. A validation meeting with Ericsson followed the alpha phase, feedback received during that meeting was incorporated into final development phase. Following the final development, four people holding different positions within the organisation were interviewed to *evaluate* the result. The interviews were divided into three parts, demonstration of the software, user scenarios and repertory grids session (see section 3.4). Finally, our *conclusion* was reached after compiling and analysing the evaluation results.

3.3 Process

During the study we realised that some parts of the original planned research method were not applicable to our investigation and therefore our plan had to be revised. We start by introducing our original plan which discloses the implementation strategy and original data collection. Subsequently we describe how the revised plan changes the approach towards data collection of the study.

Original Plan

As the thought behind design research is typified as *learning-by-doing*, the iterative approach and the evaluation feedback loop is very important (Hevner et al., 2004). In order to follow the iterative approach of design research we originally planned to divide the prototype implementation into alpha, beta and final phases and substantially base the development aims of the beta and final phases on the feedback from the alpha and beta phases. This planned approach was modified beginning with the beta phase. The modification was as a result of the recognition that repeated interviews and validation "chapters" were of less use than originally envisioned. Specifically, the first validation and interview session produced results that we believed would have guided the project somewhat outwith the narrow scope and would have been of little benefit to the subsequent development phases. Instead, we decided to continue development without being informed by feedback from the interviews and instead move the interviews to the end of the development where we could focus on them more substantially.

Given the importance of being responsive and flexible within the design research approach, we shall present here the reasoning behind our original research plan before going into more detail about the modified approach.

> The alpha implementation will be a proof of concept prototype which will be demonstrated to the company, based on the initial round of interviews but mostly on document analysis. During the demonstration of the prototype the company will be able to provide feedback which will allow us to evaluate the prototype. We will conduct interviews with key stakeholders to gather more detailed information than the feedback session can provide. Based on the fact that the interviews will be held in a semi-structured manner the result of the interviews will be abstracted and put into categories so that interview answers can be combined. To more easily categorise the answers a mind-map will be constructed with the results, this will also allow for visualisation of the interview outcome. Given the evaluation of the alpha release, the beta implementation will incorporate feedback collected combined with a new round of more structured and targeted interviews. At the end of the beta implementation we will offer a second demonstration of the prototype and again allow for feedback from the company. This release, which will be more finalised, will be evaluated by people using the software, allowing for greater feedback. The result of the third phase of development will be evaluated as the contribution of this paper. The artefact as a whole will be evaluated to answer the research question. Out of this evaluation the conclusions will be drawn.
>
> - **Initial interviews -** As outlined above
> - **Initial prototype -** Based on the interviews and requirements elicitation
> - **First-phase Validation -** The first validation will comprise a limited demonstration of the development so far and further interviews to be carried out together with a small group of stakeholders with different motives (different viewpoints). The validation serves as input to the development of the software, and also as creative input to Ericsson MBM in their exploration of more relevant and applicable requirements.
> - **Revised prototype -** The refined prototype is created to fill the refined and/or revised requirements of Ericsson MBM resulting in a product that they can use in their daily work.
> - **Second-phase Validation -** The second validation of the software is carried out as before with key people within Ericsson MBM, the group to consist of acquirers of the software and potential users. This is expected to be more formal than the previous one and will be the last chance for Ericsson MBM to modify the direction of development.
> - **Final development -** The final version is developed to fit the last requirements agreed with Ericsson MBM
> - **Final acceptance -** The final version is demonstrated and accepted

Revised Prototype Plan

The actual process which we used during development was, as explained earlier, a modification of the original iterative process. Broadly similar to the iterative plan as outlined in the bullet point list, we in the end decided to compress the process into one continuous development phase based primarily around the initial pre-development interviews. The post-iteration interviews were removed in favour of a more comprehensive session of interviews based on user-scenarios and repertory grids at the end of development. We planned also to maintain close contact with stakeholders and acquirers during development, but would not limit validation to particular scheduled meetings based around iteration phases. Given that design research has two basic activities, *build* and *evaluate* and that the purpose of design research is is the the created artefact, we would argue that the revised plan is within the scope of design research. (Hevner et al., 2004; Hevner and Chatterjee, 2010)

3.4 Data collection and Analysis

Data collection

The data collection was conducted during 35-45 minute interviews with four employees. To capture the perception of the developed prototype from different angles, interviews were conducted with a representative member from the verification team, a verification team leader, a project manager and a line manager. The interviews were conducted as follows: first the interviewees were shown a demo of the developed prototype and were given the chance to ask questions about it. Following the demonstration, they were asked to imagine and describe a hypothetical 'future version' of the STRATA tool that they would reasonably expect to exist after 6 months further development. During our time at Ericsson MBM, through previous interviews and interactions we realised that the employees had ideas of different features and improvements that would fall out of the scope of this study. Therefore we decided, during the data collection, to also include their views on a possible future version.

- 1) Current process
- 2) STRATA 1.0 (our developed prototype)
- 3) STRATA 2.0 (participant's imagined future version)

Easterby-Smith's (1980) definition of a full repertory grid is a methodology containing three components: *elements*, *constructs* and a *linking mechanism*. The grid is based upon the material represented by the first component, the *elements*, in our study represented by the three items in the bulletpoint list above. *Constructs* are words used to group and differentiate between *elements*, a *construct* is made up by two poles e.g. «low productivity–high productivity» or «traditional–novel». The constructs may be rated by *ranking*, *dichotomizing* and *rating* - the latter being the one used during this study. The third and last component of the grids is the *linking mechanism* which shows how each *element* is being assessed on each *construct* (Easterby-Smith, 1980).

Subsequently the interviewees were presented with a scenario, and were asked to describe how the scenario would unfold using each of the three elements. After describing the steps of action for the current process the interviewees were asked to describe the response for the scenario with STRATA 1.0 as an aid. Finally we asked how the scenario would proceed with the STRATA 2.0 which the interviewees themselves defined previously.

In our study, we used techniques called *PrinCom map* and *table display* to show the results, which we will explain in more detain in the next section. The constructs can be fabricated in various ways: by elicitation during discussion about the elements with participants, or by pre-defining them ourselves; in this study the constructs were pre-defined based on our reading of the literature. There are several ways to analyse the results of repertory grids which can be done either manually or by computation c.f. Easterby-Smith (1980); Tan and Hunter (2002). The rationale behind conducting the user scenario interview before the Repertory Grids session was to familiarise the interviewees with the thought of STRATA 1.0 and STRATA 2.0 before rating them. The interviewee was prompted with the following question:

> *'On a scale from one to five for the word convenience, one being low convenience and 5 being high, how would you rate: the current system, STRATA 1.0 and STRATA 2.0?'.*

After the scenario session the participants were taken through a repertory grids interview where they were asked to rate three elements with supplied constructs on a scale from **1-5**. The constructs were elicited from five original main concepts:

- **convenience**
- **useful**
- **simple**
- **efficiency**
- **collaboration**

The main concepts were chosen with the product in mind. Landseadel (1994); Staron et al. (2011) states the importance of having a *simple* user interface when it comes to testing and measurement systems. da Mota Silveira Neto et al. (2011) mentions the importance of *efficiency* to avoid having testing as the bottleneck of development, this construct also strongly ties into our research question. *Collaboration* is important within all development departments, healthy collaboration is important for the success of a project (Wurst et al., 2001). Other constructs were chosen to describe what we would like to achieve with our study, the *convenience* of being able to access the system from any computer within the organisation through a web interface. The system should be *useful* for the people working with verification and conformance, relieving them from manual labour. From these concepts several related sub-concepts, or *constructs*, were developed and added, keeping the original concepts as well.

Convenience
Ease (of use)
Time saver
Useful
Effective
Practical
Simple
Learning curve
Straightforward
Uniformity
Efficiency
Productivity
Collaboration
Communication
Teamwork

Table 1: Shows the different constructs, the original concepts are marked as bold

The sub concepts were developed by brainstorming and consultations with some literature in order to bring nuance into the data collection and interview phase.

Staron et al. (2011) found during their investigation the importance of the social issue *ease of use* and therefore bear that in mind when developing their framework. The user interface of STRATA 1.0 was developed with usability as a quality attribute, see Staron et al.'s (2011) reference to Umarji and Emurian. They also found that *ease of use* was important if a measurement system is to be successfully adopted, see Staron et al.'s (2011) reference to Jeffery and Berry.

In her paper Landseadel (1994) mentions the importance of having a *simple* user interface even when it comes to Automatic Testing Equipment. A *simple* and usable system lowers the cost for training staff in how to use the software and increases satisfaction, *productivity* and decreases the *learning curve*.

da Mota Silveira Neto et al. (2011) states in their paper that testing is the most *effective* method of quality assurance. They further highlight the need for more *efficient* and *effective* testing methods and techniques since the currently available methods make testing a very challenging process. They also found that testing is considered by some (see da Mota Silveira Neto et al.'s (2011) reference to Kolb and Muthig, and Muthig) as the bottleneck in a Software Product Line, however they offer in their paper strategies to ameliorate the bottleneck. One of their strategies is the introduction of test automation tools which will reduce the effort spent on testing, however da Mota Silveira Neto et al. (2011) raises a finger of warning since automated test execution could report false failures due to changes.

Wurst et al. (2001) states in their paper that *collaboration* between teams and *communication* within teams in an organisation is important for the success and quality of a project. As testing is a part of the development activities, importance of *collaboration* also applies between the development team and the testing team.

The constructs were given to the interviewees as undefined as possible, since we wanted to capture their understanding of the constructs filtered through the perspective of their role within the organisation. The interviewees were also assured that none of the grading is either good or bad and that the essence was to capture their perception. The user scenarios were recorded by taking notes which were later compiled. The repertory grid session was recorded by noting down the given numbers in a spreadsheet. Important to note is that the repertory grids specifically rate the *interviewee's* perception of the different elements.

Data analysis

The analysis of the repertory grids was performed using the free web-based tool WebGrids5[1] which provides analysis and presentation of supplied grids. The results of the interviews were compiled and made up a new grid with the calculated median value, given that the number of interviews was an even number the result was rounded up if the result was not a whole number. Out of the possible options provided by the the tool the "PrinCom Map" and the simple display were the most applicable for our study. Since the repertory grids captures the perception of the participants the data calls for qualitative analysis. Being repertory grids, and so a technique which is part quantitative and part qualitative, quantitative analysis could also be made, however given the low number of interviews this was not applicable.

[1]WebGrids5: http://gigi.cpsc.ucalgary.ca

4 Research outcomes

4.1 Prototype

The intention of the prototype is to further substantiate the claims made in the related literature and research in **Section 2** that the concept of an automated statistical analysis tool will improve efficiency and decision making capabilities for Ericsson MBM. This implies that the systems for performing tests and reporting results do not lie within the scope; Ericsson MBM already has systems in place that performs these functions. The prototype's job is to autonomously analyse test results, graphically present the analysis and report on interesting or concerning analysis results without the user actively requesting it. Test result information is extracted from an existing database harbouring the results, the extraction happens automatically when a new measurement is added to the database. If subscribed to, the new measurements gets analysed and stored, if a deviation is found all subscribers gets notified.

User interface

The developed system has a web-based graphical user interface that provides easy access to all employees no matter the computer. Another advantage with the web interface is that the users are able to share analyses easily by i.e. sharing a link. The interface is designed to be user-friendly and straight-forward in order to prevent a steep learning curve.

The main page provides a simple menu with the different categories to choose from. The menu is interactive and the different kind of test-types to choose from becomes visible when a category has been chosen. After a test-type has been selected the user chooses which test-configuration he/she is interested in, by default the user is able to choose between all of the different combinations possible for the type of test. The user has the possibility to filter the list for quicker access and the filtering feature also allows the user to construct custom configurations not provided by default. The analysis page allows the user to see the result of the analysis. The page contains general information about the test and the analysis. Further more the page contain two graphs containing the analysis data. The graph was designed to be quick on the eye and therefore the different datapoints in the graphs are colour coded. The base values are coded with the neutral colour blue, the trend is also coloured in a neutral colour: yellow. Alarming values are coloured red, limit indicators are coloured black. The graph is interactive which allows the user to click interesting points in the graph to acquire more information about that point of data.

In the analysis page the user is able to subscribe/unsubscribe to the selected analysis. By subscribing to the analysis the user will be notified as soon as the analysis finds interesting or alarming data (by Ericsson MBM's defined thresholds). The subscription allows the user to stay up to date with the specific type of test without actively having visit the system and check on a regular basis. When the analysis page is requested an analysis of the selected test if performed on the fly unless there is an subscription tied to the analysis. If there is an subscription the test will be analysed automatically and will already be ready when page is requested.

Extraction of raw data

On the backend the system communicates with a existing test result database. As soon as the system detects new test result data it will automatically extract that information and store it in the system in an analysis friendly format. Upon finding entirely new types of test the system will notify a nominated test administrator, who will be prompted to specify threshold values so that the analysis can be carried out correctly. Analysis will be carried out with default threshold values until the values has been properly set by the test administrator. If the system detects new test result data and there is a subscription tied to the specific test, an analysis will be carried out automatically. The system will be handling prodigious amounts of data. Tests are only interesting as long as the project is still active and/or maintained. The system has the ability to clean up old data that has not been used in a long time, also clean up old 'dead' subscriptions so that no excess space is wasted.

Analysis

The analysis is multithreaded in order to perform several analyses simultaneously, given the accessible web interface several people are able to request several analyses at the same time, this makes the concurrent analysis important. The analysis engine code is implemented in a modular fashion and is therefore very extendable, the extendability allows to easily replace the existing analysis technique or add other analysis techniques to run beside the existing one. The analysis is divided up into three steps:

- pre-analysis processing
- analysis processing
- post-analysis processing

The existing analysis technique prepares the data in the pre-analysis processing and perform simpler analysis tasks. The actual analysis processing consist of trend generation using Single Exponential Smoothing algorithm (Brown et al., 1961). The post-analysis analyse the generated trend and allows for trend prediction as a future feature. The analysis engine also allows the analysis to be analysed itself, which enables to show statistics over how well the system itself is performing. Something that could be viewed in the web interface.

4.2 User scenarios

During the user scenarios the interviewees was asked what steps would be taken given the scenario: *You run a test-suite, one of the measurement deviates by 50% from the average*.

When asked about the current process the participants gave broadly similar responses which we have consolidated together for greater clarity:

> The test would be re-run to confirm whether the same result was produced, if the problem has been located in e.g. firmware, talk to the firmware team. If the problem is not resolved the team leader will be notified and a Trouble Report[2] will be filed. The test-tool will be consulted to eliminate possible misconfigurations or errors. Alarming changes in quality would not be noticed if still within the permitted parameters, it would be discovered when manual analysis for a report is made.

For user scenarios of STRATA 1.0 and 2.0 split up by individual roles see **Table 2**

4.3 Repertory grids

The result of the repertory grids is presented in **Figure 1** as a *PrinCom map* and in **Figure 2** as a *table display*. A PrinCom map is generated using the FOCUS analysis method to sort the grid so that similar elements and similar constructs are clustered together (Shaw and Gaines, 1996).

The PrinCom map was generated by compiling the values of the different interviews and calculating the median value. Given that the number of interviews was an even number the result was rounded up if the result was not an whole number. An important thing to note about the PrinCom map is that constructs are rotated freely in unlimited dimensions, whereas the elements are shown in two dimensions (Olsson and Russo, 2004).

[2] Trouble Report: a method to file the detection, follow up and closure of bugs or problems in a project.

	STRATA 1.0	STRATA 2.0
Verification team member	The Verification team member felt that using the STRATA 1.0 tool would be much simpler since he could see the history of similar tests, and therefore simpler see the deviation from previous tests. The visual graphs also makes it easier to see if it is caused by a firmware or driver error. Or if a similar thing happened last time the driver/firmware was updated.	The Verification team member felt that a future version of STRATA could be integrated into more of the organisations existing systems and therefore saving time of finding needed information to properly address the scenario. A system that could summarise the information from most systems in one place.
Verification team leader	The Verification team leader views the system as a good tool to help in writing reports, the old process of using excel sheets will still be used but the tool will serve as a good compliment.	The team leader would like the future version become more integrated to the existing systems, and added features which will remove the need for using excel sheets.
Project manager	The Project manager thinks that the tool can help a great deal since there is no need to manually search for and extract the measurements and manually construct a graph representing the state of the project. Less room for error caused by the human factor.	The Project manger thinks that having the ability to export the analyses generated by the tool, for instance to a PDF format or spreadsheet, would improve the benefit of the tool. More types of analysis to represent different kinds of tests and different kinds of graphs.
Line manager	The Line manager thinks that the tool makes it easier to see specific areas where things went wrong, considering the trend generation. The tools makes it easier to draw conclusions. And by minimising the time for analysis pays for time spent on the development.	The Line manager sees the future version as a tool that is better at teasing out relationships and trends, and more features for more elaborate error detection. Extensions such as report generation and automatic trouble report filing is also seen as one of the potentials.

Table 2: Represents the second and third User Scenarios for each of the four participants

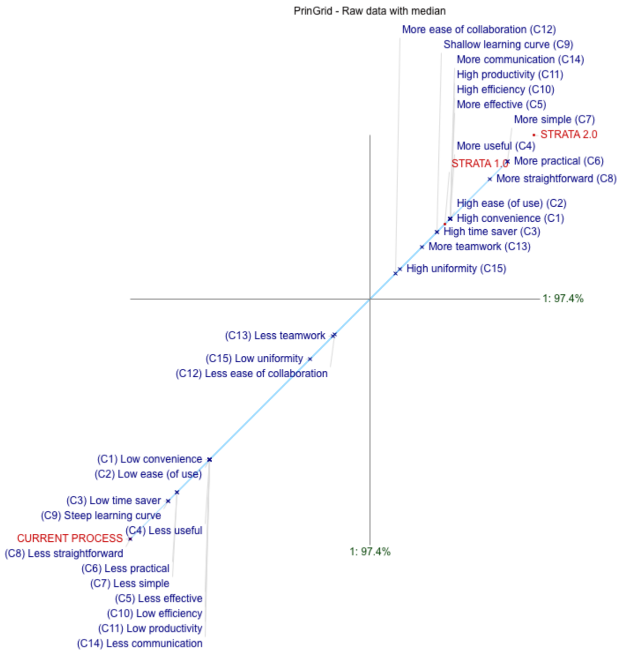


Figure 1: An example PrinCom map of the compiled results of the interviews calculated by median.

The ALL-CAPS text represents the elements, and the lower-case text represents the *constructs*. The Xs on the line represent where the *constructs* are located in relation to the *elements*. The PrinCom map shows how the three elements relate to the different constructs, the map allows for an overview of the interview results and makes it easy to see that the current process is the least favoured of the three elements. From the map one can also discern that STRATA 1.0 is favoured but the potential of STRATA 2.0 is perceived as even more so.

Figure 2 shows the calculated median values of the interview results, the columns represents the three elements and their rating on each construct (rows).

Display - Raw data with median

	CURRENT PROCESS	STRATA 1.0	STRATA 2.0	
(C1) Low convenience	2	4	5	High convenience
(C2) Low ease (of use)	2	4	5	High ease (of use)
(C3) Low time saver	2	5	5	High time saver
(C4) Less useful	2	4	5	More useful
(C5) Less effective	2	4	5	More effective
(C6) Less practical	1	4	5	More practical
(C7) Less simple	1	4	5	More simple
(C8) Less straightforward	1	5	5	More straightforward
(C9) Steep learning curve	2	5	5	Shallow learning curve
(C10) Low efficiency	2	4	5	High efficiency
(C11) Low productivity	2	4	5	High productivity
(C12) Less ease of collaboration	3	3	4	More ease of collaboration
(C13) Less teamwork	2	3	3	More teamwork
(C14) Less communication	2	4	5	More communication
(C15) Low uniformity	3	4	4	High uniformity

Figure 2: Table representation of the compiled results of the interviews calculated by median.

5 Discussion

To recap, the problem which we set out to explore in this paper was:

> *How can the procedure of data analysis be automated in order to improve the efficiency of a verification team within a large software company by introducing a central analysis tool?*

STRATA is a system which at its core automates the process of analysing data gathered from testing. The test measurements it analyses are themselves the result of automated systems which perform tests from a list of possible configurations and combinations. The primary efficiency bottleneck Ericsson MBM faced was that large parts of the post-testing analysis steps were performed manually or with loosely-integrated tools. The number of results needing to be analysed was growing every day and was already beyond the level where it was efficient or even useful to have such procedures in place for analysing the data (da Mota Silveira Neto et al., 2011). STRATA was designed to be a centralised and unified tool aimed at cutting out several of the most inefficient steps already in place. Our results showed that our interviewees, which we assumed to represent the general opinion at Ericsson MBM, perceived STRATA 1.0 to raise the productivity and improve the efficiency and effectiveness of the testing effort.

As we discussed earlier, shortly after the initial iteration phase we realised that such an approach was not the best fit for the Ericsson MBM working environment because of the mixed nature of the verification and feedback sessions. Subsequently, we had cause to reassess our research method in light of the reality of the circumstances; it became clear to us that it was best for the STRATA tool to continue in a prolonged single phase, rather than an iterative one. While it would not necessarily have caused any detrimental effect to continue as originally planned, the information we gathered from the end of the first iteration was largely tangential to the current development phase and more related to potential future applications. This meant that much of the information we gathered was something we had to put on one side anyway, so we decided it would be better to conduct all the interviews at the end when there would be a clearer idea of how much the developed tool - STRATA 1.0 - contributed to efficiency and how much more it could contribute with further development. In short, we feel that this change in method, in keeping with the principles of Design research, actually strengthened the final result.

The analysis of the results gathered from the user scenarios and the repertory grids allowed us to continue the research begun previously in the earlier papers by Kerry and Delgado (2009); Staron and Meding (2009); Tang (2010); da Mota Silveira Neto et al. (2011) and several others, and contribute our findings to the collective body of knowledge regarding automated testing and analysis.

As we considered during the review of related research, systematic and automated testing is now seen as a valuable and important tool used in many significant software development concerns. The esteem to which the software development industry holds automated testing is already great and its influence, already widespread, is still growing. This has been corroborated through many studies, including (Staron and Meding, 2009; Kerry and Delgado, 2009) amongst others. David Wade (2001) in particular stress the importance of designing the measurement gathering and analysis processes to be a close fit for the target organisation in order to make the solution as effective as possible. In this respect, STRATA certainly follows on from the prescriptions of previous research, in that it is a bespoke system designed specifically for application in Ericsson MBM.

As Staron et al. (2011); Staron and Meding (2009) and Landseadel (1994) all agree that a simple user interface that is easily understood is preferable when developing measurement and testing systems. Our results concur with these assertions as shown in our interviews where all participants thought to a greater or lesser degree that STRATA would be easy to use and learn. Furthermore, this was something which was greatly appreciated by the participants as they also all agreed that the current system in use was perceived to have a very steep learning curve.

da Mota Silveira Neto et al. (2011) brings up in their paper the challenges of testing and how it is often considered one of the greatest bottlenecks of the development process. One of the methods in their solution to ameliorate this is to migrate processes towards greater automation. Our objective was to automate and extend the final step of testing, according to Koopman et al.'s (2003) four steps of functional testing, for Ericsson MBM and by doing so improve the efficiency of their testing program. Our results show that the automation introduced by STRATA 1.0 is perceived by our interviewees to increase the efficiency and productivity of their work. The project manager, during the user scenario sessions told us that STRATA would help a great deal given that it would obviate the need to look for, extract and analyse test results manually. We concur with Neto in his claims that automation reduces the bottleneck of testing.

Our results showed that our interviews did not perceive STRATA 1.0 to improve *collaboration*, *teamwork* or *communication* as much as other factors, though they do agree they will be improved. This could be interpreted as STRATA 1.0 not directly helping them in their daily collaboration with other team members but perhaps, as this was rated higher than the other two, helping them *communicate* the results to other teams and in reports to other departments or customers. This would still be counted as collaboration according to Wurst et al. (2001), something which he deems an important issue to a development organisation. We suspect also that, as the current process requires team members to work mostly independently, collaboration and teamwork were not considered as high a priority as other factors and so STRATA 1.0 was not perceived as having as high an impact as it was on other factors.

Koopman et al. (2003) mentions that as the project grows (and so too does the testing) the likelihood of human error creeping in rises. Our result showed that a computerised *routine based* system will, according to the project manager, provide "less room for error caused by the human factor".

Staron et al. (2011) discusses how the *business management's* interest in the development phase mandates that the measurement systems should be business-centric and that gathering and analysing measurements provides the most accurate basis for decision making. Our results showed that the line manager believed that a tool (in this case, STRATA 1.0) which collects and analyses measurements will make it easier and faster to draw conclusions.

Immediately, one may see that our results - both the user scenarios from the four participants and the PrinCom maps generated from the repertory grid data for each of them - told a striking and clear story. It is an inescapable conclusion that a customised and automated system, as suggested by Staron et al. (2011); Staron and Meding (2009), is desirable in large development organisations. One may also observe that automation as suggested by da Mota Silveira Neto et al. (2011) indeed improved the efficacy of the testing performance.

As a result of the interviews during development and especially the structured interviews after the end of development, STRATA 2.0 has gathered an abundance of ideas and discussion on its future direction and application. STRATA 2.0 is desired by both us and the interviewed stakeholders to be more closely integrated with Ericsson MBM's existing systems. More specifically, as a system that could completely automatically communicate and interact with all systems within the testing chain in order to further relieve the verification departments of extra effort and error location. The extended integration and standardisation of communication will address the issues raised by Koopman et al. (2003) about human-error creeping in.

The future version of STRATA might also address the issue raised by Landseadel (1994) about the learning curve of a system by replacing a collection of several loosely-integrated systems with one single one. At the very least STRATA can reduce the awkwardness employees face when they must struggle to remember which information can be found within which system if everything is more tightly integrated and only "a click away".

6 Conclusion

This paper has been conducted in the context of Ericsson MBM as a relatively new organisation developing complex embedded systems. This endeavour requires them to work with both long development phases and intricate sets of requirement specifications. This calls for a sophisticated and comprehensive test and measurement system to allow them to develop more efficiently. Their specific problem which we decided to base our research on was that they have only partly solved this issue. As Koopman et al. (2003) show, there are four stages to testing: defining the test target and designing the test accordingly, producing data to test with, executing the test and finally, analysing and interpreting the results of the test. Ericsson MBM had an extant system which adequately addressed the first three steps, but were still lacking a suitable and scalable fourth step. As part of our study, we illustrated that the introduction of an automated test analysis system led to a significant increase in the perception of efficiency in the team.

Ericsson MBM recognised that their existing solution was inadequate for the task, particularly in light of expected future expansion of the organisation - lack of appropriate testing capability hinders efficiency (Staron and Meding, 2009). As such, they defined the initial scope of the STRATA project to focus primarily on the automatic aspect of test-analysis in order to replace a collection of different tool with a single one. A simple and intuitive user-interface was a secondary priority.

Our approach, then, was to design a system which was as autonomous as possible in order to address three coincident requirements - firstly, that the system should do as much as possible, so as to ensure that the resulting work is methodical and consistent; secondly, that such work is done as quickly and efficiently as possible; and thirdly that the users are then freed up to work on other concerns that is a more effective use of their time.

Upon completion of our tool we conducted a series of interviews in order to gather information about how suitable the department at Ericsson MBM felt the tool fit their needs. These interviews took a two-part form of user scenario discussions paired with semi-structured *repertory grids*.

Judging from the user scenarios and the repertory grid sessions it becomes clear that the different roles within the company have different goals which they see the prototype addressing and thus have different expectations from it. However our study shows that their perception of the usefulness of the prototype is in some ways similar in that they all agreed that the tool had more use than initially hoped for. During the development process, we were aware that Ericsson MBM had expectations that the tool would have the potential to be developed further. In the course of the interviews it was clear that Ericsson MBM certainly felt our tool had such potential and that they wished to expand the scope and purview of STRATA as soon as possible.

Our results show that the introduction of a customised automated analysis system can improve the efficiency of a verification department within a large organisation. STRATA 1.0 is by the Ericsson MBM representatives perceived as preferred to the previous current process and that a future – more integrated – version is perceived as carrying great potential. Staron et al. (2011) states in their study that a customised and easy-to-use measurement system is often preferred by management. Our results support Staron et al. (2011) hypothesis given the representatives attitudes towards STRATA 1.0 and 2.0.

Acknowledgements

We would like to thank our academic supervisor and our collaborators at Ericsson MBM for all the help during the study.

References

Brown, R. G., Meyer, R. F., and D'Esopo, D. A. (1961). The fundamental theorem of exponential smoothing. *Operations Research*, 9(5):pp. 673–687.

Creswell, J. W. (2008). *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*. Sage Publications, Inc.

da Mota Silveira Neto, P. A., do Carmo Machado, I., McGregor, J. D., de Almeida, E. S., and de Lemos Meira, S. R. (2011). A systematic mapping study of software product lines testing. *Information and Software Technology*, 53(5):407 – 423. Special Section on Best Papers from XP2010.

David Wade, R. J. R. (2001). *Corporate Performance Management: How to Build a Better Organization through Measurement-driven Strategic Alignment*. Elsevier Science & Technology, Boston.

Easterby-Smith, M. (1980). The Design, Analysis and Interpretation of Repertory Grids. *International Journal of Man-Machine Studies*, 13(1):3 – 24.

Hevner, A. and Chatterjee, S. (2010). Design Science Research Frameworks. In *Design Research in Information Systems*, volume 22 of *Integrated Series in Information Systems*, pages 23–31. Springer US.

Hevner, A. R., March, S. T., Park, J., and Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, 28(1):pp. 75–105.

IEEE (1990). IEEE Standard Glossary of Software Engineering Terminology. *IEEE Std 610.12-1990*, page 1.

Kerry, E. and Delgado, S. (2009). Applying software engineering practices to produce reliable, high-quality and accurate automated test systems. In *AUTOTESTCON, 2009 IEEE*, pages 69 –71.

Koopman, P., Alimarine, A., Tretmans, J., and Plasmeijer, R. (2003). Gast: Generic automated software testing. In Peña, R. and Arts, T., editors, *Implementation of Functional Languages*, volume 2670 of *Lecture Notes in Computer Science*, pages 991–991. Springer Berlin / Heidelberg.

Landseadel, P. (1994). Methodologies in user interface design. In *AUTOTESTCON '94. IEEE Systems Readiness Technology Conference. 'Cost Effective Support Into the Next Century', Conference Proceedings.*, pages 643 –648.

Olsson, C. and Russo, N. (2004). Evaluating innovative prototypes. In Fitzgerald, B. and Wynn, E., editors, *IT Innovation for Adaptability and Competitiveness*, volume 141 of *IFIP International Federation for Information Processing*, pages 181–202. Springer Boston.

Shaw, M. L. G. and Gaines, B. R. (1996). Webgrid: Knowledge elicitation and modeling on the web. http://algo.informatik.uni-freiburg.de/bibliothek/proceedings/webnet96/Html/209/209.htm.

Sommerville, I. (1996). *Software Engineering (5. ed.)*. Addison-Wesley.

Staron, M. and Meding, W. (2009). Ensuring reliability of information provided by measurement systems. In Abran, A., Braungarten, R., Dumke, R., Cuadrado-Gallego, J., and Brunekreef, J., editors, *Software Process and Product Measurement*, volume 5891 of *Lecture Notes in Computer Science*, pages 1–16. Springer Berlin / Heidelberg.

Staron, M., Meding, W., Karlsson, G., and Nilsson, C. (2011). Developing Measurement Systems: an Industrial Case Study. *Journal of Software Maintenance and Evolution: Research and Practice*, 23:89–107.

Tan, F. B. and Hunter, M. G. (2002). The Repertory Grid Technique: A Method for the Study of Cognition in Information Systems . 26:39–57.

Tang, J. (2010). Towards automation in software test life cycle based on multi-agent. In *Computational Intelligence and Software Engineering (CiSE), 2010 International Conference on*, pages 1–4.

Vaishnavi, V. K. and Kuechler, Jr., W. (2007). *Design Science Research Methods and Patterns: Innovating Information and Communication Technology*. Auerbach Publications, Boston, MA, USA, 1st edition.

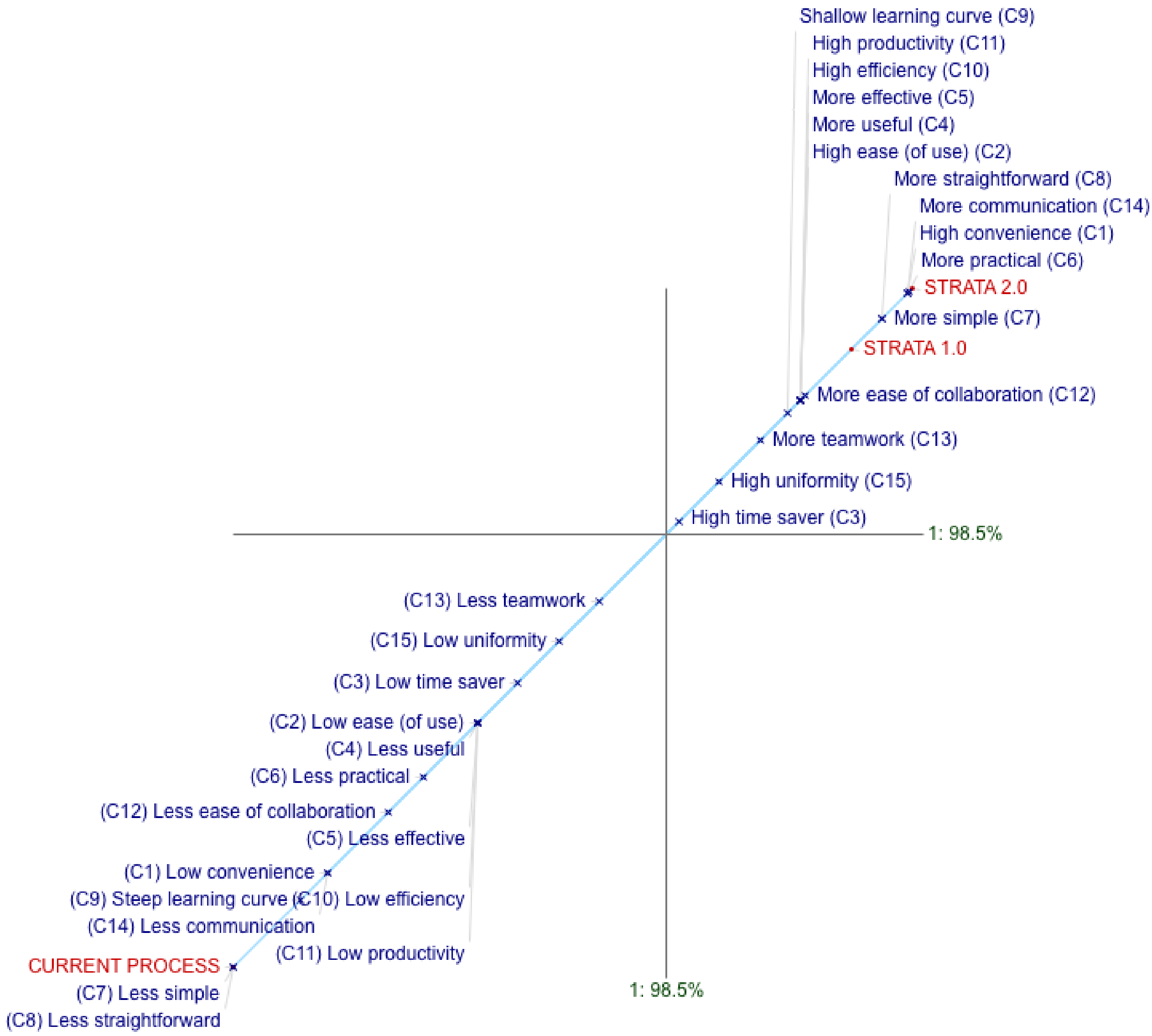
Voas, J. and Miller, K. (1995). Software Testability: the New Verification. *Software, IEEE*, 12(3):17 –28.

Wisell, D., Stenvard, P., Hansebacke, A., and Keskitalo, N. (2007). Considerations When Designing and Using Virtual Instruments as Building Blocks in Flexible Measurement System Solutions. In *Instrumentation and Measurement Technology Conference Proceedings, 2007. IMTC 2007. IEEE*, pages 1 –5.

Wurst, K., Hoegl, M., and Gemuenden, H. (2001). Collaboration within and between teams in multi-team r&d projects. In *Management of Engineering and Technology, 2001. PICMET '01. Portland International Conference on*, volume Supplement, pages 545 –552 vol.2.

A Verification team member

PrinGrid - Verification team member

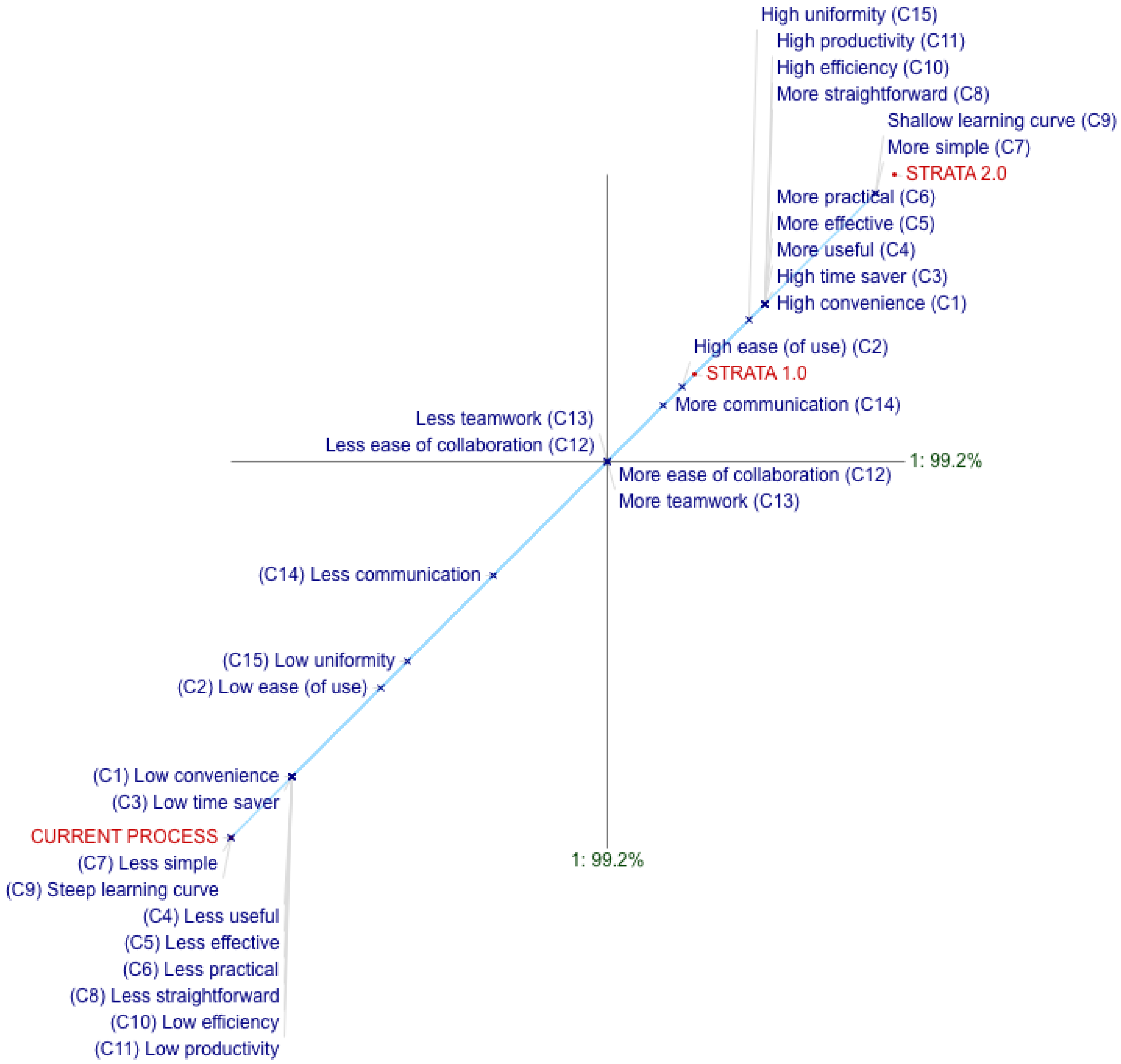


Display - Verification team member

	CURRENT PROCESS	STRATA 1.0	STRATA 2.0	
(C1) Low convenience	1	4	5	High convenience
(C2) Low ease (of use)	2	4	4	High ease (of use)
(C3) Low time saver	4	5	5	High time saver
(C4) Less useful	2	4	4	More useful
(C5) Less effective	2	4	4	More effective
(C6) Less practical	1	4	4	More practical
(C7) Less simple	1	5	5	More simple
(C8) Less straightforward	1	5	5	More straightforward
(C9) Steep learning curve	2	5	5	Shallow learning curve
(C10) Low efficiency	2	4	4	High efficiency
(C11) Low productivity	2	4	4	High productivity
(C12) Less ease of collaboration	2	4	5	More ease of collaboration
(C13) Less teamwork	2	3	3	More teamwork
(C14) Less communication	1	4	5	More communication
(C15) Low uniformity	3	4	4	High uniformity

B Verification team leader

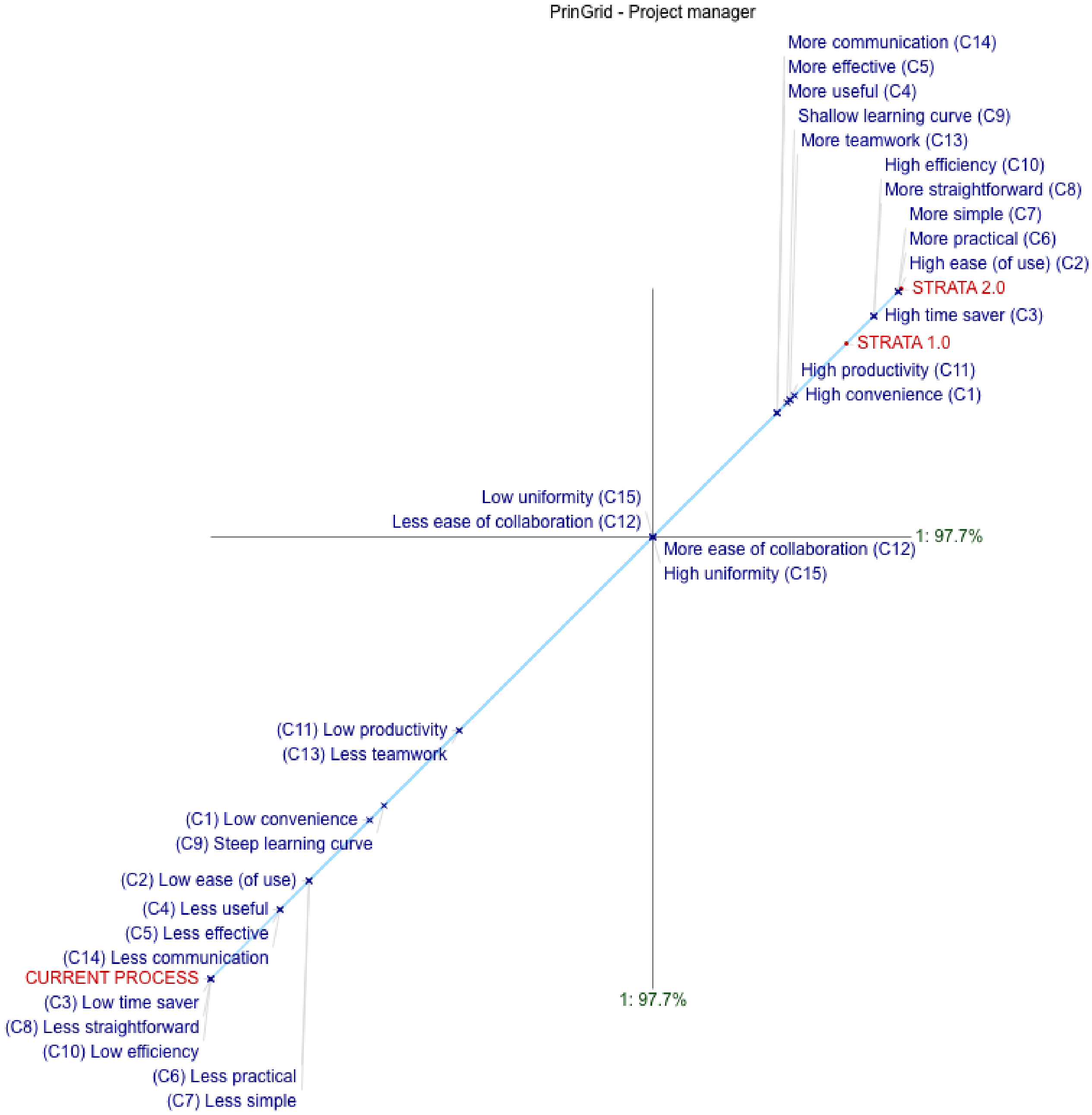
PrinGrid - Verification team leader



Display - Verification team leader

	CURRENT PROCESS	STRATA 1.0	STRATA 2.0	
(C1) Low convenience	2	4	5	High convenience
(C2) Low ease (of use)	3	4	5	High ease (of use)
(C3) Low time saver	2	4	5	High time saver
(C4) Less useful	2	4	5	More useful
(C5) Less effective	2	4	5	More effective
(C6) Less practical	2	4	5	More practical
(C7) Less simple	1	4	5	More simple
(C8) Less straightforward	2	4	5	More straightforward
(C9) Steep learning curve	1	4	5	Shallow learning curve
(C10) Low efficiency	2	4	5	High efficiency
(C11) Low productivity	2	4	5	High productivity
(C12) Less ease of collaboration	3	3	3	More ease of collaboration
(C13) Less teamwork	3	3	3	More teamwork
(C14) Less communication	3	4	4	More communication
(C15) Low uniformity	2	4	4	High uniformity

C Project manager

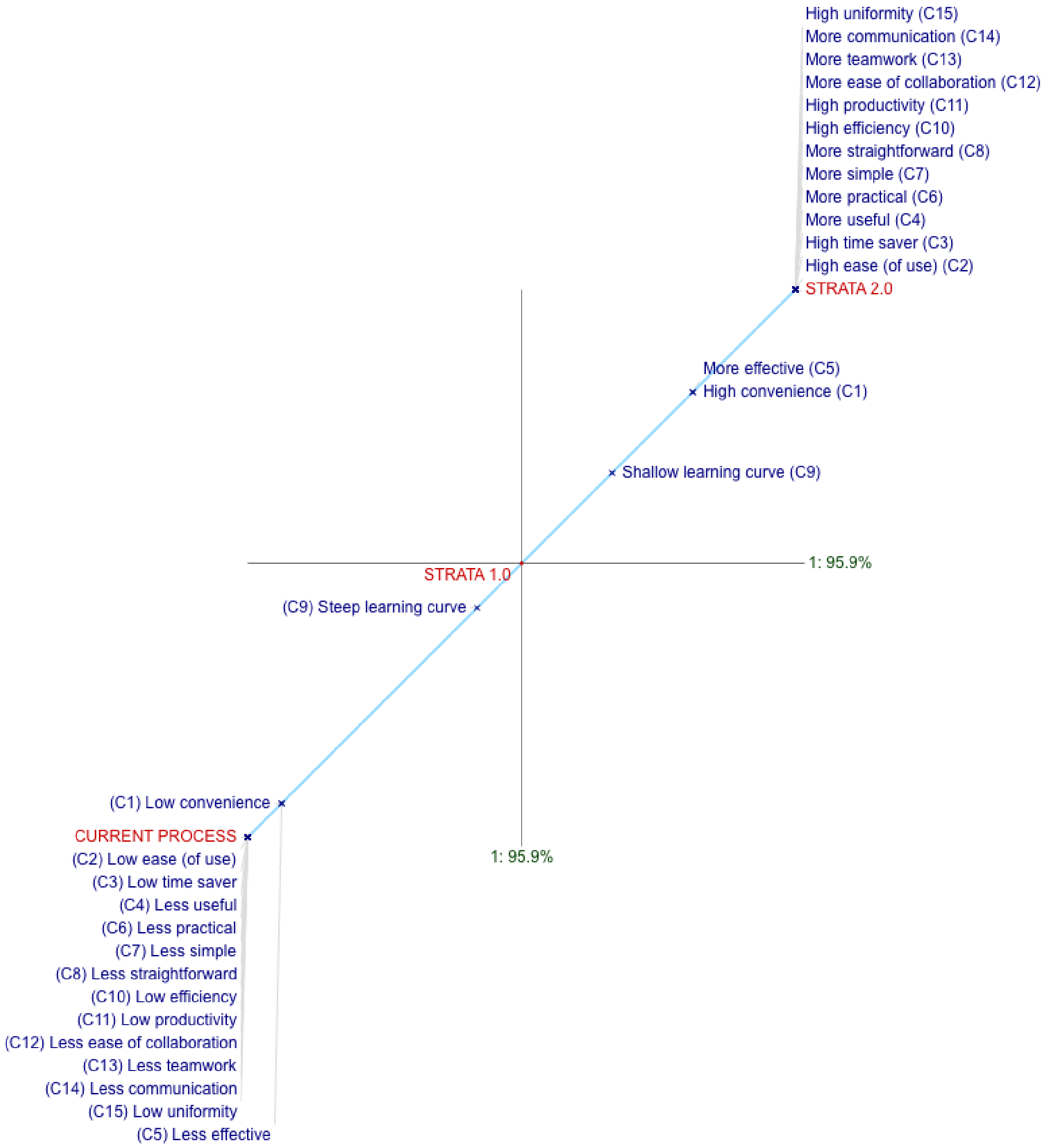


Display - Project manager

	CURRENT PROCESS	STRATA 1.0	STRATA 2.0	
(C1) Low convenience	2	4	5	High convenience
(C2) Low ease (of use)	1	4	5	High ease (of use)
(C3) Low time saver	1	5	5	High time saver
(C4) Less useful	2	5	5	More useful
(C5) Less effective	2	5	5	More effective
(C6) Less practical	1	4	5	More practical
(C7) Less simple	1	4	5	More simple
(C8) Less straightforward	1	5	5	More straightforward
(C9) Steep learning curve	2	5	4	Shallow learning curve
(C10) Low efficiency	1	5	5	High efficiency
(C11) Low productivity	2	4	4	High productivity
(C12) Less ease of collaboration	3	3	3	More ease of collaboration
(C13) Less teamwork	2	4	4	More teamwork
(C14) Less communication	2	5	5	More communication
(C15) Low uniformity	3	3	3	High uniformity

D Line manager

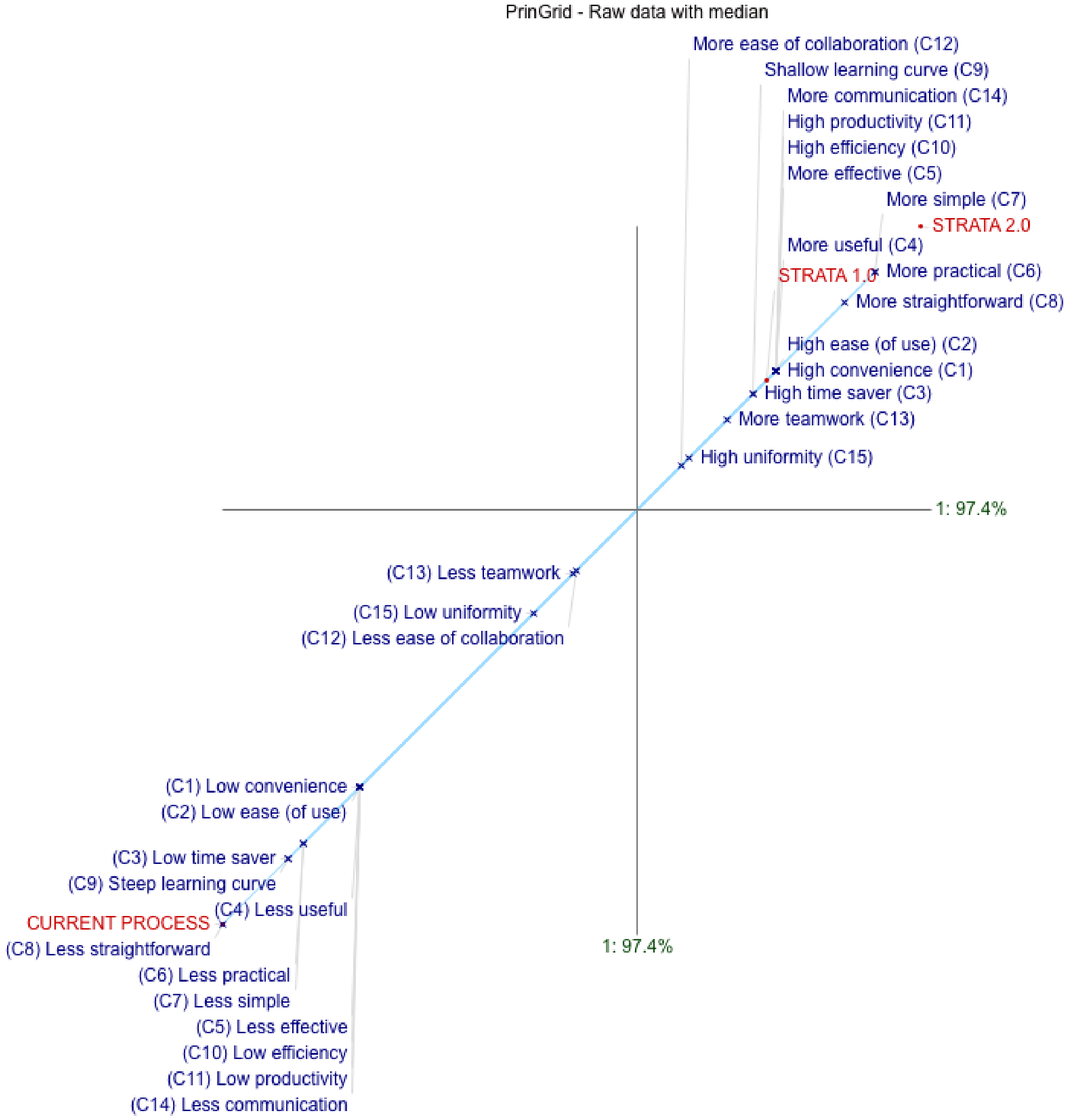
PrinGrid - Line manager



Display - Line manager

	CURRENT PROCESS	STRATA 1.0	STRATA 2.0	
(C1) Low convenience	2	3	5	High convenience
(C2) Low ease (of use)	1	3	5	High ease (of use)
(C3) Low time saver	1	3	5	High time saver
(C4) Less useful	1	3	5	More useful
(C5) Less effective	2	3	5	More effective
(C6) Less practical	1	3	5	More practical
(C7) Less simple	1	3	5	More simple
(C8) Less straightforward	1	3	5	More straightforward
(C9) Steep learning curve	1	4	2	Shallow learning curve
(C10) Low efficiency	1	3	5	High efficiency
(C11) Low productivity	1	3	5	High productivity
(C12) Less ease of collaboration	1	3	5	More ease of collaboration
(C13) Less teamwork	1	3	5	More teamwork
(C14) Less communication	1	3	5	More communication
(C15) Low uniformity	1	3	5	High uniformity

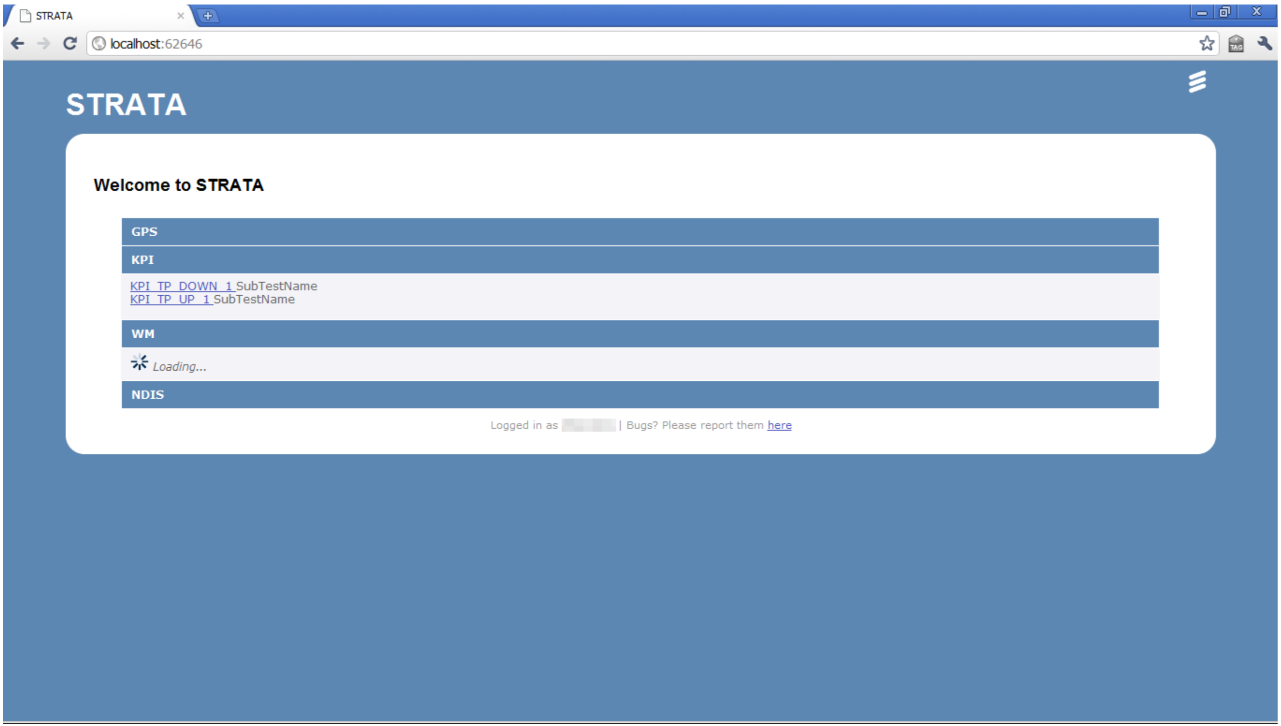
E Raw data with calculated median



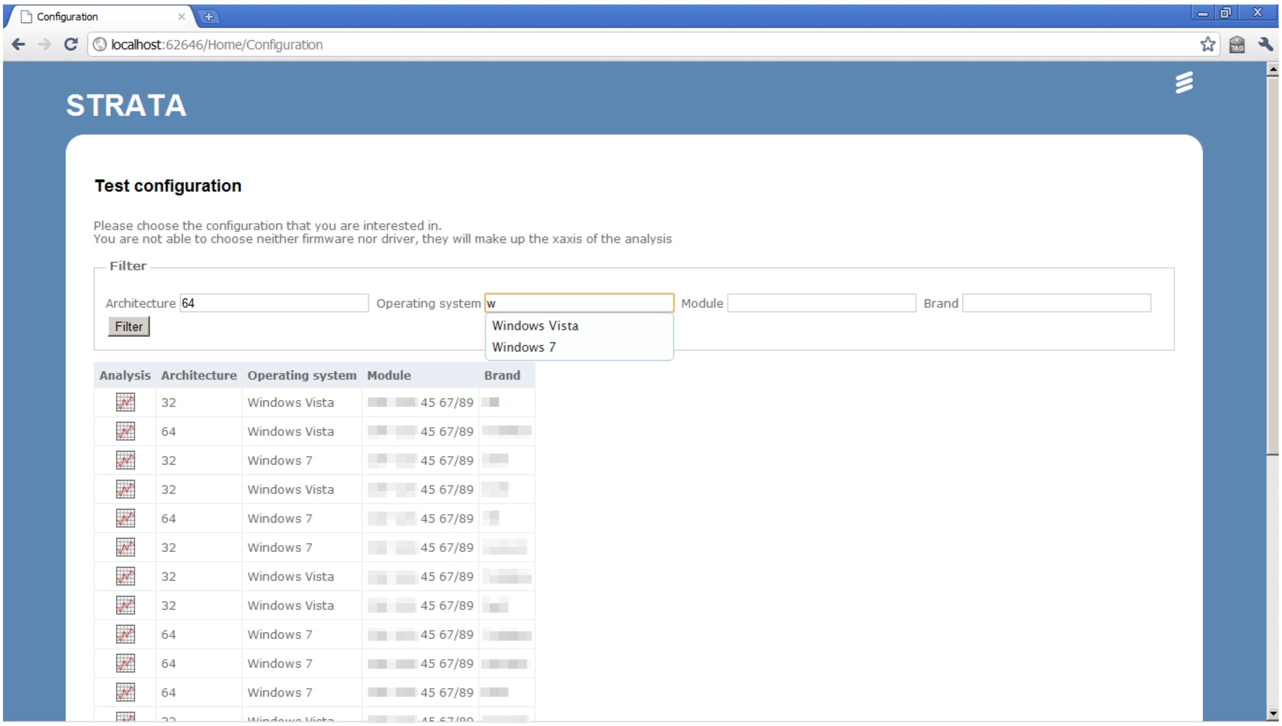
Display - Raw data with median

(C1) Low convenience	2	4	5	High convenience
(C2) Low ease (of use)	2	4	5	High ease (of use)
(C3) Low time saver	2	5	5	High time saver
(C4) Less useful	2	4	5	More useful
(C5) Less effective	2	4	5	More effective
(C6) Less practical	1	4	5	More practical
(C7) Less simple	1	4	5	More simple
(C8) Less straightforward	1	5	5	More straightforward
(C9) Steep learning curve	2	5	5	Shallow learning curve
(C10) Low efficiency	2	4	5	High efficiency
(C11) Low productivity	2	4	5	High productivity
(C12) Less ease of collaboration	3	3	4	More ease of collaboration
(C13) Less teamwork	2	3	3	More teamwork
(C14) Less communication	2	4	5	More communication
(C15) Low uniformity	3	4	4	High uniformity
	CURRENT PROCESS	STRATA 1.0	STRATA 2.0	

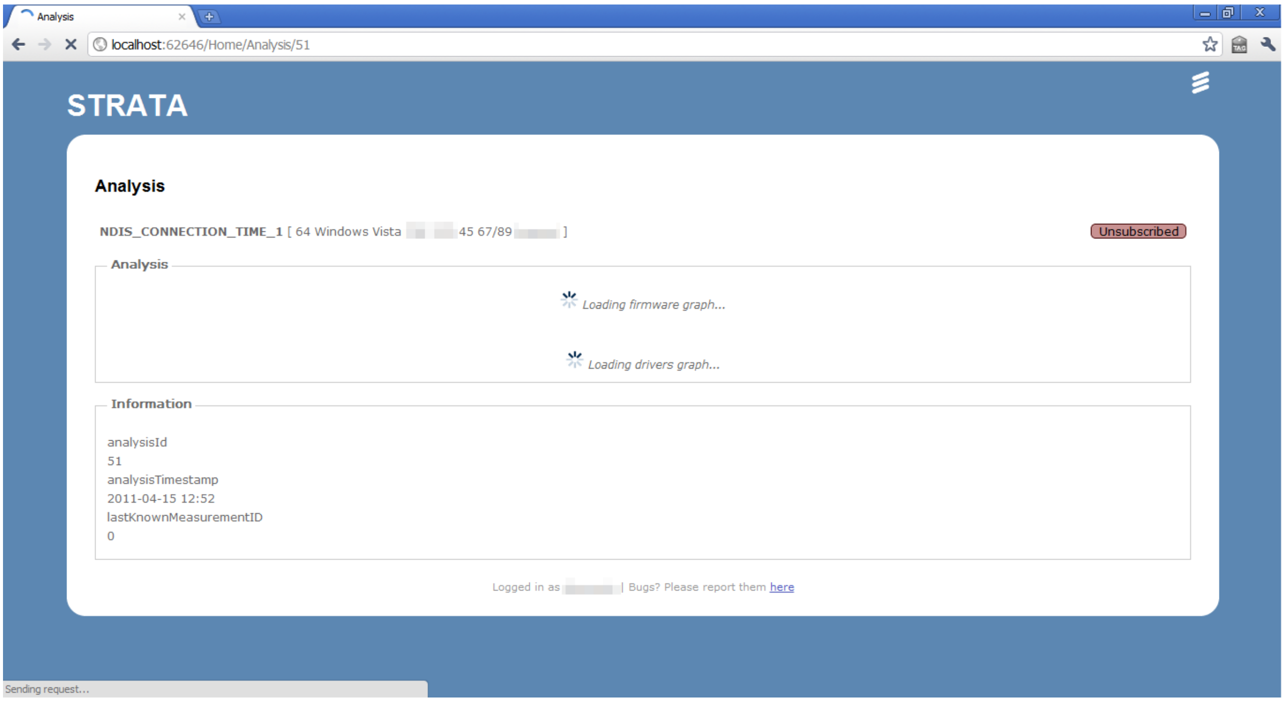
F Screenshot - Front page



G Screenshot - Configurations



H Screenshot - Analysis (loading)



I Screenshot - Analysis