The Journal of Systems and Software 89 (2014) 146–169

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Reviewing the quality of awareness support in collaborative applications

Pedro Antunes[a], Valeria Herskovic[b,*], Sergio F. Ochoa[c], José A. Pino[c]

[a] School of Information Management, Victoria University of Wellington, 23 Lambton Quay, Wellington, New Zealand
[b] Computer Science Department, Pontificia Universidad Católica de Chile, Av. V. Mackenna 4860, Santiago, Chile
[c] Computer Science Department, Universidad de Chile, Blanco Encalada 2120, Santiago de Chile, Chile

ARTICLE INFO

Article history:
Received 1 February 2013
Received in revised form 27 October 2013
Accepted 7 November 2013
Available online 15 November 2013

Keywords:
Awareness
Formative evaluation
Collaborative applications
Software quality

ABSTRACT

Awareness to users is a valuable feature of a collaborative system. Therefore, the designers of a system of this type may find it useful to receive hints on the awareness support provided by the system when it is under development or evolution. This paper proposes a tool for their use to obtain suggestions on the awareness features provided by the system and those not currently supported by it. The considered kinds of awareness were obtained from a review of a significant number of proposals from the literature. The tool is based on a checklist of design elements related to these awareness types to be applied by the application designer. The construction of this checklist was done as follows. The process started with an analysis of the types of awareness to be provided. This step ended with 54 selected design elements and six awareness types. Experts on the development of collaborative systems used their experience to provide correlations between the design elements and the types of awareness previously identified, thus encapsulating their expertise within the checklist. The proposal was applied to three existing collaborative systems and the results are presented. The obtained results suggest that the checklist is adequate to provide helpful hints that may be used to improve an application's awareness support.

1. Introduction

The words aware, cognizant, conscious, sensible, alive and awake mean having knowledge of something, according to the dictionary. *Aware* implies vigilance in observing or alertness in drawing inferences from what one experiences (Merriam-Webster, 2011). The term has been used at least since the 12th century. Much work has been done on consciousness/awareness in various areas, such as psychology and neuroscience. It even has philosophical implications, as pointed out by the Santiago theory of cognition (Maturana and Varela, 1980).

In the area of Computer Supported Collaborative Work (CSCW), Dourish and Bellotti (1992a,b) introduced the awareness term in their seminal paper. They defined it as "an understanding of the activities of others, which provides a context for your own activity." Various types of information can be included for this "understanding." Their classification is reviewed in Section 3.

Provision of awareness has proved to be a useful feature of collaborative systems (Gutwin and Greenberg, 1998, 1999, 2002; Salmon et al., 2007; Bardram and Hansen, 2010; Talaei-Khoei et al., 2011; Xiao, 2013). Awareness has been a focus of research in CSCW for over thirty years, and a thorough overview of its history was presented by Rittenbruch and McEwan (2009). Nevertheless, little research has been done to assess the quality of awareness support provided by a specific system. This assessment would be useful for concerned developers of a system encapsulating collaborative features or mobile work scenarios (Gavalas et al., 2011). Managers or users of this type of system may also be interested in knowing the extent of awareness support a system to be acquired provides.

A simple approach to do this assessment is by asking users about it. After all, users in a general sense are the final judges on the quality of a system. Questionnaires can be used for asking users after experiencing a system (MacMillan et al., 2004). Alternatively, user observation can be useful to understand how awareness is constructed. The analysis of logged interactions (Nacenta et al., 2007) and video recordings (Hornecker et al., 2008) may also provide some answers to the evaluation of awareness support. However, all these evaluation strategies require the actual participation of a group of users. Unfortunately, the participation of users is not always possible or available at the time of evaluation (Holzinger, 2005). In other scenarios, actual user participation may be expensive, if users must leave their current system aside and begin to use a candidate system just to evaluate it, including adequate training, real operation, data conversion, etc.

The approach reported in this paper does not require mandatory user participation, but requires counting on a prototype, or design,

* Corresponding author. Tel.: +56 2 23547599.
E-mail address: vherskov@ing.puc.cl (V. Herskovic).

0164-1212/$ – see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jss.2013.11.1078

of the system to be evaluated. We propose an *awareness checklist* that may be useful to obtain hints on the awareness support of collaborative applications at various stages; for instance, during the development process or during the system evolution. The checklist may be particularly convenient to use when the development team members have some knowledge but are not experts in CSCW, because it will provide a large set of suggestions from which to choose the relevant ones. That is, the checklist acts as a reminder of the types of awareness elements that a development team should consider. It should be noted that this only partially responds to the need for assessing the quality of awareness support, since – although other stakeholders are encouraged to participate – it is oriented to one main type of stakeholder: a member of the development team who plays the role of *software designer*. Therefore, the checklist does not replace traditional usability evaluation, but may complement it.

Given this orientation, we analyze what awareness really means and the types of awareness it is possible to distinguish: we conclude that six types of awareness are important to consider. On the other hand, we try to identify the relevant software design elements: 54 of them are acknowledged. Both awareness types and design elements were obtained from a literature review. A key step is afterwards to define the correlations between the design elements and the awareness types; this step is done with the help of experienced CSCW system developers. The checklist is then built with these correlations. It is important to note that the goal is not encouraging developers to incorporate unnecessary features to an application, but rather to encourage reflection about which awareness elements would be valuable in a particular scenario. The checklist format does have some inherent limitations, e.g. it provides an overview of lacking features, but does not provide detailed feedback about usability or usefulness.

The paper continues with a review of related work (Section 2). Section 3 deals with the awareness types. Section 4 presents the proposed checklist. The use of this checklist in three cases is illustrated in Section 5. Finally, Section 6 discusses the proposed approach and Section 7 concludes the paper with a summary of the obtained results.

2. Related work

2.1. Evaluation of awareness support

Evaluation of the quality of awareness support can be traced back to Formal Technical Reviews (FTR) (Fagan, 1976). They have been widely adopted in software engineering (Aurum et al., 2002; Neill and Laplante, 2003). The reviews involve several people in a formal meeting during which a software artifact is presented, discussed and approved. FTR seek to identify defects and discrepancies in the software against plans, specifications, standards and best practices. They cover the whole software development lifecycle (Laitenberger and DeBaud, 2000). Yet it is interesting to note the FTR of early life-cycle artifacts is not commonly practiced in the software industry, apparently because they seem to lack maturity (Laitenberger and DeBaud, 2000).

Johnson (1998) analyzed the impact of software reviews on quality, showing that defects can be one or two orders of magnitude less costly to remove when found in initial development stages than after distribution to the customers. Moreover, software reviews were considered effective for discovering certain soft, but nevertheless costly, defects such as logically correct but poorly structured code.

CSCW brought together two main organizational assets: technology and humans. The development of CSCW systems has for long been considered a special branch of software development concerned with: design challenges associated with organizational goals; group characteristics and dynamics; communication, coordination and collaboration; conflict resolution and decision making; social context of work; and positive and negative effects of technology on tasks, groups and organizations.

Evaluation is essential to ensure the quality of collaborative systems developments. The problem now is that evaluation must assess a very wide range of factors related with multiple stakeholders (customers, managers, individual workers, formal and informal work groups), various domains of concern (business processes, goals, tasks, group well-being, culture, just to name a few) and multiple technology components (addressing various aspects of human-oriented activities such as communication, coordination, collaboration, and of course awareness). All in all, what distinguishes evaluation in the CSCW context is the need to assess the technology impact with an eclectic perspective.

Research shows that collaborative systems evaluation is difficult to accomplish. The first reason is the complexity, cost and time involved (Antunes et al., 2012). Second, the assessments tend to be informal (Pinelle and Gutwin, 2000). A prior study revealed that almost one third of systems are not assessed in a formal way (Pinelle and Gutwin, 2000). A more recent study (Antunes and Pino, 2010) found out that only 25% of the studies adopted a positivistic assessment (encompassing laboratory experiments, surveys, empirical methods, formative evaluation, simulation and analytic methods).

Third, CSCW involves conflicting views over technology and its impact in organizations, which may require diverse assessment methods. Herskovic et al. (2007) identified twelve methods and classified them according to various criteria such as development status, scope, time span of the assessment and who participates in the assessment. Of these twelve methods, six require the participation of end users in several ways, like focus groups and observations. However, significant participation of end users in systems evaluation turns the process costly and quite difficult to manage.

Of the remaining six methods, three require modeling and analyzing the system functionality at a very low level of detail. And finally the remaining methods adapt the FTR approach to the specific CSCW context. The methods are: Groupware Heuristic Evaluation (GHE) (Baker et al., 2002), Groupware Walkthrough (GW) (Pinelle and Gutwin, 2002) and Knowledge Management Approach (KMA) (Vizcaíno et al., 2005). GHE defines a procedure for inspecting how a collaborative system conforms with eight heuristics that codify best practices in collaborative systems development (2002). GW entails stepping through task sequences to conceptually explore task goals, actions necessary to perform tasks, knowledge needed to accomplish tasks, and possible performance failures (Gutwin and Greenberg, 2000; Pinelle and Gutwin, 2002). Finally, KMA involves using a checklist to assess how the system helps knowledge circulation (Vizcaíno et al., 2005).

2.2. Evaluation and awareness

We will now delve into the three FTR methods mentioned above to unravel how they address the quality of awareness support. As previously mentioned, GHE systematizes evaluation activities around a set of heuristics (Baker et al., 2002). These heuristics define a checklist with qualities that a collaborative system should have. Six of these heuristics point toward the importance of awareness, e.g. *Provide consequential communication of an individual's embodiment*, giving awareness of who is in the workspace and what they are doing, *Provide consequential communication of shared artifacts*, highlighting what artifacts are present in the workspace, as well as the manipulations done by the users on those artifacts, and *Facilitate finding collaborators and establishing contact*, giving indications about who belongs to the group and who is around.