This is a previous version of the article published in Journal of Software: Evolution and Process. 2017, 29: e1858. https://doi.org/10.1002/smr.1858

A Comprehensive Framework for Modeling Requirements of CSCW Systems

Miguel A. Teruel[1]
LoUISE Research Group, Computing Systems Department, University of Castilla- La Mancha
Avda. España s/n, 02071 Albacete (Spain)
Phone +34 967 599 200 ext. 2624, fax +34 967 599 343
miguel@dsi.uclm.es

Elena Navarro
LoUISE Research Group, Computing Systems Department, University of Castilla- La Mancha
Avda. España s/n, 02071 Albacete (Spain)
elena.navarro@uclm.es

Víctor López-Jaquero
LoUISE Research Group, Computing Systems Department, University of Castilla- La Mancha,
Avda. España s/n, 02071 Albacete (Spain)
victor@dsi.uclm.es

Francisco Montero
LoUISE Research Group, Computing Systems Department, University of Castilla- La Mancha
Avda. España s/n, 02071 Albacete (Spain)
fmontero@dsi.uclm.es

Pascual González
LoUISE Research Group, Computing Systems Department, University of Castilla- La Mancha
Avda. España s/n, 02071 Albacete (Spain)
pgonzalez@dsi.uclm.es

Abstract. As in any other system, an accurate requirements specification is essential to developing a collaborative system, which has special kinds of requirements that are hard to specify by means of current Requirements Engineering (RE) techniques. The Collaborative Systems Requirement Modeling Framework (CSRMF) was developed to address this problem; the original CSRML (Collaborative Systems Requirement Modeling Language) was extended and modeled to properly deal with collaboration and awareness requirements. The developed CSRMF framework consists of three components: an RE modeling language able to represent collaboration among users as well as awareness needs, a set of design guidelines that drive CSCW system specification by means of five different types of diagrams, and a supporting CASE tool to specify and validate CSCW system requirements. CSRMF provides Requirements Engineers with a complete solution to the specification of awareness-demanding collaborative systems, as they can now take advantage of a language and a set of guidelines supported by a tool to guide them in specifying system requirements.

Keywords: Awareness; CASE tool; CSCW; Goal-Oriented; Requirements Engineering Process.

1 Introduction

The way in which services, applications, etc. are provided through the Internet has clearly changed in recent years with collaboration everywhere. Collaboration can be defined as working with others on a task to achieve shared goals [1]. Nowadays, if we take a look at the most visited websites, almost all in the top 100 are collaborative websites [2]. Social networks, collaborative

[1] Corresponding author

document editors, online games, everything is becoming collaborative and/or cooperative [3]. Multiplayer computer games are an excellent example of collaboration between users. Players must be aware of their current allies' status, position and/or capabilities, in order to cooperate with them to win the game [4]. These system are interesting example of CSCW systems (*Computer Supported Cooperative Work*) [5], since they rely on the extensive use of collaborative features. This CSCW label, defined as "a shorthand way of referring to a set of concerns about supporting multiple individuals working together with computer systems" was coined by Greif and Cashman in 1984 [6]. In addition to the functionality already provided by classic software applications, CSCW users can also perform tasks related to collaboration, communication and coordination (a.k.a. *3C model* [7]). Examples of this kind of task can currently be found in many CSCW applications. For instance, in Google Docs text editor [8], users are able to work together on the same document (collaboration), to chat about issues (communication) and to control access to the documents (coordination).

Because of this collaborative trend in modern software, sophisticated methods and tools are really needed to build these applications. Although comprehensive methodologies can now be used to design CSCW systems [9], [10] these methodologies do not give much attention to Requirements Engineering (RE), one of the most important stages of the software development process, and use only conventional RE techniques or obviate this stage. One of the big problems in developing CSCW systems is specifying their requirements for 3C tasks, because of the inherent complexity of this collaboration as well as the users' awareness needs.

Figure 1. Relationship between 3C model and awareness [11]

This awareness consists of the users' ability to perceive, feel or be conscious of events in the system, shared objects or users with whom to perform 3C tasks [12], [13]. Indeed, the lack of this information might make the users' work inefficient and clumsy, and at worst, infeasible [14]. In the literature, one of the most widely accepted awareness interpretations is Workspace Awareness (WA), proposed by Gutwin and defined as "the up-to-the-moment understanding of another person's interaction within a shared workspace" [15]. WA means knowing, for example, *who* is available to collaborate with, *what* the other users are doing (or what they did in the past), *where* in the shared workspace they are working, *when* an artifact was modified or *how* a certain operation happens. As can be observed, describing these WA elements in the requirements specification is a critical issue in building proper collaborative systems. Nevertheless, when dealing with a real CSCW system, the complexity of the collaboration tasks [16], the participant's roles involved [17] and the awareness needs [18] make it difficult to specify and understand them using current RE techniques, which can produce incomplete and/or imprecise requirements models [19]. According to Schmidt [20], this field, like other multidisciplinary ones, lacks the necessary scaffolding for the integration of different views from different disciplines, i.e. the need for a common framework that includes both the social and technological aspects of CSCW systems.

In order to address this issue, in this work we propose the CSRMF framework (Collaborative Systems Requirements Modeling Framework) for the specification of a whole CSCW system by modeling its 3C tasks, as well as WA features and the concept of user groups. This framework uses CSRML v2 (Collaborative Systems Requirements Modeling Language) as its specification language, an extension of the original CSRML language that supports all the WA requirements specified by Gutwin [15], a much stronger support for groups and users, and a new set of elements and relationships which make the specification of collaborative tasks much more expressive. Thanks to the additional expressive power offered by CSRML v2, not only can classic CSCW systems be specified, but also avant-garde collaborative systems, such as multiplayer games or Post-WIMP immersive applications. To make this specification easier, the CSRML CASE Tool was developed to support analysts when using CSRMF to model and verify CSCW system requirements specifications. Finally, in order to lead the specification of a CSCW system with CSRML, a set of guidelines were composed to help RE practitioners model an entire CSCW system, from user and group identification to the definition of the tasks they will perform and the awareness information needed for the tasks.

In order to show the benefits of using CSRMF the CSCW requirements of Age of Empires Online™ were modeled. This is a popular multi-user real-time strategy game which is highly demanding in terms of WA information. In fact, CSRMF was applied to the specification of several applications, such as a conference review system and a collaborative e-learning system. However, we opted for this collaborative game because it was the most complex system we had ever modeled with this framework due to its wide range of CSCW characteristics: the diversity of collaborative tasks to be performed, the huge amount of awareness information to be dealt with in real-time and the different relationships among the participants (allies, foes and neutrals). Our research questions were thus defined as follows:

- RQ1. How can a new metamodel-based framework based on the Collaborative Systems Requirements Modeling Language deal with CSCW's 3C model and Workspace Awareness features?
- RQ2. What guidelines can drive the specification of a CSCW system by using the Collaborative Systems Requirements Modeling Framework?
- RQ3. How can tool support be provided for modeling a CSCW requirements specification considering Collaborative Systems Requirements Modeling Framework guidelines?
- RQ4. Is the Collaborative Systems Requirements Modeling Framework expressive enough to model the requirements of a real awareness-demanding CSCW scenario like a mainstream game?

This paper is structured as follows. In Section 2 related work is reviewed. Section 3 summarizes the process led by empirical evaluations that resulted in the development of CSRMF and the CSRMF metamodel, modeling language and guidelines and how they are integrated into our case tool. Section 4 describes a case study using CSRMF to model a real CSCW system. Section 5 includes our most important conclusions and our intentions for further work.

2 Related Work

CSCW systems (*Computer Supported Cooperative Work systems*) [5], [7] are getting more and more attention. CSCW discipline addresses "how collaborative activities and their coordination can be supported by means of computer systems" [21]. Thus, we can find CSCW examples ranging from traditional everyday collaboration tools, such as e-mail, videoconferences, or newsgroups, to the now commonly used social networks [22], on-line games [23], collaborative text editors [24] or e-learning systems [25], either in academic or industrial fields [3]. The main purpose of CSCW is to enable users able to work collaboratively in a computerized environment as they would in a physical environment. This awareness of others' presence and activities, trivial in a physical environment, is not that easy in CSCW, but can be provided by means of Workspace Awareness (WA) elements [26].

Figure 2. Collaborative application supporting Workspace Awareness

In order to exemplify the use of Workspace Awareness techniques, Figure 2 shows a collaborative code editor called SubEthaEdit [27] featuring 3C (collaboration, communication and coordination) and WA element support, which provide the user with awareness of other participants. As can be observed in the center of Figure 2, this application has a text editor that enables several users to edit the same source code simultaneously (collaboration task). This editor provides information on who edited each part of the document by means of a color code (implementation of Gutwin's awareness question "Who is doing that?"), as well as showing the position of each user by means of the colored rectangles in the scroll bar (the amount of selected code is represented by the rectangle length). This implements Gutwin's question "Where are they looking?" In addition, the User Statistics window provides information on the actions performed by the participants (deletions, insertions and selections). It also shows when these actions were performed (implementations of “What has a person been doing?” and “When did that event happen?”). Regarding communication tasks, the Connections window makes it possible to communicate with other participants either by chat or e-mail, providing awareness of “Who is participating? Who is that?” Regarding coordination, the Document Access window shows the exact position of the users in the document (Gutwin’s “Where are they working?”) and whether they have selected a text (number between parentheses). It also supports the assignment of the document’s permissions (read/write, read only or even kick out users).

Gutwin and Greenberg [28] published an empirical study based on a usability test with several user interfaces at different WA levels. They concluded that high-level WA helped improve the performance of experimental tasks. Other studies are currently being conducted aimed at the systematization of considering WA during the development of CSCW. For instance, Figueroa-Martinez et al. [29] enriched UsiXML [30], a language for the specification of user interfaces, with WA support to integrate WA into a model-driven user interface development process. Similarly, CSRML (Collaborative Systems Requirements Modeling Language) [31] has been defined by extending the Goal-Oriented (GO) language *i** [32] to support the specification of WA as well as to model user collaboration.

A great deal of other work can be found on the development of CSCW systems, such as design guidelines [33], implementation techniques [34], empirical evaluations of CSCW systems [35], and even specific development methodologies [33]. For instance, related to CSCW systems design, particularly for mobile applications, Antunes [36][37] proposed design guidelines to help designers to identify places, users, artifacts and geo-referenced knowledge. These guidelines identify *what* information is of interest to designers and *how* user requirements may be applied. There is also an interesting study by Molina et al. [37] focused on modeling CSCW systems which describes a graphical language (and the CIAT DSL tool) for the specification of a whole groupware system by defining their different views but unfortunately did not consider WA aspects.

Despite the important amount of work already conducted on CSCW, not much attention has been paid to its requirements specification. Currently, CSCW methodologies, such as AMENITIES [33] or CIAM [37] are mainly focused on design activities. These authors specified awareness requirements by using an extension of the templates developed by Duran et al. [38], which may cause linguistic problems due to its text-only-based nature [39]. These templates are based on UML’s Use Cases [40] that do not have a specific mechanism to represent either 3C tasks or WA. Unlike current techniques such as Use Cases, which focuses on describing *what* the system functionality must be, Goal Oriented (GO) techniques ask “why” this functionality is needed by documenting it through goals, which are considered stakeholders’ intentions with regard to the objectives, properties or use of the system [41]. The use of GO techniques thus leads analysts to specify systems in a way that can be better understood by the stakeholders. They also facilitate the identification and analysis of alternative realizations of the system. Within the GO techniques, several approaches can be found such as the NFR Framework [42], KAOS [43], *i** [32] or even a unifying framework that takes advantage of the strengths of several GO methodologies [44]. The CSRMF framework described in this work is based on *i**, which is a GO approach whose main aim is to document and analyze goals and their dependences. Based on the idea that an actor depends on other actors to achieve its goals, *i** requirements models are divided into two different diagrams, namely: the Strategic Dependency Model (documenting the actor and their dependencies) and Strategic Rationale Model (detailing the goals, tasks, etc. of each actor). Additionally, *i** provides a metamodel [45] that can be extended for use in domain-specific scenarios [46].

A considerable amount of CASE tools can be found regarding tool support for GO techniques. For instance, the Organization Modeling Environment (OME) [47] provides graphical support for specifying and analyzing requirements by using several GO approaches like *i**. This tool evolved to the open-source Eclipse-based OpenOME [48], which was enriched with some new features, such as forward and backwards requirements analysis. Directly related to *i*,* J-PRiM [49] is a tool that supports modeling and analysis of requirements by means of a reengineering process. This tool guides the analysis of existing systems and their representation by using several alternative hierarchies of *i** elements.

However, when trying to specify a complex CSCW system, *i** lacks the required scalability features, advanced role management and 3C / awareness representation features [50] and so generates barely understandable requirements specifications when used for this purpose. The understandability of the two latter models was empirically evaluated [19], showing that the CSRML language, which this framework is based on, has enough expressive power to model these systems, obtaining much better results than *i**. Nevertheless, it also brought to light several other issues inherent in this first version of CSRML, such as an awareness representation mechanism based on a little-known proposal or no support for representing groups of users, together with some minor understandability issues. The CSRMF framework was therefore developed to solve these shortcomings and make it possible to specify complex CSCW systems, guiding the RE process by its guidelines and facilitating the edition and validation of models by means of the CSRML Tool, the CASE supporting software. This is a step forward in "covering the gap" between CSCW requirements, which are directly related to human activity, and computational entities, in Ackerman's words [51].

3 Collaborative Systems Requirements Modeling Framework

The CSRMF (Collaborative Systems Requirements Modeling Framework) specifies CSCW (Computer Supported Cooperative Work) systems, paying special attention to modeling 3C tasks and their underlying Workspace Awareness (WA) features. CSRMF aims at establishing a framework for the specification of CSCW requirements by the use of a common language, a tool and guidelines to reduce the difficulties among the stakeholders involved [52]. It should be noted that CSRMF focuses on requirements specification, but not on elicitation / capture stages, for which there are other techniques [53]. The framework (see Figure 3) consists of the following elements:

- CSRML v2 (Collaborative Systems Requirements Modeling Language): uses an improved version of CSRML, a Goal-Oriented Requirement Engineering language focused on specifying 3C tasks and WA features.
- Design guidelines: a set of guidelines to drive the specification of CSCW systems requirements that describe how to specify all 5 kinds of CSRMF diagram necessary for the specification of a CSCW system.
- CSRML Tool: a CASE tool that provides analysts with the necessary support for the specification of CSCW systems requirements with CSRML implemented as a Visual Studio extension.

Figure 3. CSRMF components

CSCW characteristics can be specified by the CSRMF framework as follows:

- *Group hierarchy*: A CSCW system normally has a group of users who work together to achieve common goals. In addition to *actor*[2], *groupactor* was also included in CSRMF so that both users and groups of users can be specified. When a *groupactor* is specified, it is also related to the actors that make up the group using *participation links*. Another interesting feature is that it allows analysts to specify the leading actors in the group.
- *Role management*: when developing a CSCW system it is crucial to define user capabilities. In CSRMF, these capabilities are assigned by means of a *playing link*, which specifies the condition that must be accomplished for an *actor* to play a *role*. The framework is able to represent the *role* responsible for a *task* using the *responsibility link*.
- *Collaboration among users*: the specification of collaboration is a cornerstone of CSRML v2. Owing to both the *role* and *task* elements, this framework is able to specify collaboration among

[2] All the elements are explained in Section 3.1

users as 3C tasks (collaboration, communication and coordination, as shown in Figure 2), according to the model presented in [7], which is the most widely-used in the CSCW field. Therefore, the roles (or role in case of individual tasks) participating in a task are specified by means of *participation links*, which also denote how many instances of each role are involved (cardinality).

- *Awareness*: although awareness is important, it is somewhat difficult to specify when modeling a CSCW system. CSRMF introduces the *awareness resource*, which specifies the awareness a user needs to perform a task. This is done by the elements used in the definition of an awareness resource, *present and past awareness elements,* which denote the specific awareness needs for each task, based on Gutwin's descriptive WA framework [15]. Awareness resources are thus related to tasks and roles by means of a 3-ary relationship, the *participation link*. However, Gutwin's framework elements are not the only ones considered in CSRMF, but the main characteristics of Social Awareness [54], like group members, roles and collaborative tasks, can also be specified by means of other elements in the framework
- *Goal-Orientation*: the CSRMF framework is based on *i** [32], to support goal-oriented requirements modeling and the specification of *goals* and their corresponding refinement into *resources*, *tasks* or *softgoal*s, as well as establishing *dependencies* among *roles* and elements.
- *Quality factors management*: as for any other software system, quality factors must be taken into account for the development to be successful. In CSRMF *contribution links* are used to specify *softgoals* which represent quality factors that contribute to achieving the software quality, to which *tasks* and *softgoals* can contribute.
- *Hierarchical organization:* the specification of a CSCW system is performed hierarchically, thus solving the scalability issue of *i** mentioned in Section 2. Actually, the specification is performed from high-level system goals to specific user tasks (*top-down* decomposition), which has been shown to improve the understandability of Software Engineering diagrams [55].

This framework also promotes the specification of the requirements of CSCW systems by means of 5 different types of diagram to improve the specification's readability and understandability. These diagrams, depicted in the metamodel in Figure 5, are the following:

- *Group Hierarchy Diagram (GHD)* (Figure 5a): this diagram depicts the different stakeholders (and their groups) involved in the CSCW system by using *actors* and *groupactors,* respectively. Examples of this kind of diagram are shown in Figure 9 and Figure 16.
- *Quality factors diagram (QFD)* (Figure 5b): these diagrams specify the quality factors that contribute to achieving the main softgoals (quality factors) identified in the RD diagram. Several examples can be found in Figure 13 and Figure 21.
- *System Goals Diagram (SGD)* (Figure 5c): this diagram is used to identify the goals of the CSCW system. Each of these goals will be assigned to the *actors* and *groupactors* involved in their achievement. Figure 10 and Figure 17 show some examples.
- *Responsibility Diagram (RDs)* (Figure 5d): each of these diagrams represents one of the goals identified in the SGD. The RD diagram specifies the roles played by the actors and the tasks the actors are responsible for. Figure 11 and Figure 18 show some examples.
- *Task Refinement Diagrams (TRDs)* (Figure 5e): In a TRD diagram, the tasks previously identified in RD diagrams are decomposed into individual tasks and 3C tasks that support WA features. Figure 12, Figure 19 and Figure 20 give some examples.

In Section 3.1 the CSRML v2 modeling language of the CSRMF framework will be explained by describing both its elements and relationships, corresponding to the graphical representation of the metamodel elements (Figure 4 and Figure 5). The design guidelines for the creation of CSCW specifications will be presented in Section 3.2 and the CASE Tool of the CSRMF framework is described in Section 3.3.

3.1 A modeling language: CSRML v2

Originally based on Yu's *i** [32], the CSRML v1 [31] was designed for the specification of CSCW systems by extending *i** elements and relationships with new ones that enable the specification of the 3C model and WA. It not only has CSRML v2 CSCW awareness modeling capabilities, but also solves the understandability issues of both CSRML v1 and *i** detected in [19] by means of hierarchical diagrams and the inclusion and redefinition of some of their elements. CSRML v2 is based on a metamodel made by extending the elements in v1. Figure 4 shows the hierarchy and composition relationships as well as the elements' attributes and the enumerations used (elements and relationships not present in CSRML v1 shown in red dotted lines) and Figure 5 the relationships among the elements in the diagrams (red represents CSRMF elements, blue relationships among elements, yellow represents diagrams and green represents enumerations. The color code facilitates understanding how the different elements are related. A filling pattern is also used to improve readability in black and white diagrams. Abstract class names are shown in italics.

Figure 4. CSRML v2 metamodel: detailed classes with hierarchy and aggregation relationships

Figure 5. CSRML v2 metamodel: diagrams containing elements and relationships

The awareness modeling system of this redesigned version of CSRML is based on Gutwin's WA [15]. The CSRML v2 elements used in this framework are the following: (note that the elements that have been added or modified are denoted with an asterisk (*)):

- *Actor**: a user, program, or entity with certain acquired capabilities (skills, category, and so forth) that can play a role while it executes, uses devices or is responsible for actions An actor has to play a *role* (specified by means of a *playing link*) in order to participate in a system.
- *GroupActor**: is a designator for a group composed of one or more actors whose aim is to achieve one or several goals to represent groups of users.
- *Role*: is a designator for a set of related tasks to be carried out. An actor playing a role can participate in individual or collaborative tasks (through *participation links*) and it can be responsible for the achievement of a goal (through r*esponsibility links*).
- *Goal*: answers "why?" questions. It describes a certain state of the world that an actor would like to achieve. However, a goal does not prescribe how it should be achieved.
- *Softgoal*: is a condition in the world that the actor would like to achieve, but unlike the concept of (hard) goal, the condition for its achievement is not sharply defined. A softgoal is typically a quality attribute that constrains another element, such as a goal, task or resource.
- *Task**: specifies a particular way of doing something. As can be seen in the metamodel (see Figure 4), this element has an importance level according to the task's development priority. Two types of task are identified:
  - *Abstract task*: is an abstraction of a set of concrete tasks and other elements.
  - *Concrete task*: refinements of abstract tasks and have roles responsible for their accomplishment. There are four types of concrete task: an *Individual task* that an actor can perform without any kind of interaction with other actors. *Collaboration / Communication / Coordination tasks* require two or more actors to be involved in order to perform any kind of collaboration / communication / coordination.
- *Resource:* is an entity (physical or informational) that the actor needs to achieve a goal or perform a task. The main concern about a resource is whether it is available and from whom.
- *Awareness Resource**: This element represents some awareness perceptions that a role needs to accomplish a task. As the metamodel shows (Figure 4), this element is composed of several *Awareness Elements* based on the Gamespace Awareness features identified by conducting a Thematic Synthesis on the existing awareness interpretations [56]. These elements are specialized according to their temporal category and classified according to their *importance* (*nice to have, desirable, highly desirable or mandatory*).

The modeling elements defined can be related to each other according to the CSRML v2 metamodel (Figure 5) by the following set of relationships:

- *Dependency*: is a relationship between a *depender* and a *dependee* for a *dependum*. The *depender* and the *dependee* are actors and the *dependum* can be a goal, a task, a resource, or a softgoal. The *depender* depends on the *dependee* for achieving a goal, performing a task, or using a resource. If the *dependee* fails to provide the *depender* with the required *dependum*, it becomes difficult or impossible for the *depender* to achieve the goal, perform the task, or use the resource. Based on the type of *dependum*, there are four types of dependencies: goal dependency, task dependency, resource dependency and softgoal dependency.
- *Means-end link**: a means-end link documents which softgoals, tasks, and/or resources contribute to achieving a goal. A means-end link also facilitates the documentation and evaluation of alternative ways to satisfy a goal [57], i.e., different decompositions of a goal into subgoals, tasks, and resources.
- *Task decomposition link*: a task decomposition link describes the essential elements of a task. A task decomposition link relates the task to its components, which can be any combination of sub-goals, sub-tasks, resources, and softgoals. The decomposition of a task can thus comprise sub-tasks that must be performed, sub-goals that must be achieved, resources that are needed, and softgoals that typically define quality goals for the task.
- *Contribution link*: A contribution link documents an influence from a task or softgoal to other softgoal. It is defined by means of the *kind* attribute of the corresponding metamodel element, and can be positive, *negative* or *unknown* (Figure 4).
- *Playing link*: a playing link is used to represent when an actor plays a role. This link has a *guard* condition attribute (Figure 4) that represents when a role can be played by an actor.
- *Responsibility link**: a responsibility link assigns a role (played by an actor) to a (soft)goal or task. This link represents who is the stakeholder responsible for a goal/task accomplishment.
- *Participation link**: can connect different elements according to the context (diagram) in which this relationship is used (see Section 3.2 for a complete description of the different CSRMF diagrams):
  - *TRD context*: A participation link denotes who is involved in a task. This link has an attribute to specify its *cardinality*, i.e., the number of users that can be involved in a task. It can optionally have an *awareness resource* attached to it (examples shown in Figure 19 and Figure 20). In this way, it shows that the role has a special perception need (specified by means of the awareness resource) in order to participate in the task. Without this perception, the accomplishment of the task could be negatively affected or the role could even not be able to participate in the task.
  - *GHD context:* this relationship is used to assign an actor to a *groupactor* (see Figure 16). Its cardinality shows how many actors constitute each *groupactor*.
  - *SGD context:* used to specify which actors are involved in the accomplishment of the main goals of the system (see example in Figure 17). The number of occurrences of each actor or group actor is denoted by the cardinality field.

In order to make the specification of CSCW system requirements easier using CSRML, a graphical notation was designed to create the CSRML v2 diagrams in a more intuitive manner. The graphical elements added to the original *i** notation follow Moody's prescriptive theory principles [55] in order to create a cognitively effective visual notation. The graphical definitions for the CSRML v2 elements (Figure 6) and relationships (Figure 7) are the following:

Figure 6. Graphical description of CSRML elements

Figure 7. Graphical description of CSRML relationships

- Goal, Resource and Softgoal: these elements keep the original *i** notation.

- Task: The original *i** notation for this element (an irregular hexagon) is complemented with several icons representing the kind of task (see Figure 6). The importance of a task is defined by a graphical notation [55] based on a color code (green, yellow, orange, red), green being the least and red the most important. For color-blind readers or black and white print-outs, importance is also denoted by an alternative notation based on exclamation marks (none – normal, [!] – high, [!!], very high, [!!!] – highest) to reinforce the meaning of the task's importance.
- Awareness Resource: this kind of element can be shown in the diagrams by two different graphical notations: expanded or reduced. In its expanded form, the Awareness Resource shows all the Workspace Awareness features identified by Gutwin [15], whose importance can be set (if a specific WA feature is needed) according to the relevance they have to the accomplishment of a task. This importance can be *nice to have* (N), *desirable* (D), *highly desirable* (HD) or *mandatory* (M). These awareness features are categorized into two different sections. One is related to the present and the other to the past, as in Gutwin's work. Note that as awareness is not only related to sight, but to hearing, touching, knowing, or any other means that provides users with feedback about the people collaborating with them in a shared workspace, the icon of this element is a brain containing a question mark.
- Actor, GroupActor and Role: the new graphical representations for these elements can be seen in Figure 6. Note that CSRML does not use the boundary concept from *i**. Instead, the relationships of actors and roles to tasks and goals are shown by participation links.
- Dependency, Means-end Link and Task Decomposition Link: these relationships keep the original *i** notation.
- Contribution Link: this link has a symbol representing the kind of contribution (see Figure 7). It can be either "+" (positive), "-" (negative) or "?" (unknown).
- Playing Link: this link represents its guard condition my means of a text between brackets ([guard condition]).
- Participation Link: this link represents cardinality in a UML-like style. For example:
  - 1: only one actor / role is participating
  - 1..3: from one to three actors / roles are participating
  - 2..*: a minimum of two actors / roles are participating
- Responsibility Link: the new graphical representation for this relationship can be seen in Figure 7.

3.2 CSRMF Design Guidelines

This section explains the set of guidelines (G_i) that drive the specification of CSCW systems by means of CSRML and is divided into five different types of diagram.

(G0) CSRMF roadmap

A CSCW system specified by following the CSRMF guidelines must have the diagram structure shown in Figure 8. This is the first guideline and leads CSCW analysts through the following steps:

1. Identification of the actors and groups of actors participating in the system
2. Definition of system's main goals
3. Assigning the responsibilities and specifying the main tasks
4. Refinement of tasks and definition of collaboration and awareness
5. Specification of quality factors

This procedure follows Goal-Oriented recommendations, i.e. it goes from an abstract need to a concrete one by refining the system's main goals into abstract tasks, that in turn, will be refined into less abstract ones until reaching concrete tasks that can be assigned to actors (playing the required role to participate in that task). This procedure also assumes that the main constituents of CSCW systems (actors / groups and system goals) guide the specification.

Figure 8. CSRMF model roadmap

The guidelines required to create each type of diagram are explained in the following subsections with practical examples (see Figure 9 to Figure 13). In order to create RDs, TRDs and QFDs we must follow the guidelines for these diagrams more than once until we get a diagram structure like the one shown in Figure 8. Note that when an element defined in a diagram is referenced in a different diagram, its name is shown in italics. When dealing with complex systems, the requirements specification process using the CSRMF guidelines can be performed iteratively in order to refine the model until it is properly and fully specified.

(G1) Group Hierarchy

The first step in creating a CSRMF model is to identify the participants (and groups of participants) that are going to be involved in the system. Then, for each participant / group, a new actor / *groupactor* must be added to the GHD diagram. These actors must then be related to their corresponding *groupactors* by means of participation links, whose cardinality is assigned according to the number of participants in each group. If a group has a leader, the leading actor should be specified. For example, Figure 9 shows a *groupactor* that has one or more actors and a group leader (represented by the hand icon). An example of a real scenario is shown in Section 4.2.1 (Figure 16), related to an online collaborative game.

Figure 9. Basic Group Hierarchy Diagram

(G2) Systems Goals

The tasks and goals of the CSCW system must be specified just after the creation of the GHD. The system's main tasks must be specified in this diagram and decomposed into the main goals (related to the tasks by task decomposition links). With the system goals defined, the actors and *groupactors* must be related to the goal they are involved in by means of participation links. The number of actors and *groupactors* that contribute to the accomplishment of each goal must be specified by setting the cardinality property of the corresponding participation links. For instance, Figure 10 shows a SGD that has two tasks, each decomposed into two main goals. The actors and *groupactor*, previously defined in the GHD (see Figure 9), are related to the goals they are involved in. For further details, see Figure 17 in Section 4.2.2.

Figure 10. Basic System Goals Diagram

(G3) Responsibilities and Roles

Once the system main tasks and goals have been defined, an RD diagram must be created for each task defined in the SGD decomposed into one or more sub-tasks and optionally into one or more softgoals related to quality factors. Each actor's roles must be defined here and all the sub-system roles must be added and related to the actors specified in the SGD by means of playing links. These links have a guard condition that indicates when the actor can play a role. A role must be made responsible for every task and softgoal. This can be done hierarchically by assigning a role to the main task (and consequently to all of its sub-tasks and softgoals). However, this responsibility assignment can be overridden if a different role is assigned to a sub-task or quality softgoal. For example, in Figure 11 one of the tasks of the SGD above is decomposed into two sub-tasks and two quality softgoals. In this example, Role 2 is responsible for every sub-task and quality softgoal except for *Sub-task 2*, whose responsibility falls on *Role 1.* Figure 18 in Section 4.2.3 depicts an additional example from a real system.

Figure 11. Basic Responsibilities Diagram

(G4) Task Refinement

Every sub-task in the RD main task must be refined into several goals, resources, softgoals and sub-tasks in a TRD diagram. The tasks will be decomposed by task decomposition links, following the same procedure as for goals, but using means-end links instead. This procedure must be repeated over and over until the abstract tasks are refined into concrete tasks. For example, in Figure 12 one of the sub-tasks from the RD previously specified (see Figure 11) is decomposed into two concrete tasks (an individual and a collaborative). The roles specified in the RD must be assigned to concrete tasks by means of participation links. These links have a cardinality property to indicate the number of actors playing a role necessary for a task to be performed. Note that an individual task only needs one role, but a 3C task needs more than one (or only one role with a cardinality of 2 or more). The importance of the tasks can be also specified in this diagram.

Figure 12. Basic Task Refinement Diagram

Dependencies and awareness resources can also be specified in the TRD (refer to Section 3.1 to see how dependencies work). An awareness softgoal is represented in a TRD as a 3-ary relationship, because it must be related to a task and to the role that needs awareness about other users to participate in the task. Several awareness elements (related to the past and the present) can be added to the awareness resources. The importance of every awareness element in the awareness resource can also be specified (information on awareness resources can be found in Section 3.1). As an example, Figure 12 shows an awareness resource related to *Role 2*, which participates in the *Collaboration Task* and requires ([M]) to know the history of actions performed by the users participating in the task. It has also been specified that it would be nice to have ([N]) information about the presence of other users. Two additional examples of this type of diagram can be found in Figure 19 and Figure 20 in Section 4.2.4.

(G5) Quality Factors

In order to conclude the specification of a CSCW system with CSRMF, optionally, one or more QFD diagrams could be created (depending on whether or not the system requires quality factors to be specified). Thus, if a softgoal has been added to the RDs, a new QFD must be created to specify each of these softgoals. Each softgoal defined in the RD must also be added to a QFD and refined into softgoals and tasks that contribute to its accomplishment (Figure 13) by means of contribution links. These contributions can be positive, negative or unknown, as shown in Figure 13. The specification of this type of diagram can be carried out by following international quality standards such as the ISO/IEC 25010:2011 [58]. A complementary example of this type of diagram is shown in Section 4.2.5 (Figure 21).

Figure 13. Basic Quality Factors Diagram

3.3 CASE Support: CSRML Tool

A powerful CASE tool that supports modeling and validation is a cornerstone for the success of a new technique, language, method or whatever [59] and CSRMF is no exception, as a CASE tool was created to facilitate modeling CSCW requirements (Figure 14). The CSRMF CASE tool is able to represent all its elements and relationships, support the diagram organization and validate the models created by checking whether they conform to the CSRMF metamodel (among other design restrictions). The CSRML Tool supports model validation in three different ways:

- Design-time validation: The graphical editor does not allow users to add incorrect elements to certain diagrams or to connect elements with incorrect links.
- Meta-model validation: The CSRML Tool allows users to check whether the specified model conforms to the CSRML meta-model or not.
- Other validations: The validation procedure checks other potential sources of incoherence, such as recursive tasks and goal decompositions (e.g. Task_1 is decomposed into Goal_1, which is

fulfilled by means of Task_1). Diagrams are checked for duplicated or void references to ensure their integrity. The validation system checks other aspects untestable by means of the metamodel, e.g. the correctness of the participation links, preventing users from specifying incorrect cardinalities (such as *3..1*, *-2..2*).

Figure 14. CSRML Tool running under Microsoft Visual Studio 2012 and Windows 8

This tool is integrated with Visual Studio, so several wizards and item templates have been created that support the users, as with any other Visual Studio project. For instance, by using one of these wizards, a new CSRMF project can be created that includes 5 empty diagrams (one of each type). A context-sensitive help component is available to the user which is integrated into the Visual Studio help system. This help component includes a description of all the CSRMF elements and diagrams, as well as several tutorials covering the most complex aspects of the tool.

The tool's usability has been empirically evaluated [60] using a facial expression analysis technique [61] and a user satisfaction questionnaire [62], the results of which were reported following the ISO Common Industry Format for usability test reports [63]. This evaluation performed by 28 Computer Science students concluded with 89.29% of the experimental tasks correctly completed and a satisfaction result of 6.06 out of 7, which can be considered a high score. The facial expression results helped us to identify the key aspects of the tool that need to be improved.

4 Putting CSRMF into Practice: Specification of a Real Multiplayer Collaborative Game

CSRMF was used to specify several information systems, including a multi-user text editor, a collaborative conference review system and an e-learning platform. However, despite the wide range of collaborative tasks involved, none of these systems was sufficiently awareness-demanding to need CSRMF's awareness modeling capabilities. We therefore chose an online multiplayer collaborative game for this purpose: Age of Empires Online™ (AoEO) [64]. In order to play AoEO, as in most real-time strategy games, players need constant awareness information about their allies (their location, actions and resources) in order to collaborate in a group victory. This game has a characteristic that is not demanded by most other CSCW systems: *opponents* who will collaborate in order to defeat other players and neutral entities that are not aligned with any other player. This means three sources of awareness must be specified, one regarding our allies, another for our opponents and yet another for other entities. Although the game had already been developed, its shortcomings (evidenced by the patches applied in the last year) could have been overcome by CSRMF. The following section includes an overview of the game (Section 4.1) as well as an introduction to its requirements specification using CSRMF (Section 4.2).

4.1 Age of Empires Online

AoEO is the 12th and last title of the Age of Empires™ (AoE) series, which is a collection of real-time strategy games. Originally developed by Ensemble Studios and published by Microsoft, the AoE series has been a commercial success, having sold over 20 million copies. For this work, we selected the last release of this game, which was designed to be massively played on the Internet.

Figure 15. AoEO 2vs2 skirmish game with detailed map

The *Skirmish* AoE battle game (see Figure 15) consists of creating a virtual city (inspired on an ancient civilization) and an army. Once the city has been developed, the player must defend it and try to defeat the enemies by destroying their cities, optionally with the help of one or more allied players.

We chose AoEO for the following reasons: first of all, it is a played by millions of users and is free. Secondly, collaboration among users is everywhere: players have to trade, coordinate with each other and help each other. Currently, there are several game missions hard to win (or even impossible) without the aid of other players. Finally, this game demands a lot of Workspace Awareness (WA) elements for collaboration among players.

Players can collaborate in several ways; for instance, they need to coordinate and communicate with each other (either to attack the enemies or defend their own bases) by means of a chat window (see Figure 15 bottom left corner) or flares (blue circles Figure 15). When a player needs resources that he cannot obtain by himself, he can exchange resources with allies by trading or paying tributes. However, before asking for resources, players can see information about their allies by checking the information area in the top left corner (see Figure 15). To make collaborative actions possible, this game has several WA elements to provide feedback to the players, similar to those proposed in Gutwin's framework elements [15]. For example, the most valuable element is the game map (see Figure 15), corresponding to the Overview+Detail WA display technique [65]. This element, similar to a radar view [66] shows a scaled-down version of the entire battlefield. Looking at this map, a player can see the general status of the battlefield by means of a color code (blue – user, yellow – allies, and red – enemies) and can also move through the battlefield and send position signals to his allies by clicking on the map. The map provides the user with feedback on what his allies are looking for by a set of telepointers (yellow rounded squares) [67], without talking / chatting with them (something really complicated when a skirmish is in an advanced stage).

However, this is not the only feedback received from other users, because the players are always receiving status messages and audio feedback related to what is happening on the battlefield. With this audio feedback, they are aware of the most important events in the game, such as knowing when and where the enemies are attacking a building not shown on the screen at that moment. This audio feedback is supported by a surround sound system which can provide positional feedback. Next, we describe how CSRMF's expressive power can be used for modeling a wide variety of collaborative requirements.

4.2 Modeling Age of Empires Online CSCW Requirements with CSRMF: Putting into practice CSRML v2, its guidelines and CASE tool

We now give the results of the reverse engineering process carried out to obtain AoEO requirements specification. Despite not having used the original requirements of this game, we were able to show the capabilities of the whole CSRMF framework when applied to a complex CSCW system.

4.2.1 Identifying actors and groups (G1)

As pointed out in the roadmap guideline G0, the Group Hierarchy Diagram (GHD) was defined to identify the actors and *groupactors*, as well as to assign actors to their corresponding groups by using participation links (guideline G1). As can be observed in Figure 16, the *Administration board* has one *Coordinator* and one or more *Administrators*. This group has a leader (*Coordinator*, shown by a hand in the diagram), but the other two group do not. The *Realms* consist of several users and the game's AI is formed of the different *Computers* playing against the users.

Figure 16. Group Hierarchy Diagram (GHD)

4.2.2 Specifying group-level goals and system main tasks (G2)

After defining GHD, the SGD (System Goals Diagram) was created to specify the actors' goals (guideline G2). Figure 17 shows that the tasks *Play skirmish* and *Rule empire* (the two main system tasks in this game) are decomposed into different goals related to several *groupactors*. For instance, the goal *Provide quality service* is achieved by the participation of several *Realms* of players (a realm consists of a group of players, usually from the same country) and an *Administration board* (all the game administrators). *Provide quality service* will affect the *Realms*, whose players will be provided with a better gaming experience (GX) [68]. On the other hand, the game administrators should be interested in achieving this goal because the better the quality provided, the more people will play the game. As can be seen, because of the cardinality in the participation links, one or more *Realms* could participate in achieving this goal, but only one *Administration Board*.

Figure 17. System Goals Diagram (SGD)

4.2.3 Defining roles and responsibilities (G3)

Once the GHD and SGD have been defined, the main system tasks have to be refined by using the Responsibility Diagrams (RDs) (guideline G3). Note that in this work we focus on *Play Skirmish*, because it contains the most collaborative aspects of the game. In the diagram shown in Figure 18, the *Play Skirmish* task is decomposed into subtasks and quality softgoals and the responsibility assignments are established by means of responsibility links. The *Skirmish player* role (played during a skirmish) is responsible for the *Play Skirmish* task and also for the *Attack, Build* and *Populate* sub-tasks. *User* also plays the *Skirmish player* in other sub-tasks such as *Attack* or *Develop Economy*.

Figure 18. Responsibility Diagram (RD) for *Skirmish* sub-system

4.2.4 Modeling tasks, collaboration and awareness (G4)

Following guideline G4, the next step is to refine each of the four sub-tasks shown in Figure 18 in a different Task Refinement Diagram (TRD). Due to space constraints only two are illustrated here. Figure 19 shows *Develop Economy,* an abstract task refined into individual and non-individual sub-tasks. The roles (played by actors) are assigned to tasks by participation links. The most important awareness characteristics are specified in the TRD diagrams. An awareness resource specifies knowledge about the state of the workspace and this information is relevant to the roles involved in the tasks. For instance, in Figure 19 and Figure 20, the resources provide relevant information for perception, comprehension and prediction in the workspace.

Figure 19. Task Refinement Diagram (TRD) for *Develop economy* task

Figure 20. Task Refinement Diagram (TRD) for *Attack* task

In Figure 19, from one to three *Allies* and a *Skirmish player* are needed to perform the *Do external trade* task. Note that the participation link from the *Skirmish player* has an awareness resource (*Ally's base visualization*). This is because to create a trading route between an ally's base and the player's base, the Skirmish player needs to be aware of the status of the other allies' bases (the player needs to locate the ally's town halls and choose the most appropriate to create the trading route). Thus, it is mandatory (M) to know the allies' town halls and their location (What – Artifact, and Where - Location). Also, it is desirable (D) to know who is the owner of each city hall (Who - Identity) and it would be nice (N) to have information on whether a user is building a new city hall. It can be observed that another awareness resource (*Private chat*) is related to the *Deal* task, thus helping the players to decide the amount of each resource to be sent to his allies.

However, awareness resources are not only related to the *Develop Economy* TRD. In the Attack TRD (see Figure 20) the need for feedback from other players is even higher. In order to coordinate troops before a battle, the players can use the chat (Team chat, similar to the one illustrated in the previous TRD, but here shown in a reduced form) and once this task is done, the players can find out where their allies are going to attack by Flaring the map (see Figure 15). Once the player knows that his allies are ready to attack by looking at their *Telepointers*, he can do the same. During this attack, he will receive *Audio feedback* from his allies' and his own troops. In this diagram it can also be seen how CSRMF facilitates the specification of the task's different importance levels, e.g. *Hunt* has normal importance, but *Begin attack* has the highest. Although the importance level corresponds to the tasks' development priority, in this case it also corresponds to Gaming eXperience (GX). Not implementing the *Hunt* functionality will not seriously affect GX, but it makes no sense to leave out *Attacks* in a Real Time Strategy (RTS) game.

The awareness resources specified in the previous diagrams can guide developers in choosing an appropriate design solution to satisfy certain awareness needs. For example, audio surround feedback can be considered appropriate to satisfy the *Audio feedback* awareness resource in the CSRMF requirements specification, since it provides the user with useful information about important events during the game, such as reporting when and where the player's base is being attacked. Nevertheless, other awareness resources, such as the *Ally base visualization*, could not be properly implemented with the design solution chosen in AoEO. In fact, the *Ally base visualization* awareness resource could be properly satisfied by including information on the map about the distance to the allies' base, because it is really interesting for a player to know this information when creating a new commercial route. If the game designers had used an RE technique focused on CSCW (supporting 3C tasks and WA) such as CSRMF, they would have been able to provide straightforward information to developers about the user's feedback requirements and they would have used better design solutions. Furthermore, the awareness resources included in the requirements specification could be used to implement game-difficulty selection based on the feedback provided to the user (currently, it is not available in this game). For example, the *easy* level could provide a full view of the battlefield on the map (the enemy's movements are always visible), but the *normal* level would add the typical "fog of war" of real-time strategy games. The map could be removed in the *hard* level, so that the players would only be aware of enemy attacks from the audio feedback.

4.2.5 Establishing system's quality factors (G5)

Figure 21 shows the last CSRMF capability, which is the quality interaction characteristics that can be modeled by theQFD (Quality Factors Diagram) as in guideline G5. This model specifies the quality factors that contribute to achieving a high quality gaming experience (quality of interaction). This time, as we were unable to determine the AoEO quality requirements, in this diagram a quality evaluation framework is modeled based on the factors and criteria of ISO/IEC 25010:2011 [58] standard, modified for game quality evaluation (for instance, entertainment instead of productivity). This diagram has some quality factors and criteria which are represented as softgoals and are related to the main quality softgoal by means of contribution links with positive contributions. Using these quality factors, quality assurance experts would be able to decide whether to include new awareness features. For instance, will the inclusion of a new awareness element that provides feedback on other allies' troops and resources facilitate collaboration between users or will it overload the user interface?

Figure 21. Quality Factors Diagram (QFD) for *Play Skirmish* sub-system

4.2.6 System diagrams overview (G0)

Figure 22 gives a a graphical overview of the resulting requirements model and summarizes the diagrams modeled by following the G1-G5 design guidelines, following the hierarchy of diagrams outlined in guideline G0.

Figure 22. Overview of the modeled diagrams

As can be seen, throughout the previous specifications, the special expressiveness needs of the requirements of CSCW systems (in a multi-player game) can be specified by means of the CSRMF guidelines, including the description of actors and groups, specification of awareness resources, tasks, subtasks, goals and quality. When the participants ended the last iteration of the AoEO requirements specification, they were questioned on their experience while modeling with CSRMF. The five engineers answered that they found CSRMF powerful enough to specify a real awareness-demanding CSCW system and found the guidelines particularly helpful. They also stated that the CSRML language has enough expressive power to specify CSCW systems, especially from the support and automatic validation offered by the CSRML Tool. They also appreciated the tool's context-sensitive help in supporting novice engineers when using CSRMF. They also suggested the definition of heuristics that could help novice engineers to detect the awareness needs of CSCW systems. This first evaluation of CSRMF thus means we can answer positively the fourth research question (RQ4). We now plan to conduct further evaluations with larger test groups to obtain additional feedback on the framework when used to specify other kinds of CSCW systems. We particularly wish to evaluate the use of CSRMF in the context of business intelligence systems.

5 Results and Further Work

In any software system, the requirements specification is paramount in a successful development and CSCW (Computer Supported Cooperative Work) is no exception. This type of system has some special requirements which are difficult to represent by traditional Requirements Engineering techniques [50], [69]. The aim of the present study was therefore to develop a new metamodel-based framework for the specification of CSCW requirements. Based on the Goal-Oriented CSRML (Collaborative Systems Requirements Modeling Framework) Requirement Engineering language [31], this CSRMF (Collaborative Systems Requirements Modeling Framework) consists of three main components: i) a requirements modeling language that is a new version of CSRML and allows analysts to specify user collaboration and workspace awareness requirements. ii) The specification is conducted by applying a set of guidelines that allow analysts to model the requirements of a whole CSCW system by means of 5 different types of diagram. iii) The CSRML CASE tool facilitates CSCW requirements modeling by creating and validating requirements specifications that comply with the CSRMF metamodel. This framework can thus specifiy the most complex features of CSCW systems, such as the degree of collaboration among users and the underlying awareness needs. A case study was carried out based on a cooperative multiplayer online game, highly demanding in terms of Workspace Awareness (WA) and the results highlighted the expressive power of this framework when dealing with CSCW requirements.

To sum up, the proposed framework facilitates and guides the specification of complex CSCW systems by providing: (RQ1) a highly expressive and empirically validated goal-oriented language able to specify the paramount features of a collaborative application, such as collaboration among users, awareness needs and the management of actors, roles and groups; (RQ2) a set of guidelines aimed at leading a CSCW requirements model step by step, helping to identify roles, tasks, actors and so on; and (RQ3) a CASE tool supporting the RE modeling process while using CSRMF, being able to edit and verify the five different kinds of diagrams available in CSRML. In addition, owing to the system modeled by the five SE practitioners described in Section 4 (RQ4), we are able to state that our research questions have been answered.

Our on-going work consists of three main goals: on one hand we are focusing on improving and extending the CSRML Tool by integrating CSRMF into CIAM [10], a methodological proposal for the development of user interfaces for CSCW systems. Currently, CIAM exploits Use Cases as the requirement specification technique, but no attention has yet been paid to WA at the RE stage. The integration of CSRMF will thus provide stakeholders with full support for WA features, paramount in the development of collaborative systems. Additionally, a different CSRMF CASE tool is being developed based on Eclipse EMF/GMF, so that the EMF and the current Visual Studio tool will be empirically compared when the new one reaches the necessary maturity level.

One of this work's limitations is the lack of a formal assessment of the comprehensibility of the framework, although most of its components have already been assessed individually. To address this limitation, a family of experiments is now being performed to fully assess the comprehensibility of the framework, beyond the case studies that have shown the feasibility and expressive power of the approach. Finally, in future work, the CSRML Tool is going to be extended to provide it with collaborative requirement management features [70], thus enabling distributed work teams to collaborate on CSCW requirements specifications.

We are also working on extending the CSRMF design guidelines described in this paper. Anton and Potts' heuristics [71], which are focused on identifying Goal-Oriented elements, will be extended to deal with the specific characteristics of CSCW systems. Once this has been completed, we will integrate this new support into CSRMF, thus obtaining a complete methodology for the specification of CSCW systems.

Acknowledgements. This research has been funded by the Spanish Ministry of Economy and Competitiveness and by the FEDER funds of the EU under the projects grants TIN2012-34003 and TIN2015-72931-EXP. It has also been funded by the Spanish Ministry of Education, Culture and Sports with an FPU scholarship (AP2010-0259).

References

[1] Collins. Collins English Dictionary - Complete & Unabridged, 6th ed. Collins, 2003.

[2] Alexa Internet Inc. The top 500 sites on the web, 2014. [Online]. Available: http://www.alexa.com/topsites. [Accessed: 11-Feb-2014].

[3] Medina-Dominguez F, Sanchez-Segura M-I, Moreno A, and Santin D. Collaborative tools: computer science students' skills versus software industry needs. Journal of Software: Evolution and Process 2015; 27 (3): 221–235. DOI: 10.1002/smr.1708.

[4] Ducheneaut N, Moore RJ, and Nickell E. Virtual "Third Places": A Case Study of Sociability in Massively Multiplayer Games. Computer Supported Cooperative Work (CSCW) 2007; 16 (1–2): 129–166. DOI: 10.1007/s10606-007-9041-8.

[5] Schmidt K and Bannon L. Taking CSCW seriously. Computer Supported Cooperative Work (CSCW) 1992; 1 (1): 7–40. DOI: 10.1007/BF00752449.

[6] Bannon L and Schmidt K. CSCW: Four Characters in Search of a Context, in Studies in computer supported cooperative work, J. M. Bowers and S. D. Benford, Eds. Amsterdam, The Netherlands: North-Holland Publishing Co, 1990, 3–16.

[7] Ellis CA, Gibbs SJ, and Rein G. Groupware: some issues and experiences. Communications of the ACM 1991; 34 (1): 39–58. DOI: 10.1145/99977.99987.

[8] Google Inc. Google Docs, 2001. [Online]. Available: https://docs.google.com. [Accessed: 11-Feb-2014].

[9] Garrido JL. AMENITIES: una Metodología para el Desarrollo de Sistemas Cooperativos Basada en Modelos de Comportamiento y Tareas, Dissertation, University of Granada, 2003.

[10] Molina AI, Redondo MA, Ortega M, and Hoppe U. CIAM: A methodology for the development of groupware user interfaces. Journal of Universal Computer Science 2008; 14 (9): 1435–1446. DOI: 10.3217/jucs-014-09-1435.

[11] Fuks H, Raposo AB, Gerosa M a., and Lucena CJP. Applying the 3C Model To Groupware Development. International Journal of Cooperative Information Systems 2005; 14 (02n03): 299–328. DOI: 10.1142/S0218843005001171.

[12] Schmidt K. The Problem with "Awareness": Introductory Remarks on "Awareness in CSCW." Computer Supported Cooperative Work (CSCW) 2002; 11 (3): 285–298. DOI: 10.1023/A:1021272909573.

[13] Park J, Kang T, and Yeom K. Mobile situation-aware framework for developing smart mobile software. Journal of Software: Evolution and Process 2014; 26 (12): 1213–1232. DOI: 10.1002/smr.1658.

[14] Gutwin C and Greenberg S. The effects of workspace awareness support on the usability of real-time distributed groupware. ACM Transactions on Computer-Human Interaction 1999; 6 (3): 243–281. DOI: 10.1145/329693.329696.

[15] Gutwin C and Greenberg S. A Descriptive Framework of Workspace Awareness for Real-Time Groupware. Computer Supported Cooperative Work 2002; 11 (3): 411–446. DOI: 10.1023/A:1021271517844.

[16] Tan DS, Gergle D, Mandryk R, Inkpen K, Kellar M, Hawkey K, and Czerwinski M. Using job-shop scheduling tasks for evaluating collocated collaboration. Personal and Ubiquitous Computing 2007; 12 (3): 255–267. DOI: 10.1007/s00779-007-0154-3.
[17] Selçuk YE and Erdoğan N. Role models-implementation-based approaches to using roles. Software: Practice and Experience 2011; 41 (1): 1–22. DOI: 10.1002/spe.994.
[18] Omoronyia I, Ferguson J, Roper M, and Wood M. A review of awareness in distributed collaborative software engineering. Software: Practice and Experience 2010; 40 (12): 1107–1133. DOI: 10.1002/spe.1005.
[19] Teruel MA, Navarro E, López-Jaquero V, Montero F, Jaen J, and González P. Analyzing the Understandability of Requirements Engineering Languages for CSCW Systems: A Family of Experiments. Information and Software Technology 2012; 54 (11): 1215–1228. DOI: 10.1016/j.infsof.2012.06.001.
[20] Schmidt K. Cooperative Work and Coordinative Practices. London, UK: Springer Verlag, 2011, 1–472.
[21] Carstensen PH and Schmidt K. Computer Supported Cooperative Work: New Challenges to Systems Design, in Handbook of Human Factors, K. Itoh, Ed. Tokyo, Japan: Asakura Publishing, 1999, 1–24.
[22] De Choudhury M, Diakopoulos N, and Naaman M. Unfolding the event landscape on twitter, in ACM 2012 conference on Computer Supported Cooperative Work (CSCW'12), 2012, 241–244.
[23] De Lucia A, Francese R, Passero I, and Tortora G. Development and evaluation of a system enhancing Second Life to support synchronous role-based collaborative learning. Software: Practice and Experience 2009; 39 (12): 1025–1054. DOI: 10.1002/spe.926.
[24] Tran MH, Raikundalia GK, and Yang Y. Using an experimental study to develop group awareness support for real-time distributed collaborative writing. Information and Software Technology 2006; 48 (11): 1006–1024. DOI: 10.1016/j.infsof.2005.12.009.
[25] Stahl G, Koschmann T, and Suthers D. Computer-supported collaborative learning: An historical perspective, in Cambridge Handbook of the Learning Sciences, 1st ed., R. K. Sawyer, Ed. Cambridge University Press, 2006, 409–426.
[26] Dourish P and Bellotti V. Awareness and coordination in shared workspaces, in ACM conference on Computer-supported cooperative work (CSCW'92), 1992, 107–114.
[27] TheCodingMonkeys. SubEthaEdit, 2013. [Online]. Available: http://www.codingmonkeys.de/subethaedit/. [Accessed: 11-Feb-2014].
[28] Gutwin C, Roseman M, and Greenberg S. A usability study of awareness widgets in a shared workspace groupware system, in ACM Conference on Computer Supported Cooperative Work (CSCW'96), 1996, 258–267.
[29] Figueroa-Martinez J, López-Jaquero V, Gutiérrez Vela FL, and González P. Enriching UsiXML language to support awareness requirements. Science of Computer Programming 2013; 78 (11): 2259–2267. DOI: 10.1016/j.scico.2012.09.020.
[30] Limbourg Q, Vanderdonckt J, Michotte B, Bouillon L, and López-Jaquero V. USIXML: A Language Supporting Multi-path Development of User Interfaces, in Engineering Human Computer Interaction and Interactive Systems, R. Bastide, P. Palanque, and J. Roth, Eds. Springer Berlin Heidelberg, 2004, 200–220.
[31] Teruel MA, Navarro E, López-Jaquero V, Montero F, and González P. CSRML: A Goal-Oriented Approach to Model Requirements for Collaborative Systems, in 30th International Conference on Conceptual Modeling (ER'11), 2011, 33–46.
[32] Yu ESK. Towards modelling and reasoning support for early-phase requirements engineering, in 3rd IEEE International Symposium on Requirements Engineering (ISRE'97), 1997, 226–235.
[33] Noguera M, González M, Garrido JL, Hurtado V, and Rodríguez M. System Modeling for Systematic Development of Groupware Applications, in International Conference on Software Engineering Research and Practice (SERP'06), 2006, 750–756.

[34] Viguie P, Zarate P, and Soubie J-L. CSCW implementation in enterprise: The Nibbling method, in 14th International Conference on Computer Supported Cooperative Work in Design (CSCWD'10), 2010, 77–82.
[35] Pinelle D, Gutwin C, and Greenberg S. Task analysis for groupware usability evaluation. ACM Transactions on Computer-Human Interaction 2003; 10 (4): 281–311. DOI: 10.1145/966930.966932.
[36] Antunes P. A Design Framework for Mobile Collaboration, in Encyclopedia of E-Collaboration, N. Kock, Ed. IGI Global Snippet, 2008, 133–138.
[37] Molina AI, Gallardo J, Redondo MA, Ortega M, and Giraldo WJ. Metamodel-driven definition of a visual modeling language for specifying interactive groupware applications: An empirical study. Journal of Systems and Software 2013; 86 (7): 1772–1789. DOI: 10.1016/j.jss.2012.07.049.
[38] Durán Toro A, Bernárdez Jimenez B, Ruiz Cortes A, and Toro Bonilla M. A Requirements Elicitation Approach Based in Templates and Patterns, in Workshop in Requirements Engineering (WER'99), 1999, 17–29.
[39] Sutton DC. Linguistic Problems with Requirements and Knowledge Elicitation. Requirements Engineering 2000; 5 (2): 114–124. DOI: 10.1007/PL00010344.
[40] Booch G, Rumbaugh J, and Jacobson I. The Unified Modeling Language User Guide, 2nd ed. Addison-Wesley Professional, 2005, 1–496.
[41] Pohl K. Requirements Engineering: Fundamentals, Principles, and Techniques, 1st ed. Springer, 2010.
[42] Cysneiros LM and Yu ES-K. Non-Functional Requirements Elicitation, in Perspectives on Software Requirements, J. C. S. do Prado Leite and J. H. Doorn, Eds. Springer US, 2004, 115–138.
[43] Van Lamsweerde A. Goal-oriented requirements engineering: a guided tour, in 5th IEEE International Symposium on Requirements Engineering (RE'01), 2001, 249–262.
[44] Kavakli E. Goal-Oriented Requirements Engineering: A Unifying Framework. Requirements Engineering 2002; 6 (4): 237–251. DOI: 10.1007/PL00010362.
[45] Cares C, Franch X, López L, and Marco J. Definition and Uses of the i* Metamodel, in 4th International i* Workshop (iStar'10), 2010, 20–25.
[46] Maté A, Trujillo J, and Franch X. A Modularization Proposal for Goal-Oriented: Analysis of Data Warehouses using I-star, in 30th International Conference on Conceptual Modeling (ER'11), 2011, 421–428.
[47] Yu ES-K and Yu Y. Organization Modelling Environment (OME), University of Toronto, 2000. [Online]. Available: http://www.cs.toronto.edu/km/ome/. [Accessed: 11-Feb-2014].
[48] Horkoff J, Yu Y, and Yu ES-K. OpenOME: An Open-source Goal and Agent-Oriented Model Drawing and Analysis Tool, in 5th International i* Workshop (iStar'11), 2011, 20 (iStar):, 154–156.
[49] Grau G, Franch X, and Avila S. J-PRiM: A Java Tool for a Process Reengineering i* Methodology, in 14th IEEE International Requirements Engineering Conference (RE'06), 2006, 359–360.
[50] Teruel MA, Navarro E, López-Jaquero V, Montero F, and González P. Comparing Goal-Oriented Approaches to Model Requirements for CSCW, in Evaluation of Novel Approaches to Software Engineering, L. A. Maciaszek and K. Zhang, Eds. Springer-Verlag, 2012, 169–184.
[51] Ackerman M. The Intellectual Challenge of CSCW: The Gap Between Social Requirements and Technical Feasibility. Human-Computer Interaction 2000; 15 (2): 179–203. DOI: 10.1207/S15327051HCI1523_5.
[52] Coughlan J and Macredie RD. Effective Communication in Requirements Elicitation: A Comparison of Methodologies. Requirements Engineering 2002; 7 (2): 47–60. DOI: 10.1007/s007660200004.

[53] Rahman NA and Sahibuddin S. A Systematic Mapping Study on Requirements Elicitation Techniques in Collaborative Application. Lecture Notes on Software Engineering 2013; 1 (1): 106–111. DOI: 10.7763/LNSE.2013.V1.24.

[54] Dourish P. Re-Space-ing place: “Place” and “Space” Ten Years On, in ACM Conference on Computer Supported Cooperative Work (CSCW’06), 2006, 299–308.

[55] Moody DL. The “Physics” of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering. IEEE Transactions on Software Engineering 2009; 35 (6): 756–779. DOI: 10.1109/TSE.2009.67.

[56] Teruel MA, Navarro E, González P, López-Jaquero V, and Montero F. Applying Thematic Analysis to Define an Awareness Interpretation for Collaborative Computer Games. Information and Software Technology 2016; 74 17–44. DOI: 10.1016/j.infsof.2016.01.009.

[57] Horkoff J and Yu ES-K. Comparison and evaluation of goal-oriented satisfaction analysis techniques. Requirements Engineering 2012; 18 (3): 199–222. DOI: 10.1007/s00766-011-0143-y.

[58] ISO/IEC 25010. Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models, 2011.

[59] Carrillo de Gea JM, Nicolás J, Fernández Alemán JL, Toval A, Ebert C, and Vizcaíno A. Requirements engineering tools: Capabilities, survey and assessment. Information and Software Technology 2012; 54 (10): 1142–1157. DOI: 10.1016/j.infsof.2012.04.005.

[60] Teruel MA, Navarro E, López-Jaquero V, Montero F, and González P. A CSCW Requirements Engineering CASE Tool: Development and Usability Evaluation. Information and Software Technology 2014; 56 (8): 922–949. DOI: 10.1016/j.infsof.2014.02.009.

[61] Agarwal A and Meyer A. Beyond usability: evaluating emotional response as an integral part of the user experience, in 27th international conference extended abstracts on Human factors in computing systems (CHI’09), 2009, 2919–2930.

[62] Lund AM. Measuring Usability with the USE Questionnaire. STC Usability SIG Newsletter 2001; 8 (2): 3–6. .

[63] ISO/IEC 25062. Software engineering—Software product Quality Requirements and Evaluation (SQuaRE)—Common Industry Format (CIF) for usability test reports, 2006.

[64] Microsoft Corporation. Age of Empires Online, 2012. [Online]. Available: http://www.ageofempiresonline.com. [Accessed: 11-Feb-2014].

[65] Cockburn A, Karlson A, and Bederson BB. A review of overview+detail, zooming, and focus+context interfaces. ACM Computing Surveys 2008; 41 (1): 1–31. DOI: 10.1145/1456650.1456652.

[66] Gutwin C, Greenberg S, and Roseman M. Workspace Awareness Support with Radar Views, in Conference companion on Human factors in computing systems common ground (CHI’96), 1996, 210–211.

[67] Gutwin C, Greenberg S, and Roseman M. Workspace Awareness in Real-Time Distributed Groupware: Framework, Widgets, and Evaluation, in HCI on People and Computers XI (HCI’96), 1996, 281–298.

[68] Poels K, de Kort Y, and Ijsselsteijn W. “It is always a lot of fun!”: exploring dimensions of digital game experience using focus group methodology, in Conference on Future Play (Future Play’07), 2007, 83–89.

[69] Teruel MA, Navarro E, López-Jaquero V, Montero F, and González P. An Empirical Evaluation of Requirement Engineering Techniques for Collaborative Systems, in 15th Annual Conference on Evaluation & Assessment in Software Engineering (EASE’11), 2011, 114–123.

[70] Lang M and Duggan J. A Tool to Support Collaborative Software Requirements Management. Requirements Engineering 2001; 6 (3): 161–172. .

[71] Antón AI and Potts C. The Use of Goals to Surface Requirements for Evolving Systems, in 20th International Conference on Software Enginering (ICSE’98), 1998, 157–166.