ACSIJ Advances in Computer Science: an International Journal, Vol. 4, Issue 1, No.13 , January 2015
ISSN : 2322-5157
www.ACSIJ.org

ACSIJ
WWW.ACSIJ.ORG

Study of Automated and Real-time Indicators for the Management of Global Software Development Projects

Maarit Tihinen[1], Päivi Parviainen[2], Tanja Suomalainen[3] and Juho Eskeli[4]

[1] VTT Technical Research Centre of Finland Ltd
Oulu, PO BOX 1100, 90571, Finland
maarit.tihinen@vtt.fi

[2] VTT Technical Research Centre of Finland Ltd
Espoo, PO BOX 1000, 02044 VTT, Finland
paivi.parviainen@vtt.fi

[3] VTT Technical Research Centre of Finland Ltd
Oulu, PO BOX 1100, 90571, Finland
tanja.suomalainen@vtt.fi

[4] VTT Technical Research Centre of Finland Ltd
Oulu, PO BOX 1100, 90571, Finland
juho.eskeli@vtt.fi

Abstract

Global Software Development (GSD) has become the norm in product development. In GSD projects, controlling and management activities are increasingly important as the products are developed in dynamic environments where requirements, priorities, participating sites, development processes, tools and even partners are continuously changing. Up-to-date information on project status is critical to completing GSD projects effectively. The objective of the research reported in this paper was to make progress from fixed and inflexible measurements towards dynamic, reliable and up-to-date support for decision-making in GSD. Based on the study, it is shown that dynamic measurements are enabled via the use of automated and real-time indicators with consolidated information via visualised dashboards. The paper concludes that automatically produced real-time indicators, whose data are gathered from various databases and even from different stakeholders, are a robust and feasible method to support decision-making in GSD.

Keywords: *Measurements, Metrics, Distributed development, Global software development, Project management*.

1. Introduction

Today, global software development (GSD) is a common way to develop applications and products. In fact, geographically dispersed work is a fundamental trend that has shaped software development over the past few decades [1]. Nowadays, software is a critical aspect of our society, as it is the core element of many modern products, processes and services [2]. Furthermore, the complexity and size of software have dramatically increased over time. GSD aims to meet the increasing demands for software development productivity by utilising global resources effectively and also to achieve cost advantages in order to deliver projects effectively and efficiently [3, 4]. Furthermore, GSD is being carried out more due to the availability of sophisticated development tools and network connections that enable effective collaboration across the globe. GSD means software engineering that is carried out in globally distributed settings in various geographical locations. The work can be done either within a company (multi-site development) or in collaboration between two or more companies in different locations. GSD, as used in this paper, refers to product-development activity that involves two or more companies, departments or teams that combine their competencies and technologies to create new shared value while, at the same time, managing their respective costs and risks. The entities can combine in any one of several different business relationships and for very different periods of time (adapted from [5]).

In practice, GSD projects struggle with the same problems that single-site projects do, including problems related to managing and controlling resources, quality, schedule and cost. However, distribution makes it harder to manage those problems [6-10]. For example, Bjarnason et al. [11] have pointed out that coordination and communication are vital factors in the success of a software project and in delivering the required software on time and within budget. GSD project challenges are caused by various issues, for example, less communication – especially informal communication – due to the distance between partners and

differences in the background knowledge of the partners. Thus, systematic project control and management activities are even more important in GSD.

Project management covers activities like planning, scheduling, organising, controlling and managing tasks and resources to achieve the successful completion of a set of specific project goals. Metrics and measurements support project management by providing information on project progress. In traditional measurement programs, metrics are defined at the beginning of the project and then measurement data have been collected in pre-scheduled periods, such as bi-weekly or once a month. In GSD, measurement processes and metrics have to be more dynamic because of the adaptive and changing nature of the development due to the various stakeholders and tools involved the collaboration. For example, in GSD, the data need to be collected from multiple sources, such as from different databases of various stakeholders, accurately and unobtrusively. In addition, the interpretation of measurements data is more complicated in distributed projects than single-site projects.

This paper discusses a study about the challenges in measurements in GSD and the ways to enable effective and useful metrics and measurements in GSD projects. The paper introduces the main challenges related to measurements and metrics in GSD and provides an example of technical implementation – a tool integration solution – for facilitating measurements with automated and real-time indicators in GSD. This paper points out that automatically produced real-time indicators, whose original measurement data are gathered from various databases and even from different stakeholders, provide, in practice, feasible support for decision-making in GSD.

The paper is structured as follows. First, the background of the research and the literature review of the related work are introduced in Section 2. The research design and methods used during the research are described in Section 3. Then, the challenges and the proposed solution are discussed in Section 4. After that, in Section 5, the research results are discussed along with the limitations of the research. Finally, conclusions are drawn in Section 6.

2. Background and literature review

The main purpose of measurements and metrics is to provide support for decision-making during software development [12]. The success of software development depends highly on providing the right knowledge at the right time, at the right place and for the right person [13]. In the literature, there are several papers that discuss GSD and its challenges, for example [14-16]. In addition, some papers focus on providing insights [17] or better understanding [11] which can be used by software organisations to improve their software practices in GSD. Moreover, metrics in general as well as for specific purposes have been discussed in numerous papers and books for decades. However, little GSD literature has focused on metrics and measurements or even discusses the topic. Da Silva et al. [18] report similar conclusions based on an analysis of the distributed software development (DSD) literature published from 1999–2009: they state as one of their key finding that the "*vast majority of the reported studies show only qualitative data about the effect of best practices, models, and tools on solving the challenges of DSD project management. In other words, our findings indicate that strong (quantitative) evidence about the effect of using best practices, models, and tools in DSD projects is still scarce in the literature*".

The papers that have discussed metrics for GSD usually focus on some specific aspect; for example, Korhonen and Salo [16] discuss quality metrics to support the defect management process in a multi-site organisation. Kumar et al. [19] and Yeresime et al. [20] introduce and review metrics for object-oriented design. Furthermore, Simmons and Ma [21] discuss a software engineering expert system (SEES) tool, where the software professional can gather metrics from case tool databases to reconstruct all activities in a software project, from project initiation to project termination. Misra [22] presents a cognitive weight complexity metric (CWCM) for unit testing in a GSD environment. Lotlikar et al. [3] propose a framework for global project management and governance, including some metrics with the main aim of supporting work allocation to various sites. Peixoto et al. [15] discuss effort estimation in global software engineering, and one of their conclusions is that "*GSD projects are using all kinds of estimation techniques and none of them is being consider as proper to be used in all cases that it has been used*", meaning that there is no established technique for globally distributed projects. In addition, some effort has also been invested in defining how to measure the success of GSD projects [23], and these metrics mainly focus on cost-related metrics which are done after project completion. Thus, in this paper, the GSD related challenges in current measurement practices are discussed and the tools used and the need for tool support in GSD are also introduced.

Buse and Zimmermann argue [24] that the information currently delivered by existing tools to project managers is not meeting their needs. In reality, managers must rely primarily on past experience and intuition for critical decision-making when data needs are not met, either because tools are unavailable, too difficult to use, too hard

to interpret or they simply do not present useful information on which to base decisions. According to [25] there is an urgent need for measurements of large-scale software systems, which creates a great challenge for computer science. Furthermore, nowadays there is so much data available in different databases regarding distributed projects that it is impossible to manually browse through it all. Thus, data collection and metrics visualisation should be automated whenever possible.

Despite the recognised need for measurements, they are still challenging to implement in practice. For example, according to Umarj and Shull [14], project managers argue that it is time consuming to collect metrics for the organisation level while they actually need to have metrics that are relevant for tracking the project progress. They also suggest that there usually has not been enough time budgeted for measurements, and that is why it is really difficult to get approval from stakeholders for this kind of work [14]. In addition, globally distributed development creates new challenges in terms of the measurements. For example, problems in gathering measurements data are caused by different development tools and their versions, varying work practices and cultural differences, especially in subjective evaluations. It has also been noted that distributed projects are so unique in practice (e.g. the product domain and hardware-software balance vary or different subcontractors are used in different phases of the project) that the comparison between projects is impossible.

The business environment of GSD projects differs fundamentally from local development. This kind of product development offers many opportunities, such as potential savings in development times and costs as well as gaining a closer relationship with customers. However, it is at the same time highly challenging. General risks for GSD projects include lack of openness of communication between partners, unclear assignments or specifications in terms of the work in the contract [26], lack of trust between partners, difficulties on agreeing on intellectual property rights and the reliability of the partners' development schedule [27]. Recent empirical studies have confirmed that geographic distance can have negative effects on productivity [28]. From a measurement and metrics viewpoint, the challenges derived from business environments include, for instance, the lack of a transparent and unified monitoring process across all sites, no traceability between work items and poor or no interoperability of the tools used [7, 29]. Organisations have their own processes, practices and tools (tool versions), and they are unwilling to change them [7, 29]. For example, certain work practices or cultural differences themselves cause challenges while collecting or interpreting measurement data. If historical data are available, the utilisation of the data across the sites is difficult or even impossible.

According to [30, 31], GSD projects are loaded with challenges, and measurement data are needed to back up decision-making. The most relevant challenges – which should be addressed by measurements – in GSD projects that were collected from the literature include 1) communication breakdown, 2) coordination breakdown, 3) control breakdown, 4) cost of currently available tools, 5) poor interoperability between tools and 6) lack of traceability [32]. In practice, however, measurement programs suffer both from a lack of metrics standards that reduce data comparability and from invalid and missing data, which cause delays in data analysis and reduced confidence in reports' validity [33]. Although the methods, processes and tools for collecting and analysing measurement data have substantially improved, and large volumes of data and many types of metrics exist for project managers, software projects are still difficult to predict and risky to conduct [34]. In order to provide useful information for project control and decision–making, the metrics should be defined case by case to provide visibility to the stakeholders' in terms of progress and results. In addition, they should be followed on a daily basis, which means that automated and real-time indicators are needed for project management.

Automated measurements require tool support, and in practice, there are several tool providers that supply various solutions for managing GSD projects. However, most of those solutions have been developed to communicate only with tools from the same provider. In GSD, partners and stakeholders usually change according to new collaboration settings. Thus, there are several legacy tools used in companies that the companies are not willing to change. Nowadays, the marketplace for tools that support the specific tasks in the development process is very large, but the learning curve is high. Therefore, teams feel reluctant to include a new tool or change the tools that they are currently using as part of their development process even though this could sometimes lead to a better integration with the rest of the distributed team. Other types of tools which could support the whole development process in GSD are not selected because of their price. This is limiting for small and medium enterprises (SME) which are not willing or able to pay those types of prices, especially when new and various tools could be probably used in the next collaborative situation. Wicks and Dewar [35] define tool integration as follows: "*The tool integration concerns the techniques used to form coalitions of tools to provide an environment that supports some, or all, of the activities within a*

software engineering process". There have been some solutions developed to address this challenge [29, 36, 37]; however, they do not focus on the measurement viewpoint. Wu et al. [31] have introduced a metric-based multi-agent system for the continuous monitoring and tracking of project attributes from the viewpoint of the software project manager. They focus on monitoring and controlling distributed development collaboration task status via communication control and task progress control modules implemented within the framework. They use a graphic diagram to visualise the status of collaborative tasks, their relationships and the whole project's historical progress performance for an e-development project.

3. Research design

Järvinen [38] emphasises the possibilities of case studies to examine very complicated circumstances and, in this way, to gather new information for creating new knowledge. According to Yin [39], the single-case design is eminently justifiable if the case represents (a) a critical test of existing theory, (b) a rare or unique circumstance or (c) a representative or typical case, where the case serves a revelatory or longitudinal purpose. The environments and circumstances of measurements and metrics in each organisation are unique depending on cultural, historical and technical issues and backgrounds; production processes; or collaboration modes, for example. Thus, the use of case research was selected to illustrate experiences and to gather new information and gain an understanding of complicated circumstances. Furthermore, Järvinen suggests that in constructive research, it is possible to accept a prototype or even a plan as a research outcome instead of a final product. This approach was utilised during the research process because there is not merely one exact solution that is appropriate for all situations in GSD. Thus, a proof of concept tool integration solution was developed to enable the study of measurements and metrics in different collaborative settings (e.g. industrial experiences, such as the challenges and benefits). During the research, the data and information gathering methods were interviews, observations and e-mail inquiries as well as becoming familiar with the documentation, databases, tools and Intranet of the case study companies.

The research carried out included literature studies of the challenges in measurements and metrics practices in GSD. Moreover, industrial inventory was performed for identifying and analysing the challenges faced in GSD projects. In addition, several workshops with industrial partners were arranged for identifying and specifying an example solution and developing and analysing a proposed metrics set in detail. The tool integration solution PRISMA Workbench (PSW) was used as proof of concept of technical implementation that enables dynamic measurements in GSD. During the PRISMA project [40], the industrial partners utilised and evaluated the developed solution in their real-world distributed project or in their own multisite software development projects [41]. In addition, the tool integration was tried within a few demonstrations and research settings during the PRISMA project.

In practice, the actual set of tools and amount of people involved were dependent on the context and duration of each trial. In fact, those experiments and demonstrations were also fixed on testing implementations and functionalities of the tool integration solution and so focused on further developing the solution itself. For example, the AgileReq tool (proprietary tool of one case company) was integrated based on the company's needs. In fact, one principle for the development work had been that in GSD, companies have their own practices and tools, and thus the tool integration solution should not enforce a specific process or tool set. Furthermore, feedback related to managerial issues and measurements in GSD were gathered during the experiments. It was reported that the tool integration solution offered better visibility to stakeholders – as well as of project progress – through tool support in communication, project management etc. during the trial project. Also, resource management was assessed to be more efficient because of better transparency (traceability of design actions, awareness etc.) between sites and stakeholders.

4. Measurements in GSD

This section discusses the challenges in measurements in GSD, tool integration as a solution to gather and visualise data in a GSD setting and finally dynamic measurements that enable real-time and automated indicators that support project management and the decision making in GSD.

4.1 Challenges

Umarj and Shull [14] identified several challenges faced by project managers while trying to implement measurements in practice. In the following, those challenges are introduced from a GSD perspective. While analysing the challenges in detail, it can be seen that most challenges in current measurements practices in GSD focus on the same topics as the challenges identified in one-site development. However, in the GSD setting, the challenges appear to be more complicated and new GSD-specific challenges were also identified, as follows:

C1. ***Relevance of measurements in relation to project progress***. Generally, partners and stakeholders vary between GSD projects. Each partner or stakeholder has their own needs for measurements. Also, there is a lack of a transparent and unified monitoring process across all the sites and little traceability between work items. Thus, in GSD, the relevance and the purpose of all measurements for all partners in relation to project progress is more difficult to see than in one-site projects.

C2. ***Extra work budgeting – Metrics design/selection & data collection***. In GSD, there is a need to gather measurement data from multiple sources – that is, from various partners' tools and databases. Without automation, this can mean that partners are required to collect measurement data and manually transfer data items to another tool. Without extra budgeting, partners cannot allocate enough time or resources to do the measurements properly. In addition, this kind of manual process can lead to frustration and mistakes or errors in the data. However, in GSD, the need for dynamic measurements is substantial: the metrics need to be defined case by case, and they should be followed on a daily basis.

C3. ***Data reliability caused by tools.*** In GSD, there are several legacy tools used in companies which they are not willing to change. Some of the tools are even tailored to individual partners' needs. This often means that there is no or poor interoperability of the tools used. In addition, the tools used vary based on the collaborative setting. Thus, in the worst cases, the measurement data are manually collected by stakeholders, which results in extra work (C2) and a poor understanding of measurements relevance (C1).

C4. ***Data-reliability caused by human beings.*** In GSD, work practices can vary by project partner, and the integrity and even reliability of the gathered data can also vary due to cultural differences, especially in subjective evaluations. The challenge is the same as in one-site development, but it needs to be continuously taken into consideration as partners and stakeholders change according to new collaborative settings.

C5. ***Extra work budgeting – Metrics interpretation.*** The interpretation of measurements data generally requires extra work. In GSD, the interpretation and decision-making based on measurements results are more complicated than in one-site development. For example, the decision-making requires that the interpreted information be gathered from several sources (metrics). Thus, the interpretation should be made as easy as possible through the use of combined metrics and visualised graphs.

C6. ***Training needs.*** In GSD, cultural differences and differences in background knowledge as well as stakeholders' opinions and needs for measurements should be considered in more detail when planning and implementing trainings. With successful trainings, it is possible to reduce the data reliability problems caused by human beings.

C7. ***Measurements' effect on behaviour.*** The challenge is the same in one-site and in collaborative settings: the measurements should not affect people's behaviour. This requires a careful metrics design and also trust between the partners. However, openness of communication between partners and trust are not self-evident in GSD and thus rather require specific attention.

C8. ***Metrics ethics.*** The challenge is the same in one-site and in collaborative settings.

C9. ***Responsibilities and roles are unclear.*** Because of the various partners with diverse work practices in GSD, there can be unclear assignments or specifications in terms of the work to be done. In the worst case, unclear responsibilities and roles can cause a situation where people don't know from whom they should collect measurement data and, after analysis, to whom they should report results – in other words, who is responsible for each task. In fact, unclear responsibilities often lead to a situation where no one takes up the task, and so metrics can contain missing data, indicators can show a wrong result and measurements cannot be utilised at all.

C10. ***Continuous changes.*** In GSD, partners and stakeholders change according to new collaborative settings, and that in itself creates new challenges for measurements and metrics: dynamic, reliable and up-to-date support for the decision-making process is needed.

Some of these GSD-related challenges are strongly dependent upon each other. The challenges describe the situation from different viewpoints which are all important to take into consideration while trying to find solutions to measurements problems in GSD.

Useful metrics for GSD projects were defined based on the experiences of industrial partners from several research projects [4], and they were also successfully used in GSD projects [42]. The metrics used by the PRISMA industry partners included, for example, 'test coverage' and 'test results', 'efforts spent', 'number of errors revealed' and 'requirement changes count'. The metrics were related to the processes where tool support needs in collaborative development were highlighted. In practice, the tools used in measurement processes were custom, self-developed tools, Excel spreadsheets and MS SharePoint.

The measurements that are useful for GSD projects include standard project management metrics, but the data collection and interpretation are different. As an example, the following figure (Fig. 1) shows a metric example: Budget Status. This kind of metric offers better visibility to the stakeholders regarding progress and results by providing real-time and visualised information. In GSD, the most useful metrics can change case by case depending on each collaboration setting, the stakeholders involved and their roles in the project, the tools used in development, coordination and communication etc. The Budget Status metric is one example that was monitored very often in the GSD projects of PRISMA's industrial partners. This paper is not focused on introducing the most used metrics in GSD but rather on discussing the dynamic measurements concept via the illustrative examples of the proposed metrics.

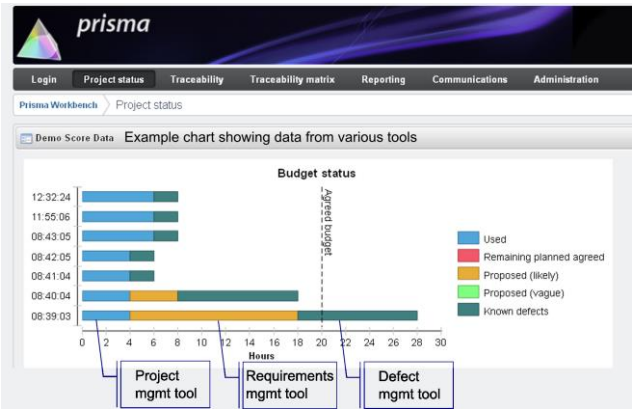

Fig. 1 A graph based on data retrieved from the integrated tools.

In GSD, the example real-time metric provides great advantages for daily project management and decision-making because the views are generated automatically and continuously, for example, on a daily basis. Thus, the project manager can control several processes – used efforts, requirements and testing processes – via one visualised graph even beyond partners' borders. The Budget Status graph shows the actual costs of the project in proportion to the agreed budget over a specific time period. The metric also gives several dynamic indicators of estimated prospective costs in each pre-scheduled period. The bars summarise the amount of costs, and each bar is composed of different cost-related data. The first block (blue) describes the *actual cumulative costs* of the project. The *agreed budget* for the project is shown clearly as a black dashed line in the middle of the graph. The red block describes the *remaining planned cost* based on the effort estimated for the requirements that have been *accepted* for implementation but not yet implemented. The orange block indicates the *proposed cost* that can be seen as very likely costs for the project. These costs are based on the effort estimated for the proposed requirements that are *estimated as likely* to be implemented, for example, a customer will want them. The green block describes the *proposed but vague costs* for the project. These costs are based on the effort estimated for the proposed requirements for which the likeliness of implementation is not known. Instead, the dark green block indicates very likely costs for the project, so-called *known defects costs*. The industrial experiences of use and benefits of these kinds of metrics are introduced in [42].

4.2 Tool integration solution

In this section, the tool integration solution PRISMA Workbench (PSW), which was developed during the PRISMA project, is briefly presented. A more in-depth description can be found from [41]. The solution allows the connection of software development tools to create company-specific software development environment instances. During the PRISMA project, the tools used as well as the need for development tools and processes that require more support in GSD were studied. Regarding tools, we analysed what kinds of artefacts, tasks or items would need to be integrated or shared by views between the partners during collaborative and distributed development, and what would be the most important issues that should be integrated in a tool integration solution. The needs focused on requirements capture and review processes, traceability needs, testing processes and project management and controlling tasks (including metrics capturing and analysing needs) [32].

One of the major design goals of PSW was to create a flexible and extendible solution that allows a configurable set of development tools to be tailored to the needs of individual partners or projects. The solution was designed and developed so that the legacy tools in the companies already in use could be easily integrated into the proposed PSW. The benefit of this type of tool integration approach is that partners can continue to use the tools they are familiar with. It is also important to remain effective even with a sudden change of tools. The caveat is that the integration of the tools has to be considered on a per tool basis. However, PSW removes the need to create point-to-point integrations between each of the tools because the solution can act as a hub where tools are connected via the integration interface. Another primary goal for PSW was that the integration of new tools had to be as easy as possible [41].

The tool integration solution was a server application developed in Java for building customised tool integration instances. The selected architecture was Service Oriented

Architecture (SOA), which provided several services implemented by tools (e.g. project management, test management, version management and requirements management). Also, a set of core services such as authentication, security and management of traceability between work items was implemented in the server application. The tool integration server was implemented using Apache Tuscany, as it provided support for implementing service-oriented architecture (SOA) and the required infrastructure for the easy development and running of applications using a service-oriented approach. The user interface via a web portal was implemented with the open source enterprise portal software Liferay, where all functionality was provided via portlets. The architecture of the framework is shown in Figure 2.

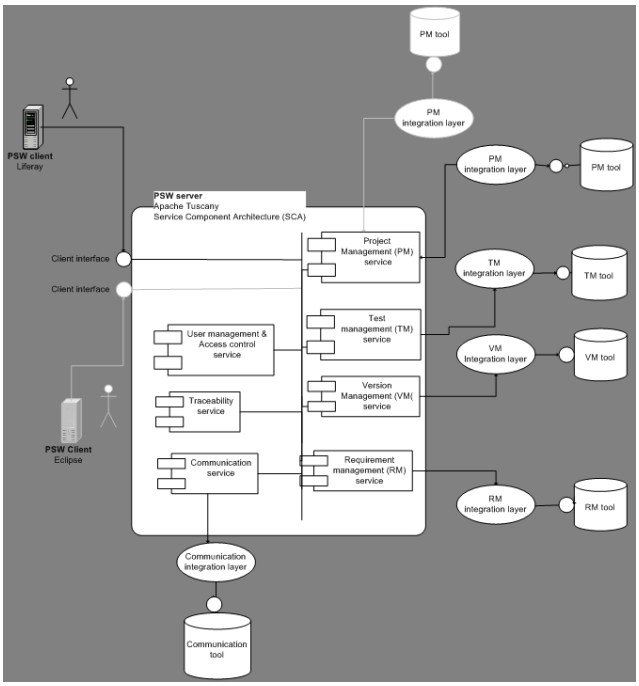

Fig. 2 The architecture of the PSW.

In the tool integration, the connections between work products (i.e. requirements) can be managed via traceability relations. The relations indicate some type of dependency between the work products. For example, a relation between a requirement and a test case can indicate that the given requirement is validated by the related test cases. The PSW provides the means to identify these relations and views to visualise them. The relations can also be used in reports, for example, to show requirements test coverage, requirements test status etc. This kind of feature is especially useful in maintaining control of the project when the work products that should be logically dependent are managed in separate tools (and/or sites).

4.3 Dynamic measurements and metrics

In dynamic measurements, metrics are defined based on the needs of each project, measurement data are collected and analysed continuously from various tools and databases (even from different stakeholders' databases) and measurement data are analysed and visualised for an easy-to-read format. In dynamic measurements, the metrics, the databases used and the analysed results of measurement data can be created, updated and changed dynamically based on the needs and setup of each project. Tihinen et al. [42] introduced a set of essential metrics that were successfully used in GSD. Some of those metrics were also implemented in the tool integration solution. The tool integration solution provides several real-time views into data that had previously been stored in various data sources, even in separate stakeholders' databases. The views were designed to increase transparency by providing visibility regarding the issues encountered in GSD. The data visualisation need was one driver for the views developed.

Regarding dynamic measurements in GSD, the Budget Status metric (shown in Fig. 1) demonstrates the indicator in which measurement data are collected from the project management tool (used effort), the requirements management tool (effort estimated for proposed vague, proposed likely and remaining planned & agreed requirements) and the defect management tool (effort estimated for known defects). The tool integration solution provided the means to identify these relations and the ways to visualise them. In GSD, these kinds of attributes are typically processed and managed in separate tools and even in different sites or are reported manually. The tool integration solution provides project status visibility from the actions of each development group to the whole development team as if everybody was working in the same location.

Some metrics correlate with each other, for example, metrics relating to tests correlate with metrics about requirements, and this needs to be taken into consideration when analysing measurement data. In fact, the real benefits of the integration solution and produced views are generated when several views can be read simultaneously. In the following figure (Fig. 3), the metrics set by the tool integration are shown in a dashboard view. The view aims to provide visibility into the project's progress by gathering and formatting data from various tools, identifying connections between the data and consolidating this information into easy-to-read dashboards.

Fig. 3 A dashboard illustration of the project status metrics.

The developed version of the tool integration solution contains a set of metrics that was identified as being beneficial for stakeholders during distributed product development. One important aspect was that the metrics should be captured from the tools used "for free" at regular intervals (meaning that no data needed to be filled in just for the sake of the measurements). Costs and budgets are good examples of metrics that can be easily captured from the tools. In addition, partners were able to define new metrics/views by utilising measurements data gathered via the tools integrated into the PSW. This was one important requirement for the solution in order to enable dynamic measurements in GSD.

In general, the interpretation of a project's comprehensive status needs a variety of metrics information – like requirements status, progress status, testing status and budget status – for making decisions based on the data. In addition, while interpreting or making decisions based on the measurement results, the distributed development implications need to be taken into account. Distributed development requires "super-balancing" – how to come to the right corrective action if, for instance, on one hand, the percent of not-accepted requirements is high and, on the other hand, the number of passed tests is lagging behind. Furthermore, it is important to consider the subjectivity of metrics in that, for example, effort estimation and differences between the backgrounds of the people (cultural or work experience) in different sites may affect the results.

5. Results and discussion

The dynamic measurements and their benefits are discussed in more detail in this section. In GSD, the dynamic measurements are important because of the nature of the work in GSD due to the various stakeholders and tools involved the collaboration. The dynamic measurements increase the transparency of GSD projects, as they provide constant visibility to the stakeholders regarding progress and results. The implemented tool integration solution enables the measurement data gathering and analysis combined from the various tools used in the projects. This enables partners to work together without having to significantly change the current environment, tools and processes.

The following table (Table 1) presents how the implemented tool integration solution with dynamic measurements can tackle the challenges discussed in section 4.1 and the kinds of benefits the dynamic measurements provide in GSD.

Table 1: The benefits of the dynamic measurements in GSD

C1: Relevance of measurements in relation to project progress: The views can be generated in real-time based on the needs of the various roles, such as project manager, test manager etc. in different GSD settings. From the viewpoint of the measurements' relevance, customisation is a key advantage. For example, the metrics can be defined together with stakeholders, ensuring that the importance of the generated metrics has been agreed upon.
C2: Extra work budgeting – metrics design/selection & data collection: Although metrics design/selection always requires additional work, the tool integration solution can help to reduce the workload via a pre-defined set of metrics. However, the proposed tool integration solution enables dynamic measurements in GSD. The solution enables the defining of new metrics by utilising measurements data gathered via the integrated tools. In addition, measurement data are automatically collected after the metrics and views definition, substantially reducing the amount of extra work for data collection.
C3: Data-reliability issues caused by tools: The tool integration solution allows a configurable set of development tools and their versions to be tailored for generating real-time indicators of individual project needs. In GSD, the benefits of using the legacy tools are major: the cost of investment of new development tools is typically too high. In addition, people are familiar with their own tools and technology, and they tend to resist changing their working platform as this takes additional time and causes extra work, too.
C4: Data-reliability issues caused by human beings: The tool integration cannot affect cultural differences; those divergences should be understood by the persons doing the measurements analysis. However, the automation of data collection and production of real-time indicators minimises data-reliability problems, as they can be detected early due to frequent measurements. Having to collect data manually can induce frustration, especially if the measurements are not understood as relevant to a person's own work. Manual collection can also lead to errors or mistakes when transferring data from one tool or form to another. The visualised graphs and consolidated information in easy-to-read dashboards make the analysis and metrics interpretation in GSD easier.

C5: Extra work budgeting – metrics interpretation: Interpretations and decisions related to actions based on the data always need to be done by humans, so the dynamic measurements do not change that. However, the automated data integrations from various tools and sources, visualised in easy-to-read dashboards, make the interpretation easier and enable the sharing of the interpretation work with partners. Thus, the dynamic measurements reduce the effort required for creating the material and make the analysis and metrics interpretation in GSD easier.
C6: Training needs: The dynamic measurements automate data collection for metrics and thus reduce the need for manual gathering and transferring of measurement data. This also reduces the need for training. However, there will always be training needs, specifically when there are differences in culture or background knowledge between partners.
C7: Measurements' effect on behaviour: The dynamic measurements partly automate measurements' actions. As data are collected from tools where data are collected as part of product development work and not as a separate action, there is less chance of affecting people's behaviour. However, it is still difficult to avoid affecting people's behaviour: "You will get it what you are measuring" still holds even with dynamic measurements.
C8: Metrics ethics: This challenge is merely a management-level issue and cannot be tackled by any external solution. However, measurement programs are more open when metrics data are made available via dynamic measurements.
C9: Responsibilities and roles are unclear: The dynamic measurements as such cannot address this kind of confusion entirely. When defining metrics and creating views, the need for metrics data is clear. In addition, unclear responsibilities or roles relating to actual measurements tasks such as data gathering, transferring, analysing and reporting have to be clarified. Furthermore, the dynamic measurements can indicate problems in responsibilities and roles via visualised and easy-to-read dashboards.
C10: Continuous changes: The dynamic measurements are essential to increasing transparency and creating real-time views between partners in GSD. Per definition, dynamic measurements are measurements that can be changed continuously based on identified needs. Dynamic measurements enable the integration of measurements data from various tools and databases as well as consolidate information into easy-to-read dashboards.

Measurements are always knowledge intensive, and so human efforts are required. In particular, metrics design and metrics visualisation have to be carefully planned. Also, metrics interpretation always requires the investment of individuals. However, it is possible that the amount of work required to be done by humans can be reduced or minimised with the proposed tool integration. The tool integration solution enables dynamic measurements in GSD. The dynamic measurements can help with most challenges related to measurements and metrics in GSD projects; metrics can be defined or updated based on the needs of each project and the demands of each project's collaboration setting. Furthermore, metrics data are automatically and continuously collected and analysed from various tools and databases, and even from different partners, thus providing visibility regarding the partners' progress. The management and control of GSD projects are demanding and complicated. Project managers should be able to perform various kinds of analysis on project data to monitor the project and take corrective actions when needed. Dynamic measurements supported by a tool integration solution offer better visibility beyond partners' borders as well as into project progress through tool support in communication, project management etc. Moreover, resource management can be supported via better transparency (traceability of design actions, awareness etc.) between sites and stakeholders.

6. Conclusions

The measurements and metrics create a means for controlling and managing a project and provide support for decision-making. This is particularly important but also challenging in GSD. Traditionally, organisations have a set of metrics that are followed in all projects. Those metrics may be updated based on the demands of development projects. In GSD, in order to provide useful information for project control and decision–making, the metrics need to be defined case by case, and they should be adapted when needed.

In this paper, dynamic measurements are defined as measurement actions where metrics are defined or updated based on the needs of each project and the demands of each project's collaboration setting. In addition, metrics data are collected and analysed continuously from various tools and databases, and measurement data are analysed and visualised. In dynamic measurements, the metrics, the databases used and the analysed results of measurement data can be created, updated and changed dynamically based on the monitoring needs of each project and the different phases and tasks. In this paper, the dynamic measurements were introduced and analysed according to the GSD-related challenges faced in current measurement practices. In addition, the benefits of the dynamic measurements during GSD projects were discussed via a proposed tool integration solution.

The tool integration solution was used as an example to show how project management can be supported by dynamic measurements in GSD. In dynamic measurements, metrics data are collected and analysed continuously from various tools and databases, and measurement data are analysed and visualised. The tool integration solution as introduced in the paper enables this aspect of dynamic measurements. The solution helps to create dynamic measurements in environments where partners, subcontractors, in-house developers etc. change according to project, and accordingly, in environments where partners' development methods, tools and tool versions change according to project.

Future research actions could be divided into theoretical and empirical research approaches. From the theoretical viewpoint, future research could include a large industrial survey of dynamic measurements in the context of GSD projects, for example. The paper discussed problems in

current measurements and metrics practices with studies of practical solutions developed and proposed to meet those challenges in GSD projects. From an empirical viewpoint, future research actions could focus on gathering experiences from various kinds of GSD environments and projects. For example, dynamic measurements could be implemented in several different GSD projects to provide real industrial environments to further examine the challenges, needs and potential solutions. One potential option could be to implement the tool integration solution in an industrial product development environment and then gather experiences via action research methods and questionnaires over a several-year period.

Acknowledgements

The dynamic measurements and experiments of the tool integration introduced in this paper were developed within the PRISMA project that was an ITEA2 project, number 07024 (PRISMA, Productivity in Collaborative Systems Development). The results were further analysed during the PROMES ITEA2 project, number 11013 (PROMES, Processes Models for Engineering of Embedded Systems) from the viewpoint of the measurements and management of distributed product development. The authors would like to thank all the partners involved in the projects for their valuable contributions and feedback during the process. Furthermore, the authors would like to thank the support of ITEA (ITEA, Information Technology for European Advancement) and Tekes (The Finnish Funding Agency for Innovation) for enabling the research.

References

[1] M. Cataldo and S. Nambiar, "The impact of geographic distribution and the nature of technical coupling on the quality of global software development projects", Journal of Software: Evolution and Process, vol. 24, (2), 2012, pp. 153-168.

[2] A. Fuggetta and E. Di Nitto, "Software process", In Proceedings of the on Future of Software Engineering, 2014, pp. 1-12.

[3] R. M. Lotlikar, R. Polavarapu, S. Sharma and B. Srivastava, "Towards effective project management across multiple projects with distributed performing centers", In Proceedings of IEEE International Conference on Services Computing (CSC'08), 2008, pp. 33-40.

[4] P. Parviainen, "Global software engineering. Challenges and solutions framework", Doctoral Dissertation, VTT Science 6, Finland, 2012, pp. 106 p. + app. 150 p.

[5] R. Welborn and V. Kasten, The Jericho Principle: How Companies use Strategic Collaboration to Find New Sources of Value, John Wiley & Sons, 2003.

[6] J. D. Herbsleb, A. Mockus, T. A. Finholt and R. E. Grinter, "Distance, dependencies, and delay in a global collaboration", In Proceedings of the ACM Conference on Computer Supported Cooperative Work, 2000, pp. 319-328.

[7] J. D. Herbsleb, "Global software engineering: The future of socio-technical coordination", In Proceedings of Future of Software Engineering FOSE '07, 2007, pp. 188-198.

[8] M. Jiménez, M. Piattini and A. Vizcaíno, "Challenges and improvements in distributed software development: A systematic review", Advances in Software Engineering, vol. Jan-2009, (No. 3), 2009, pp. 1-16.

[9] S. Komi-Sirviö and M. Tihinen, "Lessons learned by participants of distributed software development", Knowledge and Process Management, vol. 12, (2), 2005, pp. 108-122.

[10] M. Tihinen, P. Parviainen, T. Suomalainen, K. Karhu and M. Mannevaara, "ABB experiences of boosting controlling and monitoring activities in collaborative production", In Proceedings of the 6th IEEE International Conference on Global Software Engineering (ICGSE'11), 2011, pp. 1-5.

[11] E. Bjarnason, K. Smolander, E. Engström and P. Runeson, "Alignment practices affect distances in software development: A theory and a model", Proceedings of the 3rd SEMAT Workshop on General Theories of Software Engineering, 2014, pp. 21-31.

[12] V. R. Basili, "Software modeling and measurement: The goal/question/metric paradigm", Tech. Rep. CS-TR-2956, Department of Computer Science, University of Maryland, College Park, 1992, pp. 1-24.

[13] G. Ruhe, "Software engineering decision support – a new paradigm for learning software organizations", Advances in Learning Software Organization, Lecture Notes in Computer Science, Springer-Verlag, 2003, pp. 104-113.

[14] M. Umarji and F. Shull, "Measuring developers: Aligning perspectives and other best practices", IEEE Software, vol. 26, (6), 2009, pp. 92-94.

[15] C. E. L. Peixoto, J. L. N. Audy and R. Prikladnicki, "Effort estimation in global software development projects: Preliminary results from a survey", In Proceedings of International Conference on Global Software Engineering, 2010, pp. 123-127.

[16] K. Korhonen and O. Salo, "Exploring quality metrics to support defect management process in a multi-site organization - A case study", In Proceedings of 19th International Symposium on Software Reliability Engineering (ISSRE), 2008, pp. 213-218.

[17] M. Zahedi and M. A. Babar, "Towards an understanding of enabling process knowing in global software development: A case study", In Proceedings of the International Conference on Software and System Process, 2014, pp. 30-39.

[18] F. Q. B. da Silva, C. Costa, A. C. C. França and R. Prikladinicki, "Challenges and solutions in distributed software development project management: A systematic literature review", In Proceedings of International Conference on Global Software Engineering (ICGSE2010), 2010, pp. 87-96.

[19] M. Kumar, R. Thakur and M. Mann, "A tool for quality measurement of software based on object oriented design", International Journal, vol. 2, (5), 2014, pp. 140-144.

[20] S. Yeresime, J. Pati and S. K. Rath, "Review of software quality metrics for object-oriented methodology", In Proceedings of International Conference on Internet Computing and Information Communications, 2014, pp. 267-278.

[21] D. B. Simmons and N. K. Ma, "Software engineering expert system for global development", In Proceedings of 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'06), 2006, pp. 33-38.
[22] S. Misra, "A metric for global software development environment", In Proceedings of the Indian National Science Academy, 2009, pp. 145-158.
[23] B. Sengupta, S. Chandra and V. Sinha, "A research agenda for distributed software development", In Proceedings of the 28th International Conference on Software Engineering, 2006, pp. 731-740.
[24] R. P. Buse and T. Zimmermann, "Analytics for software development", Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research, 2010, pp. 77-80.
[25] Z. Mingguang, Z. Haohua, Q. Weiyi, M. Shijun and W. Chuanyi, "The measurement and evaluation for large-scale object-oriented software system", In Proceedings of the Ninth International Conference on Hybrid Intelligent Systems, HIS'09, 2009, pp. 70-73.
[26] M. Ruano- Mayoral, C. Casado- Lumbreras, H. Garbarino- Alberti and S. Misra, "Methodological framework for the allocation of work packages in global software development", Journal of Software: Evolution and Process, vol. 26, (5), 2014, pp. 476-487.
[27] J. Hyysalo, P. Parviainen and M. Tihinen, "Collaborative embedded systems development: Survey of state of the practice", 13th Annual IEEE International Symposium and Workshop on Engineering of Computer Based Systems (ECBS 2006), 2006, pp. 1-9.
[28] R. Colomo-Palacios, C. Casado-Lumbreras, P. Soto-Acosta, F. J. García-Peñalvo and E. Tovar, "Project managers in global software development teams: A study of the effects on productivity and performance", Software Quality Journal, vol. 22, (1), 2014, pp. 3-19.
[29] V. S. Sinha, B. Sengupta and S. Ghosal, "An adaptive tool integration framework to enable coordination in distributed software development", In Proceedings of the 2nd IEEE International Conference on Global Software Engineering (ICGSE), 2007, pp. 151-155.
[30] S. Komi-Sirviö, P. Parviainen and J. Ronkainen, "Measurement automation: Methodological background and practical solutions a multiple case study", In Proceedings of the Seventh International Software Metrics Symposium, METRICS 2001, 2001, pp. 306-316.
[31] C. Wu, W. Chang and I. K. Sethi, "A metric-based multi-agent system for software project management", In Proceedings of the Eight IEEE/ACIS International Conference on Computer and Information Science (ICIS 2009), 2009, pp. 3-8.
[32] J. Eskeli and J. Maurolagoitia, "Global software development: Current challenges and solutions", In Proceedings of the 6th International Conference on Software and Data Technologies, ICSOFT 2011, 2011, pp. 29-34.
[33] J. Lawler and B. Kitchenham, "Measurement modeling technology", IEEE Software, vol. 20, (3), 2003, pp. 68-75.
[34] I. D. Coman, A. Sillitti and G. Succi, "A case-study on using an automated in-process software engineering measurement and analysis system in an industrial environment", In Proceedings Of the 31st International Conference on Software Engineering, ICSE'09, 2009, pp. 89-99.
[35] M. Wicks and R. Dewar, "A new research agenda for tool integration", The Journal of Systems and Software, vol. 80, (9), 2007, pp. 1569-1585.
[36] E. Estévez, M. Marcos, U. Gangoiti and D. Orive, "A tool integration framework for industrial distributed control systems", In Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference 2005 (CDC-ECC'05), 2005, pp. 8373-8378.
[37] L. Wu, S. White, J. Helm and Y. Feng, "Distributed software development modelling and control framework", International Journal of Software Engineering & Applications, vol. 3, (5), 2012.
[38] P. Järvinen, On Research Methods, Tampere, Finland, Tampereen Yliopistopaino Oy, 2012.
[39] R. K. Yin, Case Study Research: Design and Methods, Los Angeles, SAGE Publications, 2009.
[40] PRISMA, Productivity in Collaborative Systems Development, PRISMA project (2008-2011) ITEA innovation report, URL: https://itea3.org/innovation-report/improving-productivity-for-globally-distributed-software-systems-development.html (Accessed 14.1.2015).
[41] J. Eskeli, J. Maurolagoitia and C. Polcaro, "PSW: A framework-based tool integration solution for global collaborative software development", In Proceedings of the 6th International Conference on Software Engineering Advances (ICSEA'11), 2011, pp. 124-129.
[42] M. Tihinen, P. Parviainen, R. Kommeren and J. Rotherham, "Metrics and measurements in global software development", International Journal on Advances in Software, vol. 5, (3 and 4), 2012, pp. 278-292.

Dr. Maarit Tihinen is a Senior Scientist in the digital services in context team at VTT in Oulu, Finland. She graduated in the department of mathematics from the University of Oulu in 1991. She worked as a teacher (mainly mathematics and computer sciences) at the University of Applied Sciences before coming to VTT in 2000. She completed her Secondary Subject Thesis in 2001 and received her PhD in 2014 in information processing science from the University of Oulu, Finland. Tihinen has worked in several national and international research and customer projects and has written scientific publications for international software engineering conferences and journals. Her research interests include measurement and metrics, quality management, global software development practices and digital service development practices.

Dr. Päivi Parviainen received her M.Sc. in information processing science from the University of Oulu in 1996 and her PhD in information processing science from the University of Oulu in 2013. She is currently working as a Principal Scientist in the digital services in context team at VTT in Espoo, Finland. She has worked at VTT since 1995. Over the years, she has authored over 30 publications. Her research interests include product and digital service development practices and global software development practices.

M.Sc. Tanja Suomalainen received her M.Sc. in information processing science from the University of Oulu in 2006. Thereafter, she continued her studies as a PhD student. She has worked at VTT since 2005, first as a research trainee and then after graduation as a research scientist. Her research interests include planning in software development, product roadmapping,

requirement management, global software development engineering and agile and lean software development.

M.Sc. Juho Eskeli received his M.Sc. in electrical and information engineering from the University of Oulu in 2009. He has worked at VTT since 2006 and is currently working as a research scientist in the smart lighting and integration concepts team in Oulu, Finland. His research interests include software development, software development tools, internet of things (IoT) and embedded systems.