Spreadsheets as Collaborative Technologies in Global Requirements Change Management

Waqar Hussain, Tony Clear
School of Computer and Mathematical Sciences
Auckland University of Technology
Auckland, New Zealand
waqar.hussain@aut.ac.nz, tony.clear@aut.ac.nz

Abstract— **Globally distributed stakeholders employ various collaborative technologies to manage requirements. While these technologies facilitate requirements collaboration, their perceived purpose, use and structure co-evolve over time. In this paper we report the results of a study in two global software development settings involving client-vendor relationships. In both cases we noted that the vendor and client sites appropriated spreadsheet technology in quite specific ways, for use locally and for bridging across sites. Yet these spreadsheet files were embedded within different collaborative technologies. Through close study we note how team members practices co-evolved with the spreadsheet artefacts involved in the process of managing requirements change. We note how through a single spreadsheet cell, we may see a *world* as in William Blake's *"grain of sand*". Through the evolution of a spreadsheet's structure or content we expose how seemingly incremental local changes have far wider implications for global requirements change management.**

Keywords— Global Software Development, Global Software Engineering, Requirements Change Management, Spreadsheets, Artefacts, Evolution, Collaborative Technologies, Appropriation.

I. INTRODUCTION

Requirements management is a collaboration intensive activity, which produces and consumes numerous artefacts during a project's lifespan. Developing a shared understanding and awareness of these artefacts is integral to software collaboration [1], a challenge that globally distributed teams face on a regular basis. Use of collaborative technologies and artefacts is pervasive in global software development (GSD) [2] to alleviate some of the barriers caused by distance (geographic, temporal, and cultural) [3, 4], and to facilitate consistency in understanding and managing requirements [4].

From our observations of the ubiquity of spreadsheets in GSD practice, we are interested in understanding the role and evolving nature of spreadsheets both as collaborative 'tools' and as embedded artefacts employed in requirements change management activities carried out in GSD. The logical research question that follows this interest is:

What role do spreadsheets as collaborative technologies play in enabling global requirements change management?

Although not typically viewed as a collaborative technology, Norman describes a spreadsheet as a "cognitive artifact" that can be understood and shared by a group of people, providing a point of cognitive contact that mediates cooperative work [5]. Artefacts are embedded in collaborative technologies (e.g. spreadsheets) which are meant to facilitate interactions between cross functional stakeholders. Sometimes they are the 'means to an end' e.g. (aiding the negotiation process); and other times themselves are the 'end' of an activity; being the (jointly produced) outcome of a collaboration effort (e.g. a Requirements Specification). Generally these artefacts represent tentative outcomes and their exchange is critical to the projects. However because of different processes, people and sites involved they are often 'localized' both for concept and structure [6]. We believe that collaborative artefacts are not only 'alive' in their nature, rather they progressively co-evolve and mature through collaboration, serving as facilitators of group interactions.

This paper is organized as follows; Section II backgrounds the role of collaborative artefacts and collaborative technology (CT) as an enabler of global requirements change management (GRCM). This discussion leads to the role of spreadsheets as repositories for various embedded artefacts. Section III profiles the research sites, GSD context and empirical data for the case study. Then in Section IV we present a close analysis of selected requirements change process artefacts that are embedded in spreadsheets and discuss their co-evolution over time. Section V highlights how these different local evolutions build global work practices for critical activities in global requirements management. The implications of spreadsheets within a repertoire of collaborative technologies and GSD practice are then explored. Section VI briefly summarizes and concludes the paper.

II. RELATED LITERATURE

Modern systems development necessitates a collaborative approach to software requirements management [7]. Globally dispersed stakeholders specify and manage requirements from geographical, temporal and cultural distance which renders effective requirements engineering (RE) difficult [3, 8]. Basic requirements management (RM) is considered as one of the most important but complex part of the RE process [9, 10]. Global software development makes RM even harder to perform in distributed environments [3]. The need for dedicated tool support has been identified for requirements management [10] to compensate for the lack of informal communication and collaboration. Commercial requirements management tools such as DOORS™, RequisitePro™ etc. come with basic collaboration mechanisms but a lack of deep integration between the requirements and communication environments. This causes information fragmentation across several media which, ultimately leads to a lack of common understanding of requirements [7].

For smaller scale projects with fewer coordination requirements, current RM tools, with slight modifications, have been reported to support global software development [11]. But more recent studies emphasize the need for developing better situated infrastructure and tool development [12]. This type of infrastructure can then support seamless coordination and construction of shared mental models of the problem and requirements, between distributed team members.

A. Collaborative Artifacts and Requirements Management

In today's collaborations "team members go beyond the simple coordination of still individualistic work to engage in joint activity aimed at the co-construction of collective work products" [1]. Much collaboration over distance relates to various work products; formal and semi-formal artefacts such as (requirements specifications, code, bug reports etc.) [1].

"The objectives of the software requirements management process are to develop a shared, documented understanding of the requirements (in the form of the SRS), and to enforce a mechanism to control volatility so that the system satisfactorily fulfils requirements at the time of its delivery" [7]. However with distributed stakeholders and inadequate social contact, as found in global software settings, gaining consistent understanding of requirements or managing requirements changes is very difficult without adequate tool support [4].

Artefact-based collaboration is focused on the generation and sharing of new models (artefacts) that represent the envisioned software being developed at different levels of abstraction. As a key ingredient to collaboration such artefacts are meant to aid understanding and communicating the various aspects of software at different stages [1] and facilitate achieving the RM objective of a documented and shared understanding of requirements.

B. Collaborative Technologies and Their Perceptions

The introduction of new Collaborative Technologies (CTs) in the Web 2.0 era have radically enhanced possibilities for collaboration in globally distributed projects [13]. For open source and global software projects the use of Web 2.0 applications has become quite common, generally to aid informal communication [2]. Effective CT support is a strategic initiative especially for companies with distributed resources and working in a GSD context [2].

Global practitioners "appropriate" [14] communication and collaboration technologies in particular ways to assist requirements related collaboration. CTs have become an everyday activity and are almost taken for granted in workplaces [13] yet this 'taken-for-granted', semi-invisibility leads to a lack of critical understanding. CTs adopted in global teamwork differ from more individually directed forms of IT.

Differing conceptions of IT, as outlined in [15], also shape understandings: The computational view, of "technology as algorithm" underpins the computer science discipline; The tool view of "technology as labor substitution tool" and "technology as productivity tool" underpins the commercial perspective on IT and the business rationale for IT industry research and development activities; The ensemble view regards "technology as development project" and could be said to underpin the software engineering discipline. Yet this latter conception is often stubbornly overlain by the instrumentalist 'tool' view, viz. the frequency of papers that include the word "tool' in their title, e.g. [2].

The ensemble view includes the conceptions of technology as embedded, and technology as structure. In both of those perspectives a more dynamic view is projected of technology as "an evolving system embedded in a complex and dynamic social context" [15]. This view embodies "social structures…which…have been built into the technology by designers during its development and which are then appropriated by users as they interact with the technology" [15]. We adopt these perspectives of technology as embedded and as structure, in the context of technology as 'global development project'. Through this paper we investigate how the ensemble of CTs, artefacts and practices co-evolve.

The process of active collaboration over distance requires technology support, but the manner in which collaborative technologies are themselves shaped by and constitutive of practice in GRE warrants further investigation. As noted by Lanubile et al. [2] none of the existing tools or collaborative development environments fully support all the necessary activities for GSD. Therefore choice of CTs should result from prioritization of collaboration needs by the users themselves.

C. Spreadsheets use in GSD Practice

Spreadsheets are "perhaps the best known examples of existing radically tailorable systems" [16]. The translucent nature of spreadsheets with "surprisingly strong support for cooperative development of a wide variety of applications" make them a tool of choice for many organizations [17]. GSD practitioners often adopt spreadsheets specifically for requirements management and issues/bug tracking [18, 19]. Tell and Babar [18] 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 in the GSD context. Certain agile projects have been said to use one spreadsheet to manage the entire project. Such a spreadsheet is multipurpose in nature and records "all requirements, their status, effort, responsibilities and mapping to increments, test cases and work products" [20]. As suggested by Prikladnicki et al. [21] collaborative technologies such as spreadsheets do not just organically evolve rather their structure and content is transformed through co-evolution of practice and collaboration. Spreadsheets are commonly perceived to be single-user 'tools'; not specifically built to support any of the 3C's (coordination, communication, collaboration) in collaboration models [18]. However their work "work flows across different users in fluid, informal ways and cooperation among spreadsheet users has a

TABLE 1.GSE STUDY CONTEXTUAL FACTORS

Market	**Case 1**	**Case 2**
Industry	Road Construction	Education
Sector	Government	Commercial
Product		
Maturity	Custom Development Project	Custom Development/COTS Extension
Software	Information Management System	Research Management System
Application Type	Web Application	Web Application
Size	Medium	Small
Cost	USD $3.5M	NZD $3.7M
Complexity	Medium	Medium
Requirements Change Rate (Volatility)	Low to Medium	Low to Medium
Organization		
Maturity/Certification	Mature-CMMI Level 2 Certified	Newly Formed Company
Team Size	Three to Ten	Three to Ten
Development Process		
Outsourcing	Farshore Outsourcing	Farshore Outsourcing
Methodology	Waterfall by Feature	Hybrid Agile Waterfall
Workflow	Sequential by Feature	Scrum Based
Practices	Prototyping, Webinars	User Stories, Time Boxing
Global Collaboration		
Collaborating Units	Pakistan, USA	NZ, USA, Canada
Collaborative Tools	Email, Microsoft Lync, Bug Tracker, Design Specification, Spreadsheets	Email, Skype, Requirements Specification, Bug Trackers, Spreadsheets, GoTo Meetings
Collaborative Artefacts	Contract, Web UI Prototypes, Enhancement on Design Specifications, Software Releases, Requirements Change Log	User Stories, High Level Requirements Specifications, Issue Logs, Software Releases
People		
Collaborative Roles	PMs, End User Group Members, BAs	Architect, Lead Developer, PM, PO,BAs

spontaneous, self-directed character" [17].

As observed in our case study, flexible software repositories for artefacts, especially spreadsheets facilitate requirements management activities, such as requirements analysis, negotiation, documentation and validation. Stakeholders engaged in these activities can visually contribute towards the final form of the shared requirement artefacts (e.g. requirements specifications or sprint backlogs) using various application sharing 'tools'. In GSD projects this adaptation of artefacts and practices to suit emerging needs as they are encountered, tends to be a pragmatic and incremental process warranting further investigation.

III. SITES AND METHODS

We profile here the outcomes of an exploratory case study conducted at two different GSD settings in two organizations. Table 1 compares the two studied cases and covers the global and empirical aspects of the study [22]. Similarly, Table 2 profiles the software engineering context for the studied projects. It extends the contextual factors given by [23] to include collaboration technology a notably missing element in both [22, 23].

A. Sites

Company A is an off shore software development unit in Pakistan. It is a CMMI Level–II certified small to medium sized company with almost 50 employees. In the studied project the company was working on support services development for an existing information system. Frequent interactions took place between development team members and clients at geographically dispersed locations regarding requirements change and management. This made coordinating implementation work extremely complex and subject to continuous change. Fig.1 Case 1 maps the involved sites, roles and the technologies involved for collaboration over distance.

Company B is a NZ university working as a client with a US-based vendor developing and customizing a vendor-supplied research grants management product. All development, which was performed at multiple vendor sites in the USA and Canada, took place through communication and collaboration technologies. These sites had to collaborate and exchange information frequently for decision making that required cross-site consultation. All activities related to requirements communication, negotiation and validation had to be coordinated over geographic, temporal and cultural distances. The client and vendor had no prior experience of working together and thus had limited established mutual trust. The vendor was a new start up but had prior experience and domain expertise for research and grants management software. Fig.1 Case 2 maps the sites, roles and the technologies involved in collaboration.

B. Method

In this exploratory case study qualitative data is used for analysis purposes. The two projects at two selected organizations were explored over time, through detailed, in-depth data collection. The main source of study data was derived from requirements change related artifacts (electronic and non-electronic) supported from insights into their use gained through semi structured interviews.

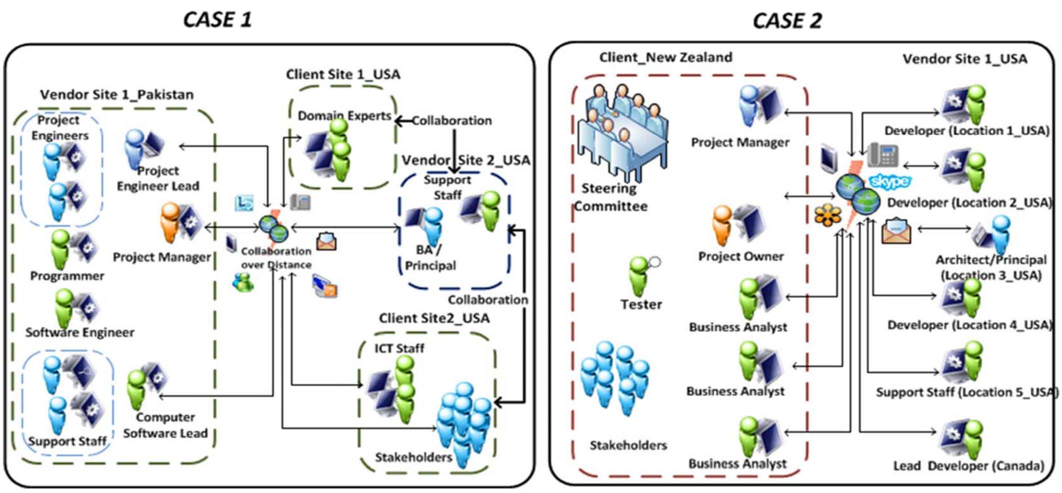


Fig. 1. Collaboration over Distance btween Pakistan and USA (Case 1) and between New Zealand, USA and Canada (Case 2)

For Case 1 in Pakistan seven members from a project core development team were interviewed twice over a period of six months. Also the proxy client and principal for the same project, who is collocated with the project client in the USA was interviewed once during the first round when he was on his regular business visit to the Pakistan site. For Case 2 in NZ, eight client stakeholders were interviewed, including members of the steering committee, project management team and end users. Case 1 in Fig.1 depicts collaboration among various team members and roles between Pakistan and the USA.

In this case the vendor must collaborate with three sets of stakeholders from the client and the client collaborates with two segments of the vendor organization. The vendor collaborates with the client's IT staff, their managers and users (shown as 'Stakeholders') the third set is the domain experts as shown in Case 1 of Fig. 1. Domain experts in this project are the extended clients who take part in requirements related activities, provide domain expertise for requirements, and perform verification and validation services for releases.

Members on the client side collaborated with two segments of the vendor organization –the onshore vendor coordination team (BA and support staff) and the offshore project development team shown in Case1 of Fig. 1. Collaboration over distance in this case often took place through emails, teleconferences, phone or mobile calls, use of application sharing through Microsoft Lync, and chat messengers. Electronic artefacts played a major supportive role in these collaboration activities as they were often exchanged as email attachments (as finished or semi-finished work products) or were embedded in the collaborative technology itself (as spreadsheets, mockups etc.) supporting these collaborations.

Case 2 in Fig.1 illustrates collaboration between client team members in NZ and various vendor locations and roles. This again is an 'atypical' virtual team composition compared to the one described in [24]. Individual vendor team members were scattered at different locations in USA and Canada but worked as a virtual team by means of the various communication and collaborative technologies. The predominant technologies used were email, Skype, GoTo meetings (with application and screen sharing), and phone calls. Another important noteworthy factor was the absence of a vendor coordination team collocated or onshore with the client.

TABLE 2. EMPIRICAL DATA FOR GSE STUDY

Empirical background	Case 1	Case 2
Main Method	Case Study	Case Study
Sub-Method	Interviews	Interviews
Background	Industry Real world	Industry Real world
Subjects of investigation	Industry Real world	Industry Real world
Empirical Focus	Empirically Based	Empirically Based
GSE Background		
Collaboration mode	Inter organizational	Inter organizational
Number of Locations	Four	Six
Location of the originator	USA	NZ
Location of the suppliers	Pakistan	USA, Canada
Perspective	Supplier	Client
Reason for outsourcing	Cost, access to pool of skilled resources	Access to expertise & product
Study		
Development methodology	Waterfall by feature	Waterfall-Agile Hybrid
Focus of the study	Requirements Change Mgt.	Requirements Change Mgt.
Success or failure?	Success of the practice described	Evidence of the GSE-related problems
Application domain	Web	Web
Claims	None	None

IV. ANALYSIS

This section analyses various artefacts related to requirements change management activities. We report our findings from a study of the two distributed development projects, Case 1 and Case 2 shown in Fig.1.

A. Artefact Analysis

For this paper, three different types of artefacts were chosen to study their evolution. These were Requirements Change Log (RCL) and Enhancements on Design Specification (EDS) from Pakistan site (Fig 1. Case 1) and High Level Requirements Specifications (HLRS) from the NZ site (Fig 1. Case2). All of these artefacts were embedded in spreadsheets.

Upon initial analysis of the codified empirical data we found ideas related to artefact evolution as well as the need to use multiple formats for working on requirements change management. The following paragraph relating to the HLRS is one of these examples.

> "…the developers have a certain template, that they like, **which we have sort of tweaked along the way,** but by and large they like screenshots , they like us to number on the screen where the changes are, on a different tab list-out those changes so it depends on what we are changing, if the screen is changing significantly then I will list all the field elements I will list the sort order , I will list the type of field , if we are changing a drop down list then I will list it separately. So it depends on the level of change, **but they have firmly asked us to spec what is different to what they have already got vs. specking from scratch**."

The sections below discuss the evolution of the RCL and EDS artefacts.

1) Requirements Change Log (Case 1):

During our exploratory case study at these sites (Pakistan, USA) we came across a few versions of this artefact (cf. table 3) which we selected as good candidates to study artefact evolution. We wanted to explore if the changes to the artefact template were influenced by a single site or were due to the (implicit or explicit) demands of cross-site collaborative activities to support requirements change related work.

TABLE 3 RCL VERSION 1- CASE1 (16/09/2006)

Change ID	Date	Module	Change Detail	Impacts

Version 1 (Table 3) created in 2006, started off with very basic information about requirements change forms created in the project. The template basically identified the change with a name and date and it had some change impact information e.g. stating which module it would involve and which module it would impact. A trivial description of the change was also provided.

i) Analysis of RCL Version 1-Case 1:

The apparent purpose RCL Version 1 was to view all requested changes in one file (as each change was specified in a separate Word document) perhaps as a means of fulfilling one of many CMMI documentation requirements. (The company was seeking Level 2 CMMI certification at the time).

TABLE 4 RCL VERSION 2 CASE1 (06/04/2009)

Change ID	Date	Module	Change Detail	Impacts
Requested by	Reason for Change		Change Type	When

Version 2 (Table 4) had four columns in addition to the ones found in Version 1. The newly added columns provided more information about the person or stakeholder group requesting the change, rationale for the change, and the phase of the project (after review, during testing, after usage of the application) in which the change was requested.

ii) Analysis of RCL Version 2-Case1:

One of the authors of this paper was involved in the decision to add these four columns into the new template (Version 2, Table 4). These suggestions were mainly based on studies conducted by Nurmuliani and Zowghi such as [25] who advocated their importance. The reason for their inclusion was to make the organization understand why these changes were occurring, and who were the most influential stakeholders suggesting changes in the project. The idea was to help ascertain the types of changes being worked on in the project and to identify stages of the project at which these changes were initiated so that attention could be directed towards areas of concern and possible improvements. Another reason was to enable the organization to perform analysis on requirements change data both during the project and at the end (e.g. in a post mortem review).

TABLE 5 RCL VERSION3 - CASE1 (02/10/2009)

CRF	Date Requested	Reference	Requested By	Total Man Hours
[Approved] Status		Deployment Status		

Version 3, Table 5 of the RCL log introduced in Oct 2009 aimed to cover the whole lifecycle of requirements change request (from inception to deployment). The important factor in this artefact was the inclusion of 'Total Man Hours' to estimate the total effort invested in a particular change.

iii) Analysis of RCL Version3-Case1:

The new RCL template (Version 3, Table 5) was used as a mechanism to manage scope creep and stop both the client and the proxy client from making small change requests to be implemented within the same time period and budget. The developers and the team lead involved valued this artefact as it guarded them from putting in 'non appreciable' effort and helped keep their schedule on track and their 'unpaid' extra hours to the minimum.

At this time, the management was challenged with the inaccuracy of effort estimation in implementing requirement changes and wanted the effort to be documented. This document was then used for the purpose of reporting and negotiating effort with the proxy client and other stakeholders for changes in budget and time etc.

The RCL Version 3, Table 5 template suggested the areas of organizational concern were impact analysis, effort estimation, and the status of development or deployment of the change.

TABLE 6 RCL VERSION 4- CASE1(05/01/2011)

Number	Project	Title	PM	DM	Lead Dev	Total
QAQC	Date Requested	Requested by	Approval Date			Deployment Status
Status	Acceptance Date	SM Cost	Billing Status			Comments

Version 4, Table 6 is the final template of the RCL artefact and the one which is being currently used between the Pakistan and the USA sites. The clear focus in this artefact is the estimate of the actual vs. estimated cost in terms of 'man hours'. The 'billing status' and 'comments' columns were added for the first time.

iv) Analysis of RCL Version4-Case1:

The RCL artefact (Version4, Table 6) has clearly evolved to include those elements that are critical for the organization to record, maintain and be aware of. It has more detail about the referenced project, team roles, change deployment and its billing status. This billing status column is meant for internal use and it records whether all changes for which the development team undertook effort were billed to the client or not. Since the owner and proxy client were at a distant location from this development site, local team members were accountable for their number of hours invested in the project. If their hours were not accounted for to the proxy client, their daily task sheet did not show them being 'productive'. This measure was used to defend team members' and to show their 'productiveness'. The template recorded the effort of all the members involved and not just the development team.

2) Requirements Change Log (Case 2):

As a contrasting set of artefacts, we discuss the alternate evolutionary paths taken by the RCL in the second case.

TABLE 7. RCL TYPE 1- CASE 2(05/01/2011)

Date	Sheet	Cell/Ref	Previous Value	New Value	Reason for Change	by

RCL Type1, Table 7 was a standard change log template with limited information about a requirements change and its reasons. It occurs as the final change summary tab in a multi sheet MS-Excel workbook encapsulating a set of requirements expressed as screenshots, with fields and annotations, and related software module information. It was used for simpler dynamic changes (e.g. demanding back-end logic or processing), rather than those that could be expressed through annotated screenshots.

i) Analysis of RCL Type 1—Case2:

The project officially used a standard document, to record and communicate requirements changes, called a requirements change form. But we did not see that form being implemented for these changes in specifications. Instead multi-worksheet spreadsheets have been predominantly used to communicate and record changes in requirements. The RCL (Type 1, Table 7) captured basic information for changes made to the specification. However it did not include information related to impact analysis, estimated effort or the current status of a particular change request. The artefact analyzed here was developed at the client site. Since some of the changes were trivial and cosmetic in nature, a limited record of changes was considered appropriate.

TABLE 8. RCL TYPE 2-VERSION 2-CASE 2(22/02/2013)

Change no	Sprint	Entity	Tab
Sub Tab	Field Name	Object Security	Live List
App Entity	Function	Workflow	Notes

The RCL Type 2, Table 8 was an alternate to the RCL Type 1Table7, and was used for more complex dynamic changes, rather than those that could be expressed through annotations on requirements screenshots. Again it was instantiated as a final change summary worksheet in a multi-tabbed workbook.

ii) Analysis of RCL Type 2-Version 2—Case2:

It is interesting to note how the RCL Type 2, Table 8 template has been enhanced to overcome some of the perceived deficiencies in the alternate variant of the template. The initial version of RCL type 2 had a structure which did not provide for all user needs. The RCL type-2 template here was slightly changed from the previous version to include a 'Sprint' column to associate a change with a particular sprint; information not captured in the original template.

These spreadsheets were shared across sites; however some local practices were adopted at the client site. The 'Sprint' column was appropriated to meet a variety of additional needs, inconsistent with its naming. It was variously used to describe a) Change status (e.g. confirmed or unconfirmed, b) severity (if it is a bug e.g. essential, helpful, cosmetic fix, etc.) or c) a descriptive explanation of the bug. Similarly the 'Notes' column was used for recording different kinds of information regarding a change. It classified the entry made in the spreadsheet as a bug, new requirement, change, defect, 'note only', 'bug+change' (to aid communication) or an actual note itself. The flexibility of the spreadsheet structure allowed these deviations from preserving strict data integrity to be made.

The positioning of both RCL variants (Type 1, Table 7 and Type 2, Table 8) is also worth noting. As the final sheet in the specification workbook, it suggests that the BAs preferred the efficiency of a unified requirements 'document' in the one workbook, rather than having to update two separate documents and maintain a traceability link between them.

The next section reviews a further artefact; Enhancement on Design Specification (EDS). In contrast to the RCL the metadata in this template remained stable. The evolutions we observed were related major content changes from the design phase onwards, as new enhancements were added dynamically.

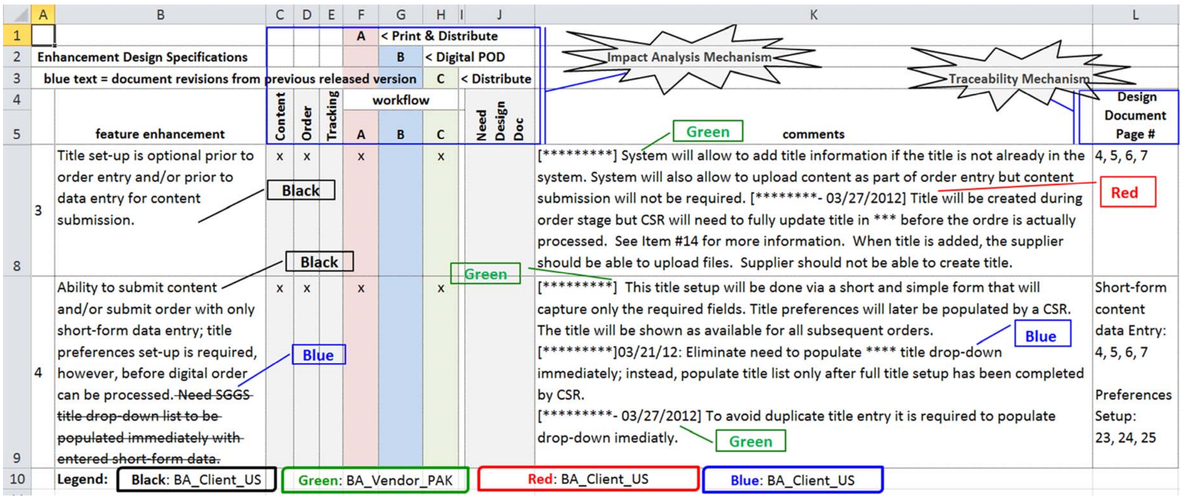
3) Enhancements on Design Specifications (Case 1)

The Enhancement on Design Specifications was used for requirements collaboration between one development site in Pakistan and two client sites in the USA. The primary purpose of the EDS sheet (see Fig 2.) was to record and communicate modifications in or additions to the agreed design specification. Members of client and vendor teams collaboratively employed this artefact for finalizing design enhancements to be implemented in the system in a further series of design review sessions. The EDS was used also as a collaborative artefact to analyze and negotiate changes in design, to develop shared understanding and record impact analysis information. It also acted as a status reporting tool and a discussion list for stakeholders.

The EDS template in Fig. 2 recorded information regarding design changes in a single worksheet file. Asterisks are placed in the text to anonymize sensitive data. Changes were numbered with a brief description identifying the software modules to which these changes belonged. It also incorporated the logic for the sequence of activities that dictated how the workflow would be implemented. The 'Need Design Doc' column indicated whether or not a new design document was needed for this change. The last column was for comments regarding a particular change.

i) Analysis of Enhancements on Design Specifications

The chosen EDS document Version 2 (Fig2.) was a mid-lifecycle-originated artefact initiated by the client. One more column, 'Design Document Page #', was added to those found in Version 1. The evolution in the structure came from the need for the traceability aspect of the specified change, not



	A	B	C	D	E	F	G	H	J	K	L
1						A	< Print & Distribute				
2		Enhancement Design Specifications					B	< Digital POD			
3		blue text = document revisions from previous released version						C	< Distribute		
4			Content	Order	Tracking	workflow			Need Design Doc		
5		feature enhancement				A	B	C		comments	Design Document Page #
8	3	Title set-up is optional prior to order entry and/or prior to data entry for content submission.	x	x		x		x		[*********] System will allow to add title information if the title is not already in the system. System will also allow to upload content as part of order entry but content submission will not be required. [********- 03/27/2012] Title will be created during order stage but CSR will need to fully update title in *** before the ordre is actually processed. See Item #14 for more information. When title is added, the supplier should be able to upload files. Supplier should not be able to create title.	4, 5, 6, 7
9	4	Ability to submit content and/or submit order with only short-form data entry; title preferences set-up is required, however, before digital order can be processed. ~~Need SGGS title drop-down list to be populated immediately with entered short-form data.~~	x	x		x		x		[*********] This title setup will be done via a short and simple form that will capture only the required fields. Title preferences will later be populated by a CSR. The title will be shown as available for all subsequent orders. [*********]03/21/12: Eliminate need to populate **** title drop-down immediately; instead, populate title list only after full title setup has been completed by CSR. [*********- 03/27/2012] To avoid duplicate title entry it is required to populate drop-down imediatly.	Short-form content data Entry: 4, 5, 6, 7 Preferences Setup: 23, 24, 25
10		Legend:	Black: BA_Client_US			Green: BA_Vendor_PAK				Red: BA_Client_US Blue: BA_Client_US	

Fig. 2. Enhancements on Design Specifications Version 2 Case 1

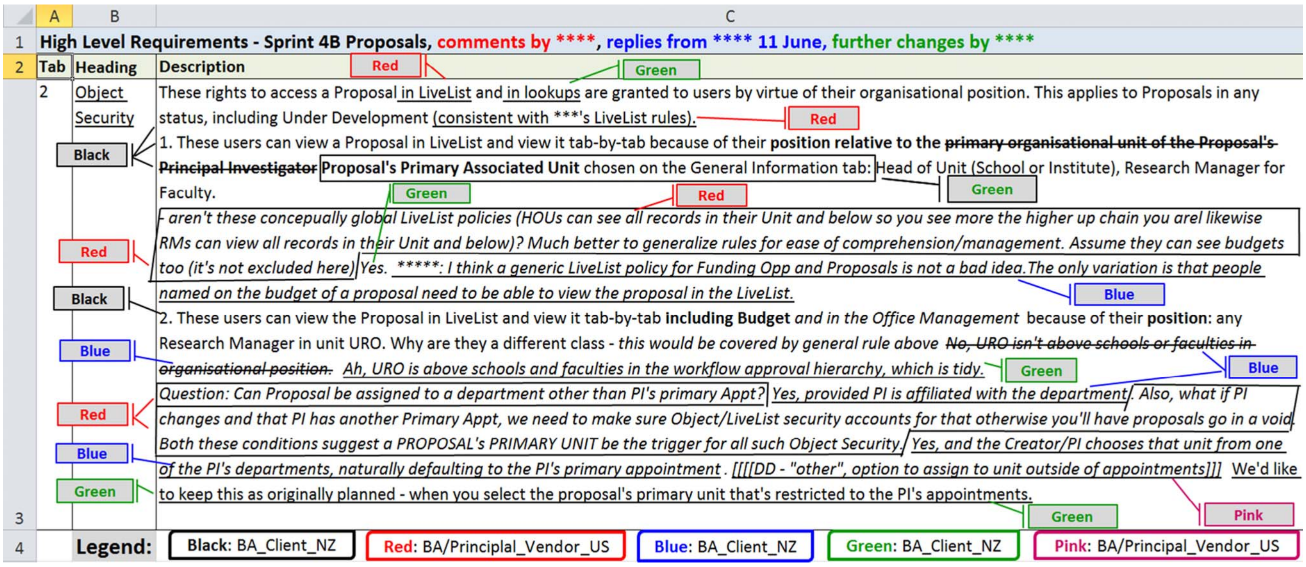
just for simplifying the process of finding relevant information. This eased locating changes in the design document (which was in the order of 80 pages in length). In some later enhancements to this column, we see more and more detail regarding the sections within the identified design page which eased finding the particular section a change referred to.

Notably, this version signified whether or not a further design document was required for this change. Individual cells of this artefact underwent constant evolution, e.g. the 'Comments' column acting as a form of discussion thread, allowed progressive changes. Discussions carried out through this column sometimes resulted in deferring a change or cancelling a request altogether. Colors were used for distinguishing various parties or roles sequentially commenting on the change. There was a date stamp on comments so that the reader might immediately distinguish the latest comments from the older ones. Initially this date-stamping practice was used only by the client but as the EDS artefact evolved the practice was found useful by the vendor and was adopted by both parties. Interestingly the vendor programmatically stamped its company name and date, (following the client's 'good' practice), to identify who had contributed the comments and to distinguish the latest comments from the older ones. So the spreadsheet technology offered not only static columns and rows but also some programmatic support for traceability.

4) High Level Requirements Specifications (Case 2)

The High Level Specification (HLRS) from Case 2, Fig.3 shared some similarities with the EDS, but also had some striking differences. We provide a brief comparison and contrast between the HLRS and the EDS below.

The HLRS was meant for communicating, clarifying, analyzing and negotiating high level requirements among cross functional stakeholders from both client and vendor sites. The artefact again consisted of a multi-worksheet (tabbed) spreadsheet file. These high level requirements were mainly related to the business flow or work logic which needed further clarification.



	A	B	C
1	High Level Requirements - Sprint 4B Proposals, comments by ****, replies from **** 11 June, further changes by ****		
2	Tab	Heading	Description
3	2	Object Security	These rights to access a Proposal in LiveList and in lookups are granted to users by virtue of their organisational position. This applies to Proposals in any status, including Under Development (consistent with ***'s LiveList rules). 1. These users can view a Proposal in LiveList and view it tab-by-tab because of their **position relative to the** ~~**primary organisational unit of the Proposal's Principal Investigator**~~ **Proposal's Primary Associated Unit** chosen on the General Information tab: Head of Unit (School or Institute), Research Manager for Faculty. - aren't these concepually global LiveList policies (HOUs can see all records in their Unit and below so you see more the higher up chain you arel likewise RMs can view all records in their Unit and below)? Much better to generalize rules for ease of comprehension/management. Assume they can see budgets too (it's not excluded here) Yes. *****: I think a generic LiveList policy for Funding Opp and Proposals is not a bad idea.The only variation is that people named on the budget of a proposal need to be able to view the proposal in the LiveList. 2. These users can view the Proposal in LiveList and view it tab-by-tab **including Budget** *and in the Office Management* because of their **position**: any Research Manager in unit URO. Why are they a different class - *this would be covered by general rule above* ~~No, URO isn't above schools or faculties in organisational position.~~ *Ah, URO is above schools and faculties in the workflow approval hierarchy, which is tidy.* *Question: Can Proposal be assigned to a department other than PI's primary Appt? Yes, provided PI is affiliated with the department. Also, what if PI changes and that PI has another Primary Appt, we need to make sure Object/LiveList security accounts for that otherwise you'll have proposals go in a void. Both these conditions suggest a PROPOSAL's PRIMARY UNIT be the trigger for all such Object Security. Yes, and the Creator/PI chooses that unit from one of the PI's departments, naturally defaulting to the PI's primary appointment . [[[[DD - "other", option to assign to unit outside of appointments]]]* We'd like to keep this as originally planned - when you select the proposal's primary unit that's restricted to the PI's appointments.
4		**Legend:**	Black: BA_Client_NZ Red: BA/Principal_Vendor_US Blue: BA_Client_NZ Green: BA_Client_NZ Pink: BA/Principal_Vendor_US

Fig. 3. High Level Requirements Specifications Case 2

The common features of both EDS and HLRS include their 'content-evolving' nature, their use of color to differentiate human identity or role, space, location, time as both snapshot and duration through traceability. The content of this document underwent a progressive set of changes, revisions and evolutions over time, but was maintained as the current live version. Both artefacts acted as 'discussion lists' consisting of comments and updates from the team members involved in the activities such as specifying, analyzing, negotiating, verifying and validating requirements.

The differentiating factor or uniqueness comes from the fact that the HLRS was a 'meta-level' requirements document which was meant for understanding the workflow or logic of various aspects of the software. It served as a 'sense-making' document for all the activities of RE (elicitation, specification, analysis, negotiation, verification, validation, management and traceability) [26], [3] providing a current state overview so that both client and vendor had a consistent pre-implementation view of the software to be delivered This practice and its supporting artefact had great utility in the GSD setting, in pragmatically documenting evolving requirements at their current state to aid shared understanding across sites.

V. Findings

The study reveals that software practices, the associated artefacts and the collaborative technology that is enabling coordination are all closely implicated and co-evolve in the process of global collaboration. We find through close analysis of embedded collaborative artefacts, that evolution of GRE practice occurs through incremental local improvements, which can gradually and progressively result in broader scale global changes in practice, akin to the processes reported by Orlikowski [27]. Artefacts such as spreadsheets provide surprisingly illustrative examples, in which even a single cell over time may tell a rich story. We observe that through local and global team practices, spreadsheet embedded artefacts such as change logs co-evolve through collaboration over requirements. We also note that spreadsheets fit within a wider repertoire of CT usage in GRE, and that they can serve as an effective bridging mechanism between core development teams and other stakeholder groups.

A. Case 1: Spreadsheets & Local and Global Requirements Change Management Practice

The evolution of the RCL in case 1 illustrates the situated evolution of the local and global dimensions of RCM practice. The initial versions of the RCL originated from a CMMI certification initiative on behalf of the vendor, and could be seen as resulting from the cultural imperatives of a global setting. Arguably we see a form of "global culture" and inter-"institutional culture" [28] in operation, wherein global capitalism imposes demands: whether in response to the client's demands for improved predictability and control of product delivery , or the vendor's marketing driven desire to position themselves as a provider of high quality services using CMMI certification as a global quality mark. The RCL represented this externally imposed reality in its initial form, where it merely recorded changes of such a magnitude that they warranted raising a formal change request [29] and initiating budget negotiations with the client. Yet over time, the RCL evolved to incorporate local and global needs. In its later forms we see an active internalization of software process improvement by the vendor team, through capturing of changes in more detail and with a lower level of granularity. This served the purpose of better management of scope creep, of protecting the developers who could explain what [hitherto unrecorded] work they had spent their time on, and of enabling additional negotiation over some clearly billable aspects of the work resulting from necessary but unexpected smaller changes.

So we see the change log evolve from a mechanism to track requirements sizeable enough to warrant additional dollar cost negotiations to very much a professional vehicle for supporting local practices and managing changes of lower granularity. Thus we see "professional culture" [28] coming to the fore, and the focus shifting from external and globally imposed processes to more internal needs driven, pragmatic and locally appropriated practices. From a series of micro-level incremental changes, we see a shift in the power balance, a lift in the professional behavior at the vendor site, increased cross-site visibility of all changes and a global accommodation of new requirements negotiation practices resulting in the possibility of fairer payment for actual work done.

B. Case 2: Spreadsheets - GRE Support in a CT Repertoire

The EDS and HLRS by contrast tell their own rather different story. We see these artefacts existing within a collaborative technology stack, which serves multiple purposes in supporting global requirements change management [35]. The collaborative technologies within which these artefacts were embedded provided the base layer of the stack. CTs such as Lync and GoToMeeting supported active screen sharing of spreadsheets and enabled discussion and synchronous editing in global meetings. Asynchronous modes of spreadsheet use were supported in Case 1 through emailing the EDS files between sites, and in Case 2 as a web based artefact repository which allowed spreadsheets to be sharable as attached files, with embedded links to the bug-tracking feature within JIRA.

The spreadsheets at the next layer of the stack, were themselves layered, through applications, workbook files, sets of individual tabbed requirements worksheets, and requirements summary worksheets, which naturally subsumed individual *cells* containing content. The requirement worksheets supported embedded screenshots as images, with annotations created using MS-Paint, (as a simple readily available and convenient graphical editor), and active links to the bug-tracker system for traceability. Spreadsheets in both cases served as a CT with the ability to flexibly capture a variety of tailorable artefact 'genres'[30] employed in GRE. Within these spreadsheet artefacts we see the complexities of GRE captured, even at the level of the individual cell. How the several dimensions are implicated in a single cell can be illustrated through a brief review of their roles in each artefact. Time is expressed as both the *current state* of understanding of requirements expressed in the HLRS worksheet, where the date of production is recorded in the first cell, and as *duration* represented by the *date stamping* and distinct coloring of each *sequential* entry within the comments cell of the EDS.

Space and location likewise are represented through the *coloring* of each entry, by *name-date* stamped *organizational* members at the same or different *locations*, through *links* to other tabbed worksheets and cells, through *mapping* of individual requirements within screenshots identifying *location* of features and changes, and through tabbed worksheets providing a logical *spatial map* of groups of requirements. *Organizational* entities and implicit *roles* are apparent in both artefacts, in which cell entries in the EDS were programmatically prefixed by *name-date* stamps for the *organization* posting the entry. In contrast *names* of contributors to the HLRS were color coded for each entry, with the *location* being implicit. The activities of GRE were supported through the communication; coordination and collaboration capabilities afforded by this combination of CTs and embedded artefacts.

These GRE activities can be seen at play within a single HLRS spreadsheet cell (cf Fig.3 c3–'Description'). Within one dialogue in the cell relating to a specific requirement we see **elicitation** (e.g. *can proposal be assigned to a dept. other than PI's primary appt? [*Response*]* Yes provided PI is affiliated); **analysis** *(...much better to generalize rules..*); **negotiation** ("DD- *other, option to assign to unit outside of appointments"*[?], [Response] *We'd like to keep this as originally planned- when you select the proposal's primary unit that's restricted to the PIs appointments*); **specification** (the cell contents themselves represent a high level specification for a designated requirement section – e.g. (*object security*); **verification** (*Why are they a different class - this would be covered by general rule above [*Response*]No, URO isn't above schools or faculties in organizational position. Ah, URO is above schools and faculties in the workflow approval hierarchy, which is tidy*). Complementing these activities was the support provided by the artefacts for the (exacerbated) traceability problems in GSD [31].

The evolution of the requirements, their originators at each site, the record of the negotiations, the linking between tabbed requirements worksheets to cells of interest, and the current state snapshot of the HLRS all served to satisfy the need for global requirements traceability.

C. Spreadsheets as Bridging Mechanisms in GRE

The ubiquity of spreadsheet usage in smaller software teams is an interesting phenomenon. In this study we saw spreadsheets within a large repertoire of CTs forming a bridging function between the core development team and the peripheral but important stakeholders such as business analysts, subject matter experts and project managers at other sites. They served a summarizing and communicative function across these groups, supporting all the activities of GRE. Spreadsheets with just enough programmability and collaborative capabilities provided a convenient, adaptable and cost effective alternative as opposed to costly installations with heavy infrastructure implications and learning curves for managing requirements While existing tools are becoming more collaboration friendly, none of the existing tools provide support for all the collaboration needs demanded in GSD [2]. Our analysis suggests that special purpose CTs in multi-site collaboration (e.g. ticket tracking systems, IDEs etc.) are complemented by more flexible and readily available CTs such as spreadsheets which cater for the collaboration needs of a wider set of stakeholders (beyond the core development teams).

D. Limitations of this study

While in this study requirements numbered in the high tens or at most in excess of a hundred, the problem with spreadsheet use for large complex software systems is their scalability. For example maintaining a traceability matrix manually poses challenges for projects with more than 2000 requirements. While one reviewer thought the workflows in this study were 'atypical' for GSD, the findings do arise from empirical data and may well be more reflective of small software companies' practices and technology use. There is the valid counter argument that undue use of Excel tools can proliferate isolated and contradictory data pools, which is hardly a desirable practice. These limitations warrant further study.

Space has precluded a fuller elaboration of the wider comparative case study in which these data collection and analysis methods have been applied, but the findings reported here have drawn upon the literature, and are empirically based. They are triangulated by the primary author's observations of practitioners in the field, and complementary interview data. The practitioners in the study employed a range of other collaborative technologies which we have not addressed here. Nor have we covered any areas of identified need or team support by contemporary cloud based technology alternatives.

E. Recommendations for Practice and Research

From this extended perspective, it is difficult to define the notion of a 'team' in GSD environments and thus to decide on the generic collaboration support that suits every project context. However the following recommendations for practice can be considered while planning a repertoire of CT support in GSD. Managers should look beyond the core development team to the CT support needs of cross functional stakeholder groups. This may involve consciously designing the most effective role for spreadsheets as bridging mechanisms between technical and non-technical stakeholder groups in the wider CT repertoire. The activities of technology use mediation [32], such as *establishment* and *reinforcement* of patterns of use for a full CT repertoire, need conscious attention. While it is more demanding to set up a suite of integrated tools, the lesser demand for set up and tailoring of spreadsheets still exists, although they can more readily evolve and offer the ability to 'fail fast' because they can easily be redesigned or tweaked to fit the needs of evolving practice. Also how spreadsheets might complement larger and more integrated tool sets without proliferating isolated and duplicate data pools, is an open question.

For researchers this study raises several questions. As noted above scalability and the limits of spreadsheets need to be tested. Yet in this project, the spreadsheets served to abstract and summarize in one location the current state of play for requirements changes across sites. While we report on two GSD cases, they may reflect wider patterns of use in small to medium scale GSD projects. Given that voluminous requirements documents often fail to achieve their goal of expressing common understandings, research is warranted into the value of this summarizing, communicative and interfacing role of spreadsheets for smaller software organizations. But such investigation cannot be divorced from the mutually shaping role of evolving GRE practices expressed through spreadsheets in this study.

VI. CONCLUSION

In this study we have reviewed two small software company GSD settings, noting that both the vendor and client sites appropriated spreadsheet technology in quite specific ways, for use locally and for bridging across sites. We observed that the role of spreadsheets as collaborative technologies was critical in enabling the collaboration demanded in global requirements change management (GRCM). Our close study over time revealed how team members' practices co-evolved with the spreadsheet artefacts involved in the process of managing requirements change. Through the micro-level evolution of a spreadsheet's structure or content we exposed how several layers of culture were implicated in a major shift of local and global practice. We further noted how spreadsheets and their contents fitted within a broader repertoire of collaborative technologies. The bridging role they played appeared vital to constructive requirements change management between distributed stakeholders and the core development team.

REFERENCES

[1] J. Whitehead, I. Mistrík, J. Grundy, and A. van der Hoek, "Collaborative software engineering: concepts and techniques," in Collaborative Software Engineering, ed: Springer, 2010, pp. 1-30.

[2] F. Lanubile, C. Ebert, R. Prikladnicki, and A. Vizcaíno, "Collaboration tools for global software engineering," Software, IEEE, vol. 27, pp. 52-55, 2010.

[3] D. Damian, "Stakeholders in global requirements engineering: Lessons learned from practice," Software, IEEE, vol. 24, pp. 21-27, 2007.

[4] V. Sinha, B. Sengupta, and S. Chandra, "Enabling collaboration in distributed requirements management," Software, IEEE, vol. 23, pp. 52-61, 2006.

[5] D. A. Norman, The design of everyday things: Basic books, 2002.

[6] M. Kuhrmann, D. M. Fernández, and M. Grober, "Towards Artifact Models as Process Interfaces in Distributed Software Projects," in Global Software Engineering (ICGSE), 2013 IEEE 8th International Conference on, 2013, pp. 11-20.

[7] M. Lang and J. Duggan, "A tool to support collaborative software requirements management," Requirements Engineering, vol. 6, pp. 161-172, 2001.

[8] P. Laurent, "Globally Distributed Requirements Engineering," in Proceedings of the 2010 5th IEEE International Conference on Global Software Engineering, 2010, pp. 361-362.

[9] D. Damian, J. Chisan, L. Vaidyanathasamy, and Y. Pal, "Requirements engineering and downstream software development: Findings from a case study," Empirical Software Engineering, vol. 10, pp. 255-283, 2005.

[10] M. Raatikainen, T. Mannisto, T. Tommila, and J. Valkonen, "Challenges of requirements engineering—A case study in nuclear energy domain," 2011, pp. 253-258.

[11] J. D. Herbsleb, "Global software engineering: The future of socio-technical coordination," in 2007 Future of Software Engineering, 2007, pp. 188-198.

[12] P. Tell and M. A. Babar, "Activity Theory Applied to Global Software Engineering: Theoretical Foundations and Implications for Tool Builders," in Global Software Engineering (ICGSE), 2012 IEEE Seventh International Conference on, 2012, pp. 21-30.

[13] Å. Cajander, M. Daniels, M. Cullhed, T. Clear, R. McDermott, and C. Laxer, "Categorizing How Students Use Collaborative Technologies in a Globally Distributed Project," in Proc. 42nd ASEE/IEEE Frontiers in Education Conference, 2012, pp. 1325-1330.

[14] G. DeSanctis and M. S. Poole, "Capturing the complexity in advanced technology use: Adaptive structuration theory," Organization science, pp. 121-147, 1994.

[15] W. J. Orlikowski and C. S. Iacono, "Research commentary: Desperately seeking the "IT" in IT research—A call to theorizing the IT artifact," Information Systems Research, vol. 12, pp. 121-134, 2001.

[16] T. W. Malone, K.-Y. Lai, and C. Fry, "Experiments with Oval: a radically tailorable tool for cooperative work," ACM Transactions on Information Systems (TOIS), vol. 13, pp. 177-205, 1995.

[17] B. A. Nardi and J. R. Miller, "Twinkling lights and nested loops: distributed problem solving and spreadsheet development," International Journal of Man-Machine Studies, vol. 34, pp. 161-184, 1991.

[18] P. Tell and M. A. Babar, "A Systematic Mapping Study of Tools for Distributed Software Development Teams," Technical Report TR-2012-1612012.

[19] D. E. Damian and D. Zowghi, "RE challenges in multi-site software development organisations," Requirements Engineering, vol. 8, pp. 149-160, 2003.

[20] C. Ebert and J. De Man, "Requirements uncertainty: influencing factors and concrete improvements," in Proceedings of the 27th international conference on Software engineering, 2005, pp. 553-560.

[21] R. Prikladnicki, S. Marczak, E. Carmel, and C. Ebert, "Technologies to Support Collaboration across Time Zones," IEEE software, vol. 29, 2012.

[22] D. Šmite, C. Wohlin, T. Gorschek, and R. Feldt, "Empirical evidence in global software engineering: a systematic review," Empirical Software Engineering, vol. 15, pp. 91-118, 2010.

[23] K. Petersen and C. Wohlin, "Context in industrial software engineering research," in Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement, 2009, pp. 401-404.

[24] J. M. Bhat, M. Gupta, and S. N. Murthy, "Overcoming requirements engineering challenges: Lessons from offshore outsourcing," Software, IEEE, vol. 23, pp. 38-44, 2006.

[25] N. Nurmuliani, D. Zowghi, and S. P. Williams, "Requirements volatility and its impact on change effort: Evidence-based research in software development projects," in Proceedings of the Eleventh Australian Workshop on Requirements Engineering, 2006.

[26] I. Sommerville, Software Engineering: Pearson/Addison-Wesley, 2011.

[27] W. J. Orlikowski, "Improvising organizational transformation over time: A situated change perspective," Information systems research, vol. 7, pp. 63-92, 1996.

[28] T. Clear, "Exploring the notion of'cultural fit'in global virtual collaborations," ACM Inroads, vol. 1, pp. 58-65, 2010.

[29] W. Hussain and T. Clear, "GRCM: a model for Global Requirements Change Management," presented at the 2nd International Requirements Engineering Efficiency Workshop (REEW 2012), Essen, Germany, 2012.

[30] J. Yates, W. J. Orlikowski, and K. Okamura, "Explicit and implicit structuring of genres in electronic communication: Reinforcement and change of social interaction," Organization science, vol. 10, pp. 83-103, 1999.

[31] J. Cleland-Huang, C. K. Chang, and M. Christensen, "Event-based traceability for managing evolutionary change," Software Engineering, IEEE Transactions on, vol. 29, pp. 796-810, 2003.

[32] T. Clear and S. G. MacDonell, "Understanding technology use in global virtual teams: Research methodologies and methods," Information and Software Technology, vol. 53, pp. 994-1011, September 2011.