Requirements Eng (2017) 22:191–214
DOI 10.1007/s00766-015-0240-4

ORIGINAL ARTICLE

A method of software requirements specification and validation for global software development

Naveed Ali[1] · **Richard Lai**[1]

Received: 20 January 2015 / Accepted: 28 October 2015 / Published online: 19 November 2015

Abstract Global software development (GSD), where software teams are located in different parts of the world, has become increasingly popular. To devise a high-quality software requirements specification (SRS), effective communication and collaboration between stakeholders are necessary for GSD. However, geographical distance, cultural diversity, differences in time zones and language barriers create difficulties for stakeholders in engaging in effective collaboration. Taking into consideration the factors involved in GSD, previous research showed that the ways by which requirements are documented and validated for collocated software development projects cannot be used effectively for GSD. In this paper, we present a method of GSD requirements specification and validation. Our method begins with generating a requirements graph to understand details of the software requirements with respect to different GSD sites. The information obtained from a requirements graph is to be contained in a requirements specification document, and then be circulated between different GSD sites for reviewing, updating and finalizing its content. Finally, the requirements contained in the specification document are to be validated by generating and comparing validation matrices at different GSD sites. Past researchers used student groups in a university environment to play the roles of stakeholders in experiments in GSD studies. We therefore validate our method by applying it to a case study of an online shopping system, where the roles of stakeholders were played by a group of students.

Keywords Software requirements specification · Global software development · Distributed teams · Requirements validation

1 Introduction

Global software development (GSD) is multi-site software development with software teams scattered across different places around the world [3, 10]. Despite the promising benefits of the lower costs of software development and access to international talent [8, 18, 25], GSD has introduced challenges for stakeholders which are not present in software projects developed in the same location (collocated projects) [9, 14, 38]. Due to development teams being in different geographical locations, differences in cultures, time zones and knowledge management practices adversely impact communication and coordination processes [1, 15, 19, 31]. Consequently, the frequency of communication, coordination and trust between the development teams decreases [2, 13]. In addition, dissimilar processes of software development, differing technical capabilities of remotely distributed team members, and the low visibility of development work carried out at different sites create additional challenges for stakeholders to tackle. Of the major challenges to successful GSD, devising a quality SRS is of greater significance.

Achieving customer satisfaction on delivered services is one of the primarily goals of every business. Of the many factors which assist stakeholders to achieve customer satisfaction, the quality of SRS plays an important role. When

✉ Richard Lai
r.lai@latrobe.edu.au

Naveed Ali
n.ali@latrobe.edu.au

[1] Department of Computer Science and Information Technology, La Trobe University, Melbourne, VIC 3086, Australia

requirements specifications are badly written, development teams often face difficulties in obtaining the required knowledge at the right time. In collocated environments, this issue can be resolved by engaging in face-to-face communication [16, 34]; however, the social, lingual, geographical and cultural differences in GSD exacerbate the problem by introducing communication pauses and delays. Consequently, many software projects fail to complete within the allocated resources, resulting in a deterioration in the relationship between customers and suppliers [11, 26, 27, 32, 35].

The issues which introduce challenges for distributed development teams in preparing and later validating the SRS document are as follows: (i) the use of dissimilar vocabulary and terminologies: development teams in different locations use different keywords to represent a similar type of requirement. As a result, requirements are often misinterpreted by remote development sites [5, 7, 24]; (ii) difficulties in knowledge exchange and transfer: the presence of geographical boundaries, different vocabularies and dissimilar standards create difficulties for development teams in exchanging and transferring project knowledge between remote development sites [29]; (iii) difficulties in handling requirements specifications: the use of different requirements specification standards brings additional challenges for development teams in handling and processing SRS documents [29, 37]; and (iv) requirements validation: the influence of culture, time zones, knowledge management and communication features of GSD also affect the requirements validation activities across geographical boundaries. As a result, development teams face difficulties in validating the contents of SRS [12, 38].

The above-mentioned challenges are not adequately addressed by the conventional methods of requirements specification and validation. In addition, the international standards of SRS do not cover GSD aspects [17], although these are marginally covered in a few textbooks [6, 36]. Thus, in this paper, we present a method of software requirements specification and validation for GSD projects. Our method facilitates stakeholders in accomplishing the following activities: (i) the systematic organization of requirements via a requirements graph helps GSD teams understand software requirements with respect to different time zones and distance, and the GSD sites at which the requirement(s) are developed; (ii) the preparation of a requirements specification document with cooperation from the members of different GSD sites; and (iii) obtaining assurance that the requirements written in the SRS document are consistent and meet the needs of stakeholders in a globally distributed environment.

Past researchers used student groups in a university environment to play the roles of stakeholders in experiments in GSD studies [4, 21, 30, 35]. Likewise, we validate our method by applying it to a case study of an online shopping system, where the roles of requirements engineers, project analysts and designers were played by a group of students. Throughout the paper, we refer to these groups of people as "stakeholders".

2 Our requirements specification and validation method

Our method is divided into five stages: (i) generation of requirements graph; (ii) preparation of SRS document; (iii) reviewing, updating and finalizing the SRS document at different GSD sites; (iv) requirements validation at different GSD sites; and (v) requirements validation between different GSD sites. The process begins with organizing the software requirements by generating a detailed requirements graph for them in stage 1. As a result, the requirements present in the requirements graph will be systematically organized with respect to the main and sub-requirements, and the GSD sites at which the requirement(s) are developed. In stage 2, the information obtained from stage 1 will be recorded in a requirements specification document and then will be circulated between different GSD sites for review, update and finalization of the contents of the specification document in stage 3. Finally, the requirements written in the specification document will be validated by generating and comparing validation matrices at different GSD sites in stages 4 and 5, respectively. The method as depicted in Fig. 1 is explained in detail in the following subsections.

2.1 Stage 1: Generation of requirements graph

To facilitate GSD teams in understanding software requirements with respect to different time zones and distance, and the GSD sites at which the requirement(s) are developed, a requirements graph will be generated in two steps by using the concept of requirements ontology. The basic definition of requirement ontology is adopted from Lee and Zhao [20] who used ontological principles to extract domain requirements by considering mutual exclusion as the predicted relationship. In real-life scenarios, cross-cutting (also called intersecting) requirements often exist which are not covered in their work. To incorporate the feature of cross-cutting requirements and to add details on GSD projects, we extend their work by defining necessary postulates for them. Ontology is a six-element set consisting of concepts, relationships, postulates, locations and time zones of GSD teams. We deal mainly with finite space (resulting in finite sets); therefore, mentioning it as an element in the definition of

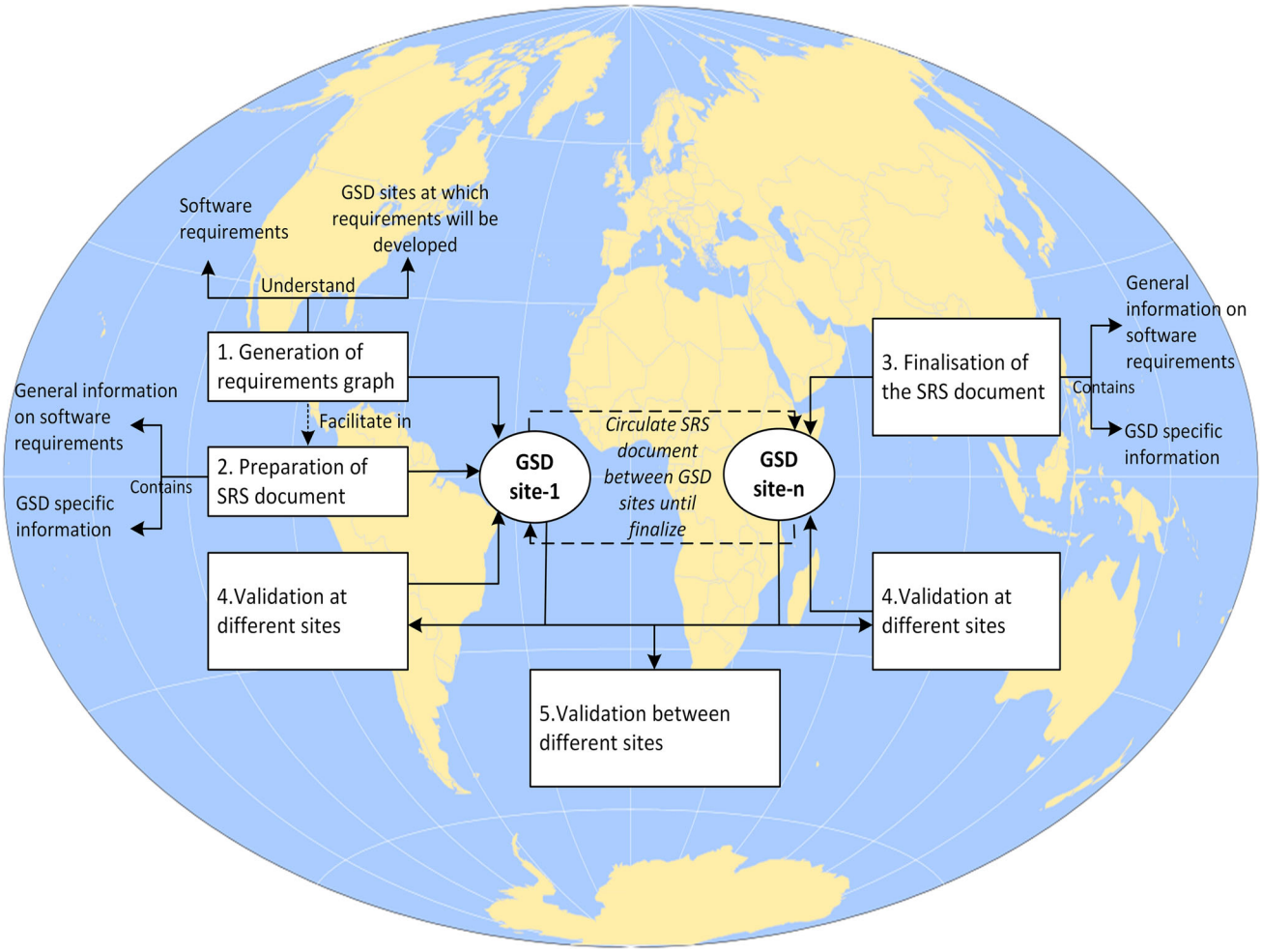

Fig. 1 Our method of software requirements specification and validation for GSD

requirements ontology is not important. Thus, ONT = (-CON, REL-CON, POST, REL-POST, GLOC, GTZONE), where CON is the collection of concepts/requirements, REL-CON is the relationship between concepts, POST is the set of rules for CON and REL, REL-POST is the relationship exists in postulates (REL1-POST ϵ REL-POST), GLOC = location of GSD team, and GTZONE = time zone of GSD team. We consider four different relationships between concepts: association (i.e. pre/post conditions), composition (i.e. part/whole), mutual exclusion and intersection. The postulates (POST) and the relationships expressed in postulates (REL-POST) are defined below:

Postulate 1: For all C_{ON1}, C_{ON2} ϵ C_{ON}, a one-to-one relationship (R_{EL}) exists between them, only if exactly one C_{ON1} relates with exactly one C_{ON2} (i.e. direct functional), or vice versa.

Postulate 2: For all C_{ON1}, C_{ON2}, C_{ON3}… ϵ C_{ON}, a many-to-one relationship (R_{EL}) exists between them, only if multiple classes (C_{ON1}, C_{ON2}, C_{ON3}…) correspond with exactly one class (C_{ON1})

Postulate 3: For all C_{ON1}, C_{ON2}, C_{ON3}…ϵ C_{ON}, a many-to-many relationship (R_{EL}) exists between them, only if multiple classes (C_{ON1}, C_{ON2}, C_{ON3}…) correspond with other multiple classes (C_{ON1}, C_{ON2}, C_{ON3}…)

Postulate 4: For all C_{ON1} and C_{ON2} ϵ C_{ON}, a symmetrical relationship (R_{EL}) exists between them, only if

$$C_{ON1}(R_{EL})C_{ON2} \Rightarrow C_{ON2}(R_{EL})C_{ON1}$$

Or

$$R_{EL}(C_{ON1}, C_{ON2}) = R_{EL}(C_{ON2}, C_{ON1})$$

Postulate 5: For all C_{ON1}, C_{ON2}, C_{ON3} ϵ C_{ON}, a transitive relationship (R_{EL}) exists between them, only if

$$C_{ON1}(R_{EL})C_{ON2}, C_{ON2}(R_{EL})C_{ON3} \Rightarrow C_{ON1}(R_{EL})C_{ON3}$$

Or

$$R_{EL}(C_{ON1}, C_{ON2}), R_{EL}(C_{ON2}, C_{ON3}) \Rightarrow R_{EL}(C_{ON1}, C_{ON3})$$

Postulate 6: For all C_{ON1} and $C_{ON3} \in C_{ON\text{-}A}$, and $C_{ON2} \in C_{ON\text{-}B}$, an injective relationship exists between them, only if

$$\{C_{ON1}(R_{EL})C_{ON2}\} \quad \text{and} \quad \{C_{ON3}(R_{EL})C_{ON2}\} \Rightarrow C_{ON1} = C_{ON2}$$

Postulate 7: For $C_{ON1} \in C_{ON}$, a reflexive relationship (R_{EL}) exists between them, only if

$$C_{ON1}(R_{EL})C_{ON1}$$

Postulate 8: For all C_{ON1}, C_{ON2}, $C_{ON3} \in C_{ON}$, a composite relationship (R_{EL}) exists between them, only if

$$F(C_{ON1}) = {}_{i=1}\cap^{2} C_{ONi}$$
$$G(C_{ONi}) = {}_{i=1}\cap^{2} C_{ONij}$$
$$(F(R_{EL})G(C_{ONi})) = F(G(C_{ONi})) = F\left({}_{i=1}\cap^{2} C_{ONij}\right) = {}_{i=1}\cap^{2}\,{}_{j=1}\cap^{2} C_{ONij}$$

Postulate 9: For all $C_{ON1}, C_{ON2}, C_{ON3} \in C_{ON}$, a distributive relationship (R_{EL}) exists between them, only if

$$\{C_{ON1}(R_{EL\text{-}A})(C_{ON2}(R_{EL\text{-}B})C_{ON3})\} = \{(C_{ON1}(R_{EL\text{-}A})C_{ON2})(R_{EL\text{-}B})(C_{ON1}(R_{EL\text{-}A})C_{ON3})\};$$

Here, $R_{EL\text{-}A}$ = Intersection operation $(\cap)$, and $R_{EL\text{-}B}$ = Union operation (U)

Or

$$\{C_{ON1} \cap (C_{ON2} \cup C_{ON3})\} = \{(C_{ON1} \cap C_{ON2}) \cup (C_{ON1} \cap C_{ON3})\}$$

Postulate 10: For all C_{ON1}, C_{ON2}, $C_{ON3} \in C_{ON}$, an associative relationship (R_{EL}) exists between them, only if

$$\{C_{ON1}(R_{EL})(C_{ON2}(R_{EL})C_{ON3})\} = \{(C_{ON1}(R_{EL})C_{ON2})(R_{EL})C_{ON3})\}$$

2.1.1 Step 1: Identification of main requirements

The graph generation process begins with the identification of the main requirements and the locations and time zones of GSD sites, at which these requirements could possibly be developed, from the information gathered during requirements elicitation and analysis. Thus, we can say that:

$$\text{Project} = \{R_{EQUIREMENTS}, R_{EL\text{-}CON}, P_{RO\text{-}DOMAIN}, G_{LOC}, G_{TZONE}\}$$

where $R_{EQUIREMENTS}$ = main functional and non-functional requirements, $R_{EL\text{-}CON}$ = part/whole relationship, $P_{RO\text{-}DOMAIN}$ = problem domain that is finite in nature, G_{LOC} = location of GSD team, G_{TZONE} = time zones of GSD team.

2.1.2 Step 2: Decomposition of the main requirements

The requirements identified in step 1 are decomposed into sub-requirements, and appropriate relationships are established between them (refer to Fig. 2). The process of requirements decomposition is conducted on the basis of ontological principles and continues until no further division is possible. Thus, we can say that:

$$R_{EQUIREMENTS} = \{C_{ON}, R_{EL\text{-}CON}, P_{OST}, R_{EL\text{-}POST}, G_{LOC}, G_{TZONE}\}$$

where C_{ON} = collection of concepts/requirements, $R_{EL\text{-}CON}$ = relationship between concepts, which can be association (i.e. pre/post conditions), composition (i.e. part/whole), mutual exclusion and intersection. P_{OST} = set of rules for C_{ON} and R_{EL}, $R_{EL\text{-}POST}$ = relationship exists in postulates, which can be one-to-one, many-to-one, many-to-many, symmetrical, transitive, injective, reflexive, composite, distributive and associative relationships, G_{LOC} = location of GSD team, G_{TZONE} = time zones of GSD team.

Figure 2 presents the structure of a requirements graph (G) which contains a collection of nodes (N) and edges (E), such that G = (N, E). In graph G, each node provides information on a particular requirement, which could be either a main or sub-requirement, and the locations and time zones of GSD teams at which the requirements are developed. However, each edge represents a relationship between two requirements. The information therefore obtained from the requirements graph will be recorded in a tabular format, where a numeric value will be assigned to each requirement for tracking purposes.

2.2 Stage 2: Preparation of SRS document

After generating the requirements graph, details on the technical and non-technical aspects of the software project will be recorded in a requirements document, using the prototype for software requirements specification. The prototype presented is based on [17] and later modified to cover GSD aspects. The following information is new in our requirements document: (i) information about the GSD sites where the requirements are developed; (ii) the locations and time zones of each GSD team; (iii) the list of communication modes, mechanisms and tools used by each GSD team for communication purposes; (iv) details about the development teams responsible for the development of certain aspects of a GSD project; (v) the list of directly and indirectly affected project modules; and (vi) the non-functional requirements which are affected due to the

lingual, temporal, cultural and geographical aspects of GSD. A detailed description of the elements of the prototype is presented in Table 1.

2.3 Stage 3: Finalization of SRS document

Once the initial version of the SRS document is generated, it will be communicated along with the requirements graph to other development sites to review its content and later update different elements of the requirements graph and SRS, if required. This process will be continued until the contents of SRS are finalized between the different GSD sites.

2.4 Stage 4: Validation at different sites

Lobo [22] provides a checklist of properties which should be satisfied to validate the requirements written in the SRS document. These properties include: acceptability—every requirement must be acceptable to the stakeholders responsible for it; ambiguity—every requirement must have only one interpretation; completeness—the requirements written in the SRS document should be complete; verifiability—every requirement must have an associated acceptance criterion to verify them after implementing the requirement; understandable—every requirement must be clear to all groups of stakeholders.

Two matrices A and B of m × n elements will be generated at different GSD sites, where m indicates the total number of requirements which need to be validated and n indicates the total number of properties which will be used to validate the requirements (refer to Eq. 1). Depending on the results of the validation process, these matrices will be filled with values "1" and "0", where 1 means "property is satisfied" and 0 "property is not satisfied".

$$A = \begin{array}{c} \\ m1 \\ m2 \\ \\ mm \end{array} \begin{array}{c} \begin{array}{cccc} n1 & n2 & n3 & nn \end{array} \\ \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & a_{m3} & \cdots & a_{mn} \end{bmatrix} \end{array} \quad B = \begin{array}{c} \\ m1 \\ m2 \\ \\ mm \end{array} \begin{array}{c} \begin{array}{cccc} n1 & n2 & n3 & nn \end{array} \\ \begin{bmatrix} b_{11} & b_{12} & b_{13} & \cdots & b_{1n} \\ b_{21} & b_{22} & b_{23} & \cdots & b_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_{m1} & b_{m2} & b_{m3} & \cdots & b_{mn} \end{bmatrix} \end{array} \tag{1}$$

2.5 Stage 5: Validation between different sites

The matrices generated in stage 4 will be compared by computing the correlation coefficient (CCp_{AB}) for them (refer to Eq. 2) [28].

$$CC_{p^{AB}} = \frac{\sum (a-\bar{a})(b-\bar{b})}{\sqrt{\sum (a-\bar{a})^2 (b-\bar{b})^2}} \tag{2}$$

In Eq. 2, A and B are the requirements property matrices generated at different GSD sites, a is the list of elements in the matrix A, $\bar{a}$ is the mean of all elements in matrix A, and the same for b and $\bar{b}$. The result of the correlation coefficient falls between -1 and 1, where results close to -1 indicate that a significant difference exists between the outcomes of different GSD sites, and the requirements written in SRS do not satisfy the validation properties listed in Sect. 2.4. However, results close to 1 indicate that a minor difference exists between the outcomes of different GSD sites, and the requirements written in SRS satisfy most of the validation properties listed in Sect. 2.4.

3 The online shopping system (OSS) case study

Client XYZ is located in Australia and has been involved in the merchandising business for the last several years, selling products to the Australian market. The client considers the needs of the customer to be very important and wants to ensure a positive relationship exists with them. The client's motive is "to sell reliable and quality products to a broad range of customers at affordable prices". Due to team work, dedication, a clear focus and well-defined marketing strategies, the client holds a respectable position in the Australian merchandising arena.

Client XYZ wants to develop an online shopping system for their organization. In the shopping system, the following features are required: (i) facilitate customers/end-users in purchasing different products, tracking their orders, viewing sellers' information, and make payments via a secure payment platform; (ii) facilitate XYZ in: selling different products, managing information about customers (i.e. shoppers) and wholesale merchandisers (i.e. sellers), manage all the orders made by the customers, and

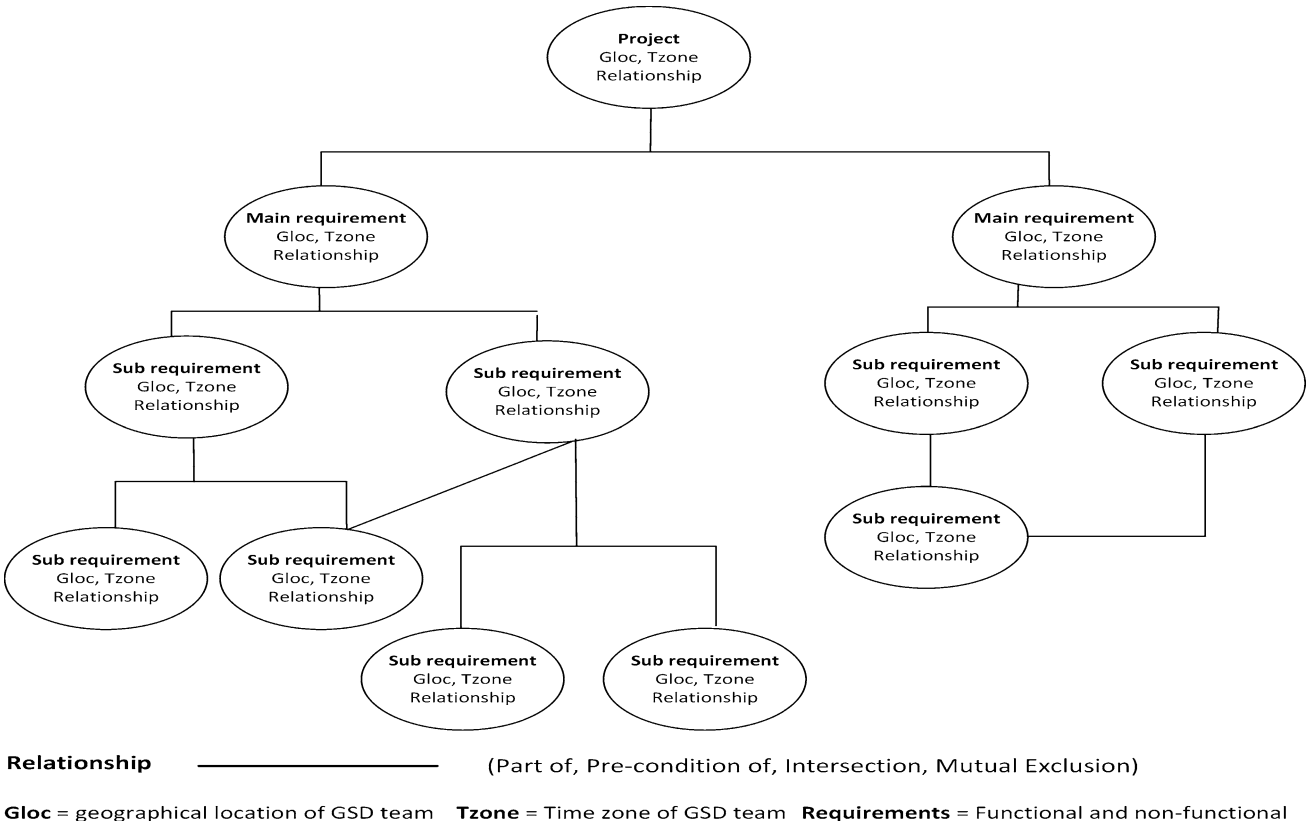

Fig. 2 Structure of a requirements graph

manage information about the products sold or still available; and (iii) easy-to-use graphical user interface (GUI).

The software organization ALPHA designs, defines and delivers a broad range of IT solutions which include: software development; hardware and software installations; network maintenance and management; desktop support and maintenance; and technological upgrades. The main site (headquarters) of ALPHA is located in Australia, and the offshore sites are located in India and China. Since its beginning, ALPHA has proven itself to be a highly committed organization which wants to deliver the best possible IT solutions at affordable prices.

Client XYZ contacted the software development organization ALPHA for the development of an online shopping system. Based on conversations with the client, the analysts and requirements engineers of ALPHA gathered and analysed details about the shopping system. After collecting and analysing the requirements of the shopping system, the analysts, requirements engineers and designers of ALPHA started the process of software requirements specification and validation. The client, requirements engineers and project analysts of ALPHA are located in Australia, and the designers are located in India.

4 Applying our method to the case study

Past researchers used student groups in a university environment to play the roles of stakeholders in experiments in GSD studies [4, 21, 30, 35]. Hence, we simulated the development of GSD requirements of XYZ using our method by creating a virtual environment for GSD within the vicinity of La Trobe University, Australia. In this virtual environment, the roles of the stakeholders were played by a group of students. With the limitations of a controlled GSD experiment, the differences in language and cultural setups were simulated by ensuring maximal participation of students from different cultures and backgrounds. Moreover, the geographical difference was simulated by ensuring that the students who performed the roles of the requirements engineer and analyst have never met the students who performed the role of the designer, and can only talk via the identified communications tools.

We selected 8 undergraduate and graduate students from the Department of Computer Science and Computer Engineering, La Trobe University, Australia, and divided them into three teams. The teams consisted of 2, 3 and 3 students who played the roles of the analyst, the requirements engineers and the designers, respectively. Basic

Table 1 Prototype for software requirements specification

Item		Description
Introduction	Purpose[a]	Describe the overall purpose of SRS and provide details about the different types of persons who will read or use the SRS document, such as document writers, developers, designers, project analysts or managers, testers and users
	Scope[a]	Provide details about the name, scope, goals and benefits of the software product that will be described in SRS
	Acronyms and definitions[a]	Define the necessary terminologies, acronyms and list of abbreviations that will be used throughout the SRS document
	References[a]	Provide details about the documents or any external pieces of information that are referred to in the SRS document. Details provided in this regard should be significant enough for a reader to access each reference
	Overview of SRS[a]	Describe the structure of the SRS document to assist the readers
General description	Product perspective[a]	Describe the overall context of the software product being detailed in SRS. Particularly, if the specified software product is a new release of the existing software, a modification to the existing systems, or a part of a larger system, then details regarding these aspects should be clearly stated in SRS
	Product functions[a]	Briefly explain the main functionalities of the software product that will be performed either by the product itself or by the user. Later, provide detailed descriptions of these functionalities in the subsequent sections
	User classes and characteristics[b]	Provide details about the different user classes that could possibly use the software product, including a list of the most and least important user classes
	Locations and time zones[b]	Record the geographical, temporal and contact details of each user class
	Communication medium used by user classes[b]	Provide details about the communication modes, mechanisms and tools that will be used for communication purposes between different user classes
	Operating environment[a]	Provide a list of software and hardware platforms that will be required to operate the software product
	Design and implementation constraints[a]	Provide details about the government policies, hardware and software limitations, language requirements, security issues and design conventions that could possibly limit the functionality of the software product
	User documentation[a]	Provide details about the user manuals and tutorials that will be delivered along with the software product
	Assumptions and dependencies[a]	List all the assumptions and dependencies that could possibly affect the requirements specified in the SRS document
Specific requirements		
Functional requirements	Introduction[a]	Provide a short description of each functional requirement
	Requirements id[a]	State the identification number of each requirement
	Priority[a]	State the priority of each requirement
	Child and parent requirements[a]	Provide details about the list of child and parent requirements
	Input[a]	List all the possible sources of input, work deadlines, valid inputs, control requirements and references to interface specifications
	Processing[a]	Define the set of operations that could be performed on input data and the parameters to achieve output. The operations include validity test for input data, execution flow, system behaviour for normal and abnormal situations, and an approach to transform input into output
	Output[a]	Provide details about the output state, including range of valid and invalid outputs, references to the interface specification document, timing to achieve output, and list of possible error messages
	Developed at user class[b]	Record details about the role and location of the user class that is responsible for the development of each requirement
	Direct and indirect project modules[b]	Keep track of project modules, including the role and location of the responsible user class that will be affected either directly or indirectly, if potential changes will occur in software product requirements

Table 1 continued

Item		Description
	External Interface	
	User[a]	Specify the list of characteristics that the software should maintain for each user interface
	Hardware[a]	Specify the list of devices that should be supported and the ways by which they are meant to be supported
	Software[a]	Specify the list of additional software products that should be required
	Comm.[a]	Specify the list of network protocols that should be used for interfacing purposes
Non-functional requirements	Performance[b]	Provide details about all the performance aspects of the software product
	Safety[b]	List any possible damages or losses that could affect the software product. In addition, details about the necessary steps to prevent damage should also be listed
	Security[b]	Specify privacy and security requirements that could affect the software product
	Quality[b]	Provide details about quality characteristics for the software product that will be significant for stakeholders. These characteristics include availability, usability, reliability, robustness, correctness, interoperability, flexibility, maintainability, adaptability and testability
Other requirements[a]		Describe any piece of requirement that is not covered elsewhere in the specification document

[a] Generic attributes, [b] GSD-specific attributes

knowledge about the experimentation process, their roles and responsibilities, and the list of things which was expected from them were given to them via information sessions. Interaction between the different student teams was performed via by a set of synchronous and asynchronous collaboration tools. In the following subsections, we describe how our method is implemented using the student groups.

4.1 Stage 1: Generation of requirements graph

To initiate the process of software requirements specification and validation, the members of the requirements engineers and the analyst teams used the requirements of a shopping system. From the information obtained, they extracted details about the main functional and non-functional requirements of the OSS, and details on the GSD sites at which these requirements could possibly be developed. In addition, they defined possible type(s) of relationship(s) which could exist among requirements. After the identification of the main requirements, they decomposed the entire set of main requirements into their constituent sub-requirements on the basis of ontological principles. The process continued until all the sub-requirements were obtained and organized to form a requirements graph (refer to Fig. 3).

All the functional requirements in the requirements graph have an associated non-functional requirement(s), a relationship(s) which exists with other requirement(s), and the location and time zones of the potential GSD teams. The functional and non-functional requirements and details on the locations and time zones of GSD teams are listed in Table 2. However, details on the relationships between the different functional requirements in Fig. 3 are listed in Table 3.

4.2 Stage 2: Preparation of the SRS document

After generating the requirements graph, the members of the requirements engineers and analyst teams started the preparation of the SRS document for the online shopping system. Thereafter, these teams prepared the initial version of the document (refer to "Appendix 2").

4.3 Stage 3: Finalization of the SRS document

Following the processes of graph generation and the preparation of the SRS document, the members of the requirements engineers and analyst teams contacted the members of the design team to discuss the project requirements. They organized a videoconferencing session and explained the content of the requirements graph and the SRS document to the members of the design team. During this process, all the members examined the requirements in a detailed manner. As a result, they were able to identify areas that required further clarification and attention. For instance, details regarding input, output and external interfaces of the "make payment" and "payment mechanism" requirements were not clear in the SRS document. All the team members made an effort to address this issue. Similarly, the team members identified different issues from the requirements graph and the SRS document, addressed them in additional videoconferencing sessions, and finalized the content of the SRS document. In total, they organized four videoconferencing sessions of approximately 30 min each.

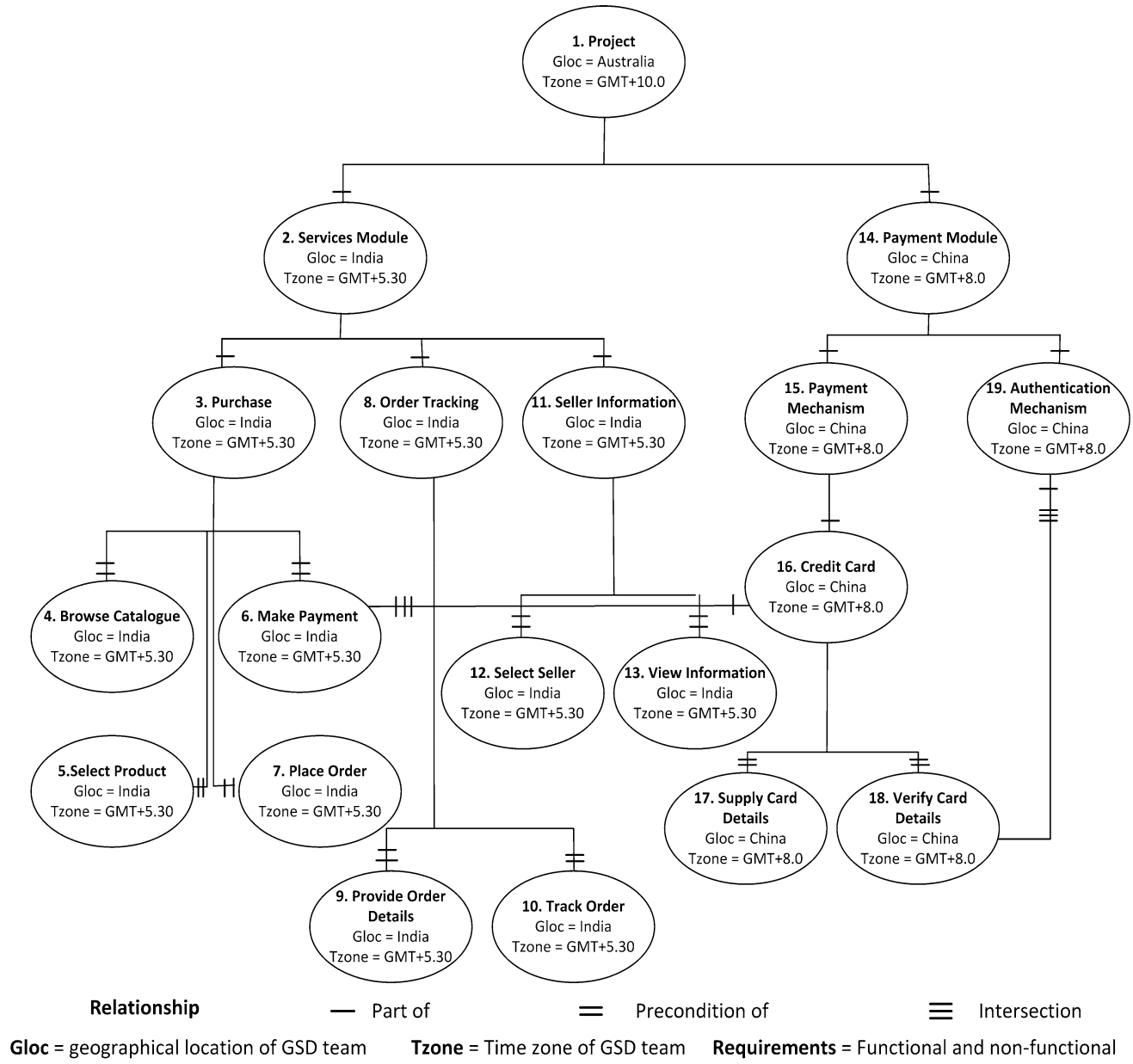

Fig. 3 Requirements graph of an online shopping system

4.4 Stage 4: Validation at different sites

After finalizing the content of the SRS document, the members of the requirements engineers, the analyst and the design teams began the process of requirements validation. They evaluated each requirement, written in the SRS document, by evaluating them with respect to the properties mentioned in Sect. 2.4. Using Eq. (1), the requirements engineers and the analyst teams generated matrix-1 (refer to Table 4) and the design team generated matrix-2 (refer to Table 5). Depending on the results of the requirements validation, the team members populated Tables 4 and 5 with 1 and 0, where 1 indicates "validation property is satisfied" and 0 "validation property is not satisfied".

4.5 Stage 5: Validation between different sites

Finally, the members of the requirements engineers, the analyst and the design teams organized a videoconferencing session to compare their matrices (refer to Tables 4 and 5). During the session, they considered one requirement at a time and compared their results by computing the correlation coefficient for them. The results obtained after applying Eq. (2) are presented in Table 6. The results vary

Table 2 Functional and non-functional requirements of the online shopping system

Requirement identifier	Functional requirements	Associated non-functional requirements	Location of GSD team	Time zone of GSD team
1	–	Performance, safety, security, usability, support, availability, localizability, reliability	Client = Australia Requirements engineer = Australia Analyst = Australia Designers = India Development team-1 = India Development team-2 = China	Australia = GMT + 10 India = GMT + 5.30 China = GMT + 8.00
2	Service module	Performance, safety, security, usability, support, availability, localizability, reliability		
3	Purchase	Performance, safety, security, availability, usability		
4	Browse catalogue	Performance, availability		
5	Select product	Performance, availability		
6	Make payment	Performance, availability, safety, security, reliability		
7	Place order	Performance, security, usability, support, availability, localizability		
8	Order tracking	Performance, availability, reliability, security		
9	Provide order details	Security, safety, availability		
10	Track orders	Performance, security, safety		
11	Seller information	Performance, availability, usability		
12	Select seller	Performance		
13	View information	Availability, usability		
14	Payment module	Performance, safety, security, usability, support, availability, localizability, reliability		
15	Payment mechanism	Performance, security, safety, availability, usability, reliability		
16	Payment via credit card	Performance, security, safety, availability, reliability		
17	Supply card details	Security, safety, availability, reliability		
18	Verify card details	Performance, security, safety, reliability		
19	Authentication mechanism	Performance, security, availability, reliability		

* Refer "Appendix 1" for details on non-functional requirements

between −1 and 1, where −1 indicates that a significant difference exists between the outcomes different GSD sites, and the requirements written in SRS do not satisfy the validation properties. However, results close to 1 indicate that minor differences exist between the outcomes of different GSD sites, and the requirements written in SRS satisfy most of the validation properties.

From the results in Table 6, the teams identified that the overall correlation for the two matrices is 0.75, which means that the matrices generated by the different teams are approximately similar and satisfied most of the validation properties. In addition, there were three requirements in the SRS document that were not validated. The correlation coefficients for these requirements are 0.61 for *requirement identifier 3*, 0.66 for *requirement identifier 8*, and 0.66 for *requirement identifier 16*. All the teams analysed each of these requirements and addressed the issues. It took 2.5 h to complete this videoconferencing session.

5 Applying the conventional method to the case study

To evaluate the effectiveness of the proposed requirements specification and validation method, the conventional RE method that does not consider GSD specifics was used for the same GSD project. To avoid learning effect, a different group of 8 undergraduate and graduate students from the Department of Computer Science and Computer Engineering, La Trobe University, Australia, were selected and later divided into three teams. The teams consisted of 2, 3 and 3 students who played the roles of the analysts, the requirements engineers and the designers, respectively.

Table 3 Relationship between the requirements of the OSS

Functional requirements	Source/destination requirements	Relationship types
2. Service module	Project (S)	Service module is a *part of* project (SR)
	Purchase (D)	Purchase is a *part of* service module (DR)
3. Purchase	Service module (S)	Purchase is a *part of* service module (SR)
	Browse catalogue (D)	Browse catalogue is a *pre-condition of* purchase (DR)
	Select product (D)	Select product is a *pre-condition of* purchase (DR)
	Make payment (D)	Make payment is a *pre-condition of* purchase AND a *part of* payment via credit card (DR)
	Place order (D)	Place order is a *pre-condition of* purchase (DR)
4. Browse catalogue	Purchase (S)	Browse catalogue is a *pre-condition of* purchase (SR)
	Not applicable (D)	– (DR)
5. Select product	Purchase (S)	Select product is a *pre-condition of* purchase (SR)
	Not applicable (D)	– (DR)
6. Make payment	Purchase (S)	Make payment is a *pre-condition of* purchase
	Payment via credit card (S)	Make payment is a *part of* (intersection) payment via credit card (SR)
	Payment via credit card (D)	Make payment is a *part of* payment via credit card (DR)
7. Place order	Purchase (S)	Place order is a *pre-condition of* purchase (SR)
	Not applicable (D)	– (DR)
8. Order tracking	Service module (S)	Order tracking is a *part of* service module (SR)
	Provide order details (D)	Provide order details is a *pre-condition of* order tracking (DR)
	Track orders (D)	Track orders is a *pre-condition of* order tracking (DR)
9. Provide order details	Order tracking (S)	Provide order details is a *pre-condition of* order tracking (SR)
	Not applicable (D)	– (DR)
10. Track orders	Order tracking (S)	Track order is a *pre-condition of* order tracking (SR)
	Not applicable (D)	– (DR)
11. Seller information	Service module (S)	Seller information is a *part of* service module (SR)
	Select seller (D)	Select seller is a *pre-condition of* seller information
	View information (D)	View information is a *pre-condition of* seller information
12. Select seller	Seller information (S)	Select seller is a *pre-condition of* seller information (SR)
	Not applicable (D)	– (DR)
13. View information	Seller information (S)	View information is a *pre-condition of* seller information
	Not applicable (D)	– (DR)
14. Payment module	Project (S)	Payment module is a *part of* project (SR)
	Payment mechanism (D)	Payment mechanism is a *part of* payment module (DR)
	Authentication mechanism (D)	Authentication mechanism is a *part of* payment module (DR)
15. Payment mechanism	Payment module (S)	Payment mechanism is a *part of* payment module (SR)
	Payment via credit card (D)	Payment via credit card is a *part of* payment mechanism (DR)
16. Payment via credit card	Make payment (S)	Payment via credit card is a *part of* make payment and payment mechanism (SR)
	Payment mechanism (S)	
	Make payment (D)	Make payment/supply card details/verify card details is a *pre-condition of* payment via credit card (DR)
	Supply card details (D)	
	Verify card details (D)	
17. Supply card details	Payment via credit card (S)	Supply card details is a *pre-condition of* payment via credit card (SR)
	Not applicable (D)	– (DR)
18. Verify card details	Payment via credit card (S)	Verify card details is a *pre-condition of* payment via credit card (SR)
	Authentication mechanism (S)	Verify card details is a *pre-condition of* authentication mechanism (SR)
	Not applicable (D)	– (DR)

Table 3 continued

Functional requirements	Source/destination requirements	Relationship types
19. Authentication mechanism	Payment module (S)	Authentication mechanism is a *part of* payment module (SR)
	Verify card details (D)	Verify card details is a *pre-condition of* authentication mechanism

Note S = source requirement, D = destination requirement, SR = relationship (in italics) between functional and source requirements, DR = relationship (in italics) between destination and functional requirements

Table 4 Validation matrix generated by the requirements engineers and analyst teams

Requirements identifier	Validation properties				
	Acceptability	Ambiguity	Completeness	Verifiability	Understandable
1	1	1	1	1	1
2	1	1	1	1	1
3	1	1	1	1	0
4	1	1	1	1	1
5	1	1	1	1	1
6	1	1	1	1	1
7	1	1	1	1	1
8	1	0	0	1	0
9	1	1	1	1	1
10	1	1	1	1	1
11	1	1	1	1	1
12	1	1	1	1	1
13	1	1	1	1	1
14	1	1	1	1	1
15	1	1	1	1	1
16	1	0	0	1	1
17	1	1	1	1	1
18	1	1	1	1	1
19	1	1	1	1	1

Basic knowledge about the experimentation process, their roles and responsibilities, and the list of things which was expected from them was given to them via information sessions. Interaction between different student teams was performed via by a set of synchronous and asynchronous collaboration tools.

In the following, we describe how the student groups used the conventional RE method to perform requirements elicitation and analysis in GSD environment.

5.1 Requirements specification

After gathering and analysing the requirements, the requirements engineers and the project analysts prepared the SRS document using [17]. Details of the SRS document are presented in "Appendix 3".

5.2 Requirements validation

After completing the requirements specification process, the requirements engineers, the analysts and the designers began the requirements validation process in a videoconferencing session. During the process, they evaluated each requirement, written in the SRS document, using the traditional contract-style requirements list by evaluating them with respect to the following properties of requirements validation: acceptability—every requirement must be acceptable to the stakeholders responsible for it; ambiguity—every requirement must have only one interpretation; completeness—the requirements written in the SRS document should be complete; verifiability—every requirement must have an associated acceptance criteria to verify them after implementing the requirement; understandable—

Table 5 Validation matrix generated by the design team

Requirements identifier	Validation properties				
	Acceptability	Ambiguity	Completeness	Verifiability	Understandable
1	1	1	1	1	1
2	1	1	1	1	1
3	1	0	1	1	0
4	1	1	1	1	1
5	1	1	1	1	1
6	1	1	1	1	1
7	1	1	1	1	1
8	1	0	0	1	1
9	1	1	1	1	1
10	1	1	1	1	1
11	1	1	1	1	1
12	1	1	1	1	1
13	1	1	1	1	1
14	1	1	1	1	1
15	1	1	1	1	1
16	1	0	0	1	0
17	1	1	1	1	1
18	1	1	1	1	1
19	1	1	1	1	1

every requirement must be clear to all groups of stakeholders [22] (refer to Table 7). They populated Table 7 with 1 and 0, where 1 indicates "validation property is satisfied" and 0 "validation property is not satisfied". Thereafter, by scanning the outcomes of Table 7, they came to conclusion that there are many requirements in the SRS document, for which some of the validation properties are not satisfied yet.

To validate the non-validated requirements, all the stakeholders (i.e. requirements engineers, project analysts and designers) organized three videoconferencing sessions at the agreed time to discuss and resolve the outstanding issues in requirements validation.

6 Discussions of the results

To validate our work, we used a case study of an OSS. A detailed description of the case study settings and a step-by-step demonstration of the proposed and the conventional RE methods were detailed in Sections 4 and 5. After completing the requirements specification and validation processes for both methods, we interviewed the student teams in five steps about their experiences in relation to the different aspects of the used methods. In step 1, we interviewed teams about the level of usefulness of the different aspects of requirements specification and validation. In step 2, we investigated the level of comfort felt by each team in working with the other teams. In step 3, we asked the teams to state the frequency of conflicts which occurred during requirements specification and validation. In step 4, we interviewed the teams about the number of rounds performed for requirements validation during the used RE methods. In step 5, we interviewed the teams about the time spent on different activities of requirements specification and validation. Finally, we analysed and discussed the data obtained during these steps.

6.1 Usefulness of our RE methods

To determine the level of usefulness of the proposed and the conventional requirements specification and validation methods, we asked members of the requirements engineer, analyst and design teams, in both project settings, to rate the different activities (also called aspects) of the used method on a scale of 1 to 4, where 1 indicates "not useful", 2 "less useful", 3 "useful" and 4 "very useful, based on their level of involvement in the various aspects. To analyze the data gathered in this step, we used Descriptive statistics to determine the level of usefulness. The results obtained after applying the Descriptive statistics are presented in Tables 8 and 9.

From the results shown in Tables 8 and 9, it can be seen that overall, the respondents from different teams rated the different aspects of the proposed requirements specification and validation as either useful or very useful for the

accomplishment of the above-mentioned aspects. The primary reasons why our method was considered useful than the conventional method are as follows: (i) the information in the requirements graph helped GSD teams in different geographical locations to identify the main and sub-requirements, the relationships between them, and details about the GSD sites at which the requirements were developed. As a result, the GSD teams found our method useful to interpret and understand the software requirements in a consistent manner; (ii) in comparison with conventional requirements specification methods, the proposed format helped GSD teams to document details about the locations and time zones of each stakeholder, to list the communication modes and the mechanisms and tools used by each group of stakeholders for communication purposes, to provide details about the development teams responsible for the development of certain aspects of a GSD project, and to produce a list of directly and indirectly affected project modules as well as a list of non-functional requirements that could be affected due to the lingual, temporal, cultural and geographical differences between GSD teams; and (iii) it helped the teams validate the contents of the SRS document and ensure the quality of the requirements written in the SRS document.

Table 6 Correlation coefficient for requirements validation

Requirements identifier	Correlation coefficient
1	1
2	1
3	0.612372
4	1
5	1
6	1
7	1
8	0.666667
9	1
10	1
11	1
12	1
13	1
14	1
15	1
16	0.666667
17	1
18	1
19	1
Overall coefficient	0.754965

6.2 Level of comfort working with other teams

To determine the level of comfort student teams felt in working with other teams, the members of the requirements engineer, analyst and design teams were asked to rate their level of comfort on a scale of 1 to 4, where 1 indicates "not comfortable", 2 "less comfortable", 3 "comfortable" and 4 "very comfortable", based on their level of involvement in the GSD project. To analyze the data gathered in this step, we used the Descriptive statistics to determine the level of comfort. The results obtained after applying the Descriptive statistics are presented below.

Table 7 Requirements validation via contract-style requirements list

Functional requirements	Validation properties				
	Acceptability	Ambiguity	Completeness	Verifiability	Understandable
Service component	1	1	1	1	1
Purchase goods	1	1	1	1	1
Browse catalogue	1	1	1	1	1
Select product	1	1	1	1	1
Make payment	1	0	0	0	0
Order placement	1	1	1	1	1
Supply order details	1	1	1	1	1
Order tracking	1	1	1	1	1
Information about seller	1	1	1	1	1
View seller information	1	0	0	0	0
Payment component	1	0	0	0	0
Payment procedure	1	0	0	0	0
Payment using credit card	1	1	1	1	1
Supply card details	1	1	1	1	1
Authentication mechanism	1	0	0	1	1

Table 8 Mean level of usefulness of different aspects of the proposed requirements specification and validation method

Different aspects	N	Mean	Median	Minimum	Maximum
Interpreting requirements	8	3.625	4	3	4
Understanding requirements	8	3.75	4	3	4
Validating requirements	8	3.625	4	3	4

Table 9 Mean level of usefulness of different aspects of the conventional requirements specification and validation method

Different aspects	N	Mean	Median	Minimum	Maximum
Interpreting requirements	8	1.714	2	1	2
Understanding requirements	8	1.571	2	1	2
Validating requirements	8	1.571	2	1	2

Table 10 Mean level of comfort expressed by the requirements engineers on the other teams in the proposed method

	N	Mean	Median	Minimum	Maximum
Analyst	3	4	4	4	4
Designers	3	3.33	3	3	4

Table 11 Mean level of comfort expressed by the requirements engineers on the other teams in the conventional RE method

	N	Mean	Median	Minimum	Maximum
Analyst	3	1.66	2	1	2
Designers	3	1.33	1	1	2

Requirements engineers versus other teams: Tables 10 and 11 present details regarding the level of comfort the requirements engineers team expressed on the other teams in the proposed and the conventional methods of RE, respectively.

Analysts versus other teams: Tables 12 and 13 present details regarding the level of comfort the analyst team expressed on the other teams in the proposed and the conventional methods of RE, respectively.

Designers versus other teams: Tables 14 and 15 present details regarding the level of comfort the design team expressed on the other teams in the proposed and the conventional methods of RE, respectively.

6.3 Frequency of conflicts

In step 3, we asked each team to state the frequency of conflicts which occurred during the different activities of RE in relation to the proposed and conventional RE methods on a scale of 1 to 4, where 1 indicates "rarely", 2 "occasionally", 3 "less frequently" and 4 "very frequently". To analyze the data obtained during this step, we applied the Descriptive statistics. The results are presented in Tables 16 and 17.

From the results shown in Tables 16 and 17, it can be seen that overall, the respondents from the different teams indicated less conflicts during requirements specification and validation in the proposed method, mainly due to the following: (i) as a result of the requirements gathered and analysed with respect to different time zones and distance, the teams find it easier and non-conflicting to organize the obtained pieces of software requirements with respect to the GSD sites at which the requirement(s) are developed; (ii) the systematic organization of software requirements helped the teams in establishing and maintaining a consistent interpretation of software requirements across different time zones and geographical locations; (iii) the availability of information on all aspects of the prospective software system in GSD environment helped the teams in obtaining the required piece of information at the right time, which is even not available in the IEEE requirements specification document [17]; (iv) although there was a lack of face-to-face contact among the teams, the identification of suitable communication modes, mechanisms, tools and a mutually convenient time for communication helped the teams engage in discussion in a virtual environment; and (v) the matrix-based process facilitated the teams to validate the requirements, initially at their local site, then evaluate the outcomes of validation process with other teams by computing correlation coefficient, and at last identify and re-validate the non-validated requirements in a collaborative environment.

6.4 Number of rounds performed for requirements validation

In step 4, we interviewed the relevant student teams in relation to the number of rounds performed for requirements validation during the different aspects of the proposed and conventional methods of RE. In Table 18, details regarding the number of requirements validated at

Table 12 Mean level of comfort expressed by the analysts on the other teams in the proposed method

	N	Mean	Median	Minimum	Maximum
Requirements engineers	2	4	4	4	4
Designers	2	3.5	3.5	3	4

Table 13 Mean level of comfort expressed by the analysts on the other teams in the conventional RE method

	N	Mean	Median	Minimum	Maximum
Requirements engineers	2	2	2	2	2
Designers	2	1.5	1.5	1	2

Table 14 Mean level of comfort expressed by the designers on the other teams in the proposed method

	N	Mean	Median	Minimum	Maximum
Requirements engineers	3	3.33	3	3	4
Analysts	3	3.33	3	3	4

Table 15 Mean level of comfort expressed by the designers on the other teams in the conventional RE method

	N	Mean	Median	Minimum	Maximum
Requirements engineers	3	1.33	1	1	2
Analysts	3	1.33	1	1	2

Table 16 Mean frequency of conflicts which occurred during different aspects of the proposed requirements specification and validation method

Different aspects	N	Mean	Median	Minimum	Maximum
Interpreting requirements	8	1.25	1	1	2
Understanding requirements	8	1.5	1.5	1	2
Validating requirements	8	1.375	1	1	2

Table 17 Mean frequency of conflicts which occurred during different aspects of the conventional requirements specification and validation methods

Different aspects	N	Mean	Median	Minimum	Maximum
Interpreting requirements	8	3.62	4	3	4
Understanding requirements	8	3.5	4	3	4
Validating requirements	8	3.62	4	3	4

Table 18 Details on the number of requirements validated

Round # of requirements validation	Proposed method	Conventional RE method
1	16 out of 19 requirements	10 out of 15 requirements
2	3 out of the remaining 3 requirements	2 out of the remaining 5 requirements
3	–	2 out of the remaining 3 requirements
4	–	1 out of the remaining 1 requirement
Total number of requirements validated	19	15

different rounds of requirements validation are presented. From the results shown in Table 18, it can be seen that the students who implemented the proposed method validated the total number of 19 requirements in two rounds of requirements validation. However, the students who implemented the conventional RE method validated the total number of 15 requirements in four rounds of requirements validation.

Table 19 Time spent on requirements specification and validation

Activities	Time spent using the proposed method	Time spent using the conventional RE method
Requirements graph	400	Did not generate
SRS report		550
Validate requirements	180	400
Total time (minutes)	581 min	950 min
Total time (hours)	9.66 h	15.83 h

6.5 Time spent on different activities of RE

In step 5, we interviewed the relevant student teams in relation to the time spent on the different activities of RE using the proposed and conventional methods of RE. In Table 19, details regarding the time spent on requirements specification and validation are presented. From the results shown in Table 19, it can be seen that the students who implemented the proposed method of requirements specification and validation spent 400 min to generate the requirements graph and prepare the software requirements specification document, and 180 min to validate requirements written in the specification document, that makes the total of 580 min (9.66 h). However, the other groups of students who implemented the conventional RE method took 550 min to prepare the requirements specification document and 400 min to do requirements validation, thus giving a total of 950 min (15.83 h).

6.6 Summary of the results

After analysing the results presented in the earlier sections, the following observation is made about the students who used the conventional RE method.

- Due to the non-availability of GSD-related information in the SRS document, the student teams find it difficult to gather information about the GSD sites at which the requirement(s) are developed, the locations and time zones of each GSD team, the list of communication modes, mechanisms and tools used by each GSD team for communication purposes, details about the development teams responsible for the development of certain aspects of a GSD project, the list of directly and indirectly affected project modules, and the non-functional requirements which are affected due to the lingual, temporal, cultural and geographical aspects of GSD. Also, the scattered presence (availability) of requirements, gathered during requirements elicitation and analysis, made it challenging for the student teams to establish a consistent interpretation and understanding of software requirements, and later validate the software requirements in a geographically distributed environment.

In comparison with the findings made about the students who used the conventional RE method, the following observations are made about the students who used the proposed method of requirements specification and validation.

- The use of graph in our method helps GSD teams understand software requirements with respect to different time zones and distance, and the GSD sites at which the requirement(s) are developed. Thereafter, it helps GSD teams across different time zones and locations in the preparation of the software requirements specification.
- The requirements specification template, proposed as part of the requirements specification and validation method, helped students (GSD teams) prepare an SRS document. Our SRS document therefore contains information about the traditional aspects of a software project [17] and the peculiarities involved in GSD, which are missing in the IEEE specification guidelines.
- The matrix-based validation helps GSD teams examine the requirements with respect to the validation properties at different GSD sites, compare the outcomes of the validation process with other teams by computing the correlation coefficients, identify the requirements which do not satisfy the validation properties, and re-evaluate the non-validated requirements in a globally distributed environment.

Overall, the students rated different aspects of the proposed method as either useful or very useful for GSD, expressed a higher level of comfort working with other teams, and the frequency of conflicts which occurred during the requirements specification and validation processes was comparatively lower than the conventional RE method. Considering the fact that students performed additional activities to perform requirements specification and validation, they are still able to finish the entire process in less time, than the groups of students who implemented the conventional RE method. Thus, we can say that our method is especially useful for those groups of stakeholders who are taking part in GSD for the first time or are not aware of the fundamental aspects and issues of RE in GSD.

6.7 Threats to validity

Our work has the following validity threats. We have selected undergraduate and postgraduate students from the Department of Computer Science and Information Technology, La Trobe University. The difference in educational

qualification can be a selection bias threat. The experiments in this demonstration case study lasted for several months; events (examination pressure, work commitments, personal issue etc.) occurred during this period could affect participant's behaviour and performance. None of the participants have previous knowledge of GSD; the results obtained in this experimental setup could be a threat to validity.

When applying our method to the case study, the students were from one university. Although we selected students from different cultural backgrounds, the work involving students from different universities with dissimilar cultural backgrounds needs to be examined. Situational specifics (e.g. location, time, supervision and role of investigator) can potentially limit the generalizability of results. Researcher's expectations and experiment bias can also be a threat to validity.

7 Related work

In the literature, only two papers have been published on GSD requirements specifications. In [23], the authors present a five-step process of requirements specifications for geographically distributed software projects. As a part of these steps, the authors suggest that the requirements specification document be circulated back and forth between the onshore and offshore development teams, until all the details in the requirements document are finalized. However, the authors in [29] proposed the use of patterns to prepare the SRS document for GSD projects. These patterns are mainly related to defining use cases, collaboration among the analysts throughout the RE process and mapping business-related terminologies to the entity attribute. With the help of these patterns, the authors aimed to address the following issues: the multiple definitions of use cases which often lead to misinterpretation; the challenges involved in knowledge transfer between development sites; and the difficulties in understanding and processing the requirements specification document. In addition to these papers, two papers have been published on requirements validation in GSD. In Yousuf et al. [39], a survey on the existing methods of requirements validation is presented. Based on the survey findings, the authors suggested that the factors of communication, control, knowledge sharing, delay and trust impact traditional methods of requirements validation and therefore cannot be used effectively for validation purposes in GSD projects. In [12], a proposal to select and utilize practices for the early validation of software requirements is presented. The authors presented a decision tree to facilitate stakeholders in the selection and utilization of different techniques and practices for requirements validation, where decisions are made on the familiarity and non-familiarity of requirements.

As a result of the contributions of this related work, the following conclusions are made: (1) due to the lack of requirements specification methods in [23, 29], a judgement about how to use these proposals and patterns to prepare an SRS document for a GSD project cannot be made; (2) in [39], the authors examined the impact of different GSD factors on the traditional methods of requirements validation and suggested that new methods are required for GSD projects; and (3) although, the decision tree facilitates stakeholders in the selection and utilization of appropriate practices for requirements validation, due to the lack of a methodical approach in [12], a judgement about how to use this proposal to validate the requirements of industrial projects cannot be made.

In the light of these findings, it can be said that these contributions lack methodical approaches to generate and validate the SRS document. We have therefore addressed these aspects in our work.

8 Conclusions and future work

The quality of software requirements specification is vital to project success. Due to the influence of cultural differences, language and communication barriers, difficulties in knowledge management and differences in time zones on software development, the ways by which requirements are documented and validated in collocated software development cannot be used effectively in a globally distributed environment. To date, there are very few research papers available which describe the work done on GSD requirements specification and validation. In this paper, we have therefore presented a method for this.

Our method is beneficial for GSD teams in the following ways: (i) the use of graph in our method helped GSD teams understand software requirements with respect to different time zones and distance, and the GSD sites at which the requirement(s) are developed; (ii) it helped GSD teams across different time zones and locations in the preparation of the SRS document. Our SRS document therefore contains information about the traditional aspects of a software project [17] and the oddness involved in GSD, which are missing in the IEEE specification guidelines; and (iii) the matrix-based validation helped GSD teams examine the requirements with respect to the validation properties at different GSD sites, compare the outcomes of the validation process with other teams by computing the correlation coefficients, identify the requirements which do not satisfy the validation properties, and re-evaluate the non-validated requirements.

To validate our method, we applied it to a case study of an online shopping system, where the roles of stakeholders were played by a group of students. Furthermore, we used

the conventional requirements specification and validation method that does not consider GSD specifics for the same GSD project, so that a comparison between the results could be made. The results showed that the proposed method helped the student teams in preparing and validating the contents of SRS document for the requirements of the online shopping system. As a result, the chances of requirements being misunderstood and misinterpreted by development teams could be possibly minimized, and the time spent searching for information about the different aspects of a GSD project could be reduced.

To examine how scalable our method is, we aim to find a commercial partner, which would be prepared to collaborate with us in experimenting our method in a real-life environment.

Acknowledgments This work is supported by a La Trobe University Postgraduate Write-up scholarship.

Appendix 1

See Table 20.

Appendix 2: Software requirements specification of online shopping system (Developed by using proposed method)

Introduction

Purpose of SRS: The SRS document describes the requirements for the online shopping system. For correspondence purposes, the reference number of this SRS document is V101-Online Shopping.

The persons who will use this document are the requirements engineers, project analysts, designers, developers and users from client XYZ. Any possible changes made in the requirements of the online shopping system will be recorded in the SRS document, and the latest version will then be used by these groups of people.

Scope of SRS: Client XYZ wants to develop a software product called an "*online shopping system*" for their organization by which they can sell different products to a broad range of customers. The shopping system must be able to: (1) facilitate customers/end-users in purchasing different products, tracking their orders, viewing sellers'

Table 20 Description of non-functional requirements

Non-functional requirements	Description
Performance	*Response time*- The system should be able to retrieve order details within 10 s
	Workload- The system should be able to support 4 pages/second
	Scalability- The system should be capable of supporting no less than 50 customers at a time when implemented
	Platform- The system should be able to operate in Internet Explorer (v. 7 and later), Mozilla Firefox (v. 2 and later), Google Chrome, and Opera
Security	The shopping system must ensure that data about different types of transactions must be processed in a secured channel
Usability	End-users with different background knowledge can easily place orders
Support	*Helpdesk support*- Regardless of the time difference between Australia and offshore sites, 24/7 support will be required for six months from the offshore sites
	Network support- Should be provided 24/7 despite geographical dispersion
	Application support- Should be provided 24/7 despite geographical dispersion
	Database support- Should be provided 24/7 despite geographical dispersion
	Administration support- Should be provided 24/7 despite geographical dispersion
	Security support- Should be provided 24/7 despite geographical dispersion
	Training support- Should be provided 24/7 despite geographical dispersion
Availability	Orders can be placed 24/7. In case of unstable internet connection, the information necessary to place orders could be send again
Localizability	Although the system will be developed at offshore locations, localizability must be ensured with respect to Australian traits
Safety	To prevent possible data damages or losses, the shopping system must have a data recovery procedure
Reliability	The system must be able to store database information on different computers to prevent it from possible losses and damage

Table 21 List of acronyms and definitions

Acronyms	Meanings
FAQ	Frequently asked questions
CRM	Customer relationship management
IEEE	The institute of electrical and electronics engineers
SRS	Software requirements specification
GUI	Graphical user interface
HTTP	Hyper text transfer protocol

information, and making payments via a secure payment platform; and (2) facilitate XYZ in selling different products, managing information about customers (i.e. shoppers) and wholesale merchandisers (i.e. sellers), managing all the orders made by the customers, and managing information about the products sold or still available.

Acronyms and definitions: (Table 21).

References: The following references are used in the SRS document.

- IEEE 830 [17] standard for writing SRS document.
- List of scenarios and use cases
- Sommerville [33]

Overview of SRS: The remaining sections of the SRS document are organized as follows.

- Section 4.2.2 defines the product perspective, functions, user classes and characteristics, locations and time zones of user classes, list of communication modes, mechanisms and tools used between user classes, operating environment, design and implementation constraints, user documentation, and assumptions and dependencies.
- Section 4.2.3 specifies the details about functional and non-functional requirements.

General description

Product perspective: The online shopping system is a new and stand-alone software product. Therefore, it is not a part of a larger system or a modification of the existing systems.

There are two basic modules/components in the online shopping system. The first module is responsible for the different types of services offered by the shopping system. However, the second module covers the security aspect of the shopping system. A detailed description of these modules and their functionalities is listed in Sect. 4.2.3.

User classes and characteristics: The users who will interact with the shopping system are the requirements engineers, project analysts, designers, developers, client XYZ and end-users. Requirements engineers, project analysts and designers are assumed to have detailed knowledge of the overall requirements, and developers are more aware about the development and technical aspects. However, easy-to-use graphical user interfaces and user documentation will be provided to educate client XYZ and other end-users about how to use the shopping system.

Locations and time zones of user classes: Details about the location and time zones of each user class are given in Table 22.

Communication modes, mechanisms and tools used by user classes: Details about the communication modes, mechanisms and tools that will be used by each user class are mentioned in Table 23.

Operating environment: The shopping system is a website and should be able to operate in Internet Explorer (v.7.0 and later), Mozilla Firefox (v.2.0 and later), Google Chrome and Opera.

Design and implementation constraints: Details about the design and implementation constraints are given in Table 24.

User documentation: Four different types of documentation will be produced during the software development life cycle.

- High-level description of the most important software processes
- Data specification report for purchase orders, order tracking, seller information, and payment and authentication mechanisms
- Online help about how to use the shopping system
- Feedback and error-reporting mechanisms to be used by the system administrator

Assumptions and dependencies: The following assumptions are made about the online shopping system

- User and management-related processes are combined at a central site, accept input and provide different services to different users at different locations
- ASP.Net will be used as a development platform and the SQL server to store database
- The shopping system will be easy to use by different groups of users
- The performance of shopping system depends on the speed of the internet

Specific requirements

There are different types of services and payment mechanisms in the online system. Detailed descriptions about their associated requirements are listed below.

Table 22 Locations and time zones of user classes

User class	Departments	Managers	Contact details	Duties
Clients	Information Technology	Mr. ABC	Australia, GMT + 10, abc@xyz.com	Technology head
	Sales/Pre-sales	Mr. DEF	Australia, GMT + 10, def@xyz.com	Senior sales officer
	Marketing	Mr. GHI	Australia, GMT + 10, ghi@xyz.com	Marketing manager
	Human Resource	Mr. JKL	Australia, GMT + 10, jkl@xyz.com	Human resource manager
	Finance	Mr. MNO	Australia, GMT + 10, mno@xyz.com	Senior financial officer
Analysts	Information Technology	Mr. PQ	Australia, GMT + 10, pq@alpha.com	Technology manager
	Business	Mr. RS	Australia, GMT + 10, rs@alpha.com	Business executive
Requirement engineers	Information Technology	Mr. TU	Australia, GMT + 10, tu@alpha.com	Requirements engineering
	Information Technology	Mr. VW	Australia, GMT + 10, vw@alpha.com	
	Business	Mr. XY	Australia, GMT + 10, xy@alpha.com	
Designers	Information Technology	Mr. AA	India, GMT + 5.30, aa@alpha.com	Product analysis and designing
		Mr. BB	India, GMT + 5.30, bb@alpha.com	
		Mr. CC	India, GMT + 5.30, cc@alpha.com	
Developers	Information Technology	Mr. DD	India, GMT + 5.30, dd@alpha.com	Development
		Mr. EE	China, GMT + 8, ee@alpha.com	
End-users	Could be any person from any part of the world			

Table 23 List of communication modes, mechanisms and tools used by user classes

Communication task	Communication aspect	User classes				
		Client	Requirement engineers	Analysts	Designers	Developers
Project discussion	Mode	Verbal	Verbal	Verbal	Verbal	Verbal
	Mechanism	Audio/ video	Audio/video	Audio/video	Audio/video	Audio/video
	Tool	Skype	Skype	Skype	Skype	Skype
Knowledge transfer and exchange	Mode	Written	Written	Written	Written	Written
	Mechanism	Messaging	Messaging	Messaging	Messaging	Messaging
	Tool	Emails	Emails and instant messaging	Emails and instant messaging	Emails and instant messaging	Emails and instant messaging

Table 24 Design and implementation constraints

Constraints	Definition
User rights and privileges	Controlled via security groups and privileges for different user classes
Back-end database	Information about services, security and management processes must be stored in a database
Training	Management processes must provide training about how to use the software product in different scenarios
HTML compliance	The product must be HTML compliant
User passwords	Depending on privilege, passwords must be assigned to each group of users

Functional requirements

I. Services

Introduction: Three different types of services are required: purchase; order tracking; and seller information

Requirement id: 2

Priority: High

Child requirement: Purchase, order tracking and seller information

Parent requirement: Online shopping system

Input: Purchase details, order tracking information, sellers' specification

Processing:

- *Purchase details* browse catalogue, select product, payment, and place order.
- *Order tracking* tracking criteria and shipping information
- *Sellers' specification* user rating and history

Output:

- *Purchase details* catalogue browsing, product selection, make payment and order placement.
- *Order tracking* track orders
- *Sellers' specification* view seller information

Developed at user class: India

Direct and indirect affected requirements: Changes in the requirements of the service module will affect the requirements of the payment module, developed at the Chinese site.

External interfaces:

- *User interface* All the GUI's must follow a similar theme and have a clear structure.
- *Hardware interfaces* The shopping system is a web-based software product that should run easily on the aforementioned web browsers, and will be hosted on a Windows server.
- *Software interfaces* Any operating system capable of running different web browsers could be used.
- *Communication interfaces* HTTP protocols must be used to facilitate communications between client and server machines.

II. Purchase

Similarly, the requirements engineers and analysts document details about other functional requirements.

Non-functional requirements

Performance requirement:

- *Response time*The system should be able to retrieve order details within 10 s
- *Workload* The system should be able to support 4 pages/second
- *Scalability* The system should be capable of supporting no less than 50 customers at a time when implemented
- *Platform* The system should be able to operate in Internet Explorer (v. 7 and later), Mozilla Firefox (v. 2 and later), Google Chrome and Opera.

Safety requirement: To prevent possible data damages or losses, the shopping system must have a data recovery procedure.

Security requirement: The shopping system must ensure that data about different types of transactions must be processed in a secured channel.

Quality attributes:

- *Usability* End-users with different background knowledge can easily use the shopping system
- *Support* Regardless of the time difference between Australia, India and China, 24/7 helpdesk, network, application, database, administration, security and training supports will be required for 6 months from the offshore sites
- *Reliability* The system must be able to store database information on different computers to prevent it from possible losses and damage
- *Localizability* Although the system will be developed in India and China, localizability must be ensured with respect to Australian traits
- *Availability* The shopping system should be accessible to users 24/7, except the specified maintenance period

Other requirements

For further details, the user should refer to the following documents.

Use case documentation

Appendix 3: Software requirements specification of online shopping system (Developed by using existing method)

Introduction

Purpose: The SRS document provides information on the requirements of the OSS. The reference number of the SRS document is SRS00-1/ONS.

The people who will use this document are the requirements engineers, project analysts, designers, developers and users from client XYZ.

Scope: Client XYZ wants to develop a software product called an "online shopping system" for their organization. The shopping system must be capable of providing the following functionalities:

- Facilitate customers/end-users in purchasing different products, tracking their orders, viewing sellers' information, and making payments via a secure payment platform.
- Facilitate XYZ in selling different products, managing information about customers (i.e. shoppers) and wholesale merchandisers (i.e. sellers), managing all the orders made by the customers, and managing information about the products sold or still available.
- Easy-to-use graphical user interface (GUI).

Table 25 List of acronyms and definitions used in the SRS document

Acronyms	Definition
FAQ	Frequently asked questions
GUI	Graphical user interface
IEEE	The Institute of Electrical and Electronics Engineers
SRS	Software requirements specification
TCP/IP	Transfer communication protocol/internet protocol

Definitions, acronyms and abbreviations: The following acronyms are used in the SRS document (refer to Table 25).

References: The following reference is used in the SRS document.

- IEEE 830 [17] standard to write the SRS document.

Overview of SRS: The SRS document is organized as follows.

- A general description presents details on the product perspective and functions, user characteristics, design and implementation constraints, and assumptions and dependencies.
- A specific requirements list which details the functional and non-functional requirements.

General description

Product perspective: This is a new system which is neither an extension of an old system nor a component of a larger system.

Product function: The overall system has two main modules that cover the different functionalities of services and payment features of the shopping system.

User characteristics: The users who will interact with the OSS are the requirements engineers, project analysts, designers, developers, and users from the client.

Design and implementation constraints: Depending on the nature of the interaction, the rights and privileges must be ensured for each group of users.

Assumptions and dependencies: The shopping system must be easy to use by different groups of users.

Specific requirements

a. Functional requirements

(i) Services

Introduction: The three types of services that will be required are: purchase, order tracking and seller information.

Input: Details of purchase, information to track orders, and sellers' specifications.

Processing: To process different types of inputs, the following mechanisms will be used:
Details of purchase- browse catalogue, select product, make payment,
Information to track orders supply order details
Sellers' specifications- seller history

Output: For each input, one of the following outputs will be generated.
Details of purchase order will be placed after browsing the available catalogue
Information to track orders- order tracking
Sellers' specifications- view information about the seller's rating and past history

External interfaces: For each of the external interfaces, the specifications listed below will be used.
User interface a clear and consistent theme should be used in all GUI's
Hardware interface a Windows server 2008–2010 will be used to host the OSS.
Software interface the versions of Mozilla Firefox, Google Chrome and Internet Explorer released after the year 2005 will be used to run the OSS.
Communication interface TCP/IP and HTTP protocols must be used for communication purposes.

(ii) Purchase

Likewise, the requirements engineers and the project analysts documented details about the other functional requirements.

b. Performance requirements

The overall performance of the shopping system mainly depends on the aforesaid hardware and software specifications.

c. Design constraints

Each of the developed subsystems must be traced back to its associated use case and non-functional requirements.

d. Attributes

Security- The possible transactions that occur in the shopping system must be processed in a secured and encrypted channel.
Safety- To minimize data loss, safety measures must be taken to prevent data.
Availability- The shopping system should be available 24 h a day, 7 days a week.
Reliability- The shopping system should be reliable

Other requirements

For further clarification, contact the project analyst.

References

1. Ali N, Lai R (2014) Managing requirements change in global software development. In: IEEE international conference on data and software engineering, Indonesia
2. Ali N, Lai R (2016) A method of requirements change management for global software development. Inf Softw Technol 70:49–67
3. Atkins D, Handel M, Hersleb J, Mockus A, Perry D, Wills G (2001) Global software development, the bell labs co-laboratory in international conference on software engineering, Toronto
4. Babar MA, Kitchenham B, Jeffery R (2008) Comparing distributed and face-to-face meetings for software architecture evaluation: a controlled experiment. Empir Softw Eng 13(1):39–62
5. Bartholomew R (2008) Globally distributed software development using an immersive virtual environment. In: IEEE international conference on electro/information technology (EIT'08), pp 355–360
6. Berenbach B, Paulish DJ, Kazmeier J, Rudorfer A (2009) Software & systems requirements engineering. In Practice, Mcgraw-Hill Professional
7. Brockmann PS, Thaumuller T (2009) Cultural aspects of global requirements engineering: an empirical chinese-german case study. In: Fourth IEEE international conference on global software engineering (ICGSE '09), pp 353–357
8. Conchuir EO, Holmström H, Ågerfalk PJ, Fitzgerald B, (2006) Exploring the assumed benefits of global software development. In: Proceedings of the 1st international conference on global software engineering, pp 159–168
9. Damian D, Zowghi D (2003) Requirements engineering challenges in multi-site software development organizations. Requir. Eng J 8:149–160
10. Ebert C, Neve PD (2001) Surviving global software development. IEEE Softw 18(2):62–69
11. Elliott J, Raynor-Smith P (2000) Achieving customer satisfaction through requirements understanding. Softw Process Technol 1780(20020):203–219
12. Heinonen S, Tanner H (2010) Early validation of requirements in distributed product development: an industrial case study. In: Proceedings of the 2010 international conference on the move to meaningful internet systems, pp 279–288
13. Herbsleb JD, Mockus A (2003) An empirical study of speed and communication in globally-distributed software development. IEEE Trans Software Eng 29(6):481–494
14. Herbsleb JD, Moitra D (2001) Global software development. IEEE Softw 18(2):16–20
15. Hsieh Y (2006) Culture and shared understanding in distributed requirements engineering. In: International conference on global software engineering; brazil, pp 101–108
16. Hull E, Jackson K, Dick J (2010) Requirements engineering. Springer Science & Business Media
17. IEEE Standard 830 (1998) Recommended practice for software requirements specifications. The Institute of Electrical and Electronics Engineers, Inc. New York
18. Johri A (2011) Sociomaterial bricolage: the creation of location-spanning work practices by global software developers, special issue on "Studying work practices in Global Software Engineering". Inf Softw Technol 53(9):955–968
19. Lai R, Ali N (2013) A method of requirements management for global software development, Advances in Information Sciences, Human and Science Publication, Volume 1, Issue 1, pp 38–58
20. Lee Y, Zhao W (2006) Domain requirement elicitation and analysis- an ontology based approach, Volume 3994/2006. In: Workshop on computational science in software engineering (CSSE'06)
21. Lloyd WJ, Rosson MB, Arthur JD (2002) Effectiveness of elicitation techniques in distributed requirements engineering. In: IEEE joint international conference on requirements engineering proceedings, pp 311–318
22. Lobo OL, Arthur JD (2005) Effective requirements generation: synchronizing early verification and validation, method and methods selection criteria, Virginia Tech
23. Lopes L, Prikladnicki R, Audy J, Majdenbaum A (2005) Requirements specification in distributed software development: a process proposal. In: Proceedings of the 38th Hawaii international conference system sciences (HICSS 05)
24. Paasivaara M, Durasiewicz S, Lassenius C (2008) Distributed agile development: using scrum in a large project. In: IEEE international conference on global software engineering (ICGSE'08), pp 87–95
25. Prikladnicki R, Audy JLN, Damian D, Oliveira TC (2007) Distributed software development: practices and challenges in different business strategies of off-shoring and on-shoring. In: Second IEEE international conference on global software engineering, pp 262–274
26. Prikladnicki R, Audy JLN, Evaristo R (2003) Global software development in practice lessons learned. Softw Process Improv Pract 8(4):267–281
27. Rickman DM (2001) A new process for requirements understanding. In 20th Conference on digital avionics system, pp 4D4/1–4D4/6
28. Rodgers JL, Nicewander WA (1988) Thirteen ways to look at the correlation coefficient. Am Stat 42(1):59–66
29. Salger F, Englert J, Engels G (2010) Towards specification patterns for global software development projects—experiences from the industry. In: Seventh international conference on the quality of information and communications technology (QUATIC), pp 73–78
30. Shami NS, Bos N, Wright Z, Hoch S, Kuan KY, Olson J, Olson G (2004) An experimental simulation of multi-site software development. In: Proceedings of the 2004 conference of the centre for advanced studies on collaborative research, pp 255–266
31. Shewell C (2000) Good business communicates across cultures: a practical guide to communication. Mastek Publications, Bristol
32. Smite D (2007) Project outcome predictions: risk barometer based on historical. In: Proceedings of the 2nd international conference on global software engineering (ICGSE'07), pp 103–112
33. Sommerville I (2007) Software engineering, 8th edn. Addison-Wesley, England
34. Sommerville I, Sawyer P (1997) Requirements engineering: a good practice guide. Wiley, New York
35. Walle VD, Campbell C, Deek FP (2007) The impact of task structure and negotiation sequence on distributed requirements negotiation activity. Conflict, and Satisfaction, published by LNCS vol 4495, pp 381–394
36. Wiegers KE (2003) Software requirements. Microsoft Press, Redmond
37. Wolf T, Dutoit AH (2005) Supporting traceability in distributed software development projects. In: Proceedings of international workshop on distributed software development
38. Wongthongtham P, Chang E, Dillon T (2007) Multi-site distributed software development: issues, solutions, and challenges. In: Proceedings of the 2007 international conference on computational science and its applications, pp 346–359
39. Yousuf F, Zaman Z, Ikram N (2008) Requirements validation techniques in GSD: a survey. In Proceedings of the IEEE multitopic conference, pp 553–557