Supporting Instructors' Decision Making in Simulator-Based Training for Crisis Management

Rita Kovordanyi
Linköping University
rita.kovordanyi@liu.se

Jelle Pelfrene
Space Application Services
jelle.pelfrene@spaceapplications.com

Henrik Eriksson
Linköping University
henrik.eriksson@liu.se

ABSTRACT

Simulator-based training is often more information-intensive and mentally overloading—both for the trainees and for the exercise staff—than a corresponding live exercise would be. In particular, massive amounts of data are produced from the simulation core, and these data are often too detailed, and too low-level to be of direct use for the human eye. The present paper describes a decision support system aimed at helping exercise instructors maintain an overview of how the exercise is progressing and how the trainees are performing. The paper describes our experience with implementing a real-time, low-key decision support system employing complex event processing, with focus on meeting the special technical challenges that are associated with the novel approach of implementing high-end, real-time processing on a low-power, 6-inch, mobile Android platform.

Keywords

Real-time decision support system, complex event processing, CRISIS, Android, tablet, phablet.

INTRODUCTION

This paper focuses on the technical challenges associated with implementing a real-time decision support system on a mobile device (Figure 1), where low power consumption and frequent processing interruptions have to be combined with requirements for heavy-duty, always-connected, real-time performance.

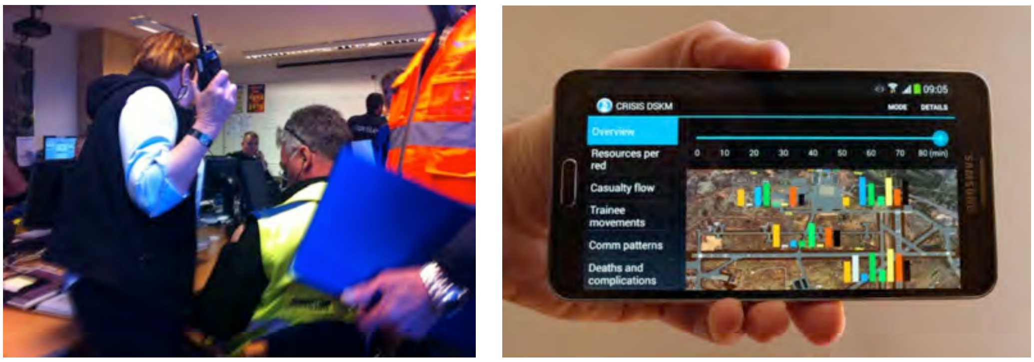

Figure 1. a. CRISIS exercise instructors are constantly on the move. **b.** We present a mobile decision support system capable of real-time processing of massive amounts of data, and giving low-key warning if deviating patterns are detected.

The work described in this paper forms part of a recently completed European-Community-funded FP7 project, CRISIS, where the objective was to develop and evaluate a virtual-reality-based simulation system for crisis management training (for further information, please visit http://idc.mdx.ac.uk/projects/crisis/).

The CRISIS system is based on the simulation core CRISIS-XVR, developed by the CRISIS partner E-Semble (http://www.e-semble.com/). In addition to the simulation core, the system comprises a number of supporting modules for planning, running, controlling, and debriefing simulated field and command-post exercises (Figure 2). This paper describes part of the overall CRISIS initiative, which was responsible for developing a decision support system aimed to support exercise staff—exercise managers and instructors, when running and debriefing a CRISIS exercise.

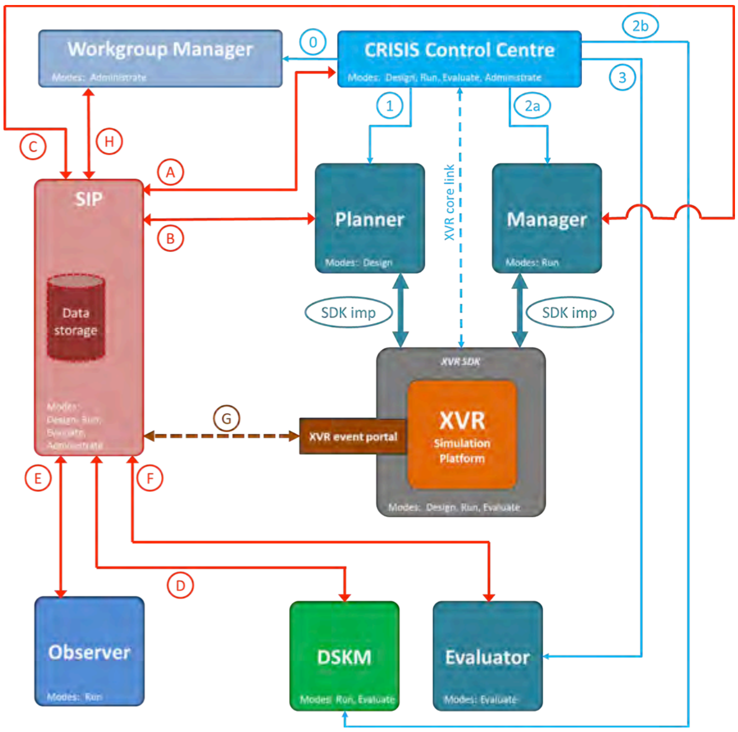

Figure 2. The overall architecture of the CRISIS system, with the Decision Support and Knowledge Management DSKM module in green in the mid-lower part of the figure. Bright red connections denote data flow, light blue lines denote control signals (stop, resume). Dark red connection show the event stream to/from the CRISIS-XVR simulation core, used as input to the DSKM. (Letters and other labels for connections denote well-defined interfaces between the various components.)

BACKGROUND

An exercise session using the CRISIS system normally involves multiple trainees using one computer each, so that the trainees can use their own joystick to move and look around in the virtual environment, and initiate virtual communicative actions independently of each other. All other forms of trainee-system interaction—for example, ordering virtual firefighters to address a fire—are mediated by one or more technical facilitators, who can directly control the CRISIS-XVR system, which is running on a dedicated XVR computer. A key characteristic of the CRISIS system is that planned events, for example, a secondary crisis scenario, complicating the present crisis, can be pre-specified in the CRISIS Planner. The plan is specified according to a Variable Uncertainty Framework (Barnett, Wong, Westley, Adderley, & Smith, 2011), to allow random variation in how and when simulated events occur. Given a particular instance of the exercise plan, the flow of simulated events in the exercise can be dynamically modified during the session both programmatically, by the CRISIS Manager module, and manually by the human technical facilitator—on orders of the instructor.

Hence, the exercise instructor can give instructions to the technical facilitator, who would then implement these instructions, for example, by pausing the exercise, if the instructor wants to have a quick debriefing of the trainees. As the instructor is not directly interacting with the CRISIS-XVR simulation core, s/he is free to walk around the session room, or walk between rooms in larger-scale exercises involving several teams of trainees, and watch the trainees' screen to see what they are currently seeing, and observe, listen to and talk to the trainees.

In this way, the exercise instructor is monitoring the trainees, moving between session rooms, conferencing with exercise staff colleagues and note-taking observers, giving advice and instructions not only to the trainees but at the same time managing the technical side of the exercise. As so much is going on, and since the instructor is continuously managing both trainees, colleagues, and the technical staff, an exercise instructor in virtual-reality-based training has generally more to do than in s/he would in a live exercise. To aggravate things, the CRISIS-XVR simulation core is capable of producing immense amounts of data that would normally not be readily available in a live exercise. The data can, for example, contain information about trainee movements (x, y, z coordinates), trainee communication (who is communicating, when), the exact position and state of all casualties, of all vehicles and avatars in the virtual-reality environment, as well as the position of all other objects in the environment, the dynamic creation and deletion of objects, et cetera. While the instructor does not need to see all this detailed information, s/he would benefit greatly from a support system that could filter, aggregate and refine this information and help the instructor get an overview of what is going on in the exercise.

Empirical studies of instructors' work

What kind of support is best suited for exercise instructors is dependent on organizational practices, which emergency procedures are to be followed, the training objectives of the exercise, and various best-practices within the organization. We have conducted a survey to elicit our end-users' view on these matters. The three end-user organizations that partook in the CRISIS project, namely Flugstodir (ISAVIA), Aeroportos de Portugal (ANA), and British Transport Police (BTP), have expressed a wish for support in the form of intelligent information filtering and presentation during After-Action Review (AAR). In addition, an informal interview with the project partner VSL Systems, who have extended experience in running large scale live exercises, indicated that real-time filtering and aggregation of the massive stream of information during the exercise might also be useful.

We have also studied instructors' work by observing live and table top exercises at KMC, Sweden, 2012 (see also (Field, Rankin, Morin, & Lemmers, 2012; Rankin, Field, Kovordanyi, & Eriksson, 2012). Our inquiries indicate that exercise instructors would greatly benefit from support for quick situation assessment, and active, but unobtrusive warnings when abnormal patterns are detected by the decision support system (Kovordanyi et al., 2012).

Theoretical research

Naturalistic decision making (NDM) refers to a movement in decision research where the focus is on intuitive, as opposed to analytic, processes as a basis for decision making (Hammond, Hamm, Grassia, & Pearson, 1987; Klein, 1993; Klein, 2008; Lipshitz, Klein, Orasanu, & Salas, 2001; Rasmussen, 1987). Natural or intuitive decision-making is assumed to mainly rely on automatic recognition processes where actions are triggered or activated by recognition of the situation at hand, without need for elaborate, conscious reasoning. Studies into how humans make decisions suggest that decision making in the field is best supported by passive systems that do not impose a theoretically-optimal strategy on the users, since such a strategy would be practically unfeasible when high-stake decisions have to be made under time-pressure—as is often the case during emergency-response operations making. Human decision makers seem instead to prefer an intuitive approach, where decisions are based on previous experience with similar situations. The role of support systems may be to highlight potentially important situational cues, and present intelligently-aggregated information to the user.

This implies that decision support systems in general and the DSKM in particular, should be designed in a low-key style, supporting the user by aggregating and refining information, and by presenting this information in an easy-to-perceive way. In essence, the support system should do the leg-work, but leave the final interpretation of the data to the human user.

TECHNICAL CHALLENGES AND SOLUTIONS

An important part of the non-functional requirements on the DSKM was high level of mobility. For this reason, we choose to run the decision support system on a small phablet (the recently marketed 5.9 inch Samsung Note 3), running Android 4.3. At the same time, running the DSKM on a small, mobile device entailed that we had to deal with the flip-side of low power consumption and frequent operating-system-enforced interruptions in processing.

Small Android devices are normally quick to enter power saving mode, and have normally limited processing power and very limited memory (RAM), meaning that apps tend to get thrown out into background processing and eventually out of memory when the user starts up another app (e.g., when taking an incoming phone call or checking a web resource).

We needed our device to always be active (never go into power saving mode). We also needed a messaging service client to run (relatively) uninterruptedly, and likewise, we needed complex event processing to infer information from the incoming stream of events continuously, and in real time. On the other hand, the GUI that the user is interacting with must be highly adaptive to device settings, and to the physical state of the device—most notably, how the device is physically rotated. This entailed that the GUI must be stopped and restarted for every device configuration change, while parts of the app needed to run without interruption.

To summarize, an Android app could be pushed into background-processing mode if the user started another application on the phone—for example, when answering an incoming phone call or checking a web page. In addition, the app would be stopped and restarted upon configuration changes, such as changes in phone settings or a physical rotation of the device. Stopping and restarting processing in these cases makes sense from the user's point of view, but poses a challenge for a real-time event processing system, as such as system is required to stay on-line, continuously listening for a stream of incoming events, and processing and displaying these events in real time.

As a solution, we opted for a system design where central components were implemented as services that would continue to run in the background while other system components got interrupted and re-started (Figure 3). In this way, every time the main application would be stopped and restarted, it would lose contact and then reestablish contact with the background services.

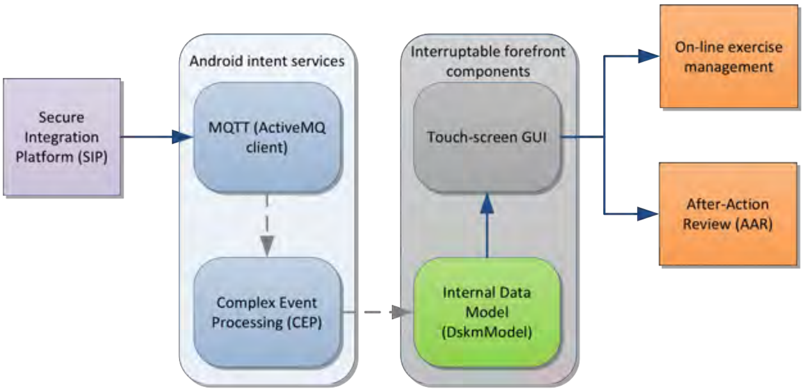

Figure 3. Four main components of the DSKM Android app: 1. MQTT intent service implementing an ActiveMQ client which is always up and connected and receives a continuous event stream from the Secure integration Platform (SIP). 2. Intent service implementing Complex Event Processing (CEP), which filters, aggregates and refines in real time the information contained in the received events. 3. The internal data model, where inferred information is time-stamped and persistently stored in proprietary data structures, allowing the user to roll back the time, and access past information. 4. The main Android activity, where the user interaction is handled. Components 3 and 4 are volatile in the sense that they will be frequently stopped and restarted by the operating system. Upon each stop and restart, the internal state of these components are serialized, written to memory, and reinstated. (Arrows denote information flow, not referencing)

At the same time, division of an app into background and foreground components entails a communication problem between these components. Normally, a major point of having a service running in the background is that the service can be functionally detached from other system components, with the service holding no references to other components to avoid having loose (undefined) references when the foreground components get stopped and get deleted from memory. For these reasons, Android intent services communicate by broadcasting (mediated by the operating system) of serialized messages to other system components, who have to register (at the operating system) as receivers. In this way, the service has no references to its receivers. This form of broadcasting is designed to mediate simple serialized messages, and is not well suited for more elaborate communication of complex information content. As we wanted the GUI and the internal data model to have a closer communication sending more elaborate data between them, we implemented the internal data model as an interruptible fragment, directly referenced by the main activity (the main component of the app).

The downside of this architectural design is that both the data model and the GUI are exposed to frequent interruptions and restarts by the operating system. An interesting quirk to this problem is that while most of these interruptions can and should be bridged by saving and reinstating these components' internal state, the app must also be capable of being really stopped—and cleanly restarted, as opposed to resumed. In some cases, it might be difficult to decide when a clean restart is appropriate, and when the app should simply be resumed.

An even more challenging issue concerns the memory capacity and the computational power of today's mobile devices. Today, that is, in year 2013, Android devices have an inherent limitation of the size of applications. Currently, a maximum of 64k method definitions can be built into an .apk-file and loaded onto an Android device. At the same time, the DSKM app is dependent on rather large libraries: Asper, which is an open-source port of Esper, used for complex event processing, AFreeChart used for data plotting, and Java Architecture for XML-binding (JAXB) needed for decoding (unmarshalling) the received events which arrive in XML-format. These libraries in turn depend on several other libraries. Together, the libraries make up a considerable memory size and, especially the libraries for XML unmarshalling, introduce a considerable latency in overall processing time.

To briefly summarize our own experience: On-line (continuously connected), real-time processing of massive event streams is stretching the envelope of today's mobile computing. It is achievable, with same care taken as to how various components are implemented, but it can only be achieved on the latest, highest-end mobile devices. The device we are currently using (Samsung Note 3) is equipped with 3 GB RAM, and utilizes four computational cores working at 2.3 GHz speed.

THE DSKM

The Decision Support and Knowledge Management system (DSKM) is one of the supporting tools incorporated into the CRISIS system (Figure 2). The DSKM is started up at the beginning of a CRISIS exercise from the CRISIS Control Centre (CCC), together with the CRISIS-XVR simulation core and other supporting modules, such as the CRISIS Manager. At the end of the exercise, the DSKM shuts down gracefully on a signal sent by the CRISIS-XVR in the form of a TrainingStopped event.

During the exercise, the DSKM receives input from the CRISIS-XVR in the form of a continuous stream of events. These events encode information, such as trainee movements, recourse locations and movements—for example, the movement of ambulances and personnel—items created—for example, that an area has been designated as the casualty assembly point—areas entered, and casualties being triaged.

The DSKM filters and combines these pieces of data into higher-level information suitable for storing, as background knowledge and so on. As the next step, the DSKM refines the received information on the basis of previously-stored information and ontological background knowledge represented as a topic map. At the final output stage, the DSKM presents several dynamically-updated visualizations containing aggregated statistics, which help the user in forming and maintaining an overview of the exercise.

The DSKM also provides data for AAR by storing its state when a flag request is issued on the command of the user. The stored DSKM state is mediated in an internal flag request. The DSKM state mediated in this way is extended subsequently with CRISIS-XVR-specific data, and this data blob is then stored in persistent store for later use for debriefing by the AAR module.

DSKM sub-modules

The intelligent core of the DSKM comprises three main sub-modules: (1) a knowledge base in Topic Map representation using an ontology of possible and planned events that can occur during the exercise, extended with inference results generated by the DSKM, (2) real-time filtering and aggregation of event streams that originate from simulator events and trainee actions during the CRISIS exercise, and (3) intelligent knowledge-based inference of aggregated information. The knowledge-management approach applied in the DSKM focuses on user-friendly querying of the above-mentioned knowledge base including inference results, using natural-language questions among other methods. In addition to this, the DSKM has an output sub-module consisting of a touch screen graphical user interface.

The DSKM implements real-time situation assessment and decision support and comprises of three main modules (Figure 5, blue components): (1) The real-time module currently running a network of Esper-statements, (2) The central knowledge base for decision support and knowledge management, consisting of a Topic Maps system, and (3) The knowledge-based reasoning module running Drools rules. The interface between the central knowledge base and the other two modules is implemented in TMAPI. This interface allows for both reading and modification of the central topic map. In addition to this intelligent core, the DSKM also hosts a touch-screen graphical user interface (Figure 5, light purple components).

Exercise instructors can use the decision support in two ways (brown boxes): During the running of the exercise and during After-Action Review (AAR). Note that exercise management can affect events in the simulation environment during the running of the exercise (feedback connection at top of figure).

To make all this happen, the DSKM utilizes a number of libraries, such as Ontopia (for managing topic map data), TMAPI (for seamless access to ontological and other forms of background knowledge), Esper (for real-time filtering of information streams), Log4J (for handling textual communication with the user), JFreeChart (for plotting aggregated data).

Implementation details

The core of the DSKM is implemented in (Android) Java, with an easily configurable (Holo Dark) Android user interface. Core DSKM functionality is complemented with an ontological background knowledge base represented as Topic Maps (TM), and rules for complex event processing and reasoning, which are implemented in other specialized formalisms.

Input events are received in real-time through SIP listeners, one listener per event type. Event types were specified in a project-specific model for internal data exchange through the secure integration platform (SIP). Events that are received can be out of temporal order; however all events are time stamped with CRISIS-XVR system time and elapsed effectively-used training time.

The received input events are translated and connected as occurrences (facts) into the body of ontological background knowledge that is maintained in the form of a topic map. These facts, together with derived knowledge will serve as a basis for question answering, which allows the user to get insight into details such as which trainee did what when. As a complement to this, the Complex Event Processing module (CEP) in DSKM will present aggregated statistics and higher-level inferred information that can be derived from these basic facts.

CEP is especially useful for high-performance correlation, aggregation and filtering of events, based on simple event attributes like position or specific actions. CEP is, however, not very well suited for higher-level semantic analysis of events. For this purpose, CEP can be combined with rule-based reasoning and ontological background knowledge, which describes how the world is structured and how things work. These combined approaches are often referred to as Semantic Event Processing, SEP, or, in the semantic web-community, simply streaming (Della Valle, Ceri, Van Harmelen, & Fensel, 2009). Similarly to CEP, SEP works on event streams, but trades performance to allow processing of events at a higher level of abstraction and semantics, taking into account background information.

The basic occurrences (facts) are fed through event-type-specific channels into the real-time complex event processing component, implemented using Asper, an Android port of Esper (see http://esper.codehaus.org/). The facts are filtered, aggregated into more complex data, and refined using background knowledge. The output is fed through data-type-specific output channels, each output channel feeding one chart or plot in the DSKM GUI. The individual charts and plots in the GUI pull changes in the respective output channels, and in this way get updated when new information is available in these channels.

Part of the DSKM screen hosts a number of charts and data plots. These include aggregated statistics of how rescue, medical, security and transport resources are utilized across the crisis scene, the casualty flow from various parts of the scene to hospitals, trainee movements (derived information on how trainees move between dedicated zones, such as primary triage area, casualty assembly point, rendezvous point, etc.), trainee communication patterns, and preventable complication and deaths.

Interaction Design

Simulator-based exercises are highly dynamic, with the instructor having to monitor multiple trainees, interact with supporting exercise staff, such as observers and technical facilitators (see Background section), constantly moving from trainee to trainee, conferencing with technical staff, and moving between session rooms (Figure 1). It is therefore important that the decision support system offered to the exercise instructor is highly mobile and easy to carry around. An additional, basic requirement is that the support system should be unobtrusive and that it should support *naturalistic decision making*. In essence, this entails that the support offered should be perceptual rather than textual: instead of enforcing analytic reasoning upon the user, the user's intuitive recognition of a situation is facilitated, by supporting what we call 'at-a-glance overview'.

At an early stage of design, we also contemplated whether we should structure the DSKM interface according to a set of predefined instructor checklist items, either inspired by the ones suggested by the International Civil Aviation Organization (ICAO) (International Civil Aviation Organization, 1991), or use a checklist similar to the ones used by medical emergency training instructors at the Centre for Teaching & Research in Disaster Medicine and Traumatology, KMC, in Sweden (Nilsson, 2013).

In a detailed survey we conducted within one of our end-user organizations, ISAVIA, instructors described which aspects of trainee actions they usually focus on when evaluating trainee performance in an exercise. For example, an overview of resource allocation was highlighted as central when assessing the progress of the exercise and grading trainee performance. On the basis of this survey, the visualizations offered by the DSKM were re-designed and are now targeted towards these user-suggested categories (see Figure 1and Figure 4).

Workflow when using the system

The tasks for an instructor during an exercise include monitoring the progress of the exercise, and assessing trainee performance (facilitated by the DSKM), both to be able to initiate direct corrective actions, such as pausing the exercise for a quick debriefing, and to create notes and temporal bookmarks as a reference for later discussion with the trainees during after-action review (also facilitated by an observer module, and the evaluator module in the CRISIS system; see Figure 2).

Working with the exercise timeline

In the DSKM, the instructor has always access to a timeline where the elapsed training time is continuously displayed (Figure 4). The instructor can use this timeline to roll-back the time and the corresponding data that is displayed, in this way re-playing important aspects of the exercise—which could, for example, be useful during after action review.

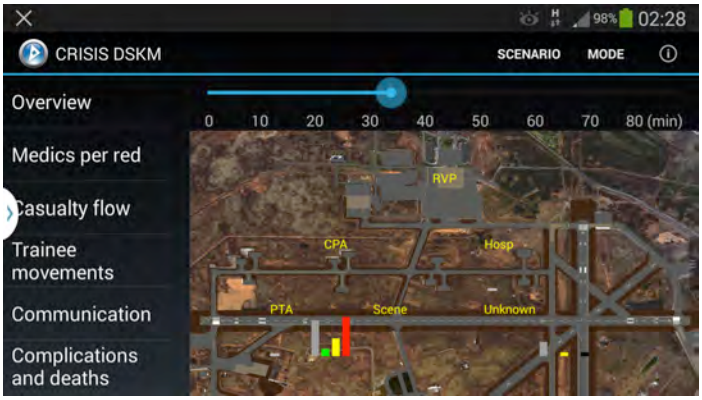

Figure 4. The instructor has always access to a timeline and an overview of casualties and resources around the crisis scene. The timeline is interactive: it can be pulled-back to display past data. When released, the timeline resumes its position on the current time.

Working with data plots

The DSKM produces a number of data visualizations containing statistics charts and data plots (see, e.g., Figure 4). These visualizations are displayed one at a time to the instructor. To see another visualization, the instructor can touch-scroll the list in the left part of the screen. The various visualizations show:

- Overview of resource allocation (e.g. how many vehicles are available at different locations around the crisis scene).
- Casualty flow (how the distribution of different categories of casualties change with time)
- Trainee movements
- Trainee communication
- Preventable complications and deaths that occur as a result of delayed medical treatment

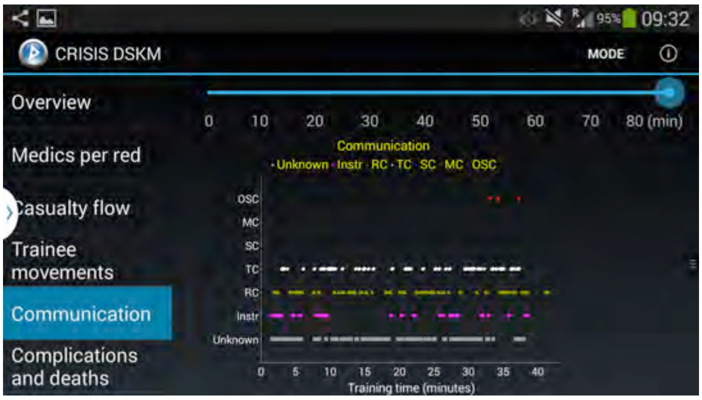

Figure 5. Visualization of trainee communication: who has been talking, and when.

Real-time, low-key warnings

In addition to offering at-a-glance-overview visualizations, the DSKM can also detect deviating patterns and issue warnings. Often, these patterns have both a spatial and temporal dimension. For example, trainees may have exhibited an abnormal communication or movement pattern, moving to an unexpected location, and neglecting to move on to the correct location. An important case is, for example, if a medical coordinator, whose duties include setting up the secondary triage area (the casualty assembly point), neglects to do so (in time). It is rather common that the medical coordinator starts out by surveying the primary triage area and continues to stay there for too long. In this case, the instructor will be notified of this deviating trainee behavior (Figure 6). The warnings issued by the DSKM are low-key in the sense that they appear as a single icon in the top status bar. If the instructor is interested in getting more information about the warning, s/he can pull down the icon to display the actual warning message.

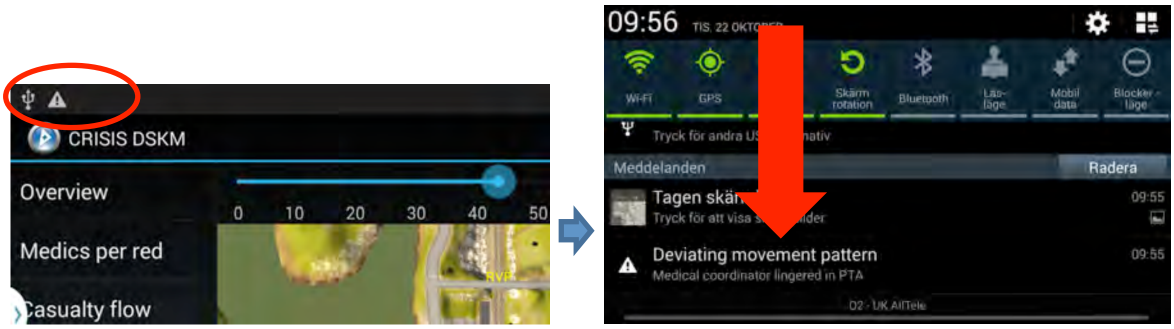

Figure 6. The DSKM has delivered a low-key warning, since a deviating pattern has been detected. In this particular instance, a trainee, the medical coordinator, has moved to and lingered for too long at the primary triage area (area where casualties are first categorized according to their injuries). As it is the responsibility of the medical coordinator to set up the secondary triage area, a warning is issued by the system, if s/she has not entered that area within a pre-defined time limit. The warning appears as an unobtrusive icon **(a)**, which can be pulled down to reveal the full warning text **(b)**.

SYSTEM EVALUATION

Early prototypes of the DSKM module were initially demonstrated and discussed at two CRISIS research workshops, which were attended by both research and end-user partners. The DSKM module was then showcased at a round-up consortium meeting.

Valuable feedback from the end-users was gathered during this last session. Our end-users acknowledged that the overall design of the system conformed to their earlier input, but the users also requested the following:

- Users should be able to configure the visualizations (e.g. location names, and bar colors).
- Likewise, the position of sub-charts (denoting various locations around the crisis scene) should be configurable.

In the final version of the system, users can configure these settings in an XML-file.

- Users whished the addition of "unknown" as a concept for those casualties that are not in any of the designated crisis scene locations (e.g., patients that have not yet been found by rescue personnel).
- Monitoring of user-defined trainee behavioral patterns, and issuing of a warning when these are violated.

While this feature—issuing of warning—is implemented in the final system, configuration of the underlying rules for complex event processing is not easily done. Temporal thresholds, and the like, as part of a rule can, however be modified without greater effort.

A final evaluation of the CRISIS system was conducted during the summer of 2013 at all end-user locations, with overall positive user feedback. All participants in the training sessions were able to operate the various interfaces within 15 minutes. Extensive user testing of the CRISIS system (and of DSKM) as a tool in everyday training remains to be done.

CONCLUSIONS AND FUTURE WORK

The DSKM has been demonstrated for and tested by our end-user organizations, but remains to be systematically evaluated in everyday use. We are committed to supporting our end-users, and expect detailed end-user feedback once our end-user partners, as well as new end-user organizations, such as the London Fire Brigade (http://www.e-semble.com/en/About_E-Semble/News/), incorporate the CRISIS system into their training regime.

We are currently contemplating on alternative uses of the DSKM; in particular, we are planning to re-use its complex event processing and visualizing capabilities as a support tool for empirical research. Initial observations indicate that trainee behavior might differ in a systematic fashion in simulator-based exercises, as compared to live exercises. The DSKM could filter and present simulator and trainee-related data, and in this way help us to highlight which variables are indicative of (correlate with) deviating trainee behavior.

ACKNOWLEDGMENTS

The research leading to these results has received funding from the European Union Seventh Framework Program (FP7/2007-2013) under grant agreement n° FP7-242474.

REFERENCES

Barnett, J., Wong, W., Westley, D., Adderley, R., & Smith, M. (2011). Startle Points: A Proposed Framework for Identifying Situational Cues, and Developing Realistic Emergency Training Scenarios. In *Proceedings of the 8th International ISCRAM Conference–Lisbon* (Vol. 1).

Della Valle, E., Ceri, S., Van Harmelen, F., & Fensel, D. (2009). It's a streaming world! Reasoning upon rapidly changing information. *Intelligent Systems, IEEE*, *24*(6), 83–89.

Field, J., Rankin, A., Morin, M., & Lemmers, A. (2012). Instructor Tools for Virtual Training Systems. In *Proceedings of ISCRAM 2012*. Vancouver, Canada.

Hammond, K., Hamm, R., Grassia, J., & Pearson, T. (1987). Direct comparison of the relative efficiency on intuitive and analytical cognition. *IEEE Transactions on Systems, Man and Cybernetics*, *17*(5), 753–770.

International Civil Aviation Organization. (1991). *Airport services manual. Part 7: Airport services planning* (p. 97). Retrieved from https://www.google.se/search?q=ICAO+9137+PART+7&ie=utf-8&oe=utf-8&aq=t&rls=org.mozilla:en-US:official&client=firefox-a&channel=fflb

Klein, G. (2008). Naturalistic Decision Making. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *50*(3), 456 –460. doi:10.1518/001872008X288385

Klein, G. A. (1993). A recognition-primed decision (RPD) model of rapid decision making. *Decision making in action: Models and methods*, 138–147.

Kovordanyi, R., Pelfrene, J., Rankin, A., Schreiner, R., Jenvald, J., Morin, M., & Eriksson, H. (2012). Real-time

Support for Exercise Managers' Situation Assessment and Decision Making. In *Proceedings of ISCRAM2012*. Vancouver, Canada.

Lipshitz, R., Klein, G., Orasanu, J., & Salas, E. (2001). Taking stock of naturalistic decision making. *Journal of Behavioral Decision Making*, *14*(5), 331–352. doi:10.1002/bdm.381

Nilsson, H. (2013). *Demand for Rapid and Accurate Regional Medical Response at Major Incidents*. Retrieved from http://www.diva-portal.org/smash/record.jsf?searchId=3&pid=diva2:579248

Rankin, A., Field, J., Kovordanyi, R., & Eriksson, H. (2012). Instructor's Tasks in Crisis Management Training. In *Proceedings of ISCRAM2012*. Vancouver, Canada.

Rasmussen, J. (1987). Cognitive control and human error mechanisms. *New technology and human error*, 53–61.