SCTA Tracer: A Distributed Environment for Standardized Awareness Support Assessments

Christoph Oemig, Tom Gross
Human-Computer Interaction Group
University of Bamberg
96047 Bamberg, Germany
c.oemig(at)acm.org, tom.gross(at)uni-bamberg.de

Abstract

Awareness support in cooperative environments has been a research issue in the area of distributed systems for computer-supported cooperative work for more than two decades. Measuring its effectiveness remains a complex task since it is difficult to grasp awareness in situ. Consequently, techniques and tools are required generating results while a user's awareness is still present. The Standardized Coordination Task Assessment (SCTA) and its tool the SCTA Tracer use freeze probes to query participants at specific points in time while working on a common task measuring and recording response times, performance, and errors. The result is visualized in a four quadrant system distinguishing illusive, ineffective, inefficient, and ideal systems. The SCTA Tracer guides awareness support researchers and designers to focus their effort on essential concepts already at early development stages. This paper shows how a smart selection of tools and techniques is integrated for this complex task.

1. Introduction

Awareness and awareness support in cooperative environments have been a research issue in the area of distributed systems for computer-supported cooperative work for more than two decades. Basically, awareness in this context means that the distributed members of a team get information about each other and each others' activities as well as relevant changes to shared workspaces and artefacts.

Studies and prototypes (e.g., group editors [1]) showed its positive effects on coordination in work groups. Over the time, many facets of awareness have been identified [2]. Yet, measuring the effectiveness of awareness support has remained a complex task—as it is to evaluate cooperative systems in general [3]. Although there are many individual approaches and methods, the lack of standard tools urges researchers to create their own. The result: specialized one-purpose tools, approaches and history repeating itself with next research effort.

In this paper we present an awareness assessment approach providing a generic tool to be used across our various research projects, especially those dealing with the development of awareness support systems. Taking a closer look at awareness itself reveals two major challenges: Awareness is ephemeral by nature and it is a secondary task. The first deals with the issue of memory: People tend to forget quickly. As one of the first, Hermann Ebbinghaus discovered and documented the exponential nature of forgetting [4] describing the decline of memory retention over time. Consequently, there is only limited time to measure awareness in order to judge on the effectiveness of an awareness support system. Therefore, we seek to measure awareness when it is still present—even in distributed settings. The second challenge addresses the fact that awareness itself is a by-product and that another process or task is needed for its creation. Many existing methods and tools disregard at least one of the two challenges. Reasons are wrong measurement timing, the absence of users, the high degree of user disruption, or tremendous preparation effort and cost due to very complex setups.

All of the above led us to develop the Standardized Coordination Task Assessment (SCTA), which analyses and categorizes an awareness support's effectiveness among the categories illusive, ineffective, inefficient, and ideal (4I). We implemented these ideas in our software tool SCTA Tracer, which can be used in collocated and distributed setups.

In the following, we briefly introduce its concept, features and application. Then we take a look at the implementation and selected technical details. Finally, we discuss early findings and point out future work.

2. Concept

2.1. Methodology

Since its initial publication in [5] the Standardized Coordination Task Assessment (SCTA-4I) grounds on the hypothesis that if somebody is aware of something, then s/he can answer questions about it quickly and without error. It consists of a standardized (primary) task and a measurement approach (for the secondary) that eventually yields a result depictable in the 4I (illusive, ineffective, inefficient, ideal) diagram. The task itself is simple and merely involves the counting of letters. Thus we achieved low preparation and setup cost/effort using a random string generator while ensuring a comparable quality and workload for the assessments. It has its roots in the research concerned with subliminal messages [6] where the counting of Bs is used as primary task. However, our approach not only contains Bs but all letters of the alphabet in upper and lower case.

The randomly ordered letters are counted by a team of at least two people who may be collocated or spatially separated. This is where the coordination effort comes into play. The counting activity and coordination creates the mental load to be measured. A configurable number of freeze probes (i.e., the halt of the task, blanking the screens and then probing the subjects for a short period of time) are used to capture awareness when it is still present. Quick questions (e.g., "Who counted Cs?", "How many Ds?", "Were Es counted?", "Which of the following letters did your partner count?") concerning the counting task are used while measuring response times. Additionally, the number of errors in relation to the number of questions (error rate) is determined. However, an answer being correct is defined as what the user counted and recalls to be counted—not the actual number of letters.

In general, quick response times and low error rates are desirable indicating reasonable awareness support. Opposed to former versions response time/forgetting time ratio and error rate make up the x- and y-axis in our (4I-) visualization. Currently, the x-axis expresses how much of the time that it takes to loose awareness information has passed. 100% of the x-axis means that awareness information is lost. Besides the above measures, the overall performance and the number or coordination errors are recorded. The number of different letters counted indicates a team's performance. Coordination errors occur for instance when team members count the same letters. Again, high performance and a low coordination error rate are desirable and indicate reasonable awareness support.

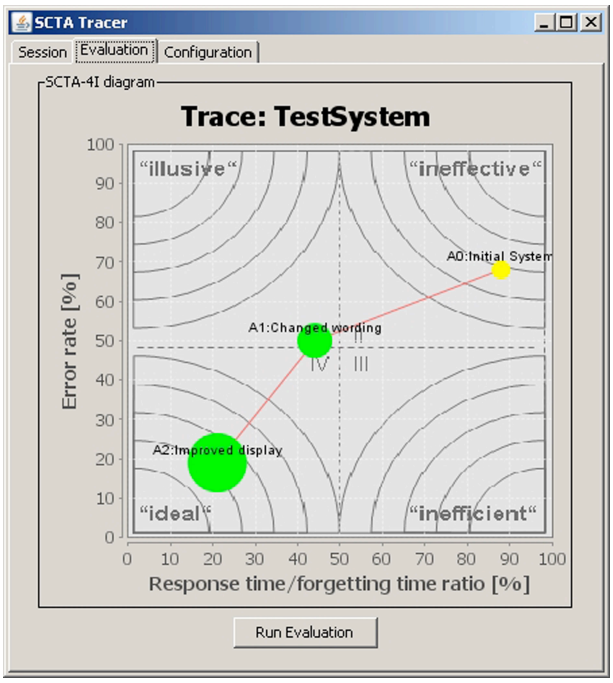

Figure 1. SCTA Tracer showing a 4I diagram.

The SCTA-4I is organized in runs, sessions, and traces:

- **Assessment run**: a single counting task with a configured duration where the four measures are recorded with a set of users. A run is interrupted by a configurable number of freeze probes.
- **Assessment session**: a set of assessment runs whose measures are aggregated to session level. Session results are depicted in the 4I diagram.
- **Assessment trace**: a trace shows the evolutionary path of the awareness support system under test. It consists of a sequence of assessment sessions.

The 4I-diagram is divided into four quadrants (cf. Figure 1). Each quadrant has a label according to the contained system type. Quadrant I contains systems with high error rates and low response time/forgetting time ratios since users present wrong answers quickly indicating illusive systems. Quadrant II encompasses systems with high error rates and high response time/forgetting time ratio indicating ineffective systems since users cannot answer questions correctly even after thinking longer. Quadrant III contains systems with high response time/forgetting time ratios but low error rates where users appear to need some time for thinking but finally come up with correct answers. Quadrant IV has correct answers provided quickly which is the characteristic of ideal systems. In our current visualization assessment sessions are

depicted as circles as opposed to former versions. These circles indicate the coordination error rate. A circle's radius conveys the team's performance (letters counted/configured duration). Reasons for the changes were the comprehension of the diagram and implementation issues. Further changes to prior versions are the use of configuration values more extensively allowing easy adaptations (e.g., using a different number of freeze probes or questions per freeze probe).

2.2. General Procedure & Evaluation

The first user to log in becomes the administrator of the software. S/he may choose loading an existing trace file or generating a new one identified by a trace name. Next, the administrator selects an existing assessment session or creates a new one identified by a session name. Finally, the administration screen opens where the administrator finds an overview of users logged in and where s/he is able to start a new assessment run. Additional tabs contain the trace's evaluation (4I diagram) and the software's configuration. Further users become regular participants of the next assessment run and get to see a wait screen. The wait screen switches to the document screen where the counting takes place when the administrator starts the assessment run. A count down appears prior the switch synchronizing the participants' counting tasks. Additionally, it collects the counting results (e.g., Bs=35) at the lower portion of the screen, which is recorded (later referred to as count data). These are needed to determine the freeze probe results. The counting task in the document screen is interrupted at configured points in time by freeze probes which switch from document screen to freeze probe screen asking the user about the overall counting activity. User, result, and response time are recorded for each answer (later referred to as response data). Afterwards the freeze probe screen switches back to the document screen. When the assessment run's configured time elapsed the application switches to the Thank-you screen from which the user may return to the initial wait screen for another assessment run.

As already mentioned the administration user interface contains an evaluation tab. This tab offers a button named "Run Evaluation" which is only enabled when there are no assessment runs active. Pressing the button launches the computation of the four already introduced measures per assessment session:

- **Average response time/forgetting time ratio**: defined as the arithmetic mean of all response times of all runs belonging to one session in relation to the configured forgetting time. It is derived from the response data:

$$x = \frac{\frac{1}{n}\sum_{r=1}^{n} responsetime_r}{forgettingtime}, \{r_1 \ldots r_n\} \in s \quad (1)$$

- **Error rate**: defined as the quotient of the total number of incorrect answer of all runs belonging to one session divided by the total number of all answers of all runs belonging to the same session. It is derived from the response data:

$$y = \frac{\sum_{r=1}^{n} incorrect_r}{\sum_{r=1}^{n} (incorrect_r + correct_r)}, \{r_1 \ldots r_n\} \in s \quad (2)$$

- **Coordination error rate**: defined as the arithmetic mean of the number of multiple counts of the same letter by different users divided by the number of all letters counted per run. It derived from the count data:

$$c = \frac{1}{n}\sum_{r=1}^{n} \frac{multicount_r}{count_r}, \{r_1 \ldots r_n\} \in s \quad (3)$$

- **Performance**: defined as the arithmetic mean of the number of letters counted per run divided by the configured assessment run time. It is derived from the count data and configuration:

$$r = \frac{\frac{1}{n}\sum_{r=1}^{n} count_r}{runduration}, \{r_1 \ldots r_n\} \in s \quad (4)$$

Thus, a session s can be described as s=(x,y,c,r). The first two become its x- and y-coordinates in the diagram; c is the colour and r the radius of the circle to be plotted for the session in the 4I-diagram.

3. Implementation

3.1. Architecture

The SCTA Tracer was developed using the Java programming language [7]. Its overall architecture works according to the mediator pattern [8]. It belongs to the object-based behavioural patterns. In this pattern a central controlling instance, the mediator, promotes loose coupling by keeping the collaborating objects (called colleagues) from referring to each other directly. The mediator controls and coordinates interaction and represents the software's overall behaviour. Colleagues obtain a reference of the mediator from a central registry. The mediator is responsible for sending/receiving information to/from the respective colleagues.

We defined two interfaces named ISCTATracerMediator and ISCTATracerColleague. Since we use Java's Remote Method Invocation (RMI) to allow the distributed use of the software, the two interfaces are Java Remote interfaces at the same time. At start-up the first main() method creates an ISCTATracerMediator instance of the remote object implementation (the stub) and tries to bind that instance to the name SCTATracerMediator in a Java RMI registry. When registered successfully, this instance launches the administration screen. Further instances try to do the same, but their registration as SCTATracerMediator will fail, due to an already registered instance. Therefore, these further instances will create objects of the ISCTATracerColleague interface. These colleague instances obtain a mediator reference using a RMI registry lookup. Now they are able to use the mediator's register() method to place their remote interface references there. Thus, the setup of the mediator pattern is complete. In the following the colleague instances setup the wait screen for their users. The mediator controls the colleagues using the ISCTATracerColleague interface—for instance, when the assessment run starts, questions of the freeze probes are to be shown, or when the assessment run is over. The colleagues use the ISCTATracerMediator interface to (un-)register for assessment runs, to send counting results, and to answer freeze probe questions.

3.2. Charts

Pushing the button "Run Evaluation" on the evaluation tab creates a 4I-diagram using an extended version of JFreeChart [9]. It is an open-source Java framework allowing the creation of complex charts of various types like XY charts (line, spline and scatter), pie charts, Gantt charts, and bar charts (horizontal and vertical, stacked and independent). Besides the creation of charts, JFreeChart allows the placement of various markers inside the resulting diagrams. However, in the case of our 4I-diagram we needed to create our own custom chart to regard our four aforementioned measures. Fortunately, JFreeChart proved to be easily extensible for this situation also due to the availability of its source code. Error rate and response time/forgetting time ratio are used as standard x- and y-coordinates. However, our need to influence an item's diameter and colour by the values of performance and coordination error rate required customization. Additionally, we needed an individual label for each item of a series to be shown while standard JFreeChart allows only labels per series (which correspond to an assessment trace; a series item corresponds to an assessment run). First, we needed to define a new dataset type that is able to contain all four measures per item (i.e., a 4-tuple, or quadruple) in addition to a session label. This dataset, named SCTADataset, extends JFreeChart's XYDataset. In order to deliver the data on screen we also needed to define a custom renderer that displays the dataset's content as 4I-diagram. Our SCTARenderer extends JFreeChart's XYLineAndShapeRenderer class to do the job. The generation of proper labels required a customized SCTAItemLabelGenerator. The JFreeChart object is finally added to a standard Java Swing container.

3.3. Persistence

For persistence a lightweight approach known as XML data binding was chosen. This allows accessing XML data using objects rather than using DOM or SAX. The Java Architecture for XML Binding (JAXB) allows mapping Java classes to XML representations. It provides two main features: the ability to marshal Java objects into XML and the inverse, that is to unmarshal XML back into Java objects JAXB allows storing and retrieving data in memory in any XML format, without the need to implement a specific set of XML loading and saving operations. JAXB is a part of Java SE platform [7].

As a first step we defined our storage format as XML schema. We planned to use one XML file per assessment trace. Therefore it became our top-level element. It only has a name attribute. A trace element may contain multiple session elements, which also have a name attribute. The session element may contain multiple run elements. A run element holds the information about the participating users, counting information and response data from the freeze probes.

The binding compiler xjc is used to generate a set of Java classes that represent the schema. These classes are filled by the application with the data collected. When the administrator chooses to save the current status of the trace then this data structure with its top element class Trace is handed over to the Marshaller object to create the XML file. On the other hand, at application start-up an existing XML file can be chosen from which the Unmarshaller object creates a data structure to be used by the application.

4. Discussion/Conclusions

The concept SCTA-4I and its tool, the SCTA Tracer, are lightweight and universally usable since they are independent of a specific awareness model and of a specific primary task. The setup of the task is straight forward allowing heavy (re–)use at very low preparation cost. Additionally, researchers are relieved from reading log files or analyzing interview data,

since the evaluation comes with the push of a button. It allows to be used in colocated and distributed settings. Opposed to questionnaires it delivers quantitative data and a visualization that helps to guide further development steps. It uses freeze probes to capture awareness when its still present and introduces a simple primary task. Some of the issues of prior versions were already resolved. The former response time (x-axis) was replaced by the response time/forgetting time ratio thus providing a fixed frame for all sessions of the trace inside the diagram. In earlier versions, the session with the highest response time was drawn at the right border of the chart, which caused misleading interpretations of the diagram. Now, x- and y-axis both use ratios The formerly rather fixed parameters became adjustable and moved to the configuration, where they can be adapted also in order to experiment with the setting itself.

But there are still difficulties: as an experimental simulation it lacks the situatedness often needed in CSCW application assessment [10]. However, we think that this situatedness is not exactly needed at these early stages of development. We suggest using our tool in addition to tests and experiments in situated settings. Another major drawback is that the user has to enter the counting results in the document screen. This is needed in order to generate the questions for the freeze probes and to check the answers. There is not really an alternate way to get hold of these counting results. One can argue that this belongs to the primary task as there also arguments that it does to the secondary.

There is future work on both the conceptual and the technical side: The SCTA-4I currently focuses on the coordination activity during the task. Future versions should include other areas of awareness information like location or presence. Additionally, we are eager to see if there are typical trace patterns and how 4I-diagrams of smaller teams relate to 4I- diagrams of larger teams using the same system under test. From a technical point of view we would like to extend our software with the Java Webstart technology that allows installing and launching it using a standard web browser. Via our website we could then provide the tool in the latest version to every researcher interested.

Acknowledgment

We like to thank all people participating in the initial setup of the assessment and all reviewers for their valuable comments and suggestions on earlier versions of this work.

5. References

[1] Dourish, P., Bellotti, V., Awareness and Coordination in Shared Workspaces. In Proceedings of the Conference on Computer-Supported Cooperative Work - CSCW'92 (Oct. 31-Nov. 4, Toronto, Canada). ACM, N.Y., 1992, pp. 107-114.

[2] Gross, T., Stary, C., Totter, A., User-Centered Awareness in Computer-Supported Cooperative Work-Systems: Structured Embedding of Findings from Social Sciences. International Journal of Human-Computer Interaction 18, 3, 2005, pp. 323-360.

[3] Grudin, J., Why CSCW Applications fail: Problems in the Design and Evaluation of Organisational Interfaces. In Proceedings of the Conference on Computer-Supported Cooperative Work - CSCW'88 (Sept. 26-28, Portland, OR). ACM, N.Y., 1988, pp. 85-93.

[4] Ebbinghaus, H., Memory: A contribution to experimental psychology. New York: Dover, 1885.

[5] Oemig, C., Gross, T. Illusive, Ineffective, Inefficient, Ideal: Standardized Coordination Task Assessments of Awareness Support. , in press.

[6] Karremans, J. C., Stroebe, W., Claus, J., Beyond Vicary's fantasies: The impact of subliminal priming and brand choice. Journal of Experimental Social Psychology, 42, 2006, pp. 792-798.

[7] Oracle Technology Network. J2SE 6.0. http://www.oracle.com/technetwork/java/index.html, 2011. (Accessed 2/8/2011).

[8] Gamma, E., Helm, R., Johnson, R., Vlissides, J. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, Reading, MA, 1994.

[9] JFreeChart. http://www.jfree.org/jfreechart/, 2011. (Accessed 5/7/2011)

[10] Twidale, M., Randall, D., Bentley, R., Situated evaluation for cooperative systems. In Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work (Oct. 22-26, Chapel Hill, North Carolina, United States). ACM, N.Y., 1994, pp.441-452.