IT University of Copenhagen

# A Systematic Mapping Study of Tools for Distributed Software Development Teams

**Paolo Tell**
**Muhammad Ali Babar**

**IT University Technical Report Series** **TR-2012-161**

**ISSN 1600–6100** **October 2012**

ISSN 1600–6100

ISBN 978-87.7949-266-0

Copies may be obtained by contacting:

IT University of Copenhagen
Rued Langgaards Vej 7
DK-2300 Copenhagen S
Denmark

Telephone: +45 72 18 50 00
Telefax: +45 72 18 50 01
Web `www.itu.dk`

# A Systematic Mapping Study of Tools for Distributed Software Development Teams

Paolo Tell, Muhammad Ali Babar
Software System Section
IT University of Copenhagen, Denmark
pate@itu.dk, maba@itu.dk

**Abstract**

*Context:* A wide variety of technologies have been developed to support Global Software Development (GSD). However, the information about the dozens of available solutions is quite diverse and scattered making it quite difficult to have an overview able to identify common trends and unveil research gaps.
*Objective:* The objective of this research is to systematically identify and classify a comprehensive list of the technologies that have been developed and/or used for supporting GSD teams.
*Method:* This study has been undertaken as a Systematic Mapping Study (SMS). Our searches identified 1958 papers out of which 182 were found suitable for inclusion in this study.
*Results:* We have identified 412 technologies reported to be developed and/or used for supporting GSD teams from 182 papers. The identified technologies have been analyzed in order to categorize them using four main classification schemas for providing a framework that can help identify the categories that have attracted significant amount of research and commercial efforts, and the research areas where there are gaps to be filled.
*Conclusions:* The findings show that whilst commercial and open source solutions are predominantly focused on specialized tools as well as platforms, research effort was concentrated on providing integrated environments, frameworks, and plug-in based solutions. Considering the findings in the context of previously proposed research agendas, some of the key challenges for GSD research (i.e., collaborative tools and innovative knowledge management systems) shows that lots of collaborative technologies have been reported, but knowledge management is being addressed by focusing on supporting aware-

ness, which is being considered as important as the three elements of 3C model (i.e., communication, collaboration, and coordination). We also conclude that future effort in this area should pay more attention to devising solutions which can fulfill several kinds of requirements necessitated by a broader set of challenges being faced by GSD practitioners rather than tackling individual issues.

*Keywords:* global software development, tool, technology, classification, systematic mapping study

---

**Contents**

## 1. Introduction

Geographical distribution of software development teams across multiple sites in one or multiple countries has become a widely accepted software development paradigm known as Global Software Development (GSD), also called Global Software Engineering (GSE) [69]. Companies have been increasingly adopting the GSD paradigm by moving their software development to places considered to be lower cost destinations, or to destinations where the required skills are more readily available [35]. As a result of the increasing popularity of GSD, more and more software development projects are being undertaken by following some kind of GSD model, thus entailing several unique aspects. In particular GSD projects are often characterized by geographical, temporal, cultural, and linguistic distances [103]. Such distances among GSD team members can result in problems such as miscommunication, delays in decision making, and a large coordination overhead [70, 35]. A key GSD challenge is thus to provide appropriate tools to help GSD teams to effectively and efficiently perform their software engineering tasks.

Given the critical role and importance of adequate tool support for GSD teams, a plethora of solutions have been provided by both industry and academia. The Software Engineering (SE) community has been particularly active in providing dozens of tools for supporting GSD teams (e.g., [PS101, PS148][137]). Considering the growing amount of literature on technologies for supporting GSD, a timely summary of all the relevant knowledge about different aspects of the reported GSD supportive technologies also becomes important in order to support future research in the area and practitioners' requirements for appropriate tool support for GSD projects.

This motivated us to carry out a comprehensive study aimed at systematically identifying, analyzing, classifying, and summarizing the technologies that have been developed and/or used for supporting GSD teams[1]. There-

[1] We came across a paper [112] that was made available on the 7th of March 2012 entitled "Tools used in Global Software Engineering: a Systematic Mapping Review", which supports our assertion of the increasing importance of the kind of work reported in this paper. A comparative discussion highlighting similarities and differences between the studies is presented in Section 2.

fore, we have performed a Systematic Mapping Study (SMS) that aimed at systematically identifying, analyzing, and classifying the technologies that have been reported in the peer-reviewed literature as being used or developed for supporting GSD teams. Through this mapping study, we have identified and classified 412 different technologies reported in 182 papers that were selected from the initially identified 1958 papers retrieved from five digital libraries. We have classified the identified 412 technologies using different classification schemes such as: the types of technologies, i.e., tools, environment, framework, plug-in, platform, and middleware; the so-called 3Cs model, which comprises the communication, collaboration, and coordination dimensions [47]; the software engineering activities supported by the identified technologies; and the type of license, thus, research outcomes, open source software (OSS), and commercial off-the-shelves products (COTS). This mapping study analyzes the current trends of the technological support available for GSD projects, and identifies potential gaps in the current available research. Moreover, by plotting the identified technologies along with the information about different categories in which they have been classified, we have provided a framework that is expected to help practitioners to identify and select appropriate technological solutions.

The remainder of the paper is organized as follows: Section 2 presents an overview of the related work and provides a detailed comparison of this work compared to a recently retrieved study [112]; Section 3 presents the methodology used, it contains all the details explaining how the systematic mapping study has been applied and the information necessary to replicate the study; Section 4 describes the outcomes following the categorization described in Section 3. Furthermore, Section 5 discusses the threats to the validity of this study; and, Section 6 presents the map summarizing the results and the conclusions. Finally, Appendix A is dedicated to present additional details regarding the concrete search strings used in each digital library, and Appendix B lists the references in which each of the technologies retrieved were found and the primary source of information (i.e., either the paper reporting the research project, or the reference web-site).

At this point it is important to explain how citations are used in this work. As mentioned briefly, 412 publications were found to be relevant to this mapping study; this set of articles—primary studies—has been indexed in an independent list of references recognizable for the string 'PS' prefixing the indexes.

Moreover, it should be noted that the set of primary studies in which a specific technology was found, may not contain the article that describes the technology. In such case, when referencing the tool, the reference used would be the most appropriate one, not necessarily the primary study. The set of primary studies in which it was retrieved, and the primary source of information for each technology are listed in Appendix B

## 2. Related Work

An increasing realization of the importance of providing appropriate tool support for GSD teams has resulted in dozens of research and commercial tools being developed and used [PS137, PS101, PS148]. Researchers have reported extensive surveys of tools for GSD teams [137]. During the execution of this systematic mapping study other secondary studies have been identified and retrieved. Their nature ranges from surveys and reviews to more systematic methodologies like mapping studies and literature reviews. Table 1 provides an overview of such publications. An overview of commercial and open source technologies and tools for distributed software development teams has been reported in [PS133]. Lanubile et al. have reported an overview of collaboration tools for global software engineering in [PS101]. Sarma et al. [119] have reviewed coordination technologies and proposed a framework to categorize them; a classification driven by the software life cycle has also been proposed [PS106]. The authors of [PS148] systematically identified and reported awareness support for GSD. The publication of extensive literature reviews and classification framework shows that there is a growing number of tools supporting GSD teams.

Table 1: Secondary studies overview

| Focus | Retrieval methodology | Ref |
|---|---|---|
| Collaboration tools | Manual | [PS101] |
| Tools and services | Manual | [PS106] |
| Technologies and tools | Manual | [PS137] |
| Awareness Support | Systematic literature review | [PS148] |
| Challenges and Solutions | Systematic literature review | [PS052] |
| Tools | Survey | [PS133] |
| Project management | Systematic mapping study | [PS054] |
| Awareness, theory and practice | Review | [PS125] |

Besides the above-mentioned survey papers (Table 1) that we came across during the search process of this mapping study, A paper reporting a mapping study of GSD tools [112] became available online when we were finalizing the reporting step of our mapping study. The two pieces of work were carried out independent of each other by two teams of researchers without any knowledge of each others' initiatives on the same topic using the same methodology. Despite the fact that both efforts were motivated by the same main research question (i.e., "which software tools are available to support Global Software Engineering?" [112]) and used the same research methodology, the two studies differ in several ways; which should provide the potential readers with an opportunity to benefit from interesting similarities and differences in the findings from the two independent efforts. In the following paragraphs, we provide a brief comparison of the two studies in order to highlight their similarities and differences; however, a detailed comparison between the two studies is out of the scope of this piece of work.

The authors of [112] decided to limit the search for papers published from 2000 to 2010. They argue that software technologies quickly become obsolete that was why they decided to start their search from 2000. However, we did not place such restriction as we believe that a thorough overview should try to incorporate older publications. Whilst we did find publications reporting technologies for which there was no publication dated later than 2000, there were certain software technologies that were initially published earlier than 2000 and were still being utilized by GSD practitioners. For example, we found 10 publications reporting Lotus Notes, the first of which dated 1996 [PS080], and the most recent ones were published in 2010 [PS043, PS137, PS133], or SLIM with two occurrences, first in 1997 [PS079] and the last in 2009 [PS168]). An obvious result of limiting the search from 2000 was that the technologies reported before 2000 did not appear in the findings reported in [112].

Another difference between two studies is the number of digital libraries searched. We decided to carry out our searches for the relevant papers in five well known digital libraries compared to four digital libraries used for [112], which left out Springer Link. Our decision to include Springer link led us to retrieve 246 potentially relevant papers from the Springer Link digital library; out of these 246 papers, 32 were included in our mapping study. The inclusion of the 32 relevant papers from Springer link resulted in more comprehensive list of the papers reporting technologies for GSD support.

Both studies also vary in terms of the categorization schemes used for classifying the identified technologies. In our case, we utilized four distinct main classifications schemes based on: the form of technological support, the widely accepted 3C model, the intent pursued by the technology, and the type of license (see Section 3.5). Whereas, their classification scheme is mainly based on the SWEBOK categories complemented with what have been called features as independent characteristics. Features, in their categorization, included license, communication, awareness, control and coordination, knowledge management, and socio-cultural. If on the one hand the SWEBOK categories might be considered a similar classification to one schema used in our study as focusing on the intent for which a technology has been designed or utilized; on the other hand, the classification schema based on the 3C model is considered as a framework for analyzing technologies. Moreover, we assert that the analysis based on the type of technology was found very useful in providing interesting insights about the trends and the directions followed by both industry and academia in the studied area. As an example, plug-in technologies have been heavily explored since 2006, and there has recently been a new workshop series collocated with the International Conference on Software Engineering (ICSE); for further discussions on trends, we direct a reader to the concluding sections of this article. Finally, both include tool license category; and, the authors of [112] have provided an additional analysis for each tool based on the presence of an evaluation in the context of GSD that helps understand the maturity of the technologies and of the field itself.

The two studies also differ in terms of the number of primary studies included in each of them. Our mapping study has incorporated 182 publications (66 in [112]) from which 412 unique software technologies were extracted (132 in [112]). Looking more carefully into the identified primary studies and tools, we found that out of 66 primary studies identified by the work reported in [112], 24 were also retrieved by our search (36%); further, out of 132 tools, 100 were also found by us (75%). However, analyzing these data from another perspective, even if our study considered no time frame for the initial publication date and have been updated in a second stage to update all the results up to October 2011, we retrieved 158 more primary studies and 312 additional tools. It has also to be emphasized that, even though looking at the search strings of the two pieces of work the one used in our study should be a superset thus retrieving all the publications retrieved by [112], 40 primary studies (61%) are missing from our search resulting in 32 tools. We investigated this issue by checking the publications that we retrieved but excluded but we could not find any more overlapping primary studies. Hence, we can conclude that one plausible explanation for this situation can be the reliability of the search engine provided by the searched digital libraries.

This comparison of the both studies also shows that some of our decisions (e.g., no limit on the search year and searching Spring link) enabled us to access more relevant papers that resulted in a more comprehensive overview of the current state of the art of the area—objective of a mapping study. For example, our study found Skype 7 times in publications dated later than 2006 (e.g., [PS007, PS127, PS123]) and Jazz Band 6 times in publications dated later than 2006 (e.g., [PS022, PS045, PS125]). Neither of these technologies were identified in [112]. One possible explanation for such omission can be the different choices in terms of keywords used for both studies.

## 3. Systematic Mapping Study

Since its introduction in 2004 [87], Evidence-Based Software Engineering (EBSE) has become a widely used research approach in software engineering. Learning from other disciplines like medicine, where large numbers of results are already reported and need to be identified and aggregated, Systematic Literature Reviews (SLRs) and Systematic Mapping Studies (SMSs) have been increasingly adopted.

SLRs and SMSs are two methodologies that slightly differ from each other. As clearly explained by Kitchenham in [86], the main difference between the two approaches lies in the scope of the investigated research questions. While during the conduction of an SLR the aim is to precisely answer one or more focused research questions analyzing and discussing the paper's results, in the case of SMSs the objective is to provide a broad overview of a research area, establish if research evidence exists on a topic, provide an indication of the quantity of such evidence, and decide whether or not to perform further investigations. Therefore, the keywords chosen are less focused to allow a larger number of publications to be retrieved, and, since classification of the retrieved evidence is the ultimate goal of SMSs, a categorization stage is included in the process.

Industry and academia are constantly looking more at the GSD phenomenon. On the one side, the research community have identified many risks and safeguards (e.g., [93]); on the other side, many solutions have been proposed to lessen or even overcome related issues (e.g., [PS052]). Several papers have been published reporting existing tools used to lessen issues in the GSD environment; for example, collaboration has been the focus in

[PS101][137], and coordination was analyzed in [69]. After conducting a pilot manual search to find existing SLRs or SMSs focusing on GSD, it was clear that a thorough and systematic survey related to software technologies that have been developed for or have been used by companies for supporting practitioners applying the GSD paradigm was missing. Therefore, this work aims at filling the aforementioned gap by means of a systematic mapping study. In fact, to provide innovative tool support for GSD teams, a first important step to is to have a clear overview of the context.

The remainder of this section provides a detailed description of the methodology followed to carry out this SMS (Section 3.1). We designed, evaluated, and followed a study protocol as suggested by Kitchenham [86]. In particular, Section 3.2 outlines the research questions investigated in this study and describe the search strategy that was used to seek the primary studies; Section 3.3 enlists the criteria used to determine which studies were to be included in, or excluded from the study; followed by the definition of procedures used to apply them. Finally, Section 3.4 describes how the required information was extracted from the primary studies and, in Section 3.5, a detailed description of the classification schemas used to categorize the retrieved technologies is provided.

### *3.1. Methodology*

Initially introduced in software engineering by Kitchenham [86] and later detailed by Petersen in [111], SMSs are a methodology used to provide a broad overview of a particular research area, to establish if research evidence exists on a topic, and to report an indication of the quantity of such evidence [86].

The SMS was performed in a time frame of one year incrementally building a knowledge base. Once ready to finalize the study, the knowledge base allowed us, after formalizing the definitive protocol as prescribed in [86], to quickly analyze the data by heavily drawing from it.

A protocol was defined in which the research questions were detailed and the process followed to report the results explained. The process followed was based on a slightly modified version of the original guidelines proposed in [111] in which the fundamental methodology steps as well as the expected outcomes from each of them are outlined. It can be seen from Figure 1 that the screening phase has been broken down into two separated phases. In the initial stage relevant papers were identified by analyzing publications title and abstract (full-text if needed); and, in the second stage, which entailed a full-text analysis, the concrete technologies reported in each of the relevant papers were discovered and recorded. After the second screening stage, the primary studies were selected and an unstructured set of terms used to describe technologies was created and subsequently processed in the keywording stage in which the classification schemas were created. The remainder of the process followed the guidelines reported in [111].

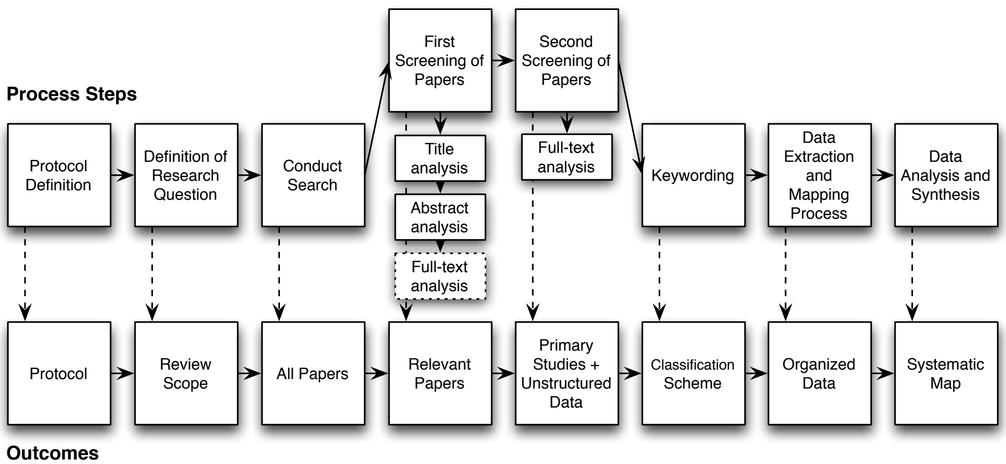



Figure 1: Systematic Mapping Study process (inspired from [111] and [34])

It is important to clarify that usually SMSs and SLRs are performed by extracting data that is strongly related to the overall topic of the papers retrieved. As it will become clearer later in this section, we decided to use the SMS methodology to structure the study in a well-established process able to mitigate potential biases almost inevitable in unstructured literature reviews. By following a systematic process to study a particular topic, a researcher can summarize "all existing information about some phenomenon in a thorough and unbiased manner" [86]. However, the purpose for undergoing this study was to identify all kinds of concrete technologies reported or mentioned in scientific publications, which is a hard concept to frame in a search string. Thus, by means of a systematic study we were able to substantially narrow down existing literature (Figure 1, the first screening phase) by focusing on the context and concepts investigated (i.e., GSD and technology). This initial skimming allowed us to retrieve concrete technologies from a small subset of publications (Figure 1, second screening phase). It is important to highlight that, rather than nature of the primary studies, what significantly provided input data to this piece of work were the names of the concrete technologies reported or mentioned in the primary studies retrieved. Besides the demographic information presented in Section 4.1, the main analysis reported in this study focuses on the technologies. As an example [PS057] focused on identifying requirements challenges in multi-site software development organizations. This study mentioned Microsoft NetMeeting was used by a GSD team during a field study. In the context of this study [PS057] contributed the name of a tool being used by GSD teams not information about requirements challenges nor details about the tool itself. In fact, in such cases information about the tool has been sought from sources independent from the set of primary studies in which the given tool was mentioned. To obtain such information, we consulted both additional literature (e.g., a publication reporting the details of the specific tool, which was not retrieved by our searches) and online sources (e.g., the main web site advertising a commercial tool to the potential users).

Finally, it is appropriate to describe the responsibilities of each of the two researchers that participated in this study. The first author was responsible for designing and conducting the study; while, the second author, was in charge of supervising every stage of the study. Tasks undertaken by the second author therefore ranged from the validation of the protocol, the assessment of decisions taken at each stage, the verification of a random subset of papers at each inclusion/exclusion step, and the complete assessment of the final set of primary studie. During each of these tasks, in case of disagreements, consensus meeting were held either to align the direction taken by the study or to discuss doubtful publications/technologies. We did not experience any case in which, after these meetings, consensus was not achieved.

### *3.2. Research questions and search strategy*

The research questions guiding this SMS are hereafter listed:

- (**RQ1**) *"What are the venues that have published papers reporting technologies for supporting GSD?"* Whilst there is only one dedicated conference to GSE topic (i.e., International Conference on Global Software Engineering (ICGSE)), the work related to GSD is published in several conferences, workshops, and journals. Apart from the software engineering community, there are other communities that are contributing to support broader GSD-like categories (e.g., distributed teams, virtual teams, and computer supported cooperative work) producing results that can be applicable to the GSD context. Thus we identify sources other researchers should review when looking at technologies supporting GSD.

- (**RQ2**) *"Which are the technologies that have been reported in peer reviewed literature for supporting GSD teams?"* One of the main objectives of this research is to provide an overview of the technologies that have been developed or used for supporting GSD teams. By technologies we mean the tools and tool support aimed at reducing the distances that characterize the GSD paradigm. Such technologies can be in the form of a tool, framework, middleware, etc. Hence, our second research question is:
- To effectively contribute to the community with an innovative solution, it is important to clearly understand the current state of the art, to see the directions to which different researchers are moving. Through the following questions, we aim to structure and categorize the results: ***(RQ3)*** *"What are the forms of the provided technological support?"*; ***(RQ4)*** *"Which of the 3C dimensions*[2]*—i.e., communication, collaboration, and coordination—have been tackled by these technologies?"*; ***(RQ5)*** *"Which are the software development activities supported by the reported technologies?"*.

Three categories of search approaches have been proposed in the literature: manual searches, automated searches, and hybrid searches [141]. The manual search (e.g., [85]) has been extensively used, however, it usually entails extensive effort in examining irrelevant studies without giving any guarantee of being more accurate in retrieving the relevant publications [80]; additionally, adopting the manual search would influence RQ1 a priori. The hybrid one (e.g., [74]) is used to give the review a high sensitivity[3] and pre-

[2]Introduced in 1991 by Ellis et al. [47], the 3Cs refer to key areas that need to be tackled in order to support group interactions; afterwards, their model (called the 3C model) became widely accepted for analyzing tools employed to support computer-mediated interactions.

[3]Sensitivity for a given topic, in SLRs, is defined as the proportion of relevant studies retrieved for the topic [141].

cision[4]. In SLRs, these properties are considered to be of major importance. However, it has been reported that if the research questions are not focused enough, the precision drops instead of rising [141]. Considering that in SMSs the topic sought is very broad, hybrid searches were excluded and the literature search was performed automatically by querying online digital libraries.

A critical phase in performing an automatic search is the identification of the keywords to be used when building a query string. The rationale behind their selection is hereafter explained. The main topics of the study are clearly inferable from the research questions: on the one side the concept of global software development; on the other side, the ones related to the software technologies. It has been observed that researchers use different

Table 2: List of keywords

| | |
|---|---|
| A1: global software development | A2: distributed software development |
| A3: global software engineering | A4: distributed software engineering |
| A5: distributed team | A6: distributed teams |
| A7: virtual team | A8: virtual teams |
| A9: outsourcing | |

| | | |
|---|---|---|
| B1: tool | B2: framework | B3: middleware |
| B4: tools | B5: frameworks | B6: middlewares |

terms to refer to GSD depending on the focus of their work [141]; thus, we decided to include all of the commonly used expressions: global/distributed software development/engineering, outsourcing. Moreover, even if the focus of the study is strongly related to the GSD context, many relevant studies from the Computer Supported Cooperative Work (CSCW) community have been published without specifically referring to the GSD topic; this is why we decided to add two more keywords, namely, distributed team and virtual team. On the other hand, we tried to capture different software technologies by using the keywords tool, framework and middleware. We believed that these terms would have been sufficient to retrieve publication related to software technologies. Table 2 contains the full list of keywords used in this stage of the pilot study. As can be seen, additional terms were added to

[4]Precision for a given topic, in SLRs, is the proportion of retrieved articles that are relevant studies [141].

overcome the issue that not all digital libraries apply stemming to the query tokens. Finally, the resulting query string consisted of the following boolean expression: (A1 OR ... OR A9) AND (B1 OR ... OR B6).

A pilot search was performed on IEEE Xplore to assess the quality of the search string against the identified set of known relevant studies. After conducting the search, a keyword (i.e., technology) initially excluded turned out to be necessary. In fact, without using *technology* in the second group of keywords, relevant known publications were missing[5]. Further analysis was performed on the extracted data and it became clear that the keyword *outsourcing* was introducing unrelated results while others, necessary to better defining technologies, were missing (i.e., *plug-in*, *environment*, and *platform*). Because *frameworks* and *middlewares* were not bringing additional material, they also were removed. Therefore, the original query was modified accordingly resulting in the following search string:

```
(A1 OR ...OR A8) AND (B1 OR ...OR B4 OR technology OR technologies OR
plugin OR plugins OR plug-in OR plug-ins OR environment OR platform)
```

Table 3 presents the results of the pilot search indicating the number of hits found by different queries. Subsequently, the search was conducted on all digital libraries to assess the number of publications that needed to be considered in the study thus an estimation of the effort required. We strongly advice researchers interested in utilizing this methodology to carefully allocate resources to this phase for tweaking the search string as necessary. The search string that we ultimately used for the study is the results of many iterations and quick analysis of the results after each iteration. Table 4 summarizes the data related to the final pilot search, which resulted in the set of papers we processed further for this work.

Due to the absence of any standards and the limitations that some digital libraries have, the concrete search string used on each specific engine potentially differed considerably in terms of structure. However, these modifications were applied after ensuring the semantic equivalence among the search strings used for different search engines. The list of final search strings executed on different search engines is reported in Appendix A. Where allowed by search engines, the searches were conducted only on the titles, abstracts

[5]e.g.: Souren Paul and Sumati Ray. Manifested Intra-Group Conflict in Collaborative Technology Supported Multi-Cultural Virtual Teams: Findings from a Laboratory Experiment. (2010) pp. 1-11

Table 3: Results from pilot search performed on IEEE Xplore

| **Original** | **+ technology** | **- outsourcing** | **+ plug-in + environment + platform** |
|---|---|---|---|
| 497 | 858 | 390 | 480 |

Table 4: Results from complete pilot search

| **ACM** | **IEEE** | **Springer** | **InterScience** | **ScienceDirect** | **Total** |
|---|---|---|---|---|---|
| 160 | 480 | 246 | 950 | 122 | **1958** |

and metadata. In fact, performing the search on the full text often has the side effect of retrieving unrelated studies that match the query via references. The digital libraries used to gather the primary studies are hereafter reported: *ACM Digital Library*[6]; *IEEE Xplore*[7]; *Springer Link*[8]; *Wiley InterScience*[9] and *ScienceDirect*[10]. Finally, the set of known relevant publications used to assess the quality of the query string during the various iterations of the pilot study was: [PS049, PS006, PS103, PS165, PS137, PS148, PS149, PS132].

### *3.3. Study selection*

As suggested in [86], for each identified publication, the bibliographic data presented in Table 5 were extracted. To provide a thorough overview, we decided not to apply any timeframe constraint during the search and the papers selection activities. In fact, from previous experiences and during the pilot study we noticed that certain software technologies still utilized by GSD practitioners, were also reported in older publications. This fact cannot be neglected and provides a rough estimation of the longevity and success of a tool. Taking for instance the example provided in Section 2 of Lotus Notes for which we found 10 publications the oldest of which dated 1996 [PS080],

[6]ACM Digital Library. http://dl.acm.org/
[7]IEEE Xplore. http://ieeexplore.ieee.org/
[8]Springer Link. http://www.springerlink.com/
[9]Wiley InterScience. http://www.interscience.com/
[10]ScienceDirect. http://www.sciencedirect.com/

Table 5: Extraction form: bibliographic information

| Initial data | |
|---|---|
| **ID** | A progressive ID for reference reasons |
| **Title** | The title of the publication |
| **Author(s)** | The list of author(s) |
| **Year** | The year of the publication |
| **Type** | Conference, journal or book |
| **Name of database(s)** | The list of databases where the publication was retrieved |
| **Publisher** | The name of the publishing journal or conference |
| **Abstract** | The abstract of the publication |

it could be inferred that this platform has unique features considered to be very important in the context of GSD. An analysis of these tools could therefore provide an interesting work aimed at identifying requirements needed in specific kind of tools as well as flaws in others allowing to build a features list that could be used as a baseline to evaluate similar technologies.

We decided to exclude publications written in non-English languages, panel discussions, presentations slides, and tutorials. All the retrieved papers not related to the topic explored in the reported research were also excluded.

To assess the relevance of each primary study to the GSD topic, an iterative procedure was followed: all primary studies were analyzed based on their title and abstract and, when necessary to clear any doubts, their full text. Information to fill in Table 6 was extracted from each of the papers included in this study. Publications retrieved during the search phase but

Table 6: Extraction form: inclusion/exclusion

| Inclusion/Exclusion data | |
|---|---|
| **Data checker** | The name of the screener |
| **Included/Excluded** | Whether or not the paper has to be included |
| **Reason** | A brief reason for exclusion |
| **Additional note** | |

dealing with the following topics were excluded:

- teaching,
- supply chain management,

- logistics,
- distributed software,
- ASP (Application Service Provider),
- e-research.

The studies purely related to the outsourcing paradigm and not applicable to the context of GSD were also excluded as well as the studies describing approaches, theoretical frameworks, challenges, risks, etc. Based on the results from the pilot search, we concluded that the papers using the keyword *outsourcing* rarely reported any concrete technology. Hence, we decided to exclude this keyword from our final set of keywords with which the search strings for this study were composed. During this phase, the second researcher checked a random subset of the studies for verification purposes. After this iteration, the final group of the relevant studies was acquired and processed for the second screening phase.

The goal of this phase was to assess which papers would have be part of the set of primary studies. To be included in this study a paper needed to satisfy two requirements, which are also reflected in the search string. First of all the study needs to be on GSD or GSD related topic (see set of keywords prefixed by the 'A' letter); secondly, it needed to report concrete technology names. If the retrieved papers did not conform to these two requirements, they were excluded. As an example publications reporting general technologies like blog, version control system, instant messengers, and so on were all excluded. At this stage, the full text of the papers was reviewed; for each paper that satisfied the inclusion criteria all technologies were recorded through pairs composed by the id of the primary study and the name of the technology. During this phase, additional papers were excluded. Consensus meeting were held also during this phase to discuss and resolve any disagreements.

### *3.4. Data Extraction and Keywording*

The objective of this phase was to improve the raw data extracted during the second screening procedure in order to retrieve for each technology all the data reported in Table 7. As suggested by Petersen et al. [111], during the previous screening stage, a keywording activity was performed by assigning a set of keywords and concepts reflecting the information provided in each primary study that could facilitating the filling of Table 7.

The input data to this phase were pairs composed of primary study id and technology name enriched by a set of labels created during the keywording stage. An initial activity was performed to eliminate duplicates and consolidate data. Therefore, pairs became tuples containing the name of the technology and all the primary study references in which it was found. Sets of labels were also merged. This provided us the possibility of performing an initial verification over the data; in fact, in case of labels reporting contradictory information, the primary studies were re-checked to solve discrepancies. Moreover, to fill the gaps in case of missing information necessary to complete Table 7, additional searches were performed online. To obtain such information, we consulted both additional literature (e.g., a publication reporting the details of the specific tool, which was not retrieved by the searches) and online sources (e.g., the main web site advertising a commercial tool to the potential users).

For each of the identified technologies, we stored all the information retrievable from the publication as presented in Table 7 which, in addition to the one previously extracted, provided the necessary data to have quantitative evidence. All the information was stored in a spreadsheet from where

Table 7: Data Extraction form: classification information

| **Additional data** | |
|---|---|
| **Contribution type** | The type of contribution (e.g., standalone tool, framework, middleware) |
| **3C characteristics** | Collaboration, coordination and communication |
| **Purpose** | Purpose for which the technology was designed |

the second researcher was able to randomly check the extracted data. Again, in case of disagreement, a consensus meeting was held. Afterwards, each technology was classified to find the answers to the research questions RQ3, RQ4 and RQ5. Therefore, as described in detail in the next section, four aspects were considered: technology type; 3C classification model; supported software engineering activity; and, type of license.

### *3.5. Classification schemas*

In this section the construction and evolution of the categorizations used throughout this systematic mapping study will be extensively described by analyzing each classification schema.

From the initial pilot search it was clear that standalone tool, framework, and middleware were not the only forms of technologies that have been developed for or have been reported to be used by GSD teams. Rather, it appeared necessary to include environment, platform, and plug-in as keywords in the final query string. Therefore, the categories emerged as a by-product of the various iterations of the pilot study. Hence, the categories used to shape the answer to RQ3 were: standalone tool, framework, middleware, environment, platform and plug-in. The rationale followed to categorize the reported technologies is based on the following definitions:

- *standalone tool:* an independent software application fulfilling a specific design intent (e.g., Skype);
- *framework:* a 'semi-complete' application that provides an integrated set of domain-specific structures and functionality [121] (e.g., CAISE [30] found in [PS148]);
- *middleware:* a software that lies between the operative system and the applications, whose task is to provide gluing mechanisms and functionalities often enabling cross-application interactions (e.g., the nameless technology presented in [PS108]);
- *environment:* a development environment that comprises a set of processes and programming tools used to create software products. Moreover, an integrated development environment is a subset of this group, which identifies a development environment having a unified interface (e.g., SourceForge, the public face of the Collab.Net platform);
- *platform:* a set of generic components that form a common structure, from which a set of derivative products can be developed [48] (e.g., Collab.Net and IBM Jazz); and,
- *plug-in:* a software component that interacts with an existing software application through the use of well defined APIs, often designed to enhance it by adding new functionalities (e.g., Jazz Band [27]).

The geographical distance among GSD teams leads to an increased level of communication, collaboration, and coordination burden that needs the use of dedicated technological support. In 1991, Ellis et al. [47] identified communication, collaboration, and coordination as key areas that needed to be tackled to support group interactions; afterwards, their model (called the 3C model) became widely accepted to analyze tools employed to support computer-mediated interactions. To find an answer to RQ4, one classification was based on the 3C model, which has been widely used by other researchers to study specific categories of tools (e.g., awareness support in [PS148]) and focuses on the main characteristics that technological support should address when dealing with computer-mediated interaction: communication, collaboration and coordination. Technologies were thus categorized by identifying for each technology the predominant 3C attributes (i.e., communication, collaboration, and coordination). The definitions we followed to categorize them are based on the ones proposed in [PS148] but directly applied to technologies. Therefore, we selected the operational definitions of the terms for the reported work as the following:

- *communication* - when the technology brings improvements to the way messages and information are exchanged among people, reducing gaps, ambiguity, or the effort needed to understand, establish, or continue a conversation;
- *collaboration* - when the technology brings improvements to the shared space or to the way users interact with shared artifacts synchronous or asynchronously; and,
- *coordination* - when the technology brings improvements to the support offered for people managing themselves, or being aware of the activities and its effects to the collaboration.

To grasp a better understanding of the tool support available for GSD teams, we decided to provide a further classification. This classification aims at helping us answer RQ5 by categorizing the reported technological support according to specific software engineering activities such tools are supporting or—more generally—the purpose for which they have been designed. For our initial classification, we used the types of GSD challenges identified in the SLR reported by Jimenez et al. in 2009 [78]. According to their systematic review, the GSD challenges to be addressed were classified based on specific topics such as communication, group awareness, software configuration management, knowledge management, coordination, collaboration, project and process management, process support, quality and measurement, and risk management. During the data extraction, these categories were also confirmed by our results. Communication, coordination, and collaboration were widely reported and represent one of the main categorization schemas of this study (Section 4.4), while most of the remaining categories of [78] represent the elements we found referenced the most (Figure 8). Even if treated separately in Section 4.4, communication has been kept also in this classification to identify all the retrieved tools that are designed to provide concrete means of communication. In fact, geographical, temporal, cultural, and linguistic distances normally characterize GSD projects [103]; these distances usually lead to a significantly increased complexity for project teams that are expected to face several kinds of new challenges [70, 35], and an improved communication via better practices or more appropriate tools has often been identified and reported by researchers to be of great impact in lessening these distances [108]. Therefore, herein we are presenting the communication tools that have been retrieved. Compared to the communication category discussed above, we are not capturing the characteristic supported by tools but the intent for which they have been designed; tools belonging to this group need to clearly provide means for communicating.

During the data extraction, we modified the initial classification as we found that the categories were not able to cover all the identified technologies for supporting GSD teams. The software configuration management group[11] was too wide, which is why we decided to split it into two new categories: version control systems (e.g., Git, Rational ClearCase), dealing with the management of changes in source code artifacts; and, artifact management (e.g., Google Docs), including more general purpose tools for dealing with artifacts. Project and process management was restricted to project management only; process support became process management. Quality and measurement were renamed as quality management; and, group awareness was renamed to awareness to avoid restricting the scope to only one of the different kinds of awareness identified by Gutwin et al. [60] for collaborative work and widely accepted as categorization scheme (e.g., [PS148]). Moreover, additional themes emerged for which we inserted specific groups that are listed below:

- implementation;
- software design;
- requirement management;
- software integration;
- software inspection;
- bug/issue management;
- office and productivity applications;
- tools for remote access;
- decision support systems;
- collaboration platforms; and,
- application integration systems.

[11] In [78], the software configuration management group identifies issues related to the control of the source code. Therefore, coordination and synchronization problems due to team distribution and traceability.

## 4. Results and Discussion

In this section and its sub-sections, we present and discuss the results from the analysis of the data extracted from the papers included in this mapping study. The results are presented with respect to the research questions that this mapping study aimed to answer by following the classification schemas introduced in the previous section.

Figure 2 presents the different phases of the papers selection process showing the number of papers included at each stage of the SMS. During the initial

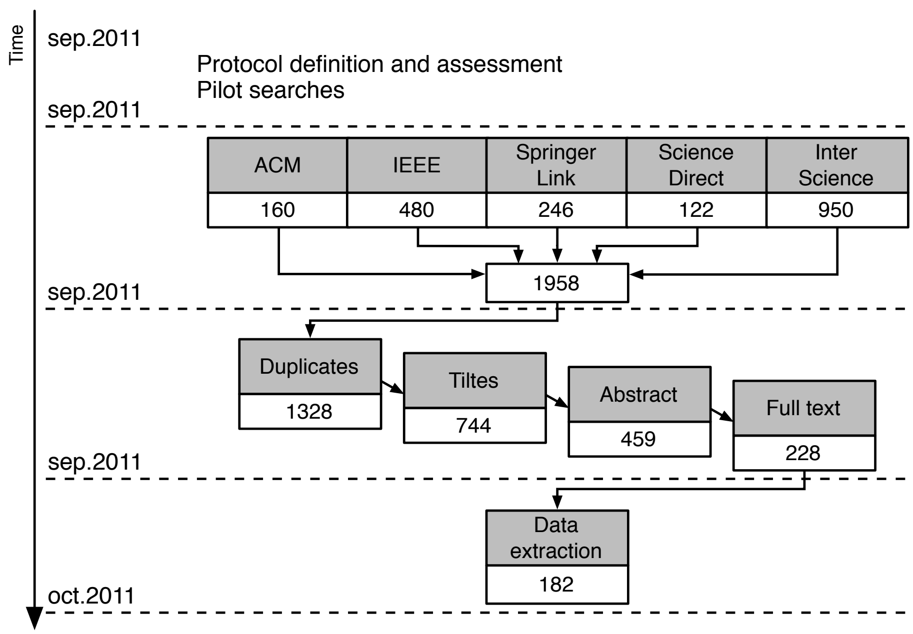



Figure 2: Screening phases overview

stages of the papers screening process, there were certain papers for which the inclusion or exclusion decision was postponed until the data extraction stage. This allowed us, having a complete picture of the data base of the candidate papers, to better assess their relevance to this study. It became clearer that many of them, rather than being focused on software engineering, were addressing virtual or distributed teams and their outcomes or contributions were not applicable to the software engineering context. Since final inclusion and exclusion decisions are difficult and much prone to bias, the final set of primary studies were scrutinized by both researchers. Our strategy of keeping the doubtful papers proved to be correct as some papers included important data which were only discovered upon an analysis of the full text; the rest of the doubtful papers were excluded before the data extraction stage.

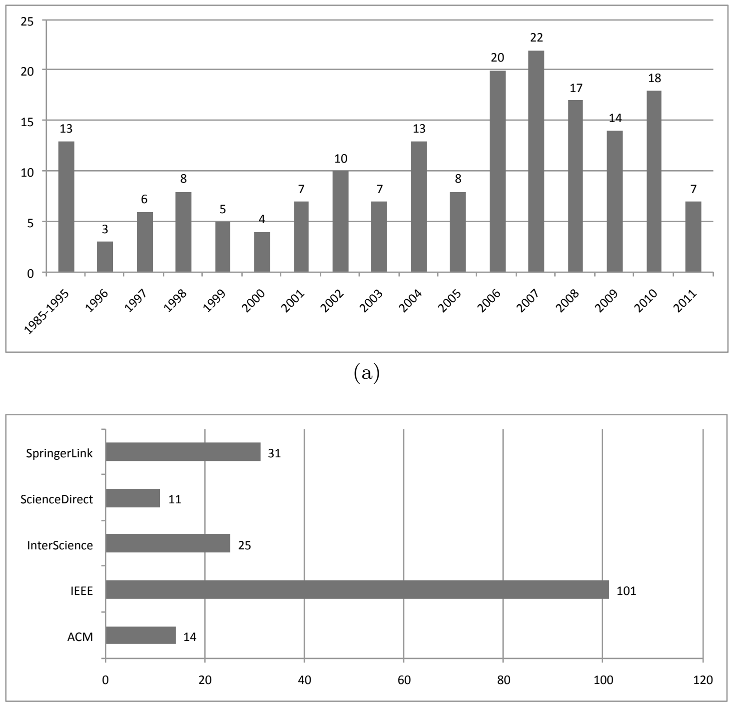
This section is structured as follows. We present the demographic data by synthesizing bibliographic information in Section 4.1 (RQ1: "what are the venues that have published papers reporting technologies for supporting GSD?"). After this, the remaining sections report the findings based on the analysis of the data extracted from the papers included in this study. The identified technologies are organized according to the refined classification schemas described in Section 3.5 (i.e., form of technological support, 3C model, and intent pursued by the technology) with respect to the research questions hereafter reproduced:

- RQ3: "what are the forms of the provided technological support?";
- RQ4: "which of the 3C dimensions have been tackled by these technologies?"; and,
- RQ5: "which are the software development activities supported by the reported technologies?".

Finally, in Section 6, by combining RQ3, RQ4, and RQ5, we address RQ2, i.e., "which are the technologies that have been reported in peer reviewed literature for supporting GSD teams?".

### *4.1. Demographics of included primary studies*

Figure 3 shows the primary studies distribution according to (a) the year of publication, and (b) the digital libraries from which the papers were retrieved. From Figure 3-(a), we see that there has been a continuous stream of published papers reporting technologies supporting GSD and distributed teams over the years, and this trend has grown significantly since 2006. The graph plotting the papers based on the digital libraries (Figure 3-(b)) reveals that a large majority of the papers have been published by IEEE. This means that IEEE has sponsored more conferences and workshops which published papers reporting technologies supporting GSD than any another publisher. 10 papers (out of 101) were published in IEEE journals—primarily *IEEE Software* and *Software Engineering Journal*. The remaining papers were pub-




(a)

(b)

Figure 3: Number of relevant studies: (a) per year, (b) per digital library

lished in different conferences ranging from *ICGSE*[12] (19 papers) to *WET ICE*[13], *COMPSAC*[14] and *ICSE*[15]. Table 8 provides more details about the publication venues of the papers included in this mapping study. We have listed only those venues in which more than two papers were published. It is quite interesting to note that there were 108 different venues from which the papers were retrieved showing that there are a large number of venues (i.e., 108) that publish papers reporting technologies supporting GSD. Out of 182 primary studies, 122 papers were published in conferences, 58 in journals, and 2 as a book chapters.

[12]ICGSE: International Conference on Global Software Engineering

[13]WET ICE: International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises

[14]COMPSAC: International Computer Software and Applications Conference

[15]ICSE: International Conference on Software Engineering

Table 8: Venues

| Venue | # | Type | Refs |
|---|---|---|---|
| International Conference on Global Software Engineering (ICGSE) | 19 | c | [PS096, PS158, PS028, PS123] [PS060, PS106, PS056, PS171] [PS155, PS016, PS017, PS102] [PS159, PS161, PS001, PS144] [PS052, PS009, PS133] |
| Software Process: Improvement and Practice | 8 | j | [PS119, PS175, PS063, PS006] [PS069, PS044, PS103, PS128] |
| Int. Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE) | 8 | c | [PS024, PS065, PS070, PS091] [PS109, PS036, PS061, PS012] |
| Int. Conf. on Computers, Software, and Applications (COMPSAC) | 6 | c | [PS153, PS023, PS178, PS005] [PS077, PS117] |
| Hawaii Int. Conf. on System Sciences (HICSS) | 5 | c | [PS072, PS157, PS132, PS064] [PS130] |
| Int. Conf. on Software Engineering (ICSE) | 5 | c | [PS149, PS085, PS038, PS136] [PS059] |
| Software: Practice and Experience | 5 | j | [PS122, PS049, PS163, PS039] [PS125] |
| Int. Conf. on Collaborative Computing (CollaborateCom) | 4 | c | [PS015, PS087, PS154, PS045] |
| Agile Conference | 3 | c | [PS018, PS167, PS114] |
| Software Engineering Journal | 3 | j | [PS088, PS142, PS025] |
| Journal of Software Maintenance and Evolution: Research and Practice | 3 | j | [PS003, PS035, PS054] |
| IEEE Software | 3 | j | [PS101, PS002, PS137] |

### *4.2. Technologies*

Based on the analysis of the data extracted from the 182 papers included in this mapping study, we identified 412 unique software technologies which included research outcomes, open source software (OSS), and commercial off-the-shelves (COTS) products. The information about the license arrangements (e.g., research, OSS, or COTS) of the retrieved technologies was obtained from the reviewed papers or from the web if not reported in the respective papers. Analysis presenting classification diagrams also detailing the distribution of technology over the years (i.e., Figure 6, 7, and 8) do not consider this distinction. Figure 4 shows the percentages of the reported technologies in the three categories representing the licensing arrangements.

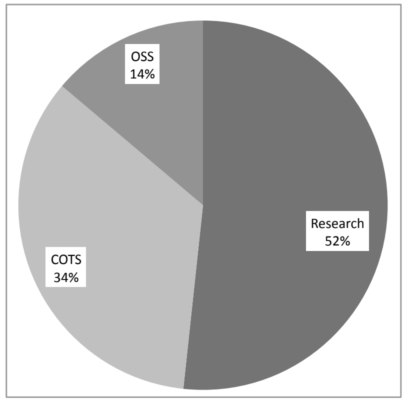



Figure 4: Technology origin

In the remainder of this section, the technologies supporting GSD have been categorized according to the three classifications introduced in Section 3.5 and hereafter reported:

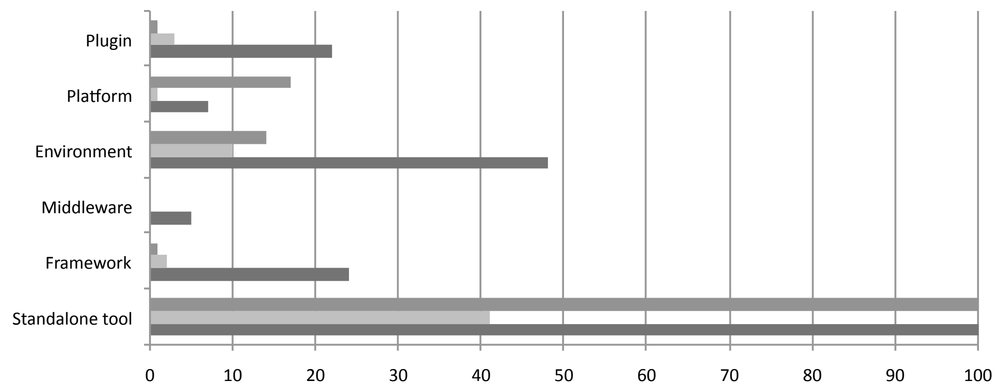
- the form of technological support for GSD;
- the 3C model; and,
- the software development activity for which each of the retrieved technology was designed.

*4.3. Classification based on the form of technological support*

All of the identified results were categorized according to this classification; Figure 5 shows the number of technologies associated with each group.

To grasp a better understanding of directions followed by both research and industry, the number of tools that have been identified during the extraction process was mapped according to the year in which they first appeared in publications (Figure 6). As an example, following this criteria, a standalone tool reported ten times like Lotus Notes only increments the count of platforms reported in 1996—year of the retrieved primary study that first reported it.

It is clear that standalone tools are the type of technological support provided the most. However, we can also see a clear difference related to how researchers are investigating the field. On the one hand, standalone tools




| | Standalone tool | Framework | Middleware | Environment | Platform | Plugin |
|---|---|---|---|---|---|---|
| COTS | 109 | 1 | 0 | 14 | 17 | 1 |
| OSS | 41 | 2 | 0 | 10 | 1 | 3 |
| Research | 107 | 24 | 5 | 48 | 7 | 22 |

Figure 5: Classification: form of technology

and platforms are dominated by commercial and open source solutions; on the other hand, the number of research-based technologies is larger when it comes to environments, frameworks, and plug-ins. This result could be interpreted in different ways. However, we believe that the simplest explanation in this case can support the evidence: integration has been heavily discussed in research agendas and vision papers [69, 126]; moreover, collaborative development environments have been also suggested as possible solution to face some of the well known challenges [17]. Finally, observing results in the case of middleware solutions requires effort in terms of time and human resources that rarely can be afforded by researchers. Figure 5 shows that companies are still providing technological support in the form of standalone tools; however, since 2002, they started to sell comprehensive platforms to support distributed teams and, in some cases, specifically for software engineering teams. IBM and Microsoft are two of the main players in this respect. To support geographically dispersed groups of people, IBM initially proposed Lotus suite and afterwards concentrated its effort on supporting software engineering teams with the development of the Jazz platform and all Rational add-ons dealing with specific software engineering processes. Microsoft, on the other hand, developed SharePoint, which is a more general solution, focused on enhancing communication in GSD teams; and the Visual Studio Team System, which is a set of tools designed to support the entire life-cycle of application development.

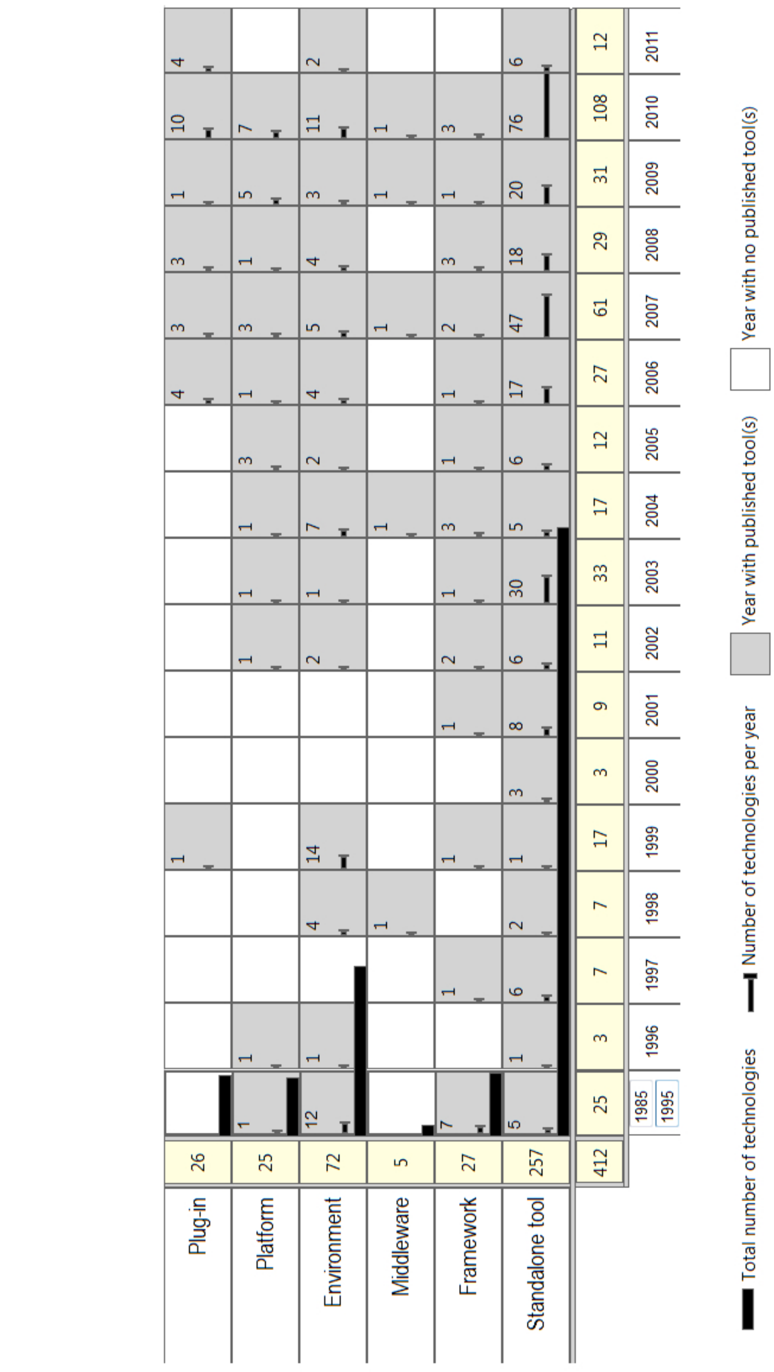



Figure 6: Classification: form of technology over the years

Tool integration has always been an area of great attraction for industry and academia in order to reduce the burden of using multiple tools for one particular activity or different activities of a particular software development phase. One of the most common approaches to integrate a diverse set of tools is the plug-in paradigm that can help to either smoothly integrate pre-existing tools into an integrated environment for supporting software development or enhance technologies already in use with new functionalities. Given the known advantages of plug-in based integration, there has been an increased interest in developing software engineering tools as plug-ins for well established platforms (e.g., Eclipse and Visual Studio) since 2006. The remainder of this section will be dedicated to the description of the plug-ins reported in the papers included in this mapping study.

Throughout the review, we found references to 26 plug-ins. A summary is provided in Table 9. All articles were published after 2005 with the exception of one publication—dated 1999—reporting *Merlin ToolChain*: a plug-in for integrating project management, requirements management, configuration management, and test management tools. Almost all of them have been designed to be deployed on the Eclipse environment. *XPairtise* [123], *Sangam* [71], and *Distributed Pair Programmers Editor* [PS046] are solutions designed for supporting pair programming; *UNISYS* is a distributed modeling tool based on Rational Rose; *eConference* is a text-based distributed meeting system [PS040]. *Jazz Band* [27], *Palantir* [118], *ProjectWatcher* [61], *Automatic Status Updates (ASU)* [PS095], *Stellation* for Eclipse and, *CollabVS* [41] for Visual Studio are attempts to introduce and enhance awareness as well as increase collaboration. *FineVMS* [36] and [75] are also Eclipse plug-ins developed to increase context awareness by leveraging version control systems (VCS): the former introduces new functionalities to the VCS allowing a fine-grained control over the artifacts, the latter provides a plug-in that shows concurrent changes in VCSs by leveraging the annotation system provided by Eclipse. Along these lines, *Lighthouse* [33]—Eclipse plug-in—

[16] merlintoolchain.sourceforge.net
[17] eclipse.org/mylyn
[18] sangam.sourceforge.net
[19] eclipse.org/technology/archived.php
[20] subclipse.tigris.org
[21] tagsea.sourceforge.net
[22] ibm.com/support/docview.wss?uid=swg21131368

Table 9: Plug-ins overview.
Acronyms: ASU "Automatic Status Updates", CRI "Continuum of Relevance Index", DPPE "Distributed Pair Programmers Editor"

| Name | Env | Main | Refs |
|---|---|---|---|
| n/a | Eclipse | [21] | [PS148] |
| n/a | Eclipse | [75] | [PS148] |
| ASU | Eclipse | [PS095] | [PS095] |
| CASI | Eclipse | [127] | [PS148] |
| CollabVS | Visual Studio | [41] | [PS045, PS133] |
| CRI | Eclipse | [105] | [PS113, PS125, PS148] |
| Deep Intellisense | Visual Studio | [72] | [PS022] |
| DPPE | Eclipse | [PS046] | [PS046] |
| eConference | Eclipse | [PS040] | [PS040, PS039, PS101] |
| EGRET | Eclipse | [125] | [PS144] |
| FineVMS | Eclipse | [36] | [PS148] |
| Hipikat | Eclipse | [31] | [PS095, PS022, PS125] |
| Jazz Band | Eclipse | [27] | [PS149, PS148, PS038, PS161] [PS045] |
| Lighthouse | Eclipse | [33] | [PS148] |
| Merlin ToolChain | Eclipse | *url*[16] | [PS137, PS056, PS036, PS133] |
| Mylyn | Eclipse | *url*[17] | [PS113, PS095, PS022, PS125] |
| Palantir | Eclipse | [118] | [PS002, PS161, PS148, PS095] [PS045, PS135, PS130, PS125] |
| ProjectWatcher | Eclipse | [61] | [PS135, PS130] |
| Remail | Eclipse | [PS022] | [PS022] |
| Sangam | Eclipse | *url*[18] | [PS149, PS046] |
| Stellation | Eclipse | *url*[19] | [PS149] |
| Subclipse | Eclipse | *url*[20] | [PS133] |
| TagSEA | Eclipse | *url*[21] | [PS125] |
| Team Tracks | Visual Studio | [40] | [PS125] |
| UNISYS | Rational Rose | *url*[22] | [PS081] |
| XPairtise | Eclipse | [PS152] | [PS152] |

presents an approach to avoid conflicts during collaborative development activities by proposing a concept called emerging design; and *CASI* [127], a plug-in of Lighthouse, leverages the emerging design concept by making developers aware of potential direct and indirect implications occurring in the source code being modified by them. To lessen information overload problems, some plug-ins have been developed to filter information presented to developers; examples of this approach can be seen in *Continuum of Relevance Index (CRI)* [105], *Mylyn* (previously called Mylar [84]) and *Deep Intellisense* [72]. Similarly, *Hipikat* [31], *TagSEA*, *Team Tracks* [40], and *Remail* [PS022] aim at increasing artifact awareness and understanding by introducing collaborative annotation features, by tracking how often and in relation with what source code parts are visited or, in the case of Remail, by linking code snippets to mail based discussions. Distributed requirement management integrated in the IDE has been tackled (i.e., *EGRET* [125]) as well as version control systems integration in *Subclipse* for Subversion. Finally, a plug-in based on the Eclipse and Jazz platform presented in [21] is reported in [PS148]; this plug-in leverages a social network called FriendFeed and aims at integrating social networking features into the development environment offering micro-blogging and forum-related functionalities.

### *4.4. Classification based on the 3C model*

Technologies can support any of the 3C attributes. As a result, technologies can be grouped into sets, each of which support one or more of the 3C attributes. This can be effectively visualized by means of Venn diagrams as shown in Figure 7. The use of a Venn diagram was necessary to plot this information as the sets are not disjoint; technologies, in fact, may exhibit more than one predominant characteristic thus the possibility of showing intersections led us to the use of this type of diagram. The numbers in parenthesis identify the number of technologies for each group that are research-based technologies, whereas the numbers outside parenthesis show the overall number including all licensing types. To assess the presence of these characteristics we analyzed the descriptions provided in the primary studies, or, if not sufficient, the primary sources (see Appendix B). In case of contradicting information, we classified the tools according to the primary sources.

Taking *Communication* as example, it can be seen that 141 technologies were categorized as having communication features (61 of which coming from research projects); 48 are supporting solely communication (9 of which are research outcomes); whereas, 51 also tackle collaboration and 33 cover all the 3Cs.

It is clear that the collaboration dimension of the 3C model has been addressed the most followed by the coordination and the communication dimensions almost equally addressed. Considering solely the research outcomes, we can see that communication is the characteristic addressed the

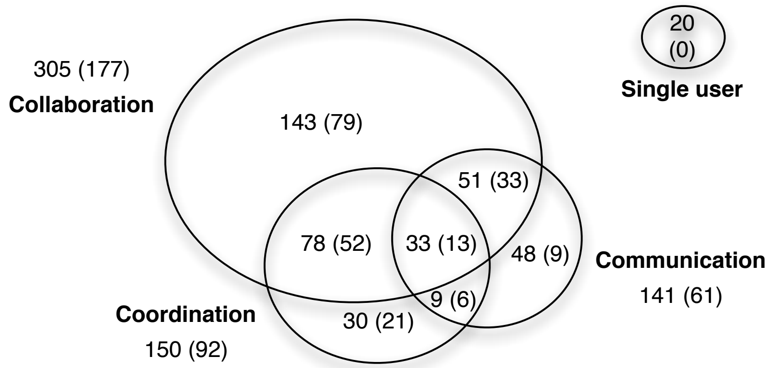



Figure 7: Classification: 3C model

least among the 3Cs. A plausible explanation can relate this finding to the fact that the market is overwhelmed by technologies supporting pure communication. Moreover, this dimension does not raise much interest per se as compared to when it is paired with collaboration or coordination. Interestingly, researchers reported also 20 *single user* tools that were used by practitioners adopting the GSD paradigm even though they were not specifically developed to support any of the 3C attributes. Tools belonging to this group, either COTS or OSS, range from modeling tools (e.g., Argo UML) to integrated development environment (IDE) (e.g., Eclipse, NetBeans), or text editors (e.g., vi, emacs); moreover, Microsoft Excel, which belongs to this group, has been reported to be used for requirements management and bug/issue tracking [PS057, PS172]. Artifacts generated by these applications are usually edited collaboratively in an asynchronous manner by storing them in a shared common repository.

In the remainder of this section, an example taken from the research outcomes is presented. To select the candidate to describe, we applied the following rationale: (i) research outcomes have been isolated from both COTS and OSS products; (ii) tools have been ordered considering the number of primary studies referencing each tool; and, (iii) in the case of multiple candidates at this stage, they were all selected.

*Collaboration*

All those systems that allow multiple people to work on the same task that do not provide much support in terms of coordination and communication belong to this category. Traditional versioning systems and many tools for managing artifacts in a more general manner are examples of technologies categorized as pure collaborative systems. Further, tools automating processes (i.e., building tools) are also included among the tools enabling collaboration. IBIS [PS103] is the research-based technology that has been referenced the most. It is a web-based tool developed to support geographically distributed inspection teams. The aim of the tool is to minimize synchronous activities to reduce coordination issues. It supports asynchronous communication between participants applying a flexible redesigned inspection process.

*Coordination*

Project and artifact management tools, technologies to improve awareness as well as calendaring applications are all systems that fit into this category. The technologies referenced the most among the research outcomes which have coordination as main attribute, are FASTDash and TAMRI. The former [15] represents an attempt to improve group awareness by providing a dynamic visualization of team members activities; whereas, the latter [PS102] is a tool able to support project managers in identifying suitable task allocations during the planning process of GSD teams.

*Communication*

It is quite challenging to support communication properly in distributed environments. One of the major difference between co-located and distributed teams is the inability to engage in informal communication with co-workers, and this shortcoming is identified as one of the main reasons for distributed projects to fail. In [138], statistical information regarding informal communication is provided. They report that 25% to 70% of the communication done in co-located settings is informal and half of the time involved artifacts; further, 88% of these conversations were joined by a third person. The communication group comprises all tools developed to provide a pure means for communicating either textually, verbally or by using video support. Research effort included in this group tend to possess less features than the COTS and OSS alternatives. However, most often they have been designed to support specific functionalities making them unique in their genre. The authors of [PS019] provide a detailed list of available alternatives. Rear View Mirror [19] is the research outcome cited the most; it is a presence-based instant messenger with features that help to support virtual teams.

*Collaboration and Coordination*

The technology cited the most in this group is Palantir [118], a plug-in based on the Eclipse IDE. The tool allows basic code analysis to visually inform developers about changes and dependences to identify potential issues in the code.

*Communication and Coordination*

Decision support systems like TeamSpirit [PS041], a web-based group decision support system; and, TeamVote[PS065], a client based software integrating the most important collective choice strategies belong to this category. However, the research-based technology cited the most is CWS-IM [PS135]: an augmented instant messenger that supports the concept of collaborative working spheres. Contacts, instead of being presented as it happens in normal IMs, are grouped around the concept of task, providing an easy to use and more coherent way of communicating and coordinating among team members.

*Collaboration and Communication*

Jazz Band [27] is a plug-in that nicely fits into this category. It was specifically designed to enhance Eclipse for supporting collaboration by also providing communication capabilities. The plug-in provides a view containing team members' avatars and related information including their status. Textual and vocal communication can be initialized through a contextual menu and, by means of additional plug-ins, markers are placed in the code showing the line currently edited by each participant as well as currently edited artifacts in the package explorer view. Finally, chat transcripts are anchored inside the related code.

*Collaboration, Coordination and Communication*

Many commercial vendors have been working on providing full 3C support. To reach this goal, the most representative technologies are platform systems designed to lessen GSD issues, e.g., IBM Rational suite based on the Jazz platform, and the Microsoft SharePoint suite. Furthermore, environments to support developers have been widely reported (e.g., SourceForge and EzForge) as well as packages of office and productivity applications like Google Apps. Moreover, a tool possessing all the 3C attribute is Jira: a bug, issue and project tracking system for software development. Besides supporting collaboration and coordination functionalities to track issues, Jira provides a fully capable communication system to facilitate discussion over specific issues. The research-based technology referenced the most is Sysiphus [PS017]; a distributed environment designed to provide a coherent infrastructure for artifacts aimed at enhancing and encouraging communication between participants.

### *4.5. Classification based on the activities supported*

Figure 8 classifies the tools based on the main purpose of each of the tool extracted from the included primary studies and the year in which the tool was reported for the first time.

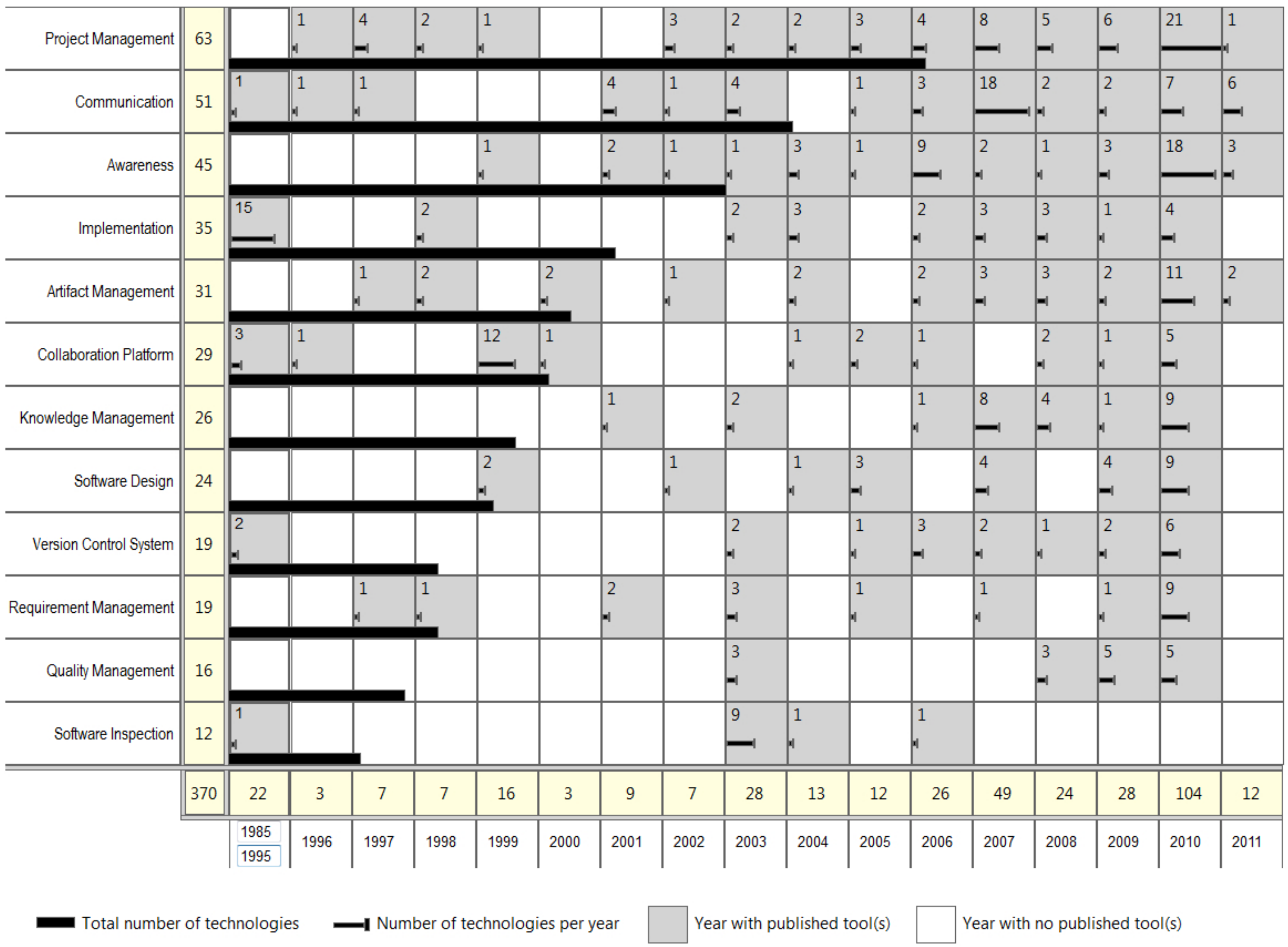



Figure 8: Classification: intent

As an example, let us consider Eclipse: it is classified in the *Implementation* group and it has been referenced in papers from 2004 to 2010 eight times; however, Eclipse has been counted as one unit only in 2004 under the *Implementation* group. Figure 8 provides the quantitative information about the number of tools that have been developed for or have been reported to be used by GSD teams.

Figure 8 only plots those groups of tools which we cited at least ten times. In the following sub-sections, the groups supporting different software development activities are specified along with a representative set of tools. We briefly discuss tools, where applicable, from each of the different license arrangements (i.e., COTS, OSS, and outcomes of a research project). The groups of tools with less than ten citations are briefly reported at the end of this section.

*Project management*

Software projects are known to be relatively more difficult to manage as compared to projects in other engineering disciplines. Project management activities become even more complex when dealing with teams that are distributed. This characteristic of GSD can pose serious threats to the successful delivery of GSD projects [PS149]. This category covers (but is not restricted to) tools used to handle the complexity of projects. For example, estimation and planning support, scheduling, resource allocation, cost control and budget management represent some examples of areas addressed by these technologies. Commercial and open source examples of project management tools are Microsoft Project, IBM Rational Team Concert, Borland Application Lifecycle Management, GitHub, and SourceForge. Research-based technologies are represented by Genesis [PS077], a web-based platform providing a non intrusive support that is mainly focused on enhancing coordination and collaboration; and, MILOS [PS110], a system supporting the dynamic coordination of distributed software development teams providing project planning functionalities and workflow technologies.

*Communication*

Compared to open source and commercial solutions, research-based technologies were retrieved less: hardly any solution coming from the academia can provide a suitable replacement for this category as the COTS and OSS provide many more functionalities than the ones developed to investigate specific properties of such technologies. eConference [PS040] is an attempt to introduce a distributed meeting system on top of the Eclipse Rich Client Platform (RCP) leveraging the eXtensible Messaging and Presence Protocol (XMPP). The authors of [PS165] provide a summary of some commercial and open source tools used for communication purposes; others are reported in [PS019]. Technologies cited the most are Skype, Internet Relay Chat (IRC), Microsoft NetMeeting, Lotus SameTime, and Cisco WebEx for the OSS and COTS sets.

*Awareness*

Jiménez et al. [78] use the awareness cluster to capture tools designed to provide information to members of a virtual team regarding project status changes or notification systems. However, we believe that, in a context where face-to-face interaction becomes harder—i.e., GSD—awareness should identify tools that improve the knowledge of the working context that a member of a virtual team has [134]. Thus, we have categorized tools according to the definition provided in [42] where awareness is defined as the "understanding of the activities of others, which provides a context for your own activity". Therefore, the objective of introducing awareness is to allow people to collaborate smoothly by being able to assess contributions to group activities. Hardly technologies used in or developed for environments adopting the GSD paradigm are designed without providing awareness support; however, technologies included in this category are specifically addressing this challenge. In [PS148], tools for supporting awareness are systematically identified and reported. Examples developed by the research community are: ToxicFarm [PS076], a framework for hosting virtual teams providing diverse services to support them; FASTDash [15] and Jazz Band [27], which were previously described. Examples of COTS and OSS are reported less and, the ones found, are date-related systems offering collaborative functionalities like the calendars provided by Google and Lotus.

*Implementation*

Tools used to write and edit code artifacts have been classified under the implementation group. They vary from Integrated Development Environments (IDE) to group text editors, and text editor like vi and emacs especially used by open source communities members. We extracted several software development implementation related tools, both commercial and open source, from the primary studies reviewed in this study. Some of the most notable ones are Microsoft Visual Studio, SharePoint Designer, Eclipse, NetBeans, and Sun Java Studio Enterprise. On the research side, WebMake is an example of a research-based technology aimed at improving development environments by exploring the feasibility of providing a software development environment deployed on the WWW. Technologies reported in this group by academia differ greatly from the ones available as COTS and OSS. We noticed that researchers are mainly focused on introducing collaboration functionalities in software development environments. An interesting group of research outcomes is represented by the group editors. To properly collaborate remotely, especially when adopting agile methodologies in the context of GSD, synchronous editing functionalities are often sought and rarely supported by the current technologies (e.g., distributed pair programming when applying eXtreme Programming). We identified a significant number of research-based technologies belonging to this group most of which are now outdated. For coordination support in distributed software development environments we found CES [58], DistEdit [89], DSEE [92], GROVE [47], Mercury [82], and Pan [6]. Whereas, more recent ones are: Sangam, an Eclipse plug-in that enables complete sharing of workshops to simulate co-located pair programming described in [71] and now available as a OSS; Moomba [PS143], enabling distributed extreme programming; and, Tukan [124] and XPairtise [PS152] supporting distributed pair programming.

*Artifact management systems*

Artifact management systems include technologies that are developed to handle artifacts consumed or generated during the software development activities such as requirements documents, test cases, project plans, and design diagrams. TraVis [PS053] from the research community is an example of these systems; it leverages the use of dependences among artifacts and their user to allow the visualization and analysis of different relationships. Rational Asset Manage, Google Docs, and ToolChain represent the commercial and open source counterparts.

*Collaboration platform*

This category includes systems that have been designed to provide integration between applications to form a consistent platform for collaboration; they are often the technology underneath well-known comprehensive multi-purpose collaboration systems. Commercially available examples are IBM Jazz and IBM Lotus platforms; CollabNet whose public face is the well-known open source web site SourceForge.net. On the other hand, LiveNet [PS003] is a good example of research project fitting in this group; it allows the creation and management of flexible shared workspaces that can be used for different purposes (e.g., to support architecture evaluation processes [PS003], or the arrangement of distributed meetings [PS004]).

*Knowledge management*

Knowledge management system refers to systems designed to manage the knowledge consumed and/or generated during software development, which is a knowledge-intensive undertaking. Archiving and sharing experiences, reasoning, and, generally speaking, information is an important process to allow people taking over existing work as well as independently understanding the rationale behind decisions allowing savings in terms of time and costs. Experience Browser [PS085] and Expertise Browser [99] are two similar systems developed by the academia to assist users in identifying experts in relation to specific artifacts or tasks; Collaborative Panel Administrator [PS015] is a framework developed with similar intents, which also provides the ability to validate the results given by the system. Zwiki and TWiki from the OSS community, and commercial products like TeamWeaver and Lotus Connections are some representative of this group.

*Software design*

Tools supporting software design activities like modeling are a fertile ground for research in the context of GSD. In fact, traditional tools are not able to support team members when collaborating with each other in a distributed fashion. Tools belonging to this group consist of all the applications used for supporting software design. Borland Together and Rational Rose are the commercially available examples reported the most; while ArgoUML is the only open source example mentioned. Camel [26], Libra-on-Chat [PS178], Odyssey [20], and STEVE [PS050] are the results of research projects. Similar to the tools identified for supporting software implementation, the representatives from the academia in this group are specifically built to investigate different ways of supporting distributed collaboration.

*Version control systems*

Considering the number of revision systems encountered, we decided to isolate version control systems in an independent group detached from the more generic one related to artifact management. In software engineering, source code control is a fundamental concern that team members strongly rely on for versioning code artifacts. Famous representatives from OSS are: CVS, SVN, Git, and Mercurial; COTS examples are: Rational ClearCase, Visual Paradigm TeamWork Center, and Microsoft Visual Source Safe. While, research-based technologies are represented by RCS [PS163], ADAMS [PS016], and the peer-to-peer tool proposed in [PS116]. Compared to OSS and COTS, research outcomes usually investigate particular features of such systems. ADAMS, for example, is a version control system designed to increase traceability by providing a fine-grained management of artifacts; whereas, the authors of [PS116], provided a complete peer-to-peer system for versioning artifacts able to avoid any centralized service supporting highly distributed contexts.

*Requirement management*

Requirement management is already considered a difficult field in traditional software engineering. Many challenges have to be taken into consideration when gathering requirements from stakeholders and when managing them; traceability of requirements throughout the software life-cycle also represents an interesting research area. However, when dealing with distributed teams, geographical, temporal, cultural, and linguistic distances [103] need to be properly addressed to restrict the issues to only the ones known to be challenging in traditional software engineering. Tools reported in this group refer therefore to systems giving support to teams in the handling of requirements. Examples of commercially available tools are Rational Requirements Composer, Rational RequisitePro as well as Microsoft Excel (extensively used by development teams). No open source technology was retrieved. Whereas, examples of research projects are ABREV [PS032], an agent based system to monitor requirements evolution; Arena II [PS150] and RM-Tool [PS100], which are web based system supporting distributed requirement management; and, Arena-M [PS150], which is a mobile version of Arena II.

*Quality management*

Whilst an organization may be able to release products faster by adopting the GSD model, the distances that characterize this paradigm may also have serious negative impact on the quality of the products. Thus, the choice and use of appropriate tools for quality management is very important. This group of tools includes (but is not limited to) tools for analyzing, monitoring, measuring and testing software quality. Examples are tools for memory debugging, for memory leaks detection, profilers, and test managers. CAST, HP Quality Center, Rational PurifyPlus, Rational Quality Manager, and Rational Test are commercial tools; OSS examples are CheckStyle, EMMA, FindBug, and Junit. We did not find any research-based technology for this group. Compared to traditional co-located development, dealing with software quality in a distributed environment does not entail many differences. Therefore, restricting the search to distributed development had the side effect of loosing the research projects that would have fit into this category.

*Software integration*

Automation tools are very important to release team members from the burden of performing cumbersome and repetitive tasks on artifacts in GSD projects. Automating such task improves also reliability increasing the confidence in outcomes shared among teams. The software integration group identifies systems that support automatic building and configuration. Continuous integration usually also comprises versioning systems for which we reserved a unique group. The majority of these tools are OSS including: Ant, AntHill, autoconf, automake, Apache Continuum, CruiseControl, make, and Maven. Further, a research project providing an infrastructure that allows the remote control of building and testing processes called SoftFab [PS159] was found.

*Bug and issue management systems*

During the data extraction, we also retrieved bug and issue management systems. This kind of tools are well-known to developers collaborating in open source initiatives. In such settings where communication is limited, collaboration and coordination are encouraged and supported by different practices. Open source projects heavily use these tools whose most famous representatives are Bugzilla, Mantis, and Trac; while, commercial tools found explicitly addressing this issue are Jira and Rational ClearQuest. Microsoft Excel was also reported to be used for bugs and issues management. PIMS [PS074] is the only research project retrieved addressing bugs and issues handling. Used to support processes at Fujitsu, PIMS is based on a web system that allows the automatic problem tracking through the use of workflows.

*Process management*

This category includes tools for software process modeling and workflow management. Support for these activities is normally included in project management systems; however, from the primary study we identified the following research projects: FlowManager [PS005], the workflow management system used by the Genesis platform; pmlcheck [PS010], which is used to examine the correctness of process models; SPEARMINT [115] and XCHIPS [116]; a framework for integrating process models defined using different notation [PS153]; and, a framework for process awareness [PS047].

*Software inspection*

When collaborating remotely, the possibility of inspecting code artifacts either for quality assurance activities or for collaboration purposes is necessary. Tools for software inspection are extensively studied by research groups including: ASSIST [PS031]; HyperCode [110]; XATI [68]; IBIS reported in [PS103], one of which functionalities is to support code reviews through a web interface; and, ISPIS [PS097], a framework able to support software inspection processes by allowing the integration of external tools.

*Office and productivity applications*

Six commercial tools have been reported to be used in GSD environments, which have been grouped under this category. Such tools in fact are used to support office related tasks and are normally not related to any software development activity. With the advent of cloud computing it can be seen that most of these software solutions have been enhanced in order to become tools able to support concurrent multiple use. Used for different purposes, the representative that we retrieved from the primary studies are: Google Apps, Microsoft Office, Microsoft SharePoint Online, Microsoft PowerPoint, SharePoint Insights, and Zoho Office Suite.

*Application integration systems*

Application integration systems have also been reported. Five solutions, all research projects, have been retrieved for this group: Eureka Software Factory (ESF) and ISM [PS142]; systems designed to provide a federation of components interacting through a specific communication framework (e.g., [PS158] and Eleggua [25]); and, Software Bus [PS169], a system designed to allow the integration of heterogeneous software engineering applications.

*Others*

Finally, there are some tools that do not properly fit in any of the above described groups; they include:

- tools for remote access;
- decision support systems; and,
- risk analysis tools.

Tools that overcome physical distance belong to the first group. The four solutions we found in the primary studies are: pcAnywhere, Remote Administrator, Remote Desktop, and the open source VNC. These tools allow remote collaboration by directly accessing and taking over control of remote workstations. The second group is composed of TeamSpirit [PS041] and TeamVote[PS065] described in Section 4.4. Risk Barometer [PS155] a tool for predicting project outcomes relying on historical data is the only tool we found that addressed risk analysis support.

## 5. Threats to validity

A systematic mapping study such as this one has to consider several threats to validity. In this section we report the potential threats to the validity of this research and describe how we have tried to mitigate their potential impact.

One of the reasons for performing a systematic study is to get a higher reliability with regards to the *completeness of the retrieved primary studies*. When performing an automatic search, two factors may influence the set of primary studies about to be retrieved: the choice related to which sources to query, and the selection of the query string to use. We performed the search on five digital libraries, which are the ones related to the most relevant publishers in software engineering (i.e., IEEE, ACM, Springer, Wiley, and ScienceDirect). A good practice to improve the results in terms of completeness—which however we did not follow—is to include also digital libraries that are meta search engines and index publications from different external sources. Examples of these libraries are ISI Web of Knowledge[23] and Compendex[24]. However, for this kind of comprehensive search, research teams need to have significant funding support because their institutions are unlikely to have such comprehensive subscriptions. A critical step often

[23] ISI Web of Knowledge: `isiknowledge.com`

[24] Compendex: `digital.library.wisc.edu/1711.web/compendex`

overlooked in systematic studies based on automatic searches is the correct *identification of keywords*. An erroneous selection at this stage can result in an incomplete retrieval of primary studies and, the diversity of terminology used by researchers to refer to the concepts, can make this task even more challenging. Techniques, like the quasi-gold standard [141], have been proposed to address this issue; however, we decided to manually select them by reviewing similar literature reviews (e.g., [PS148]) and by relying on our personal knowledge. However, to assess the quality of the key phrases selected, we took two safety measures: on the one hand, we followed the suggestion proposed by Kitchenham [86] of performing a pilot search; and, on the other hand, we used the results of the pilot search to check the presence of previously identified relevant papers, which eventually unveiled the need to include additional terms. These modifications of the set of keywords resulted in an iterative process during the pilot search phase as described in Section 3.2. Finally, we take the opportunity to report a malfunction encountered with Wiley InterScience digital library that retrieved an enormous number of studies some of which did not contain any of the provided keywords and were clearly unrelated. This set of papers included publication from the history field as well as the medical field which needed to be removed during the screening stage. Due to this malfunction we believe that the calculation of indexes, like precision and sensitivity [141] to evaluate the quality of the searches and search string used, lost relevance.

Once publications are retrieved, the subsequent phases towards the reporting of the results are not anymore partially supported by applications but are entirely left to the researchers, thus *subjective and potentially biased*. Furthermore, as reported by Turner in [135], when handling a large number of primary studies, data extraction and aggregation problems may arise. To reduce this threat, all activities were checked by a second researcher as described in Section 3.1; and, in case of disagreements, consensus meetings were held to meet approbation.

Considering that not all possible sources were used to perform the searches and that the use of manual search techniques was avoided, this review may have missed some relevant papers. Nevertheless, the primary studies identified and the number of retrieved information seems sufficient to gain a deep understanding of the topic investigated.

Finally, as elaborated in Section 3.1, the process used in this study has deviated from the one proposed in [111]. The data we were interested in (i.e., the concrete tools) could not be framed in the query string. Therefore, the real objective needed to be abstracted by using keywords identifying the concept sought (e.g., tools, plug-ins, frameworks, and so on). We iteratively built the search string and we are confident that the terms used properly cover the available form of software technological support available. However, we started with the assumption that a publication reporting a concrete technology would have used one of these keywords. It might be therefore argued that there is no real correlation between the primary studies and the data extracted. The aim of this study was to identify and report in a comprehensive way the technological support known and reported in scientific publications; for this purpose, we leveraged the systematic mapping study technique to restrict the set of publications to the ones related to the global software development field that are reporting technologies.

## 6. Conclusions

Over the last two decades, an increasing number of companies, large as well as small, have been adopting geographically distributed software development paradigm that has made software development a multi-site, multi-culture, and global undertaking. All forms of geographical distribution of software development activities have come to be known as Global Software Development (GSD). This approach promises many benefits as well as presenting several technical and socio-technical challenges. One of these challenges is to provide globally distributed software development teams with appropriate tools. Over the years, a large number of tools have been developed and used for supporting GSD projects by both industry and academia. The main motivation of this work was to provide an overview of the state-of-the-art of GSD tools (i.e., technologies developed and/or used for supporting GSD teams) in order to help guide the future research in this area and practitioners in their search for appropriate GSD tools.

We have identified, analyzed, and summarized the existing information about different aspects of GSD tools through a systematic mapping study, which is one of the key research methodologies of Evidence-Based Software Engineering (EBSE) [111]. This mapping study has identified and classified 412 technologies that have been developed and/or used for supporting GSD projects from 182 papers selected from an initial set of 1958 papers published in peer-reviewed literature by querying five digital libraries. We applied well-defined clustering schemas to provide a mapping of the reported technologies for GSD support. The findings from this study are expected to provide useful and important information to GSD researchers and practitioners interested in information about different aspects of the technologies developed and/or used for GSD projects. We have structured and presented the findings from this SMS in a way to help readers to easily find the answers to our research questions (i.e., Section 4.1 is dedicated to RQ1, Section 4.3 answered RQ3, Section 4.4 covered RQ4, and Section 4.5 focused on RQ5). In the following paragraphs, we discuss the way this mapping study has addressed the RQ2 and emphasize its main contributions.

In order to answer RQ2, we have structured and presented the information about the identified technologies in several ways. We provide a few maps (Figures 10 to 13) on which all the identified technologies have been classified according to the four classification schemas used throughout this study. We also present more specific views of the findings by filtering out the technologies which were reported in publications older than 2010. This kind of concise overview of the GSD technologies reported during the last two years can provide practitioners with information that can be useful in searching and selecting appropriate technologies for supporting GSD teams. Moreover, we also provide a complete list of all the identified technologies in Appendix B along with the reference(s) from which each technology has been extracted. We believe that such a list can facilitate a quick browsing through the 412 technologies found in the primary studies included in this systematic mapping study. Furthermore, to gather the information contained in this study for each technology the primary source reporting its information was also retrieved (i.e., either the paper reporting the research project, or the reference web-site); such information is included in the list presented in Appendix B by either reporting the web site or by underlying the primary reference. Finally, we relate some of the well-known research agendas for GSD related topics with the findings from this study in order to pursue our goal to guide future research based on the findings from this piece of work.

As stated in the previous paragraph, we have provided visual summaries of the identified technologies by plotting them on a map after merging all of the four classification schemas to help easily understand the findings. Our main purpose of presenting the identified technologies through a classification map was to provide researchers and practitioners with a framework that can help researchers to identify the gaps in tool support for GSD projects, and to guide practitioners to choose an appropriate technology to support a particular task or to find a suitable alternative to the one that they may currently be using. In doing so, we noticed that, especially in the case of practitioners using the presented map, having the list of tools representing the entire historical evolution of a specific area might provide too much information. Therefore, we decided, once Figure 10 and 11 were designed, to filter and present the information about a relatively smaller subset of the identified technologies. Given the fact that the lifetime of support technologies is usually short, we chose to highlight the ones that have been reported from 2010. This decision has resulted in an outcome comprising 43% of the overall set of technologies (179 out of 412). Figures 10 to 13 not only answer RQ2 of this study (i.e., "which are the technologies that have been reported in peer reviewed literature for supporting GSD teams?"), but also provide a classification framework that can support analytical comparison of the identified technologies for supporting GSD projects.

The technologies shown in Figures 10 to 13 have been classified according to the schemas reported in Section 3.5. The columns in the figures represent the classification based on the main purpose of a particular technology. Each column has been divided in six blocks identifying the form of support, i.e., from top: standalone tools, environment, framework, plug-in, platform, and middleware. Furthermore, the tools reported in each block are listed in three distinct sections indicating the licensing types of different tools (i.e., research, OSS, and COTS). Finally, a glyph has been placed underneath each technology name to represent the classification based on the 3C model. The glyphs are composed of three rectangles identifying communication, collaboration, and coordination, which are blackened to signify that the related characteristic is proper of the tool. A meta-model describing how to read the map of the technologies has been provided in Figure 9 and has been repeated in each map for the reader's convenience. To help understand Figure 10 to Figure 13,

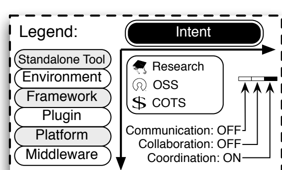



Figure 9: Meta-model for Figure 10 to Figure 13

we describe how to read the information about two technologies plotted in those figures: Eclipse and Rational Suite. Eclipse is reported in the column

*Implementation*; placed in the *environment* box with the other open source projects; and, considering the glyph, it is considered a single user tool, one of those designed without specifically considering the unique requirements of the 3C dimensions. The latter, Rational Suite, is reported under *Project Management* in the *platform* box; it is listed in the commercial section, and its glyph is blackened for all the dimensions of the 3C model. In fact, Rational Suite is a comprehensive platform by IBM that can be augmented by all components of the Rational product family.

In the following paragraphs, we discuss the findings from this study with respect to the GSD research agenda points proposed in the papers by Whitehead [137], Herbsleb [69], and Sengupta et al. [126]. Based on the findings from this study, we can conclude that the research interest focusing on GSD has been continuously increasing over the last few years. This has resulted in a large variety of technological solutions ranging from standalone tools to integrated environments, from frameworks to solutions based on plug-ins, from middleware to platforms. However, there is no one-size-fits-all kind of solution, i.e., so-called a silver bullet. It is inevitable that different people/teams/companies will adopt different practices and tools when undertaking their software development efforts with geographically distributed teams. Whilst the evidence reported in the literature shows that GSD initiatives can be successful, the challenges caused by the well known GSD distribution factors (i.e., geographical, temporal, cultural, and linguistic) need to be successfully overcome through appropriate technologies support in order to fully fulfill the goals of GSD endeavors. The importance of appropriate technological support for GSD teams has also been highlighted in all the three research agenda papers (i.e., [137], [69], and [126]), which identify the collaborative software development tools and innovative knowledge management systems as two of the main GSD research challenges that need to be addressed.

For the collaborative tools category, we found many proposed solutions (e.g., Stellation and CollabNet) including IBM's and Microsoft's platforms that can support all phases of a software development lifecycle (e.g., IBM Rational Suite based on the Jazz platform and Visual Studio Team System). Such comprehensive solutions are able to substantially mitigate several kinds of challenges that usually characterize GSD context (e.g., traceability from the requirement elicitation stages to the testing ones). However, the final goal of providing virtual proximity should be pursued with an innovative perspective without focusing on specific cases but trying to provide reusable solutions, that would be flexible enough to accommodate the specific needs of different practitioners adopting the GSD paradigm. Compared to the past, software development has become an increasingly collaborative activity and hardware advancements have enabled previously unimaginable potential (e.g., broadband communication infrastructures and new computing delivery models). We found only few attempts at providing integration at a broad level; and none of them leverage alternative computing models like cloud computing.

For the knowledge management systems, we did not find many systems addressing the issues related to knowledge management as proposed by Whitehead, Herbsleb, and Sengupta et al.: systems to explore, leverage, and enrich project memory [69]; to recover and integrate knowledge from formal and informal artifacts as well as to maintain informal knowledge in a human independent way [126]; and record organizational knowledge [137]. Most of these concerns, however, appear to have been addressed from the end user's perspective in applications that have been designed to improve awareness of contextual information, and of individual and group activities. Whilst it is important to make technical, socio-technical, and contextual knowledge potentially accessible for GSD teams, it is also equally important to be aware of the existence of knowledge through appropriate mechanisms of creating and maintaining awareness. Four different types of awareness have been identified by the Computer-Supported Cooperative Work (CSCW) community in the case of co-located environment. These have been summarized and related to each other by Gutwin et al. in [60]: workspace awareness, informal awareness, group-structural awareness, and social awareness.

We believe that these four types of awareness directly relate to the needs of improved knowledge management identified by the research agenda papers (e.g., the knowledge of who should coordinate with whom is directly related to the group-structural awareness), and, if appropriately supported, would represent an important step towards bridging the gap between co-located and distributed environments. Based on the findings from this mapping study and an emerging position of other researchers (e.g., [55]), we posit that awareness should be elevated to the level of the dimensions of the 3C model [134]. Nonetheless, starting by being addressed as challenging dimensions to be tackled when providing technological support for GSD teams, communication, coordination, collaboration, and awareness continue to be considered the main aspects that need to be supported; an indication of the potential need for a change in the 3C model since it was proposed in [47]? Many solutions have been designed and demonstrated in promising evaluations. Existing tools have also been enhanced by means of plug-ins (e.g., Palantir [118]), which in some cases have been accepted by practitioners insomuch that successful research projects have become freely available in the form of OSS and continue to be improved by online communities (i.e., Mylyn[25]). We have also observed that standalone tools continue to be the most leveraged form of technological support available to GSD teams even though the use of such tools increases context switches, which many practitioners consider as one of the main sources of frustration [126]; and, practitioners are still keen on using standalone tools that do not support any kind of computer-mediated teamwork (e.g., Microsoft Excel reported to be used for both requirement and bug/issue tracking [PS172, PS057]).

The findings from this study and our own observations from other GSD literature and practice have also enabled us to conclude that the future research on technological support for GSD teams would greatly benefit from allocating more effort to devising technological solutions that can fulfill several kinds of requirements necessitated by different dimensions of a GSD context rather than those which tackle only very specific issues without paying much attention to a broader set of challenges being faced by GSD practitioners. For example, a system providing awareness information should be designed by taking into consideration the importance of providing information for other recipients thus adopting standardized communication mechanism; such a system should provide APIs or be based on service-oriented solutions that allow external systems to make use of the generated information. That is why it should not be a surprise that the well known IDEs like Eclipse and Visual Studio are heavily leveraged for building add-on features since they expose most of their functionalities by means of APIs and are ideal systems to be enhanced and leveraged.

For supporting the future similar studies and contributing to the growing body of knowledge about conducting literature reviews systematically (i.e., SMS or SLR), we not only report the complete and detailed information about the search strings used for searching the relevant primary studies from different search engines, we also provide an annotated commentary about the challenges that we faced and the workarounds that we devised for overcoming such challenges and mitigating their potential effects on our effort to search as many of the relevant primary studies as possible. We believe that this kind

[25]Known as Mylar during its research stage [84].

of information can be a useful and quick reference that can be leveraged by other researchers interested in performing automatic searches in the context of Evidence-Based Software Engineering; in Appendix A, we have reported the limitations, lessons learned, and tips for working with each of the search engines that were used during the search stages of this study.

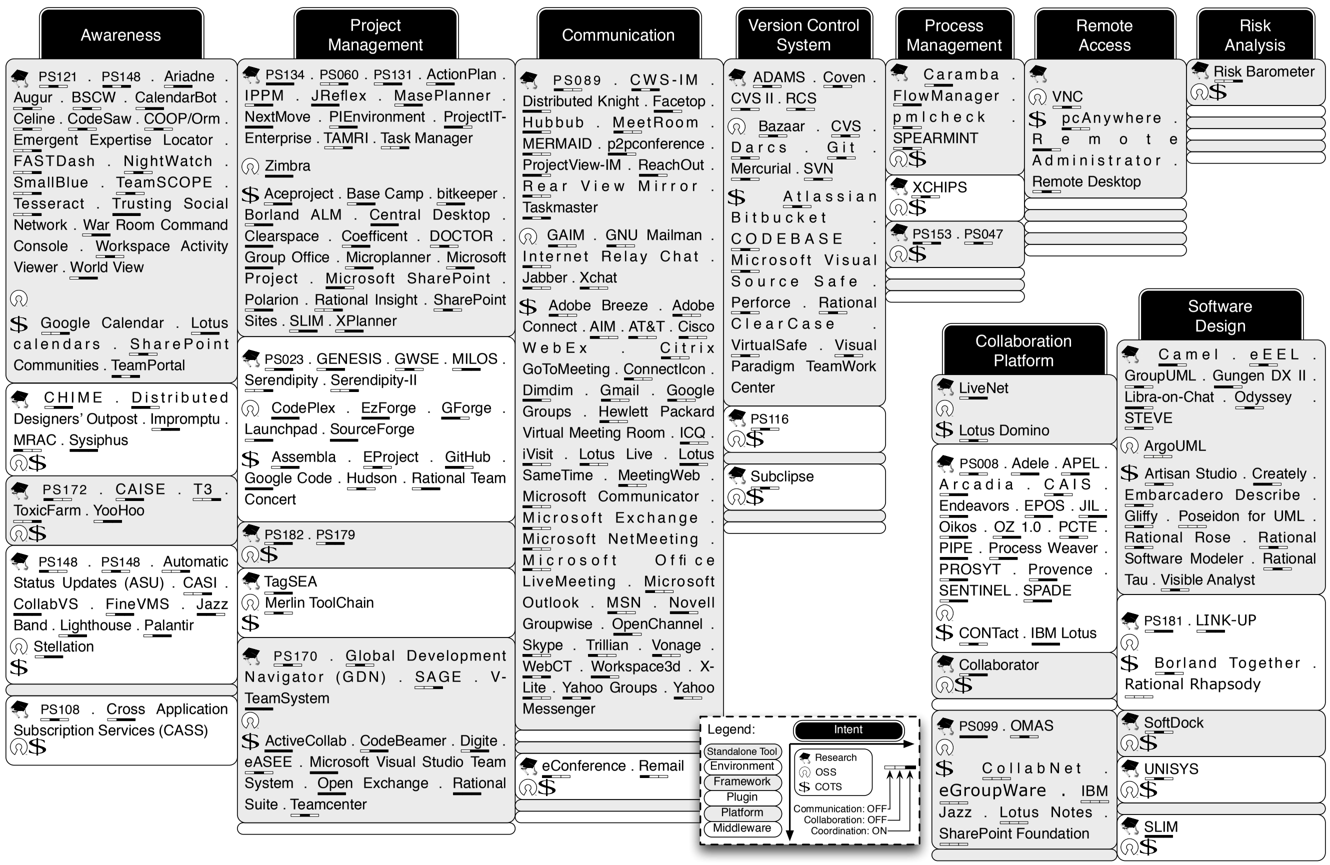



Figure 10: Map$_{01}$. All the technologies retrieved are presented applying all the four classification schemas utilized throughout this SMS.

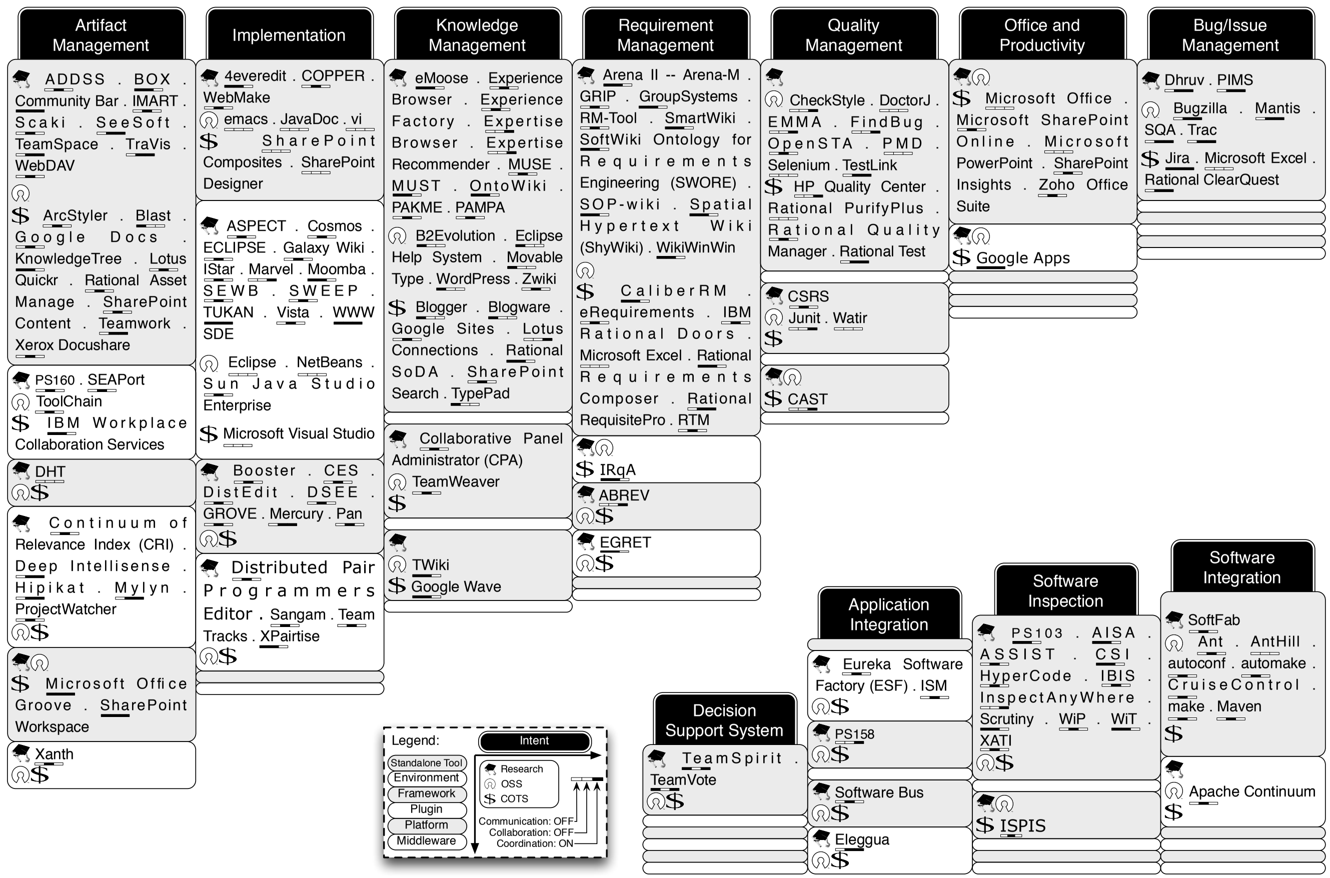



Figure 11: Map$_{02}$. All the technologies retrieved are presented applying all the four classification schemas utilized throughout this SMS.

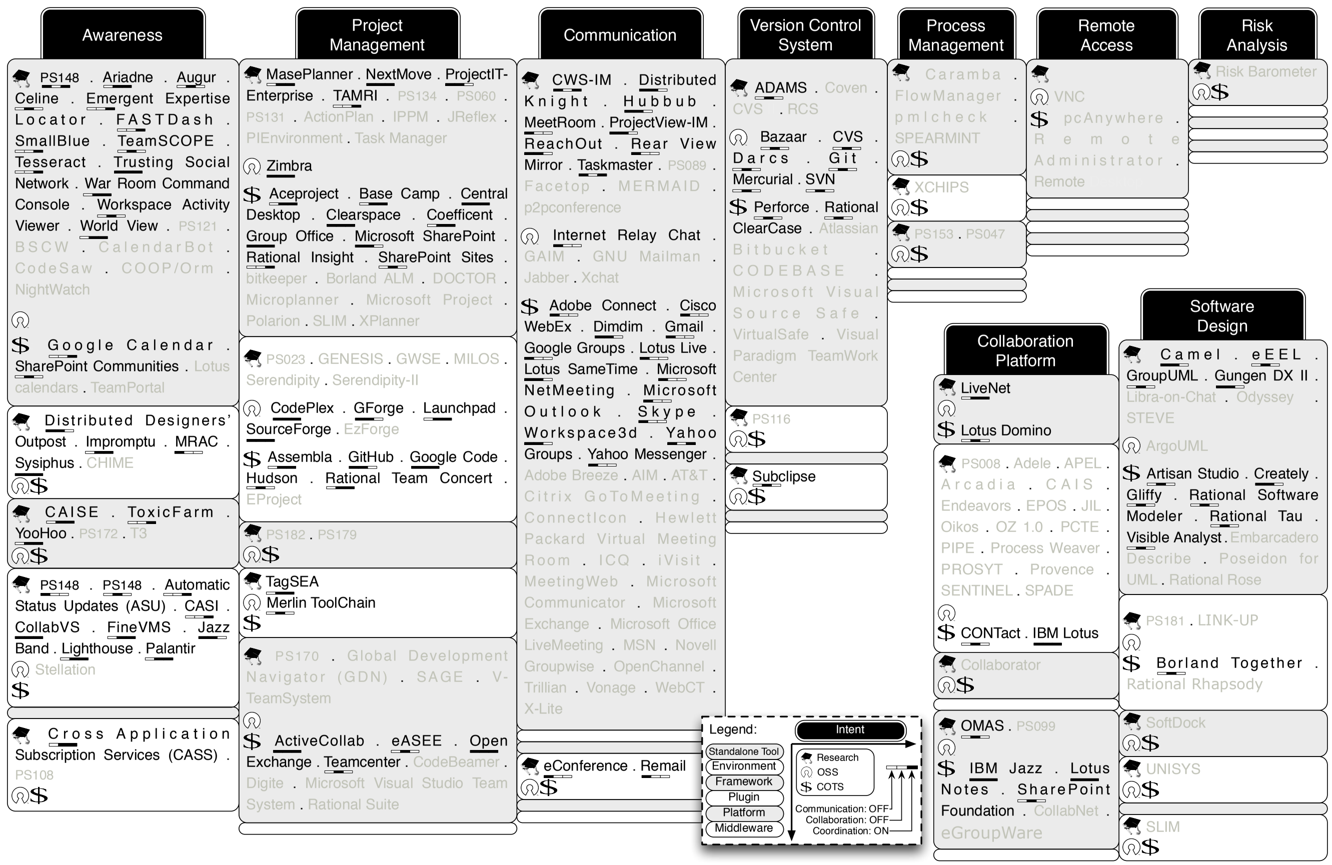



Figure 12: $Map_{03}$ A filtered version of $Map_{01}$; technologies retrieved in publications older than 2010 have been shaded highlighting the ones reported in the last 2 years.

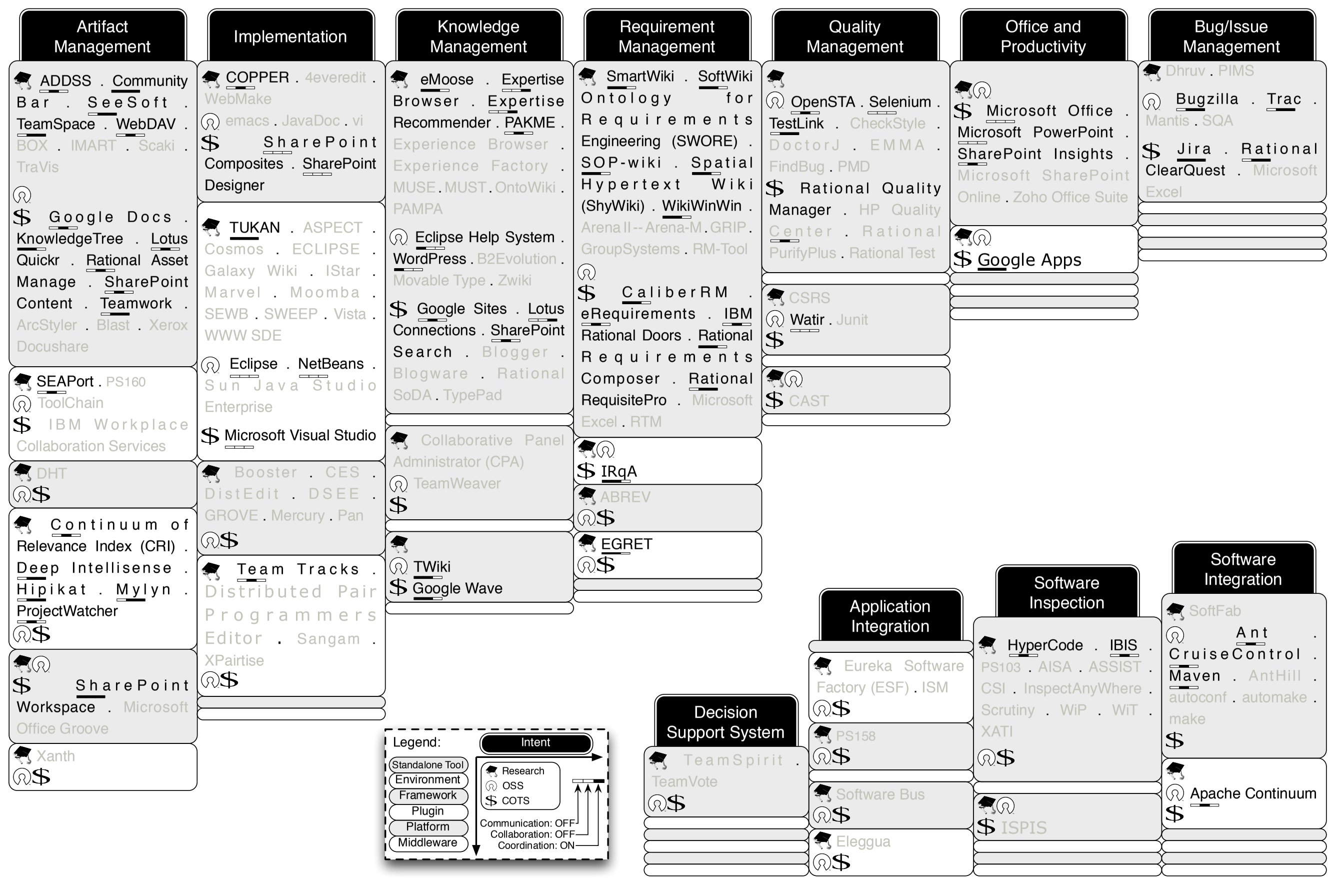



Figure 13: $Map_{04}$ A filtered version of $Map_{02}$; technologies retrieved in publications older than 2010 have been shaded highlighting the ones reported in the last 2 years.

**Acknowledgment**

We would like to note down our appreciation for the useful comments received from Klaas-Jan Stol on an earlier draft of this report. A particular thanks is owed to Barbara Kitchenham for the fruitful discussions and her neat observations. Finally, a last acknowledgment goes to Kostas Pantazos for the timely contribution provided.

## 7. References

[1] B. Al-Ani, E. Trainer, R. Ripley, and A. Sarma. Continuous coordination within the context of cooperative and human aspects of Software Engineering. *Proceedings of the 2008 international workshop on Cooperative and human aspects of software engineering*, 2008.

[2] V. Ambriola, P. Ciancarini, and C. Montangero. Software process enactment in Oikos. In *Proceedings of the fourth ACM SIGSOFT symposium on Software development environments*, pages 183–192, New York, NY, USA, 1990. ACM.

[3] A. Ankolekar, K. Sycara, J. Herbsleb, R. Kraut, and C. Welty. Supporting online problem-solving communities with the semantic web. In *Proceedings of the 15th international conference on World Wide Web*, pages 575–584, New York, NY, USA, 2006. ACM.

[4] W. Appelt. WWW Based Collaboration with the BSCW System. In *Proceedings of the 26th Conference on Current Trends in Theory and Practice of Informatics on Theory and Practice of Informatics*, pages 66–78, London, UK, UK, 1999. Springer-Verlag.

[5] M. A. Babar and I. Gorton. A Tool for Managing Software Architecture Knowledge. *Sharing and Reusing Architectural Knowledge - Architecture, Rationale, and Design Intent, 2007. SHARK/ADI '07: ICSE Workshops 2007. Second Workshop on*, pages 11–11, 2007.

[6] R. A. Ballance, S. L. Graham, and M. L. Van De Vanter. The Pan language-based editing system. *ACM Trans. Softw. Eng. Methodol.*, 1(1):95–127, 1992.

[7] S. Bandinelli, A. Fuggetta, C. Ghezzi, and L. Lavazza. SPADE: An environment for software process analysis, design, and enactment. *Software process modelling and technology*, pages 223–247, 1994.

[8] N. S. Barghouti and B. Krishnamurthy. Using event contexts and matching constraints to monitor software processes. In *Proceedings of the 17th international conference on Software engineering*, pages 83–92, New York, NY, USA, 1995. ACM.

[9] V. R. C. G. Basili and H. D. Rombach. Experience Factory. In *Encyclopedia of Software Engineering*. John Wiley & Sons, Inc., 2002.

[10] N. Belkhatir, J. Estublier, and W. Melo. A Support to Large Software Development Process. In *Software Process, 1991. Proceedings. First International Conference on the*, pages 159–170, Oct. 1991.

[11] V. Bellotti, N. Ducheneaut, M. Howard, and I. Smith. Taskmaster: recasting email as task management. In *Workshop: Redesigning Email for the 21st Century, CSCW*, 2002.

[12] B. Berliner and others. CVS II: Parallelizing software development. In *Proceedings of the USENIX Winter 1990 Technical Conference*, page 352, 1990.

[13] J. T. Biehl and B. P. Bailey. Improving interfaces for managing applications in multiple-device environments. In *Proceedings of the working conference on Advanced visual interfaces*, pages 35–42. ACM, 2006.

[14] J. T. Biehl, W. T. Baker, B. P. Bailey, D. S. Tan, K. M. Inkpen, and M. Czerwinski. Impromptu: a new interaction framework for supporting collaboration in multiple display environments and its field evaluation for co-located software development. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 939–948, New York, NY, USA, 2008. ACM.

[15] J. T. Biehl, M. Czerwinski, G. Smith, and G. G. Robertson. FASTDash: a visual dashboard for fostering awareness in software teams. *Proceedings of the SIGCHI conference on Human Factors in computing systems*, 2007.

[16] G. A. Bolcer and R. N. Taylor. Endeavors: a process system integration infrastructure. In *Proceedings of the Fourth International Conference on the Software Process (ICSP '96)*, pages 76–76, Washington, DC, USA, 1996. IEEE Computer Society.

[17] G. Booch and A. Brown. Collaborative development environments. *Advances in Computers*, 59:2–29, 2003.

[18] N. Boulila. Group support for distributed collaborative concurrent software modeling. *Automated Software Engineering, 2004. Proceedings. 19th International Conference on*, pages 422–425, 2004.

[19] D. G. Boyer, M. J. Handel, and J. Herbsleb. Virtual community prescence awareness. *SIGGROUP Bull.*, 19:11–14, Dec. 1998.

[20] R. M. M. Braga, C. M. L. Werner, and M. Mattoso. Odyssey: A Reuse Environment based on Domain Models. In *Proceedings of the 1999 IEEE Symposium on Application - Specific Systems and Software Engineering and Technology*, pages 50–50, Washington, DC, USA, 1999. IEEE Computer Society.

[21] F. Calefato, D. Gendarmi, and F. Lanubile. Embedding social networking information into jazz to foster group awareness within distributed teams. In *Proceedings of the 2nd international workshop on Social software engineering and applications*, pages 23–28, New York, NY, USA, 2009. ACM.

[22] F. Calefato, F. Lanubile, and T. Mallardo. Peer-to-peer remote conferencing. *IEE Seminar Digests*, 2004(912):34–38, 2004.

[23] J. Camacho, J. Favela, and V. M. Gonzalez. Supporting the management of multiple activities in mobile collaborative working environments. In *Proceedings of the 12th international conference on Groupware: design, implementation, and use*, pages 381–388, Berlin, Heidelberg, 2006. Springer-Verlag.

[24] R. Capilla, F. Nava, S. Pérez, and J. C. Due n as. A web-based tool for managing architectural design decisions. *SIGSOFT Softw. Eng. Notes*, 31(5), 2006.

[25] R. Casallas, N. L o pez, and D. Correal. Eleggua: An Event Infrastructure for Application Cooperation. In K. Turowski and J. M. Zaha, editors, *Component-Oriented Enterprise Applications, Proceedings of the Conference on Component-Oriented Enterprise Applications (COEA 2005)*, pages 109–123. GI, 2005.

[26] M. Cataldo, C. Shelton, Y. Choi, Y.-Y. Huang, V. Ramesh, D. Saini, and L.-Y. Wang. CAMEL: A Tool for Collaborative Distributed Software Design. *Global Software Engineering, 2009. ICGSE 2009. Fourth IEEE International Conference on*, pages 83–92, 2009.

[27] L.-T. Cheng, S. Hupfer, S. Ross, and J. Patterson. Jazzing up Eclipse with collaborative tools. In *Proceedings of the 2003 OOPSLA workshop on eclipse technology eXchange*, pages 45–49, New York, NY, USA, 2003. ACM.

[28] C. M. Chewar, E. Bachetti, D. S. McCrickard, and J. E. Booker. Automating a design reuse facility with critical parameters: lessons learned in developing the link-up system. In *In Proceedings of the Conference on Computer-Aided Design of User Interfaces (CADUI*, pages 236–247, 2004.

[29] M. C. Chu-Carroll and S. Sprenkle. Coven: brewing better collaboration through software configuration management. In *Proceedings of the 8th ACM SIGSOFT international symposium on Foundations of software engineering: twenty-first century applications*, pages 88–97, New York, NY, USA, 2000. ACM.

[30] C. Cook, N. Churcher, and W. Irwin. Towards Synchronous Collaborative Software Engineering. In *Proceedings of the 11th Asia-Pacific Software Engineering Conference*, pages 230–239, Washington, DC, USA, 2004. IEEE Computer Society.

[31] D. Cubranic, G. C. Murphy, J. Singer, and K. S. Booth. Hipikat: A Project Memory for Software Development. *Software Engineering, IEEE Transactions on*, 31:446–465, June 2005.

[32] G. Cugola, E. Di Nitto, C. Ghezzi, and M. Mantione. How to deal with deviations during process model enactment. In *Proceedings of the 17th*

*international conference on Software engineering*, pages 265–273, New York, NY, USA, 1995. ACM.

[33] I. A. da Silva, P. H. Chen, C. Van der Westhuizen, R. M. Ripley, and A. van der Hoek. Lighthouse: coordination through emerging design. In *Proceedings of the 2006 OOPSLA workshop on eclipse technology eXchange*, pages 11–15, New York, NY, USA, 2006. ACM.

[34] I. F. da Silva, P. A. da Mota Silveira Neto, P. O'Leary, E. S. de Almeida, and S. R. de Lemos Meira. Agile software product lines: a systematic mapping study. *Software: Practice and Experience*, 41(8):899–920, 2011.

[35] D. Damian and D. Moitra. Guest Editors' Introduction: Global Software Development: How Far Have We Come? *IEEE Software*, 23(5):17–19, 2006.

[36] A. De Lucia, F. Fasano, R. Oliveto, and D. Santonicola. Improving context awareness in subversion through fine-grained versioning of Java code. In *Ninth international workshop on Principles of software evolution: in conjunction with the 6th ESEC/FSE joint meeting*, pages 110–113, New York, NY, USA, 2007. ACM.

[37] C. De Souza. From technical dependencies to social dependencies. In *In: Proceedings of the Social Networks Workshop at the 2004 International Conference on CSCW*, pages 6–10, 2004.

[38] B. Decker, E. Ras, J. Rech, P. Jaubert, and M. Rieth. Wiki-Based Stakeholder Participation in Requirements Engineering. *Software, IEEE*, 24(2):28–35, Mar. 2007.

[39] U. Dekel and J. D. Herbsleb. Pushing relevant artifact annotations in collaborative software development. In *Proceedings of the 2008 ACM conference on Computer supported cooperative work*, pages 1–4, New York, NY, USA, 2008. ACM.

[40] R. DeLine, A. Khella, M. Czerwinski, and G. Robertson. Towards understanding programs through wear-based filtering. In *Proceedings of the 2005 ACM symposium on Software visualization*, pages 183–192, New York, NY, USA, 2005. ACM.

[41] P. Dewan and R. Hegde. Semi-synchronous conflict detection and resolution in asynchronous software development. In *ECSCW 2007*. 2007.

[42] P. Dourish and V. Bellotti. Awareness and coordination in shared workspaces. *CSCW '92 Proceedings of the 1992 ACM conference on Computer-supported cooperative work*, pages 107–114, 1992.

[43] M. Dowson. Integrated Project Support with IStar. *Software, IEEE*, 4(6):6–15, Nov. 1987.

[44] S. Dustdar. Caramba—A Process-Aware Collaboration System Supporting Ad hoc and Collaborative Processes in Virtual Teams. *Distributed and Parallel Databases*, 15:45–66, 2004.

[45] K. Ehrlich, C.-Y. Lin, and V. Griffiths-Fisher. Searching for experts in the enterprise: combining text and social network analysis. In *Proceedings of the 2007 international ACM conference on Supporting group work*, pages 117–126, New York, NY, USA, 2007. ACM.

[46] S. Eick, J. Steffen, and E. J. Sumner. Seesoft-a tool for visualizing line oriented software statistics. *Software Engineering, IEEE Transactions on*, 18(11):957–968, Nov. 1992.

[47] C. A. Ellis, S. J. Gibbs, and G. Rein. Groupware: some issues and experiences. *Communications of the ACM*, 34(1):39–58, 1991.

[48] E. Engström and P. Runeson. Software product line testing - A systematic mapping study. *Information and Software Technology*, 53:2–13, Jan. 2011.

[49] J. Estublier, P. Cunin, N. Belkhatir, M. Amiour, and S. Dami. Architectures for process support system interoperability. In *Proceedings of the 5th International Conference on The Software Process, The International Software Process Association Press, Lisle, IL*. Citeseer, 1998.

[50] J. Estublier and S. Garcia. Process model and awareness in SCM. In *Proceedings of the 12th international workshop on Software configuration management*, pages 59–74, New York, NY, USA, 2005. ACM.

[51] C. Fernstrom. Process Weaver: adding process support to UNIX. In *Software Process, 1993. Continuous Software Process Improvement, Second International Conference on the*, pages 12–26, Feb. 1993.

[52] C. Fernström and L. Ohlsson. ESF - an approach to industrial software production. In *Proceedings of the fourth conference on Software engineering environments: research and practice*, pages 17–28, Upper Saddle River, NJ, USA, 1989. Ellis Horwood.

[53] D. Ferreira and A. Silva. An Enhanced Wiki for Requirements Engineering. In *Euromicro Conference, 2002. Proceedings. 28th*, pages 87–94, Aug. 2009.

[54] G. Fitzpatrick, P. Marshall, and A. Phillips. CVS integration with notification and chat: lightweight software team collaboration. In *Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, pages 49–58, New York, NY, USA, 2006. ACM.

[55] H. Fuks, A. Raposo, M. Gerosa, M. Pimentel, and C. Lucena. The 3c collaboration model. *The Encyclopedia of E-Collaboration, Ned Kock (org)*, pages 637–644, 2007.

[56] J. Gintell, J. Arnold, M. Houde, J. Kruszelnicki, R. McKenney, and G. Memmi. Scrutiny: A Collaborative Inspection and Review System. In *Proceedings of the 4th European Software Engineering Conference on Software Engineering*, pages 344–360, London, UK, UK, 1993. Springer-Verlag.

[57] S. Greenberg and C. Gutwin. Providing artifact awareness to a distributed group through screen sharing. In *Proceedings*, pages 99–108. New York, NY, USA, 2006.

[58] I. Greif, R. Seliger, and W. E. Weihl. Atomic data abstractions in a distributed collaborative editing system. In *Proceedings of the 13th ACM SIGACT-SIGPLAN symposium on Principles of programming languages*, pages 160–172, New York, NY, USA, 1986. ACM.

[59] J. Grundy and J. Hosking. Serendipity: Integrated Environment Support for Process Modelling, Enactment and Work Coordination. *Automated Software Engineering*, 5(1):27–60, 1998.

[60] C. Gutwin, S. Greenberg, and M. Roseman. Workspace Awareness in Real-Time Distributed Groupware: Framework, Widgets, and Evaluation. In *Proceedings of HCI on People and Computers XI*, pages 281–298, London, UK, 1996. Springer-Verlag.

[61] C. Gutwin, K. Schneider, D. Paquette, and R. Penner. Supporting Group Awareness in Distributed Software Development. In *Engineering Human Computer Interaction and Interactive Systems*. 2005.

[62] B. Hakami. ISM: a knowledge-based project support system. In *Proceedings of the fourth conference on Software engineering environments: research and practice*, pages 217–232, Upper Saddle River, NJ, USA, 1989. Ellis Horwood.

[63] J. Hall, P. Hitchcock, and R. Took. An overview of the ASPECT architecture. *Integrated project support environments*, pages 86–99, 1985.

[64] M. Hallin, P. Gr u nbacher, and S. Biffl. Tailoring a COTS Group Support System for Software Requirements Inspection. In *Proceedings of the 16th IEEE international conference on Automated software engineering*, pages 201–201, Washington, DC, USA, 2001. IEEE Computer Society.

[65] K. M. Hansen and C. H. Damm. Instant collaboration: using context-aware instant messaging for session management in distributed collaboration tools. In *Proceedings of the second Nordic conference on Human-computer interaction*, pages 279–282, New York, NY, USA, 2002. ACM.

[66] L. Harjumaa and I. Tervonen. A WWW-Based Tool for Software Inspection. In *Proceedings of the Thirty-First Annual Hawaii International Conference on System Sciences - Volume 3*, pages 379–379, Washington, DC, USA, 1998. IEEE Computer Society.

[67] L. Harjumaa and I. Tervonen. Virtual Software Inspections over the Internet. In *Proc. of the 3rd ICSE Workshop on Software Engineering over the Internet*, pages 30–40, 2000.

[68] H. Hedberg and L. Harjumaa. Virtual software inspections for distributed software engineering projects. In *Proc. of the ICSE Int. Workshop on Global Software Development*. Citeseer, 2002.

[69] J. Herbsleb. Global software engineering: The future of socio-technical coordination. *2007 Future of Software Engineering*, 2007.

[70] J. Herbsleb and D. Moitra. Global software development. *IEEE Software*, 18(2):16–20, 2001.

[71] C.-W. Ho, S. Raha, E. Gehringer, and L. Williams. Sangam: a distributed pair programming plug-in for Eclipse. In *Proceedings of the 2004 OOPSLA workshop on eclipse technology eXchange*, pages 73–77, New York, NY, USA, 2004. ACM.

[72] R. Holmes and A. Begel. Deep intellisense: a tool for rehydrating evaporated information. In *Proceedings of the 2008 international working conference on Mining software repositories*, pages 23–26, New York, NY, USA, 2008. ACM.

[73] R. Holmes and R. J. Walker. Promoting developer-specific awareness. In *Proceedings of the 2008 international workshop on Cooperative and human aspects of software engineering*, pages 61–64, New York, NY, USA, 2008. ACM.

[74] E. Hossain, M. A. Babar, and H.-y. Paik. Using Scrum in Global Software Development: A Systematic Literature Review. In *Global Software Engineering, 2009. ICGSE 2009. Fourth IEEE International Conference on*, pages 175–184, 2009.

[75] C. L. Ignat. Annotation of concurrent changes in collaborative software development. In *Intelligent Computer Communication and Processing, 2008. ICCP 2008. 4th International Conference on*, pages 137–144, Aug. 2008.

[76] E. Isaacs, A. Walendowski, and D. Ranganthan. Hubbub: a sound-enhanced mobile instant messenger that supports awareness and opportunistic interactions. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Changing our world, changing ourselves*, pages 179–186, New York, NY, USA, 2002. ACM.

[77] L. Jaccheri, J. Larsen, and R. Conradi. Software process modeling and evolution in EPOS. In *Software Engineering and Knowledge Engineering, 1992. Proceedings., Fourth International Conference on*, pages 574–581, June 1992.

[78] M. Jiménez, M. Piattini, and A. Vizcaíno. Challenges and improvements in distributed software development: a systematic review. *Advances in Software Engineering*, 2009:3:1–3:16, 2009.

[79] P. M. Johnson. An instrumented approach to improving software quality through formal technical review. In *Proceedings of the 16th international conference on Software engineering*, pages 113–122, Los Alamitos, CA, USA, 1994. IEEE Computer Society Press.

[80] M. Jorgensen and M. Shepperd. A Systematic Review of Software Development Cost Estimation Studies. *IEEE Transactions on Software Engineering*, 33(1):33–53, 2007.

[81] G. E. Kaiser, P. H. Feiler, and S. S. Popovich. Intelligent Assistance for Software Development and Maintenance. *Software, IEEE*, 5(3):40–49, 1988.

[82] G. E. Kaiser, S. M. Kaplan, and J. Micallef. Multiuser, Distributed Language-Based Environments. *Software, IEEE*, 4(6):58–67, 1987.

[83] M. Kantor and D. Redmiles. Creating an Infrastructure for Ubiquitous Awareness. In *Eight IFIP TC 13 Conference on Human-Computer Interaction (INTERACT 2001)*, pages 431–438, 2001.

[84] M. Kersten and G. C. Murphy. Using task context to improve programmer productivity. In *Proceedings of the 14th ACM SIGSOFT international symposium on Foundations of software engineering*, pages 1–11, New York, NY, USA, 2006. ACM.

[85] B. Kitchenham, O. P. Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman. Systematic Literature Reviews in Software Engineering: A Systematic Literature Review. *Information and Software Technology*, 51(1):7–15, 2009.

[86] B. Kitchenham and S. Charters. Guidelines for Performing Systematic Literature Reviews in Software Engineering (version 2.3). Technical report, Keele University and University of Durham, 2007.

[87] B. A. Kitchenham, T. Dybå, and M. Jorgensen. Evidence-Based Software Engineering. In *Proceedings of the 26th International Conference*

*on Software Engineering*, pages 273–281, Washington, DC, USA, 2004. IEEE Computer Society.

[88] E. Knauss, O. Brill, I. Kitzmann, and T. Flohr. SmartWiki: Support for high-quality requirements engineering in a collaborative setting. In *Wikis for Software Engineering, 2009. WIKIS4SE '09. ICSE Workshop on*, pages 25–35, May 2009.

[89] M. J. Knister and A. Prakash. DistEdit: a distributed toolkit for supporting multiple group editors. In *Proceedings of the 1990 ACM conference on Computer-supported cooperative work*, pages 343–355, New York, NY, USA, 1990. ACM.

[90] T. Kreifelts, E. Hinrichs, and G. Woetzel. Sharing to-do lists with a distributed task manager. In *Proceedings of the third conference on European Conference on Computer-Supported Cooperative Work*, pages 31–46, Norwell, MA, USA, 1993. Kluwer Academic Publishers.

[91] F. Lanubile and T. Mallardo. Tool support for distributed inspection. In *Computer Software and Applications Conference, 2002. COMPSAC 2002. Proceedings. 26th Annual International*, pages 1071–1076, 2002.

[92] D. B. Leblang and R. P. Chase Jr. Computer-Aided Software Engineering in a distributed workstation environment. *SIGSOFT Softw. Eng. Notes*, 9(3):104–112, 1984.

[93] A. Lopez, J. Nicolas, and A. Toval. Risks and Safeguards for the Requirements Engineering Process in Global Software Development. *Global Software Engineering, 2009. ICGSE 2009. Fourth IEEE International Conference on*, pages 394–399, 2009.

[94] S. Lukosch, D. Veiel, and J. M. Haake. Enabling Context-adaptive Collaboration for Knowledge-intense Processes. In *ICEIS (5)'09*, pages 34–41, 2009.

[95] B. Magnusson and U. Asklund. Fine grained version control of configurations in COOP/Orm. In I. Sommerville, editor, *SOFTWARE CONFIGURATION MANAGEMENT*, pages 31–48. Springer Berlin / Heidelberg, 1996.

[96] V. Mashayekhi, J. M. Drake, W. T. Tsai, and J. Riedl. Distributed, collaborative software inspection. *Software, IEEE*, 10(5):66–75, Sept. 1993.

[97] D. W. McDonald and M. S. Ackerman. Expertise recommender: a flexible recommendation system and architecture. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 231–240, New York, NY, USA, 2000. ACM.

[98] S. Minto and G. C. Murphy. Recommending Emergent Teams. In *Proceedings of the Fourth International Workshop on Mining Software Repositories*, pages 5–5, Washington, DC, USA, 2007. IEEE Computer Society.

[99] A. Mockus and J. Herbsleb. Expertise Browser: a quantitative approach to identifying expertise. In *Software Engineering, 2002. ICSE 2002. Proceedings of the 24rd International Conference on*, pages 503–512, May 2002.

[100] R. Morgan and F. Maurer. MasePlanner: A Card-Based Distributed Planning Tool for Agile Teams. In *Proceedings of the IEEE international conference on Global Software Engineering*, pages 132–138, Washington, DC, USA, 2006. IEEE Computer Society.

[101] P. Mukherjee, A. Kovacevic, M. Benz, and A. Schürr. Towards a Peer-to-Peer Based Global Software Development Environment. In *Software Engineering*, pages 204–216, 2008.

[102] H. Natsu, J. Favela, A. Moran, D. Decouchant, and A. Martinez-Enriquez. Distributed pair programming on the Web. In *Computer Science, 2003. ENC 2003. Proceedings of the Fourth Mexican International Conference on*, pages 81–88, Sept. 2003.

[103] J. Noll, S. Beecham, and I. Richardson. Global software development and collaboration: barriers and solutions. *ACM Inroads*, 1(3):66–78, 2010.

[104] P. Oberndorf. The Common Ada Programming Support Environment (APSE) Interface Set (CAIS). *Software Engineering, IEEE Transactions on*, 14(6):742–748, June 1988.

[105] I. Omoronyia, J. Ferguson, M. Roper, and M. Wood. A 3-Dimensional Relevance Model for Collaborative Software Engineering Spaces. In *Proceedings of the International Conference on Global Software Engineering*, pages 204–216, Washington, DC, USA, 2007. IEEE Computer Society.

[106] C. O'Reilly, D. Bustard, and P. Morrow. The war room command console: shared visualizations for inclusive team coordination. In *Proceedings of the 2005 ACM symposium on Software visualization*, pages 57–65, New York, NY, USA, 2005. ACM.

[107] C. O'Reilly, P. Morrow, and D. Bustard. Improving conflict detection in optimistic concurrency control models. In *Proceedings of the 2001 ICSE Workshops on SCM 2001, and SCM 2003 conference on Software configuration management*, pages 191–205, Berlin, Heidelberg, 2003. Springer-Verlag.

[108] J. Paer and B. Fitzgerald. Flexible And Distributed Software Processes: Old Petunias In New Bowls? *Communications of the ACM*, 2006.

[109] S. Palaniappan and L. Ling. A novel SVG application in UML system modelling. In *6th SVG Open Conference, Nuremberg, Deutschland*, 2008.

[110] D. Perry, A. Porter, M. Wade, L. Votta, and J. Perpich. Reducing inspection interval in large-scale software development. *Software Engineering, IEEE Transactions on*, 28(7):695–705, July 2002.

[111] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson. Systematic Mapping Studies in Software Engineering. *12th International Conference on Evaluation and Assessment in Software Engineering (EASE)*, 2008.

[112] J. Portillo-Rodriguez, A. Vizcaíno, M. Piattini, and S. Beecham. Tools used in Global Software Engineering: a Systematic Mapping Review. *Information and Software Technology*, (0):–, 2012.

[113] A. Ribak, M. Jacovi, and V. Soroka. "Ask before you search": peer support and community building with reachout. In *Proceedings of the 2002 ACM conference on Computer supported cooperative work*, pages 126–135, New York, NY, USA, 2002. ACM.

[114] R. Ripley, A. Sarma, and A. Van Der Hoek. Using visualizations to analyze workspace activity and discern software project evolution. Technical report, 2006.

[115] P. Rösch, J. Zettel, R. Kempkens, and R. Webby. The Spearmint software process modeling environment. In *Proceedings of the 20th International Conference on Software Engineering*, pages 54–57, 1998.

[116] J. Rubart, J. M. Haake, D. A. Tietze, and W. Wang. Organizing shared enterprise workspaces using component-based cooperative hypermedia. In *Proceedings of the 12th ACM conference on Hypertext and Hypermedia*, pages 73–82, New York, NY, USA, 2001. ACM.

[117] A. Sarma, L. Maccherone, P. Wagstrom, and J. Herbsleb. Tesseract: Interactive visual exploration of socio-technical relationships in software development. In *Proceedings of the 31st International Conference on Software Engineering*, pages 23–33, Washington, DC, USA, 2009. IEEE Computer Society.

[118] A. Sarma, Z. Noroozi, and A. Van Der Hoek. Palantir: raising awareness among configuration management workspaces. *Software Engineering, 2007. ICSE 2007. 29th International Conference on*, pages 444–454, 2003.

[119] A. Sarma, D. Redmiles, and A. v. d. Hoek. Categorizing the Spectrum of Coordination Technology. *Computer*, 43:61–67, 2010.

[120] J. Satzinger and L. Olfman. A research program to assess user perceptions of group work support. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 99–106, New York, NY, USA, 1992. ACM.

[121] D. Schmidt and F. Buschmann. Patterns, frameworks, and middleware: their synergistic relationships. In *Software Engineering, 2003. Proceedings. 25th International Conference on*, pages 694–704, May 2003.

[122] H. Schnädelbach, A. Penn, P. Steadman, S. Benford, B. Koleva, and T. Rodden. Moving office: inhabiting a dynamic building. In *Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, pages 313–322, New York, NY, USA, 2006. ACM.

[123] T. Schummer and S. Lukosch. Supporting the social practices of distributed pair programming. *14th International Workshop of Groupware, CRIWG 2008, September 14, 2008 - September 18, 2008*, 5411 LNCS:83–98, 2008.

[124] T. Schümmer and J. Schümmer. Support for distributed teams in extreme programming. In *Extreme programming examined*, pages 355–377. Addison-Wesley Longman Publishing Co., Inc., 2001.

[125] B. Sengupta. Enabling Collaboration in Distributed Requirements Management. *Software, IEEE*, 23(5):52–61, 2006.

[126] B. Sengupta, S. Chandra, and V. Sinha. A research agenda for distributed software development. *Proceedings of the International Conference on Software Engineering*, page 740, 2006.

[127] F. Servant, J. A. Jones, and A. van der Hoek. CASI: preventing indirect conflicts through a live visualization. In *Proceedings of the 2010 ICSE Workshop on Cooperative and Human Aspects of Software Engineering*, pages 39–46, New York, NY, USA, 2010. ACM.

[128] T. Shigenobu, T. Yoshino, and J. Munemori. Evaluation and application of creativity collaboration support system gungen dx ii for consensus-building among users. *International Journal of Information Technology and Decision Making*, 6(3):475–490, 2007.

[129] D. B. Simmons and C.-S. Wu. Plan Tracking Knowledge Base. In *24th International Computer Software and Applications Conference*, pages 299–304, Washington, DC, USA, 2000. IEEE Computer Society.

[130] M. Stein, J. Riedl, S. o. r. J. Harner, and V. Mashayekhi. A case study of distributed, asynchronous software inspection. In *Proceedings of the 19th international conference on Software engineering*, pages 107–117, New York, NY, USA, 1997. ACM.

[131] D. Stotts, J. Smith, and K. Gyllstrom. Support for Distributed Pair Programming in the Transparent Video Facetop. In C. Zannier, H. Erdogmus, and L. Lindstrom, editors, *Extreme Programming and Agile Methods - XP/Agile Universe 2004*, pages 150–192. Springer Berlin / Heidelberg, 2004.

[132] S. M. Sutton Jr and L. J. Osterweil. The design of a next-generation process language. *SIGSOFT Softw. Eng. Notes*, 22(6):142–158, 1997.

[133] R. N. Taylor, F. C. Belz, L. A. Clarke, L. Osterweil, R. W. Selby, J. C. Wileden, A. L. Wolf, and M. Young. Foundations for the Arcadia environment architecture. *SIGSOFT Softw. Eng. Notes*, 13(5):1–13, 1988.

[134] P. Tell and M. A. Babar. Requirements for an infrastructure to support Activity-Based Computing in Global Software Development. In *International Conference on Global Software Engineering Workshop (ICGSEW '11)*, Nov. 2011.

[135] M. Turner, B. Kitchenham, D. Budgen, and O. Brereton. Lessons learnt undertaking a large-scale systematic literature review. *Proceedings of EASE'08, British Computer Society*, 2008.

[136] M. van Genuchten, C. van Dijk, H. Scholten, and D. Vogel. Using Group Support Systems for Software Inspections. *Software, IEEE*, 18(3):60–65, 2001.

[137] J. Whitehead. Collaboration in software engineering: A roadmap. *Future of Software Engineering (FOSE '07)*, 2007.

[138] S. Whittaker, D. Frohlich, and O. Daly-Jones. Informal workplace communication: what is it like and how might we support it? *CHI '94: Proceedings of the SIGCHI conference on Human factors in computing systems*, 1994.

[139] K. Wong, W. Blanchet, Y. Liu, C. Schofield, E. Stroulia, and Z. Xing. JRefleX: towards supporting small student software teams. In *Proceedings of the 2003 OOPSLA workshop on eclipse technology eXchange*, pages 50–54, New York, NY, USA, 2003. ACM.

[140] D. Yang, D. Wu, S. Koolmanojwong, A. Winsor Brown, and B. Boehm. WikiWinWin: A Wiki Based System for Collaborative Requirements Negotiation. In *Hawaii International Conference on System Sciences, Proceedings of the 41st Annual*, page 24, Jan. 2008.

[141] H. Zhang, M. A. Babar, and P. Tell. Identifying relevant studies in software engineering. *Information and Software Technology*, 53:625–637, June 2011.

## 8. Primary Studies

[PS001] B Al-Ani. Questions Regarding Knowledge Engineering and Management. In *Global Software Engineering (ICGSE), 2010 5th IEEE International Conference on*, pages 324–329, August 2010.

[PS002] B Al-Ani and D Redmiles. Supporting Trust in Distributed Teams through Continuous Coordination. *Software, IEEE*, PP(99):1, 2009.

[PS003] Muhammad Ali Babar. A framework for groupware-supported software architecture evaluation process in global software development. *Journal of Software Maintenance and Evolution: Research and Practice*, pages 207–229, 2010.

[PS004] Muhammad Ali Babar, Barbara Kitchenham, and Ross Jeffery. Distributed versus face-to-face meetings for architecture evalution: a controlled experiment. In *Proceedings of the 2006 ACM/IEEE international symposium on Empirical software engineering*, pages 252–261, New York, NY, USA, 2006. ACM.

[PS005] L Aversano, A Cimitile, P Gallucci, and M.L. Villani. Flow-Manager: a workflow management system based on Petri nets. In *Computer Software and Applications Conference, 2002. COMPSAC 2002. Proceedings. 26th Annual International*, pages 1054–1059, 2002.

[PS006] Lerina Aversano, Andrea De Lucia, Matteo Gaeta, Pierluigi Ritrovato, Silvio Stefanucci, and Maria Luisa Villani. Managing coordination and cooperation in distributed software processes: the GENESIS environment. *Software Process: Improvement and Practice*, 9(4):239–263, 2004.

[PS007] Aditya and Johri. Sociomaterial bricolage: The creation of location-spanning work practices by global software developers. *Information and Software Technology*, 53(9):955–968, 2011.

[PS008] Oscar Ardaiz-Villanueva. On-Demand Collaborative Work Environments Based on Grid Technologies for Virtual Projects. In Wei-ming Shen, Kuo-Ming Chao, Zongkai Lin, Jean-Paul Barthès, and Anne James, editors, *Computer Supported Cooperative Work in Design II*, pages 131–140. Springer Berlin / Heidelberg, 2006.

[PS009] G.N. Aranda, A. Vizcaí andno, R.R. Palacio, and A.L. Morá andn. What Information Would You Like to Know about Your Co-worker? A Case Study. In *Global Software Engineering (ICGSE), 2010 5th IEEE International Conference on*, pages 135–144, August 2010.

[PS010] Darren C Atkinson, Daniel C Weeks, and John Noll. Tool support for iterative software process modeling. *Information and Software Technology*, 49(5):493–514, 2007.

[PS011] R.P. Biuk-Aghai. Visualization of Web-based workspace structures. In *Web Information Systems Engineering, 2000. Proceedings of the First International Conference on*, pages 302–309 vol.1, 2000.

[PS012] R.P. Biuk-Aghai. Visualizing structural and behavioural aspects of virtual collaboration. In *Enabling Technologies: Infrastructure for Collaborative Enterprises, 2001. WET ICE 2001. Proceedings. Tenth IEEE International Workshops on*, pages 279–284, 2001.

[PS013] J.-P.A. Barthes. OMAS - A flexible multi-agent environment for CSCWD. In *Computer Supported Cooperative Work in Design, 2009. CSCWD 2009. 13th International Conference on*, pages 258–263, April 2009.

[PS014] A.L. Bartell and K.A. Brown. Secrets to managing a large documentation project virtually – process, technology, and group ethos: Lessons learned. In *Professional Communication Conference, 2009. IPCC 2009. IEEE International*, pages 1–6, July 2009.

[PS015] C Behrens, D Bassu, and H-s Shim. Collaborative Panel Administrator: A framework and implementation for expert panel lifecycle management. In *Collaborative Computing: Networking, Applications and Worksharing, 2007. CollaborateCom 2007. International Conference on*, pages 408–417, November 2007.

[PS016] Bernd Bruegge, Andrea De Lucia, Fausto Fasano, and Genoveffa Tortora. Supporting Distributed Software Development with fine-grained Artefact Management. In *Global Software Engineering, 2006. ICGSE '06. International Conference on*, pages 213–222, October 2006.

[PS017] Bernd Bruegge, Allen H Dutoit, and Timo Wolf. Sysiphus: Enabling informal collaboration in global software development. In *Global Software Engineering, 2006. ICGSE '06. International Conference on*, pages 139–148, October 2006.

[PS018] S Berczuk. Back to Basics: The Role of Agile Principles in Success with an Distributed Scrum Team. In *AGILE 2007*, pages 382–388, August 2007.

[PS019] M.K. Brown, B Huettner, and C James-Tanny. Choosing the Right Tools for Your Virtual Team: Evaluating Wikis, Blogs, and Other Collaborative Tools. In *Professional Communication Conference, 2007. IPCC 2007. IEEE International*, pages 1–4, October 2007.

[PS020] E. Bugli Innocenti. A distributed and transport environment for software engineering built on networking facilities: the PCTE distribution service. In *Distributed Computing Systems in the 1990s, 1988. Proceedings., Workshop on the Future Trends of*, pages 33–37, September 1988.

[PS021] Muhammad Ali Babar, Barbara Kitchenham, Liming Zhu, Ian Gorton, and Ross Jeffery. An empirical study of groupware support for distributed software architecture evaluation process. *Journal of Systems and Software*, 79(7):912–925, 2006.

[PS022] A. Bacchelli, M. Lanza, and V. Humpa. RTFM (Read the Factual Mails) - Augmenting Program Comprehension with Remail. In *Software Maintenance and Reengineering (CSMR), 2011 15th European Conference on*, pages 15–24, March 2011.

[PS023] S Bowen and F Maurer. Designing a distributed software development support system using a peer-to-peer architecture. In *Computer Software and Applications Conference, 2002. COMPSAC 2002. Proceedings. 26th Annual International*, pages 1087–1092, 2002.

[PS024] F Bergenti, M Mari, and M Garijo. Collaborator - enabling enterprise collaboration through agents. In *Enabling Technologies: Infrastructure for Collaborative Enterprises, 2004. WET ICE 2004. 13th IEEE International Workshops on*, pages 41–46, June 2004.

[PS025] G.S. Blair, J. Malik, J.R. Nicol, and J. Walpole. A synthesis of object-oriented and functional ideas in the design of a distributed software engineering environment. *Software Engineering Journal*, 5(3):193–204, May 1990.

[PS026] M Baentsch, G Molter, and P Sturm. Booster: A WWW-based prototype of the global software highway. In *Services in Distributed and Networked Environments, 1995., Second International Workshop on*, pages 156–165, June 1995.

[PS027] Michael Baentsch, Georg Molter, and Peter Sturm. WebMake: Integrating distributed software development in a structure-enhanced Web. *Computer Networks and ISDN Systems*, 27(6):789–800, 1995.

[PS028] A Boden, B Nett, and V Wulf. Coordination Practices in Distributed Software Development of Small Enterprises. In *Global Software Engineering, 2007. ICGSE 2007. Second IEEE International Conference on*, pages 235–246, August 2007.

[PS029] P Bjoern and J Simonsen. Joint enterprise and the role of the intermediator: challenges managing groupware in global virtual teams. In *Database and Expert Systems Applications, 2004. Proceedings. 15th International Workshop on*, pages 609–615, August 2005.

[PS030] I.Z. Ben-Shaul and G. E. Kaiser. A configuration process for a distributed software development environment. In *Configurable Distributed Systems, 1994., Proceedings of 2nd International Workshop on*, pages 123–134, March 1994.

[PS031] M.A. Babar, D Winkler, and S Biffi. Evaluating the Usefulness and Ease of Use of a Groupware Tool for the Software Architecture Evaluation Process. In *Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on*, pages 430–439, September 2007.

[PS032] C Chang and L Cai. Agent based requirements evolution over the Internet. In *Applications and the Internet Workshops, 2001. Proceedings. 2001 Symposium on*, pages 83–88, 2001.

[PS033] Keith C.C. Chan and Lawrence M.L. Chung. Integrating Process and Project Management for Multi-Site Software Development. *Annals of Software Engineering*, 14:115–143, 2002.

[PS034] Raymond O. Colbert, Diane S. Compton, Randy L. Hackbarth, James D Herbsleb, Laurie A. Hoadley, and Graham J. Wills. Advanced services: Changing how we communicate. *Bell Labs Technical Journal*, 6(1):211–228, 2001.

[PS035] Rafael Capilla, Juan C. Dueñas, and Francisco Nava. Viability for codifying and documenting architectural design decisions with tool support. *Journal of Software Maintenance and Evolution: Research and Practice*, 22(2):81–119, 2010.

[PS036] G. Cugola and C. Ghezzi. Design and implementation of PROSYT: a distributed process support system. In *Enabling Technologies: Infrastructure for Collaborative Enterprises, 1999. (WET ICE '99) Proceedings. IEEE 8th International Workshops on*, pages 32–39, 1999.

[PS037] Rubby Casallas, Oscar González, and Nicolás López. Dealing with Scalability in an Event-Based Infrastructure to Support Global Software Development. In Dirk Draheim and Gerald Weber, editors, *Trends in Enterprise Application Architecture*, pages 100–111. Springer Berlin / Heidelberg, 2007.

[PS038] Marcelo Cataldo and James D Herbsleb. End-to-end features as meta-entities for enabling coordination in geographically distributed software development. In *Proceedings of the 2009 ICSE Workshop on Software Development Governance*, pages 21–26, Washington, DC, USA, 2009. IEEE Computer Society.

[PS039] Fabio Calefato and Filippo Lanubile. Using frameworks to develop a distributed conferencing system: an experience report. *Software: Practice and Experience*, 39(15):1293–1311, 2009.

[PS040] Fabio Calefato, Filippo Lanubile, and Mario Scalas. Porting a distributed meeting system to the Eclipse communication framework. In *Proceedings of the 2007 OOPSLA workshop on eclipse technology eXchange*, pages 46–49, New York, NY, USA, 2007. ACM.

[PS041] Minder Chen, Yiching Liou, Ching-Wen Wang, Yi-Wen Fan, and Yan-Ping Jeffery Chi. TeamSpirit: Design, implementation, and evaluation of a Web-based group decision support system. *Decision Support Systems*, 43(4):1186–1202, 2007.

[PS042] G.S. Cowan and R.G. Reynolds. Evolving distributed software engineering environments. In *Reliable Distributed Systems, 1998. Proceedings. Seventeenth IEEE Symposium on*, pages 151–157, October 1998.

[PS043] D Christian and S Rotenstreich. An Evaluation Framework For Distributed Collaboration Tools. In *Information Technology: New Generations (ITNG), 2010 Seventh International Conference on*, pages 512–517, April 2010.

[PS044] Renata Mendes de Araujo and Marcos R. S. Borges. The role of collaborative support to promote participation and commitment in software development teams. *Software Process: Improvement and Practice*, 12(3):229–246, 2007.

[PS045] P. Dewan, P. Agrawal, G. Shroff, and R. Hegde. Experiments in distributed side-by-side software development. In *Collaborative Computing: Networking, Applications and Worksharing, 2009. CollaborateCom 2009. 5th International Conference on*, pages 1–10, November 2009.

[PS046] Jacek Dajda and Grzegorz Dobrowolski. How to build support for distributed pair programming. In *Proceedings of the 8th international conference on Agile processes in software engineering and extreme programming*, pages 70–73, Berlin, Heidelberg, 2007. Springer-Verlag.

[PS047] S Dustdar and H Gall. Process awareness for distributed software development in virtual teams. In *Euromicro Conference, 2002. Proceedings. 28th*, pages 244–250, 2002.

[PS048] Stephen E. Dossick and Gail E. Kaiser. CHIME: a metadata-based distributed software development environment. *SIGSOFT Softw. Eng. Notes*, 24:464–475, October 1999.

[PS049] Andrea De Lucia, Fausto Fasano, Rocco Oliveto, and Genoveffa Tortora. Fine-grained management of software artefacts: the ADAMS system. *Software: Practice and Experience*, 40(11):1007–1034, 2010.

[PS050] A De Lucia, F Fasano, G Scanniello, and G Tortora. Enhancing collaborative synchronous UML modelling with fine-grained versioning of software artefacts. *Journal of Visual Languages & Computing*, 18(5):492–503, 2007.

[PS051] Daniela Damian, Filippo Lanubile, and Teresa Mallardo. An Empirical Study of the Impact of Asynchronous Discussions on Remote Synchronous Requirements Meetings. In Luciano Baresi and Reiko Heckel, editors, *Fundamental Approaches to Software Engineering*, pages 155–169. Springer Berlin / Heidelberg, 2006.

[PS052] Fabio Q. B. da Silva, Catarina Costa, A. Cesar C. Franca, and Rafael Prikladinicki. Challenges and Solutions in Distributed Software Development Project Management: A Systematic Literature Review. In *Global Software Engineering (ICGSE), 2010 5th IEEE International Conference on*, pages 87–96, August 2010.

[PS053] Cleidson De Souza, Tobias Hildenbrand, and David Redmiles. Toward Visualization and Analysis of Traceability Relationships in Distributed and Offshore Software Development Projects. In Bertrand Meyer and Mathai Joseph, editors, *Software Engineering Approaches for Offshore and Outsourced Development*, pages 182–199. Springer Berlin / Heidelberg, 2007.

[PS054] Fabio Q B da Silva, Rafael Prikladinicki, A Csar C Frana, Cleviton V. F. Monteiro, Catarina Costa, and Rodrigo Rocha. An evidence-based model of distributed software development project management: results from a systematic mapping study. *Journal of Software Maintenance and Evolution: Research and Practice*, 2011.

[PS055] Schahram Dustdar. Collaborative Knowledge Flow - Improving Process-Awareness and Traceability of Work Activities. In Dimitris Karagiannis and Ulrich Reimer, editors, *Practical Aspects of Knowledge Management*, pages 389–397. Springer Berlin / Heidelberg, 2002.

[PS056] K Dullemond, B van Gameren, and R van Solingen. How Technological Support Can Enable Advantages of Agile Software Development in a GSE Setting. In *Global Software Engineering, 2009. ICGSE 2009. Fourth IEEE International Conference on*, pages 143–152, July 2009.

[PS057] Daniela E Damian and Didar Zowghi. RE challenges in multi-site software development organisations. *Requirements Engineering*, 8:149–160, 2003.

[PS058] Katherine M. Everitt, Scott R Klemmer, Robert Lee, and James A. Landay. Two worlds apart: bridging the gap between physical and virtual media for distributed design collaboration. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 553–560, New York, NY, USA, 2003. ACM.

[PS059] Jon Froehlich and Paul Dourish. Unifying Artifacts and Activities in a Visual Tool for Distributed Software Development Teams. In *Proceedings of the 26th International Conference on Software Engineering*, pages 387–396, Washington, DC, USA, 2004. IEEE Computer Society.

[PS060] Sebastiao B Fonseca, Cleidson R B de Souza, and David F Redmiles. Exploring the Relationship between Dependencies and Coordination to Support Global Software Development Projects. In *Global Software Engineering, 2006. ICGSE '06. International Conference on*, page 243, October 2006.

[PS061] A Ferscha. Workspace awareness in mobile virtual teams. In *Enabling Technologies: Infrastructure for Collaborative Enterprises, 2000. (WET ICE 2000). Proeedings. IEEE 9th International Workshops on*, pages 272–277, 2000.

[PS062] A. Fernandez. Scaki -the scaffolding Wiki. In *Database and Expert Systems Applications, 2004. Proceedings. 15th International Workshop on*, pages 271–275, August 2004.

[PS063] Alejandro Fernandez, Badie Garzaldeen, Ines Grützner, and Jürgen Münch. Guided support for collaborative modeling, enactment and simulation of software development processes. *Software Process: Improvement and Practice*, 9(2):95–106, 2004.

[PS064] A Feng, T Kikuno, and K Torii. Updating semantic information to support coordination in distributed software development environments. In *System Sciences, 1993, Proceeding of the Twenty-Sixth Hawaii International Conference on*, pages 13–22 vol.4, January 1993.

[PS065] A Ferscha and C Scheiner. Collective choice in virtual teams. In *Enabling Technologies: Infrastructure for Collaborative En-*

*terprises, 1999. (WET ICE '99) Proceedings. IEEE 8th International Workshops on*, pages 96–101, 1999.

[PS066] R Fernandez, J Soriano, X Larrucea, A.L. Martinez, and J.M. Gonzalez-Barahona. Towards the improvement of the software quality: An Enterprise 2.0 architecture for distributed software developments. In *Distributed Framework and Applications, 2008. DFmA 2008. First International Conference on*, pages 52–59, October 2008.

[PS067] J.C. Grundy, M.D. Apperley, J.G. Hosking, and W.B. Mugridge. A decentralized architecture for software process modeling and enactment. *Internet Computing, IEEE*, 2(5):53–62, September 1998.

[PS068] Amar Gupta, Igor Crk, and Rajdeep Bondade. Leveraging temporal and spatial separations with the 24-hour knowledge factory paradigm. *Information Systems Frontiers*, 13:397–405, 2011.

[PS069] Daniel M. German. The GNOME project: a case study of open source, global software development. *Software Process: Improvement and Practice*, 8(4):201–215, 2003.

[PS070] J Grundy, J Hosking, and R. Mugridge. Coordinating distributed software development projects with integrated process modelling and enactment environments. In *Enabling Technologies: Infrastructure for Collaborative Enterprises, 1998. (WET ICE '98) Proceedings., Seventh IEEE International Workshops on*, pages 39–44, June 1998.

[PS071] I Gorton, I Hawryszkiewycz, and K Ragoonaden. Collaborative tools and processes to support software engineering shift work. *BT Technology Journal*, 15:189–198, 1997.

[PS072] I Gorton, I Hawryszkiewycz, K Ragoonaden, C Chung, Shijian Lu, and G Randhawa. Groupware support tools for collaborative software engineering. In *System Sciences, 1997, Proceedings of the Thirtieth Hawaii International Conference on*, pages 157–166 vol.2, January 1997.

[PS073] M Geisser, T Hildenbrand, F Rothlauf, and C Atkinson. An Evaluation Method for Requirements Engineering Approaches in Distributed Software Development Projects. In *Software Engineering Advances, 2007. ICSEA 2007. International Conference on*, page 39, August 2007.

[PS074] Jerry Z. Gao, Fukao Itaru, and Y Toyoshima. Managing Problems for Global Software Production – Experience and Lessons. *Information Technology and Management*, 3:85–112, 2002.

[PS075] Eric Gilbert and Karrie Karahalios. CodeSaw: A Social Visualization of Distributed Software Development. In Cécilia Baranauskas, Philippe Palanque, Julio Abascal, and Simone Barbosa, editors, *Human-Computer Interaction – INTERACT 2007*, pages 303–316. Springer Berlin / Heidelberg, 2007.

[PS076] Claude Godart, Pascal Molli, Gerald Oster, Olivier Perrin, Hala Skaf-Molli, Pradeep Ray, and Fethi Rabhi. The ToxicFarm Integrated Cooperation Framework for Virtual Teams. *Distributed and Parallel Databases*, 15:67–88, 2004.

[PS077] M Gaeta and P Ritrovato. Generalised environment for process management in cooperative software engineering. In *Computer Software and Applications Conference, 2002. COMPSAC 2002. Proceedings. 26th Annual International*, pages 1049–1053, 2002.

[PS078] Werner Geyer, Heather Richter, Ludwin Fuchs, Tom Frauenhofer, Shahrokh Daijavad, and Steven Poltrock. A team collaboration space supporting capture and access of virtual meetings. In *Proceedings of the 2001 International ACM SIGGROUP Conference on Supporting Group Work*, pages 188–196, New York, NY, USA, 2001. ACM.

[PS079] Z. Haag, R. Foley, and J. Newman. Software process improvement in geographically distributed software engineering: an initial evaluation. In *EUROMICRO 97. 'New Frontiers of Information Technology'., Proceedings of the 23rd EUROMICRO Conference*, pages 134–141, September 1997.

[PS080] I T Hawryszkiewycz and I Gorton. Distributing the software process. In *Software Engineering Conference, 1996. Proceedings of 1996 Australian*, pages 176–182, July 1996.

[PS081] Shihong Huang, Vaishali Gohel, and Sam Hsu. Towards interoperability of UML tools for exchanging high-fidelity diagrams. In *Proceedings of the 25th annual ACM international conference on Design of communication*, pages 134–141, New York, NY, USA, 2007. ACM.

[PS082] S.S. Hussain and J.M. Haake. Process support for context-based adaptations in a collaboration environment. In *Collaborative Technologies and Systems (CTS), 2010 International Symposium on*, pages 265–274, May 2010.

[PS083] H. Harroud, A. Karmouch, and R. Impey. SIP-based agent architecture for collaborative applications. In *Communications, Computers and signal Processing, 2003. PACRIM. 2003 IEEE Pacific Rim Conference on*, pages 518–521 vol.2, August 2003.

[PS084] A. Hac and Dongchen Lu. Architecture and implementation of a multimedia conference system. In *Emerging Technologies and Applications in Communications, 1996. Proceedings., First Annual Conference on*, pages 50–53, May 1996.

[PS085] James D Herbsleb, Audris Mockus, Thomas A. Finholt, and Rebecca E. Grinter. An empirical study of global software development: distance and speed. In *Proceedings of the 23rd International Conference on Software Engineering*, pages 81–90, Washington, DC, USA, 2001. IEEE Computer Society.

[PS086] Marleen Huysman, Charles Steinfield, Chyng-Yang Jang, Kenneth David, Mirjam Huis in T Veld, Jan Poot, and Ingrid Mulder. Virtual Teams and the Appropriation of Communication Technology: Exploring the Concept of Media Stickiness. *Computer Supported Cooperative Work (CSCW)*, 12:411–436, 2003.

[PS087] S.S. Hussain, D. Veiel, J.M. Haake, and S. Lukosch. Facilitating understanding of team-based adaptation policies. In *Col-*

*laborative Computing: Networking, Applications and Worksharing (CollaborateCom), 2010 6th International Conference on*, pages 1–8, October 2010.

[PS088] David Hutchison and Jonathan Walpole. Current research. Eclipse - a distributed software development environment. *Software Engineering Journal*, 1(2):88–92, March 1986.

[PS089] A. IIac and Dongchen Lu. Design and implementation of the user interface in a multimedia conference system. In *Communication Technology Proceedings, 1996. ICCT'96., 1996 International Conference on*, pages 183–186 vol.1, May 1996.

[PS090] Chyng-Yang Jang, Charles Steinfield, and Ben Pfaff. Virtual team awareness and groupware support: an evaluation of the TeamSCOPE system. *International Journal of Human-Computer Studies*, 56(1):109–126, 2002.

[PS091] G. E. Kaiser and S.E. Dossick. Workgroup middleware for distributed projects. In *Enabling Technologies: Infrastructure for Collaborative Enterprises, 1998. (WET ICE '98) Proceedings., Seventh IEEE International Workshops on*, pages 63–68, June 1998.

[PS092] Jukka Kääriäinen, Juho Eskeli, Susanna Teppola, Antti Välimäki, Pekka Tuuttila, and Markus Piippola. Extending Global Tool Integration Environment towards Lifecycle Management. In Robert Meersman, Pilar Herrero, and Tharam Dillon, editors, *On the Move to Meaningful Internet Systems: OTM 2009 Workshops*, pages 238–247. Springer Berlin / Heidelberg, 2009.

[PS093] S. Kolvenbach, W. Grather, and K. Klockner. With Collaborative Task Management towards Lightweight Distributed Engineering Processes. In *Euromicro Conference, 2002. Proceedings. 28th*, pages 456–463, September 2006.

[PS094] M Kuechler and A Kunz. HoloPort - A Device for Simultaneous Video and Data Conferencing Featuring Gaze Awareness. In *Virtual Reality Conference, 2006*, pages 81–88, March 2006.

[PS095] Abayomi King and Kelly Lyons. Automatic status updates in distributed software development. In *Proceeding of the 2nd international workshop on Web 2.0 for software engineering*, pages 19–24, New York, NY, USA, 2011. ACM.

[PS096] T.V. Krishnamurthy and S Subramani. Ailments of Distributed Document Reviews and Remedies of DOCTOR (DOCument Tree ORganizer Tool) with Distributed Reviews Support. In *Global Software Engineering, 2008. ICGSE 2008. IEEE International Conference on*, pages 210–214, August 2008.

[PS097] Marcos Kalinowski and Guilherme Horta Travassos. ISPIS: A Framework Supporting Software Inspection Processes. In *Proceedings of the 19th IEEE international conference on Automated software engineering*, pages 392–393, Washington, DC, USA, 2004. IEEE Computer Society.

[PS098] Karl E. Kurbel and Brandon Ulrich. Tooling Support for ISD. In *The Making of Information Systems*, pages 533–566. Springer Berlin Heidelberg, 2008.

[PS099] Kolja Lehmann, Lawrence Cabac, Daniel Moldt, and Heiko Rolke. Towards a Distributed Tool Platform Based on Mobile Agents. In Torsten Eymann, Franziska Klügl, Winfried Lamersdorf, Matthias Klusch, and Michael Huhns, editors, *Multiagent System Technologies*, pages 179–190. Springer Berlin / Heidelberg, 2005.

[PS100] Michael Lang and Jim Duggan. A Tool to Support Collaborative Software Requirements Management. *Requirements Engineering*, 6:161–172, 2001.

[PS101] F Lanubile, C Ebert, R Prikladnicki, and A Vizcaino. Collaboration Tools for Global Software Engineering. *Software, IEEE*, 27(2):52–55, March 2010.

[PS102] A Lamersdorf and J Münch. TAMRI: A Tool for Supporting Task Distribution in Global Software Development Projects.

In *Global Software Engineering, 2009. ICGSE 2009. Fourth IEEE International Conference on*, pages 322–327, July 2009.

[PS103] Filippo Lanubile, Teresa Mallardo, and Fabio Calefato. Tool support for geographically dispersed inspection teams. *Software Process: Improvement and Practice*, 8(4):217–231, 2003.

[PS104] N.G. Lester and F.G. Wilkie. Evaluating UML tool support for effective coordination and communication across geographically disparate sites. In *Software Technology and Engineering Practice, 2004. STEP 2004. The 12th International Workshop on*, pages 8 pp.–64, September 2004.

[PS105] E Ly. Distributed Java applets for project management on the Web. *Internet Computing, IEEE*, 1(3):21–26, May 1997.

[PS106] R Martignoni. Global Sourcing of Software Development - A Review of Tools and Services. In *Global Software Engineering, 2009. ICGSE 2009. Fourth IEEE International Conference on*, pages 303–308, July 2009.

[PS107] Brent McKnight and Nick Bontis. E-improvisation: collaborative groupware technology expands the reach and effectiveness of organizational improvisation. *Knowledge and Process Management*, 9(4):219–227, 2002.

[PS108] Marco Mangan, Marcos Borges, and Cláudia Werner. Increasing Awareness in Distributed Software Development Workspaces. In Gert-Jan De Vreede, Luis Guerrero, and Gabriela Marín Raventós, editors, *Groupware: Design, Implementation, and Use*, pages 84–91. Springer Berlin / Heidelberg, 2004.

[PS109] F Maurer and B Dellen. Internet based software process support. In *Enabling Technologies: Infrastructure for Collaborative Enterprises, 1998. (WET ICE '98) Proceedings., Seventh IEEE International Workshops on*, pages 27–32, June 1998.

[PS110] F Maurer, B Dellen, F Bendeck, S Goldmann, H Holz, B Kotting, and M Schaaf. Merging project planning and Web enabled dynamic workflow technologies. *Internet Computing, IEEE*, 4(3):65–74, May 2000.

[PS111] Bertrand Meyer. Design and code reviews in the age of the internet. *Commun. ACM*, 51:66–71, September 2008.

[PS112] Frank Maurer and Harald Holz. Integrating Process Support and Knowledge Management for Virtual Software Development Teams. *Annals of Software Engineering*, 14:145–168, 2002.

[PS113] Walid Maalej and Hans-Jörg Happel. A Lightweight Approach for Knowledge Sharing in Distributed Software Teams. In Takahira Yamaguchi, editor, *Practical Aspects of Knowledge Management*, pages 14–25. Springer Berlin / Heidelberg, 2008.

[PS114] A. Miller. A Hundred Days of Continuous Integration. In *Agile, 2008. AGILE '08. Conference*, pages 289–293, August 2008.

[PS115] David K M Mak and Philippe B Kruchten. Task Coordination in an Agile Distributed Software Development Environment. In *Electrical and Computer Engineering, 2006. CCECE '06. Canadian Conference on*, pages 606–611, May 2006.

[PS116] P Mukherjee, C Leng, W.W. Terpstra, and A Schurr. Peer-to-Peer Based Version Control. In *Parallel and Distributed Systems, 2008. ICPADS '08. 14th IEEE International Conference on*, pages 829–834, December 2008.

[PS117] H Mimura and H Maezawa. SEWB: distributed software development environment on workstations. In *Computer Software and Applications Conference, 1989. COMPSAC 89., Proceedings of the 13th Annual International*, pages 428–432, September 1989.

[PS118] K Mueller and P Pfeifer. Codebase: A commercially developed code management system and code transfer facility. *Computer Physics Communications*, 57(1-3):239–243, 1989.

[PS119] Michael Meisinger, Andreas Rausch, and Marc Sihling. 4everedit — Team-based Process Documentation Management. *Software Process: Improvement and Practice*, 11(6):627–642, 2006.

[PS120] C Maidantchik, G B Xexéo, and A R C Rocha. A WWW software development environment to support cooperative and spread working groups. *Computer Physics Communications*, 110(1-3):181–191, 1998.

[PS121] D Nutter and C Boldyreff. Evaluation of an awareness distribution mechanism: a simulation approach. In *Enabling Technologies: Infrastructure for Collaborative Enterprises, 2004. WET ICE 2004. 13th IEEE International Workshops on*, pages 235–240, June 2005.

[PS122] Christian Nentwich, Wolfgang Emmerich, Anthony Finkelstein, and Andrea Zisman. BOX: Browsing objects in XML. *Software: Practice and Experience*, 30(15):1661–1676, 2000.

[PS123] T Niinimäki and C Lassenius. Experiences of Instant Messaging in Global Software Development Projects: A Multiple Case Study. In *Global Software Engineering, 2008. ICGSE 2008. IEEE International Conference on*, pages 55–64, August 2008.

[PS124] John Noll and Walt Scacchi. Supporting distributed configuration management in virtual enterprises. In Reidar Conradi, editor, *Software Configuration Management*, pages 142–160. Springer Berlin / Heidelberg, 1997.

[PS125] Inah Omoronyia, John Ferguson, Marc Roper, and Murray Wood. A review of awareness in distributed collaborative software engineering. *Software: Practice and Experience*, 40(12):1107–1133, 2010.

[PS126] A C Pithon, M R Brochado, F S Sandonato, and B M Teixeira. - Collaborative Analysis among virtual teams: an experience. In D T Pham, E E Eldukhri, and A J Soroka, editors, *Intelligent Production Machines and Systems*, pages 113–119. Elsevier Science Ltd, Oxford, 2006.

[PS127] R. Phalnikar, V.S. Deshpande, and S.D. Joshi. Applying Agile Principles for Distributed Software Development. In *Advanced Computer Control, 2009. ICACC '09. International Conference on*, pages 535–539, January 2009.

[PS128] Maria Paasivaara, Sandra Durasiewicz, and Casper Lassenius. Using scrum in a globally distributed project: a case study. *Software Process: Improvement and Practice*, 13(6):527–544, 2008.

[PS129] Kyoung Park, Jason Leigh, Andrew Johnson, and Yongjoo Cho. Design of Awareness Interface for Distributed Teams in Display-Rich Advanced Collaboration Environments. In Gert-Jan De Vreede, Luis Guerrero, and Gabriela Marín Raventós, editors, *Groupware: Design, Implementation, and Use*, pages 105–120. Springer Berlin / Heidelberg, 2004.

[PS130] R.R. Palacio, A.L. Moran, A Vizcaino, and V.M. Gonzalez. Knowledge Flow as Facilitator for Getting into Collaboration in Distributed Software Development. In *Hawaii International Conference on System Sciences, Proceedings of the 41st Annual*, pages 1–10, January 2011.

[PS131] M.R. Peters, R.J. Ocker, and M.B. Rosson. Collaboration in a Distributed World: Technological Support for Partially Distributed Teams. In *Advanced Learning Technologies, 2008. ICALT '08. Eighth IEEE International Conference on*, pages 273–277, July 2008.

[PS132] S Paul and S Ray. Manifested Intra-Group Conflict in Collaborative Technology Supported Multi-Cultural Virtual Teams: Findings from a Laboratory Experiment. In *Hawaii International Conference on System Sciences, Proceedings of the 41st Annual*, pages 1–11, January 2010.

[PS133] J. Portillo-Rodriguez, A Vizcaino, C Ebert, and M Piattini. Tools to Support Global Software Development Processes: A

Survey. In *Global Software Engineering (ICGSE), 2010 5th IEEE International Conference on*, pages 13–22, August 2010.

[PS134] A Pasala, A Sethuraman, S Niranjani, R.P. Gorthi, and K.B. Gadde. Context-aware mobile assistant agents in software project management. In *TENCON 2008 - 2008 IEEE Region 10 Conference*, pages 1–6, November 2008.

[PS135] R.R. Palacio, A. Vizcaí andno, A.L. Morá andn, and V.M. Gonzá andlez. Tool to facilitate appropriate interaction in global software development. *Software, IET*, 5(2):157–171, April 2011.

[PS136] T Riechert and T Berger. Leveraging semantic data Wikis for distributed requirements elicitation. In *Wikis for Software Engineering, 2009. WIKIS4SE '09. ICSE Workshop on*, pages 7–13, May 2009.

[PS137] J.P. Rodríguez, C Ebert, and A Vizcaino. Technologies and Tools for Distributed Teams. *Software, IEEE*, 27(5):10–14, September 2010.

[PS138] Gregorio Robles and Jesus M. Gonzalez-Barahona. Geographic location of developers at SourceForge. In *Proceedings of the 2006 international workshop on Mining software repositories*, pages 144–150, New York, NY, USA, 2006. ACM.

[PS139] Felix Rodriguez, Michael Geisser, Kay Berkling, and Tobias Hildenbrand. Evaluating Collaboration Platforms for Offshore Software Development Scenarios. In Bertrand Meyer and Mathai Joseph, editors, *Software Engineering Approaches for Offshore and Outsourced Development*, pages 96–108. Springer Berlin / Heidelberg, 2007.

[PS140] Madhu C. Reddy, Bernard J. Jansen, and Rashmi Krishnappa. The role of communication in collaborative information searching. *Proceedings of the American Society for Information Science and Technology*, 45(1):1–10, 2008.

[PS141] Nicholas C. Romano, Jay F. Nunamaker, Robert O. Briggs, and Douglas R. Vogel. Architecture, design, and development of an HTML/JavaScript Web-based Group Support System. *Journal of the American Society for Information Science*, 49(7):649–667, 1998.

[PS142] T. Rodden, P. Sawyer, and I Sommerville. Vista: a user interface for a distributed object-oriented software engineering environment. *Software Engineering Journal*, 7(1):25–34, January 1992.

[PS143] Michael Reeves and Jihan Zhu. Moomba – A Collaborative Environment for Supporting Distributed Extreme Programming in Global Software Development. In Jutta Eckstein and Hubert Baumeister, editors, *Extreme Programming and Agile Processes in Software Engineering*, pages 38–50. Springer Berlin / Heidelberg, 2004.

[PS144] C. Solis and N. Ali. Distributed Requirements Elicitation Using a Spatial Hypertext Wiki. In *Global Software Engineering (ICGSE), 2010 5th IEEE International Conference on*, pages 237–246, August 2010.

[PS145] S Sakata. Multimedia and multi-party desktop conference system (MERMAID) as groupware platform. In *TENCON '94. IEEE Region 10's Ninth Annual International Conference. Theme: Frontiers of Computer Technology. Proceedings of 1994*, pages 739–743 vol.2, August 1994.

[PS146] Saonee Sarker. Knowledge Transfer and Collaboration in Distributed U.S.-Thai Teams. *Journal of Computer-Mediated Communication*, 10(4):00–00, 2005.

[PS147] J.L. Smith, S.A. Bohner, and D.S. McCrickard. Toward introducing notification technology into distributed project teams. In *Engineering of Computer-Based Systems, 2005. ECBS '05. 12th IEEE International Conference and Workshops on the*, pages 349–356, April 2005.

[PS148] Igor Steinmacher, Ana Chaves, and Marco Gerosa. Awareness Support in Global Software Development: A Systematic Review Based on the 3C Collaboration Model. In Gwendolyn Kolfschoten, Thomas Herrmann, and Stephan Lukosch, editors, *Collaboration and Technology*, pages 185–201. Springer Berlin / Heidelberg, 2010.

[PS149] Bikram Sengupta, Satish Chandra, and Vibha Sinha. A research agenda for distributed software development. In *Proceedings of the 28th international conference on Software engineering*, pages 731–740, New York, NY, USA, 2006. ACM.

[PS150] N Seyff, C Hoyer, E Kroiher, and P Grunbacher. Enhancing GSS-based requirements negotiation with distributed and mobile tools. In *Enabling Technologies: Infrastructure for Collaborative Enterprises, 2004. WET ICE 2004. 13th IEEE International Workshops on*, pages 87–92, June 2005.

[PS151] D.B. Simmons. Measuring and tracking distributed software development projects. In *Distributed Computing Systems, 2003. FTDCS 2003. Proceedings. The Ninth IEEE Workshop on Future Trends of*, pages 63–69, May 2003.

[PS152] Till Schümmer and Stephan Lukosch. Supporting the Social Practices of Distributed Pair Programming. In Robert Briggs, Pedro Antunes, Gert-Jan De Vreede, and Aaron Read, editors, *Groupware: Design, Implementation, and Use*, pages 83–98. Springer Berlin / Heidelberg, 2008.

[PS153] Jin Sa and E Maslova. A unified process support framework for global software development. In *Computer Software and Applications Conference, 2002. COMPSAC 2002. Proceedings. 26th Annual International*, pages 1065–1070, 2002.

[PS154] G. Shroff, A. Mehta, P. Agarwal, and R. Sinha. Collaborative development of business applications. In *Collaborative Computing: Networking, Applications and Worksharing, 2005 International Conference on*, page 6 pp., 0 2005.

[PS155] D Smite. Project Outcome Predictions: Risk Barometer Based on Historical Data. In *Global Software Engineering, 2007. ICGSE 2007. Second IEEE International Conference on*, pages 103–112, August 2007.

[PS156] Jeffrey B. Schmidt, Mitzi M. Montoya-Weiss, and Anne P. Massey. New Product Development Decision-Making Effectiveness: Comparing Individuals, Face-To-Face Teams, and Virtual Teams*. *Decision Sciences*, 32(4):575–600, 2001.

[PS157] I Samarah, S Paul, and S Tadisina. Knowledge Conversion in GSS-Aided Virtual Teams: An Empirical Study. In *Hawaii International Conference on System Sciences, Proceedings of the 41st Annual*, page 344, January 2008.

[PS158] V.S. Sinha, B Sengupta, and S Ghosal. An Adaptive Tool Integration Framework to Enable Coordination in Distributed Software Development. In *Global Software Engineering, 2007. ICGSE 2007. Second IEEE International Conference on*, pages 151–155, August 2007.

[PS159] Hans Spanjers, Maarten ter Huurne, Bas Graaf, Marco Lormans, Dan Bendas, and Rini van Solingen. Tool Support for Distributed Software Engineering. In *Global Software Engineering, 2006. ICGSE '06. International Conference on*, pages 187–198, October 2006.

[PS160] M Suchocki. Web-based collaboration: a key tool for the management of corporate information. In *Database and Expert Systems Applications, 2004. Proceedings. 15th International Workshop on*, pages 276–280, August 2004.

[PS161] Anita Sarma and André van der Hoek. Towards Awareness in the Large. In *Global Software Engineering, 2006. ICGSE '06. International Conference on*, pages 127–131, October 2006.

[PS162] J Suzuki and Y Yamamoto. Leveraging distributed software development. *Computer*, 32(9):59–65, September 1999.

[PS163] Walter F. Tichy. Rcs — a system for version control. *Software: Practice and Experience*, 15(7):637–654, 1985.

[PS164] S.S.F. Tse, G Moore, and S Narang. Self-Service Collaborative Applications: Automated Generation and Deployment of Applications Using a Visual Interface and Specification Tool. In *Collaborative Technologies and Systems, 2006. CTS 2006. International Symposium on*, pages 326–333, May 2006.

[PS165] M. Rita Thissen, Jean M. Page, Madhavi C. Bharathi, and Toyia L. Austin. Communication tools for distributed software development teams. In *Proceedings of the 2007 ACM SIGMIS CPR conference on Computer personnel research: The global information technology workforce*, pages 28–35, New York, NY, USA, 2007. ACM.

[PS166] Halbana Tarmizi, Matt Payne, Cherie Noteboom, Chi Zhang, Lucas Steinhauser, Gert-Jan De Vreede, and Ilze ZIgurs. Technical and Environmental Challenges of Collaboration Engineering in Distributed Environments. In Yannis Dimitriadis, Ilze ZIgurs, and Eduardo Gómez-Sánchez, editors, *Groupware: Design, Implementation, and Use*, pages 38–53. Springer Berlin / Heidelberg, 2006.

[PS167] V Trapa and S Rao. T3 - tool for monitoring agile development. In *Agile Conference, 2006*, pages 6 pp.–248, July 2006.

[PS168] Christian Thum, Michael Schwind, and Martin Schader. SLIM—A Lightweight Environment for Synchronous Collaborative Modeling. In Andy Schürr and Bran Selic, editors, *Model Driven Engineering Languages and Systems*, pages 137–151. Springer Berlin / Heidelberg, 2009.

[PS169] M. Verrall and T. Bingen. The basis for the Software Bus solution to the inter-operation problems of heterogeneous tools in a distributed SEE. In *Software Engineering Environments Conference, 1993. Proceedings*, pages 72–78, July 1993.

[PS170] P. Wongthongtham, E. Chang, and T. S. Dillon. Ontology-based multi-agent system to multi-site software development.

In *Proceedings of the 2004 workshop on Quantitative techniques for software agile process*, pages 66–75, New York, NY, USA, 2004. ACM.

[PS171] Igor Scaliante Wiese and Elisa Hatsue Moriya Huzita. IMART: An Interoperability Model for Artifacts of Distributed Software Development Environments. In *Global Software Engineering, 2006. ICGSE '06. International Conference on*, pages 255–256, October 2006.

[PS172] Dindin Wahyudin, Matthias Heindl, Benedikt Eckhard, Alexander Schatten, and Stefan Biffl. In-Time Role-Specific Notification as Formal Means to Balance Agile Practices in Global Software Development Settings. In Bertrand Meyer, Jerzy Nawrocki, and Bartosz Walter, editors, *Balancing Agility and Formalism in Software Engineering*, pages 208–222. Springer Berlin / Heidelberg, 2008.

[PS173] E. James Whitehead. Collaborative Authoring on the Web: Introducing WebDAV. *Bulletin of the American Society for Information Science and Technology*, 25(1):25–29, 1998.

[PS174] Gamel O. Wiredu. Understanding the functions of teleconferences for coordinating global software development projects. *Information Systems Journal*, 21(2):175–194, 2011.

[PS175] Timo Wolf, Thanh Nguyen, and Daniela Damian. Does distance still matter? *Software Process: Improvement and Practice*, 13(6):493–510, 2008.

[PS176] Merrill E. Warkentin, Lutfus Sayeed, and Ross Hightower. Virtual Teams versus Face-to-Face Teams: An Exploratory Study of a Web-based Conference System*. *Decision Sciences*, 28(4):975–996, 1997.

[PS177] Wenpeng Xiao, Changyan Chi, and Min Yang. On-line collaborative software development via wiki. In *Proceedings of the 2007 international symposium on Wikis*, pages 177–183, New York, NY, USA, 2007. ACM.

[PS178] Dongmei Xu, J Kurogi, Y Ohgame, and A Hazeyama. Distributed Collaborative Modeling Support System Associating UML Diagrams with Chat Messages. In *Computer Software and Applications Conference, 2002. COMPSAC 2002. Proceedings. 26th Annual International*, pages 367–372, July 2009.

[PS179] Bin Xu, Xiaohu Yang, and A Ma. Role based cross-project collaboration in multiple distributed software design projects. In *Computer Supported Cooperative Work in Design, 2008. CSCWD 2008. 12th International Conference on*, pages 177–182, April 2008.

[PS180] Pavan Yara, Ramaseshan Ramachandran, Gayathri Balasubramanian, Karthik Muthuswamy, and Divya Chandrasekar. Global Software Development with Cloud Platforms. In Olly Gotel, Mathai Joseph, Bertrand Meyer, Wil Aalst, John Mylopoulos, Michael Rosemann, Michael J. Shaw, and Clemens Szyperski, editors, *Software Engineering Approaches for Offshore and Outsourced Development*, pages 81–95. Springer Berlin Heidelberg, 2009.

[PS181] Yun Yang and P. Wojcieszak. Supporting distributed software development processes in a Web-based environment. In *Technology of Object-Oriented Languages and Systems, 1999. TOOLS 31. Proceedings*, pages 292–295, 1999.

[PS182] Li-Na Zhao, Yin Zhang, Xiu-Zi Ye, and San-Yuan Zhang. Distributed software engineering management. In *Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on*, pages 39–44 Vol. 1, August 2005.

## Appendix A. Search strings

Following the exact query strings used for each digital library. Limitations and lessons learned by performing the searches are reported.

*Legend*

| | |
|---|---|
| A1: global software development | A2: distributed software development |
| A3: global software engineering | A4: distributed software engineering |
| A5: distributed team | A6: distributed teams |
| A7: virtual team | A8: virtual teams |

| | | | |
|---|---|---|---|
| B01: technology | B02: technologies | B03: tool | B04: tools |
| B05: framework | B06: middleware | B07: plug-in | B08: plug-ins |
| B09: plugin | B10: plugins | B11: platform | B12: environment |

*Logic string*

```
( A1 or...or A8 ) and ( B01 or...or B12 )
```

*ACM*

Link: http://dl.acm.org/.

Difficult to reach but present, also ACM has an advanced search page where users can directly insert the query without constraints regarding the number of terms used. Moreover, a special tag (i.e. *Owner:ACM*) can be used to limit the search to articles published by ACM avoiding the ones indexed but published by others. To download the citations a crawler was developed, for the web site does not allow the download of multiple citations. Moreover, in the past, third party software like *Mendeley*[26] or *Zotero*[27] were available to support the automatic download of references; however, at the time we performed the searches, they were not working properly. Moreover, Mendeley is able to retrieve only the snippet of the abstract prompted in the results page and citations have to be selected one by one and downloaded page by page (20 at a time). On the other hand, Zotero was another option and it would have actually been better considering that it is able to retrieve the full abstract but, after using it on more than one page, the system was blacklisting the ip for some days.

```
(Owner:ACM) AND
(Abstract:"A1" OR...OR Abstract:"A8" OR Title:"A1" OR...OR Title:"A8") AND
(Abstract:"B01" OR...OR Abstract:"B12" OR Title:"B01" OR...OR Title:"B12")
```

*IEEE Xplorer*

Link: http://ieeexplore.ieee.org/.

Using the command search (reachable from the advanced search), it is possible to insert specific tags to limit the search to titles and abstracts. To

[26] http://www.mendeley.com/

[27] http://www.zotero.org/

download the citations it is possible to modify the page link overcoming the limit of 100 publications per page (500 publications per page has been tested and works smoothly).

```
(("Abstract":"A1") OR...OR ("Abstract":"A8") OR
("Document Title:"A1") OR...OR ("Document Title":"A8")) AND
(("Abstract":"B01") OR...OR ("Abstract":"B12") OR
("Document Title":"B01") OR...OR ("Document Title":"B12"))
```

*SpringerLink*

Link: http://www.springerlink.com/.
Using the new web interface and looking at the generated URL it is possible to see that the tag *ab* can be used to limit the search to titles and abstracts only; however, the engine imposes a constraint on the number of allowed keywords which cannot exceed 10. Therefore, the query had to be split resulting in a not negligible amount of duplicates. The new web interface incorporates a button to download citations but apparently the functionality has not yet been added as the button is disabled; thus, to download the citations we followed the same approach we used for the ACM digital library.

```
ab:(( A1 OR...OR A8 ) AND ( <( B01 OR B02 )|...|( B11 OR B12 )> ))
```

*ScienceDirect*

Link: http://www.sciencedirect.com/.
Using the expert search (reachable from the advanced search), it is possible to insert specific tags to limit the search to titles and abstracts and keywords. Even if not an issue encountered during this study, it has to be noticed that ScienceDirect allows only the download of the first 1000 publications, which means that it will probably be necessary to split the query covering smaller year ranges. To avoid facing this issue in the future we successfully adapted the crawler to see if it was applicable also to this search engine.

```
(abs("A1") OR...OR abs("A8") OR ttl("A1") OR...OR ttl("A8")) AND
(abs("B01") OR...OR abs("B12") OR ttl("B01") OR...OR ttl("B12"))
```

*Wiley InterScience*

Link: http://onlinelibrary.wiley.com/.
After testing the search engine we discovered that the structure to be used for limiting the search to specific fields was defined by these expressions: *in Article Titles* for the title and *in Abstract* for the abstract. Therefore, we composed the query and used in the main search field.

```
("A1" in Article Titles OR...OR "A8" in Article Titles OR
"A1" in Abstract OR...OR "A8" in Abstract) AND
("B01" in Article Titles OR...OR "B12" in Article Titles OR
"B01" in Abstract OR...OR "B12" in Abstract)
```

# Appendix B. Tools

**NoName** [PS153]
**NoName** [PS158]
**NoName** [PS134]
**NoName** [PS023]
**NoName** [PS182]
**NoName** [PS121]
**NoName** [PS060]
**NoName** [PS172]
**NoName** [PS108]
**NoName** [PS116]
**NoName** [PS047]
**NoName** [PS179]
**NoName** [PS099]
**NoName** [PS008]
**NoName** [PS160]
**NoName** [PS131]
**NoName** [PS148][21]
**NoName** [PS170]
**NoName** [PS089][PS084]
**NoName** [PS181]
**NoName** [PS148][54]
**NoName** [PS148][75]
**NoName** [PS103][136]
**4everedit** [PS119]
**ABREV** [PS032]
**Aceproject** [PS043][aceproject.com]
**ActionPlan** [PS105]
**ActiveCollab** [PS137][PS133][PS101][activecollab.com]
**ADAMS** [PS049][PS016][PS050]
**ADDSS** [PS035][PS133][24]
**Adele** [PS036][10]
**Adobe Breeze** [PS019][adobe.com/resources/breeze]
**Adobe Connect** [PS043][PS168][adobe.com/products/adobeconnect.html]
**AIM** [PS085][PS165][PS019][aim.com]
**AISA** [PS103][130]
**Ant** [PS101][PS098][ant.apache.org]
**AntHill** [PS098][urbancode.com/html/products/anthillpro]
**Apache Continuum** [PS101][PS129][continuum.apache.org]
**APEL** [PS036][49]
**Arcadia** [PS036][133]
**ArcStyler** [PS139][www.arcstyler.com]
**Arena II – Arena-M** [PS150]
**ArgoUML** [PS081][argouml.tigris.org]
**Ariadne** [PS002][PS053][PS060][PS001][PS125][PS148][37]
**Artisan Studio** [PS101][atego.com/products/artisan-studio]
**ASPECT** [PS142][63]
**Assembla** [PS101][PS137][PS133][assembla.com]
**ASSIST** [PS031][PS103]
**AT&T** [PS165][teleconference.att.com]
**Atlassian Bitbucket** [PS106][bitbucket.org]
**Augur** [PS059][PS125]
**autoconf** [PS069][gnu.org/software/autoconf]
**automake** [PS069][gnu.org/software/automake]
**ASU** [PS095]
**B2Evolution** [PS019][b2evolution.net]
**Base Camp** [PS043][basecamp.com]
**Bazaar** [PS101][bazaar.canonical.com]
**bitkeeper** [PS069][bitkeeper.com]
**Blast** [PS165][blast.com/cgi-bin/blast.com/view_services.cgi?request=detail&prod_num=1UNX107]
**Blogger** [PS019][blogger.com]
**Blogware** [PS019][ShutDown]
**Booster** [PS026]
**Borland ALM** [PS106][borland.com/products]
**Borland Together** [PS104][PS164][borland.com/us/products/together]
**BOX** [PS122]
**BSCW** [PS161][4]
**Bugzilla** [PS106][PS069][PS095][PS098][PS101][bugzilla.org]
**CAIS** [PS142][104][PS103]
**CAISE** [PS148][30]
**CalendarBot** [PS085]
**CaliberRM** [PS144][borland.com/products/caliber]
**Camel** [PS137][PS052][PS133][PS054][26]
**Caramba** [PS055][44]
**CASI** [PS148][127]
**CAST** [PS106][castsoftware.com/products/cast-application-intelligence-platform]
**Celine** [PS148][50]
**Central Desktop** [PS043][centraldesktop.com]
**CES** [PS064][58]
**CheckStyle** [PS106][PS098][checkstyle.sourceforge.net]
**CHIME** [PS048]
**Cisco WebEx** [PS165][PS111][PS127][PS180][PS168][PS019][PS014][PS101][webex.com]
**Citrix GoToMeeting** [PS165][PS127][PS019][citrix.com/gotomeeting]
**Clearspace** [PS043][jivesoftware.com/social-business/beyond/clearspace]
**CODEBASE** [PS118]
**CodeBeamer** [PS139][intland.com/products/codebeamer]
**CodePlex** [PS101][codeplex.com]
**CodeSaw** [PS075]
**Coefficent** [PS043][n/a]
**CollabNet** [PS149][PS139][PS106][PS154][collab.net]
**CPA** [PS015]
**Collaborator** [PS024]
**CollabVS** [PS045][PS133][41]
**Community Bar** [PS125][57]
**ConnectIcon** [PS034]
**CONTact** [PS082][PS087][94]
**CRI** [PS113][PS125][PS148][105]
**COOP/Orm** [PS161][95]
**COPPER** [PS133][102]
**Cosmos** [PS025]
**Coven** [PS149][29]
**Creately** [PS101][creately.com]
**CASS** [PS125][83]
**CruiseControl** [PS101][PS098][cruisecontrol.sourceforge.net]
**CSI** [PS103][96]
**CSRS** [PS103][79]
**CVS** [PS101][PS148][PS147][PS128][PS182][PS139][PS106][PS069][cvs.nongnu.org]
**CVS-Watch** [PS161][12]
**CWS-IM** [PS135][PS130]
**Darcs** [PS101][darcs.net]
**Deep Intellisense** [PS022][72]
**Dhruv** [PS113][3]
**DHT** [PS124]

en-us/excel]

**Microsoft Exchange** [PS165][microsoft.com/exchange]

**Microsoft NetMeeting** [PS086][PS165][PS031][PS046][PS057][PS052][PS112][microsoft.com/en-us/download/details.aspx?id=23745]

**Microsoft Office** [PS137][PS019][office.microsoft.com]

**Microsoft Office Groove** [PS107][PS166][groove.net]

**Microsoft Office LiveMeeting** [PS019][office.microsoft.com/en-us/live-meeting]

**Microsoft Outlook** [PS165][PS137][PS014][office.microsoft.com/en-us/outlook]

**Microsoft PowerPoint** [PS052][office.microsoft.com/en-us/powerpoint]

**Microsoft Project** [PS151][PS079][microsoft.com/project]

**Microsoft SharePoint** [PS137][PS106][PS043][PS068][PS014][PS101][sharepoint.microsoft.com]

**Microsoft SharePoint Online** [PS180][sharepoint.microsoft.com/en-us/sharepoint-online]

**Microsoft Visual Source Safe** [PS165][PS182][en.wikipedia.org/wiki/Microsoft_Visual_SourceSafe]

**Microsoft Visual Studio** [PS106][PS114][PS125][microsoft.com/visualstudio]

**Microsoft Visual Studio Team System** [PS056][n/a, microsoft.com/visualstudio/11/en-us/lifecycle]

**MILOS** [PS110][PS108][PS109][PS112]

**Moomba** [PS143]

**Movable Type** [PS019][movabletype.org]

**MRAC** [PS135][122]

**MSN** [PS165][PS019][PS126][msn.com]

**MUSE** [PS140]

**MUST** [PS140]

**Mylyn** [PS113][PS095][PS022][PS125][eclipse.org/mylyn]

**NetBeans** [PS106][PS125][netbeans.org]

**NextMove** [PS115][PS052][PS054]

**NightWatch** [PS161][107]

**Novell Groupwise** [PS165][novell.com/products/groupwise]

**Odyssey** [PS108][20]

**Oikos** [PS036][2]

**OMAS** [PS013]

**OntoWiki** [PS136]

**Open Exchange** [PS043][PS106][open-xchange.com]

**OpenChannel** [PS034]

**OpenSTA** [PS101][opensta.org]

**OZ 1.0** [PS030]

**p2pconference** [PS103][22]

**PAKME** [PS133][5]

**Palantir** [PS002][PS161][PS148][PS095][PS045][PS135][PS130][PS125][118]

**PAMPA** [PS151][129]

**Pan** [PS064][6]

**pcAnywhere** [PS165][symantec.com/pcanywhere]

**PCTE** [PS020][PS142]

**Perforce** [PS101][perforce.com]

**PIEnvironment** [PS044]

**PIMS** [PS074]

**PIPE** [PS116][101]

**PMD** [PS098][pmd.sourceforge.net]

**pmlcheck** [PS010]

**Polarion** [PS139][PS106][polarion.com]

**Poseidon for UML** [PS168][gentleware.com]

**Process Weaver** [PS036][51]

**ProjectIT-Enterprise** [PS144][53]

**ProjectView-IM** [PS135][23]

**ProjectWatcher** [PS135][PS130][61]

**PROSYT** [PS036]

**Provence** [PS036][8]

**Rational Asset Manage** [PS137][ibm.com/software/rational/products/ram]

**Rational ClearCase** [PS137][PS151][PS106][PS096][ibm.com/software/awdtools/clearcase]

**Rational ClearQuest** [PS137][PS151][ibm.com/software/awdtools/clearquest]

**Rational Insight** [PS137][ibm.com/software/rational/products/insight]

**Rational PurifyPlus** [PS151][ibm.com/software/awdtools/purifyplus]

**Rational Quality Manager** [PS137][PS133][ibm.com/software/rational/products/rqm]

**Rational Rhapsody** [PS139][ibm.com/software/awdtools/rhapsody]

**Rational Requirements Composer** [PS137][PS133][ibm.com/software/awdtools/rrc]

**Rational RequisitePro** [PS151][PS172][PS144][ibm.com/software/awdtools/reqpro]

**Rational Rose** [PS153][PS104][PS151][PS081][PS101][ibm.com/software/awdtools/developer/rose]

**Rational SoDA** [PS151][ibm.com/software/awdtools/soda]

**Rational Software Modeler** [PS101][ibm.com/software/awdtools/modeler/swmodeler]

**Rational Suite** [PS139][PS106][ibm.com/software/rational]

**Rational Tau** [PS137][PS133][ibm.com/software/awdtools/tau]

**Rational Team Concert** [PS101][PS137][PS133][PS125][jazz.net/projects/rational-team-concert]

**Rational Test** [PS151][ibm.com/software/awdtools/test/realtime]

**RCS** [PS163][PS079]

**ReachOut** [PS148][113]

**Rear View Mirror** [PS085][PS135][19]

**Remail** [PS022]

**Remote Administrator** [PS165][eset.com/us/business/products/remote-administrator]

**Remote Desktop** [PS165][windows.microsoft.com/en-us/windows-vista/Connect-to-another-computer-using-Remote-Desktop-Connection]

**Risk Barometer** [PS155]

**RM-Tool** [PS100]

**RTM** [PS079][serena.com/solutions/requirements-management]

**SAGE** [PS093]

**Sangam** [PS149][PS046][71][sangam.sourceforge.net]

**Scaki** [PS062]

**Scrutiny** [PS103][56]

**SEAPort** [PS125][13]

**SeeSoft** [PS125][46]

**Selenium** [PS133][PS101][seleniumhq.org]

**SENTINEL** [PS036][32]

**Serendipity** [PS108][59]

**Serendipity-II** [PS067][PS070]

**SEWB** [PS117]

**SharePoint Communities** [PS137][sharepoint.microsoft.com/en-us/product/capabilities/communities]

**SharePoint Composites** [PS137][sharepoint.microsoft.com/en-us/product/capabilities/composites]

**SharePoint Content** [PS137][sharepoint.microsoft.com/en-us/product/capabilities/content]

**SharePoint Designer** [PS137][sharepoint.microsoft.com/en-us/product/related-technologies]

**SharePoint Foundation** [PS137][sharepoint.microsoft.com/en-us/product/Related-Technologies]

**SharePoint Insights** [PS137][sharepoint.microsoft.com/en-us/product/capabilities/insights]

**SharePoint Search** [PS137][sharepoint.microsoft.com/en-us/product/capabilities/search]

**SharePoint Sites** [PS137][sharepoint.microsoft.com/en-us/product/capabilities/sites]

**SharePoint Workspace** [PS137][office.microsoft.com/en-us/sharepoint-workspace]

**Skype** [PS018][PS111][PS123][PS127][PS007][PS019][PS126][skype.com]

**SLIM** [PS079][n/a]

**SLIM** [PS168]
**SmallBlue** [PS148][45]
**SmartWiki** [PS144][88]
**SoftDock** [PS162]
**SoftFab** [PS159]
**Software Bus** [PS169]
**SWORE** [PS144][PS136]
**SOP-wiki** [PS144][38]
**SourceForge** [PS149][PS101][PS106][PS138][sourceforge.net]
**SPADE** [PS036][7]
**Spatial Hypertext Wiki (ShyWiki)** [PS144]
**SPEARMINT** [PS063][115]
**SQA** [PS028][n/a]
**Stellation** [PS149][eclipse.org/technology/archived.php]
**STEVE** [PS050]
**Subclipse** [PS133][subclipse.tigris.org]
**Sun Java Studio Enterprise** [PS139][developers.sun.com/jsenterprise]
**SVN** [PS137][PS106][PS101][PS148][PS139][PS133][subversion.tigris.org]
**SWEEP** [PS042]
**Sysiphus** [PS017][PS038][PS133][PS125]
**T3** [PS167]
**TagSEA** [PS125][tagsea.sourceforge.net]
**TAMRI** [PS102][PS052][PS054]
**Task Manager** [PS093][90]
**Taskmaster** [PS135][11]
**Team Tracks** [PS125][40]
**Teamcenter** [PS101][siemens.com/teamcenter]
**TeamPortal** [PS034]
**TeamSCOPE** [PS148][PS147][PS086][PS090]
**TeamSpace** [PS147][PS078][PS052][PS054][PS061]
**TeamSpirit** [PS041]
**TeamVote** [PS065]
**TeamWeaver** [PS113][teamweaver.org]
**Teamwork** [PS043][n/a]
**Tesseract** [PS148][117]
**TestLink** [PS133][PS101][testlink.org]
**ToolChain** [PS092]
**ToxicFarm** [PS148][PS076]
**Trac** [PS101][PS095][PS133][trac.edgewall.org]
**TraVis** [PS073][PS053]
**Trillian** [PS019][trillian.im]
**Trusting Social Network** [PS009]
**TUKAN** [PS108][PS046][PS133][124]
**TWiki** [PS137][PS139][PS133][twiki.org]
**TypePad** [PS019][typepad.com]
**UNISYS** [PS081][ibm.com/support/docview.wss?uid=swg21131368]
**V-TeamSystem** [PS083]
**vi** [PS069][vim.org]
**VirtualSafe** [PS106][n/a]
**Visible Analyst** [PS101][www.visible.com/Products/Analyst]
**Vista** [PS142]
**Visual Paradigm TeamWork Center** [PS139][visual-paradigm.com]
**VNC** [PS128][e.g., tightvnc.com]
**Vonage** [PS127][vonage.com]
**War Room Command Console** [PS148][106]
**Watir** [PS101][watir.com]
**WebCT** [PS146][blackboard.com]
**WebDAV** [PS173][PS068]
**WebMake** [PS026][PS027]
**WikiWinWin** [PS144][140]
**WiP** [PS103][66]
**WiT** [PS103][67]
**WordPress** [PS101][PS019][wordpress.org]
**Workspace Activity Viewer** [PS002][PS101][114]
**Workspace3d** [PS101][tixeo.com]
**World View** [PS002][PS001][PS101][PS148][1]
**WWW SDE** [PS120]
**X-Lite** [PS111][counterpath.com/x-lite.html]
**Xanth** [PS091]
**XATI** [PS096][68][PS103]
**Xchat** [PS019][xchat.org]
**XCHIPS** [PS063][116]
**Xerox Docushare** [PS106][docushare.xerox.com]
**XPairtise** [PS152]
**XPlanner** [PS018][PS098][xplanner.codehaus.org]
**Yahoo Groups** [PS043][PS126][groups.yahoo.com]
**Yahoo Messenger** [PS165][PS007][PS019][messenger.yahoo.com]
**YooHoo** [PS135][PS148][73]
**Zimbra** [PS043][zimbra.com]
**Zoho Office Suite** [PS180][zoho.com]
**Zwiki** [PS021][PS004][zwiki.org]

ASU: Automatic Status Updates
CPA: Collaborative Panel Administrator
CRI: Continuum of Relevance Index
CASS: Cross Application Subscription Services
GDN: Global Development Navigator
SWORE: SoftWiki Ontology for Requirements Engineering