Designing Interactive Health Care Systems: Bridging the Gap Between Patients and Health Care Professionals

Lisa Graham – *IEEE Student Member,* Mohammad Moshirpour – *IEEE Student Member,*
Michael Smith – *Senior IEEE Member*, Behrouz H. Far – *IEEE Member*

***Abstract*—As patients become more proactive about their health and turn to technologies such as the Internet to acquire knowledge, the patient-health care professional relationship has been changing. Traditionally, information has flowed from health care professional to patient, but change to a two-way dialogue is taking place. In this study, we examine a high level design of a perceived medical system and determine the implications of adding patients as active contributors. The main challenge of modifying existing systems to incorporate patient interaction is preserving system integrity. We propose a systematic approach to support scaling health care systems while preserving system integrity. Distributed systems such as personal health records and eHealth systems provide two ways in which patients can become more involved with their own health care with or without the involvement of health care professionals. It is important that modifications to such systems do not compromise patient record integrity regardless of whether the patient is working alone or with their health care professional. The lack of central control in distributed systems added to the complexity of health systems poses challenges for design and modification. Of particular interest is the identification of emergent behavior (behavior not explicitly specified in the specifications) in distributed systems not explicitly defined in the requirements of its individual components. Use of the new emergent behavior detection (EBD) tool offers potentially considerable cost savings by proactively identifying such behaviors during the design rather than the deployment phase of a project. Based on high level message sequence charts, the EBD tool highlighted a data synchronization issue between the main database and the patient's interface to the system. This provides valuable feedback of the early health system design which benefits future design development.**

Keywords—public health; health care; personal health systems; eHealth systems; distributed systems; emergent behavior; scenario-based software engineering; message sequence chart; proactive care; patient-provider relationships

I. INTRODUCTION

Patients are becoming more interested in managing their own health, which is leading to changes in traditional information dissemination from the health care professional to the patient [1]. In managing this change, it is necessary to design software applications to support both the health care professional and the patient in the clinic and out. Personal health records and eHealth systems provide two ways in which the patient can be involved in health management. First, we will look at existing definitions for each of these concepts, examine some concepts from literature for each, and see how they are related.

There is no currently agreed upon definition for eHealth [2], but health and technology are two common themes in this area [3]. Previous work has shown there exists little evidence to support claims of cost effectiveness and patient outcome improvements through eHealth systems [4] even through eHealth systems have been implemented (such as eHealth Ontario [5]) or are in the process of being implemented (such as the European Commission eHealth Network [6]). In one study, the authors recommend evaluating new health technologies comprehensively from both social and technological standpoints to achieve an optimal result [4]. Considering both social and technological factors is a large undertaking, so we only focus on one aspect of the technological perspective here – emergent behavior in the design of new health software.

Personal health records can be defined as private, secure, and confidential electronic systems which range in complexity and allow users to access, manage, and share health information of their own and those for whom they are authorized [7, 8]. Some personal health systems provide standalone data for tracking of, for example, physical activity, diet, weight, and sleep (such as FitBit and MyFitnessPal [9, 10]) allowing the patient to track information independent of a health care professional. Other personal health and eHealth systems integrate guidance from a health care professional [8]. Such interconnected systems provide more significant benefits [8], one of which is improved communication between health care professionals and patients [11]. In the preliminary design presented here, communication facilitation between patient and health care professional via a software tool is expected to increase patient knowledge and involvement in a health program.

However, modifying existing systems to allow or increase patient interaction can be a challenging task. It is important to ensure modifications will not compromise system integrity and lack of central control in distributed systems poses challenges such as emergent behavior [12-17]. Emergent behavior is behavior in a synthesized model of the distributed system not explicitly specified in its specification. Emergent behavior arises when there is a state in which the system component cannot determine which course of action to take. For instance, deadlock is a form of emergent behavior.

L. Graham is with the Schulich School of Engineering Biomedical Engineering Program (email: lmgraham@ucalgary.ca).
M. Moshirpour is with the Department of Electrical and Computer Engineering (email: mmoshirp@ucalary.ca).
M. Smith is with the Schulich School of Engineering Biomedical Engineering Program, Department of Electrical and Computer Engineering, and Department of Radiology (email: smithmr@ucalgary.ca).
B. H. Far is with the Department of Electrical and Computer Engineering (email: far@ucalgary.ca).
University of Calgary, 2500 University Drive NW, Calgary, Alberta, Canada. T2N 1N4

978-1-4799-2131-7/14/$31.00 ©2014 IEEE

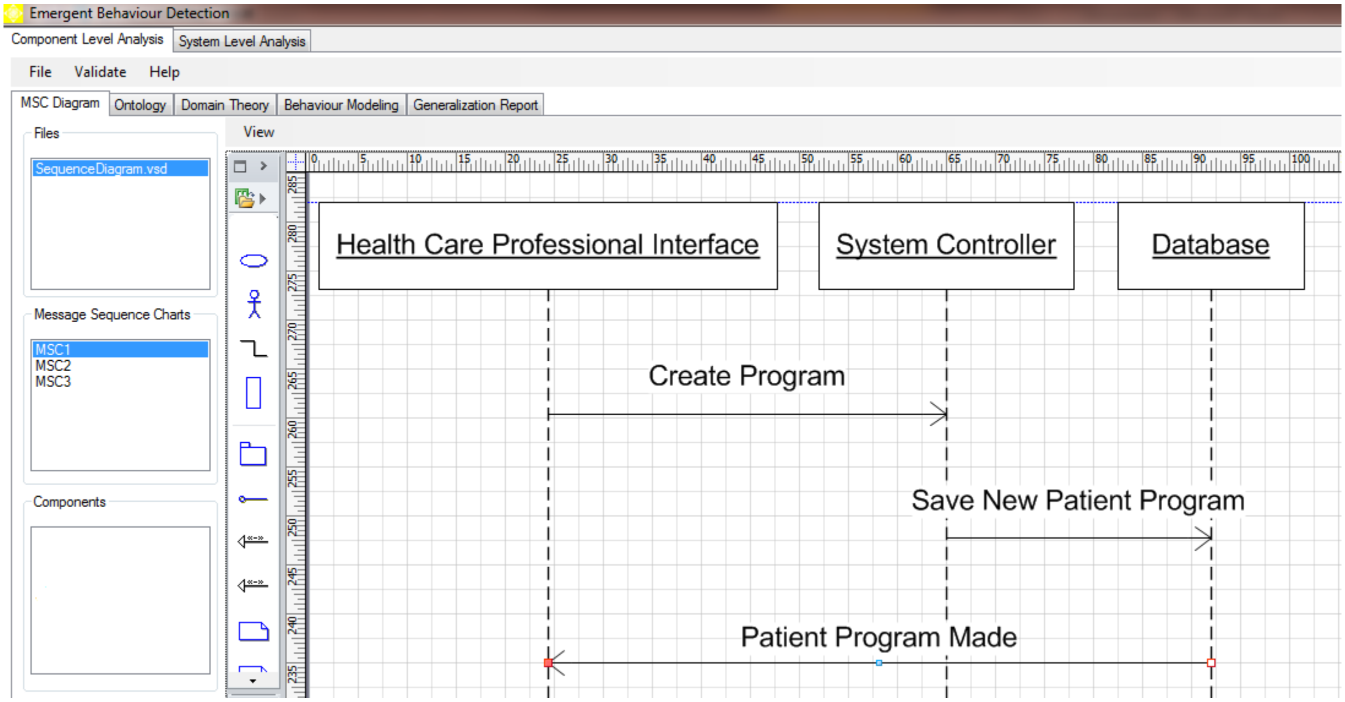

Fig. 1. *Message sequence chart for creation of new patient program shown in the emergent behavior detection tool. The health care professional can modify the patient's file, but the patient can neither view nor contribute.*

Although emergent behavior is not always problematic, there are many cases in which emergent behavior becomes synonymous with unwanted behavior of the system [12-17].

Literature suggests that detecting unwanted behavior during the design phase of a project is up to 20 times less expensive than finding that behavior during the deployment phase [18]. Unfortunately, manual review to reconcile requirement and design documents may not efficiently detect all the design flaws and emergent behaviors when components interact as part of a large and complex system.

To ensure integrity of software systems, methodologies using scenario-based software engineering have been derived. The emergent behavior detection (EBD) tool was developed using this approach and is used to analyze a variety of software systems in areas such as robotics and engineering [14, 19]. This tool performs automated analysis of design artifacts and software requirements using the message sequence chart (MSC) formal notation developed by the International Telecommunication Union (ITU) [20].

In this paper, we demonstrate the use of the EBD tool in the complex context of a personal health system. Previous work with the EBD tool has not covered the patient element. The design of the system presented here is very high level and is based on private conversations with people involved in community health programs for cancer patients [21, 22]. The EBD tool is used to detect possible emergent behavior arising when an existing patient program is modified to become a system including both health care professionals and patients.

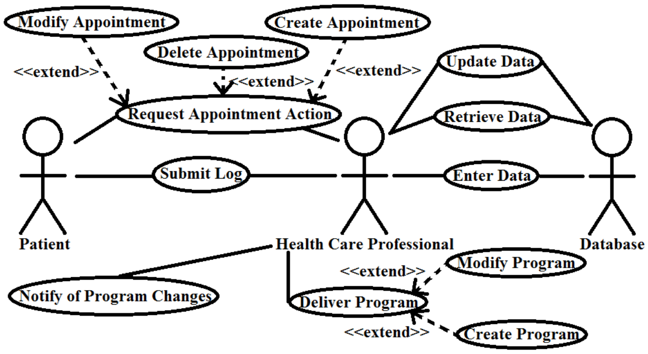

Fig. 2. *Use case diagram of the existing system. Patient interactions are done either on paper or verbally with the health care professional. The health care professional can access the database and manually enters information collected from the patient.*

The remainder of this paper is organized as follows: in Section 2, the details of the original and modified personal health systems are presented as use case diagrams with key features of the systems expressed in MSC formal notation. This allows us to use software analysis methodology and the EBD tool in Section 3 to identify emergent behavior that might occur during the move to the modified personal health system. Section 4 presents results and conclusions and future work are presented in Section 5.

II. CASE STUDY: PERSONAL HEALTH SYSTEM

First, we provide details of an existing personal health system with use case diagrams and message sequence charts (MSC). Then, the modified system is presented through use case diagrams and MSCs.

A. Existing System

The existing system has interactions between patients and health care professionals (HCP). The main events include recording information on paper (patient), submitting logs (patient to HCP), and entering data into a database (HCP).

Figure 2 illustrates appointments and program execution (comprised of creation and maintenance). Note that the HCP is a central figure in the system. Their roles include transferring data from patients to the database and creating and maintaining the program. Although the patient has two points of entry into the system – requesting an appointment action and submitting logs – both are mediated by a HCP.

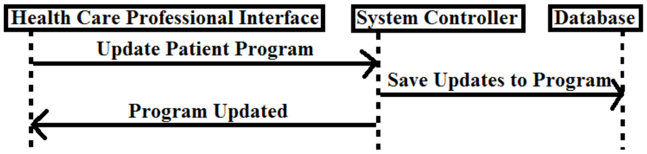

Fig. 3. *Message sequence chart for updating an existing patient program. Again, the health care professional is able to modify the patient's file, but the patient has no way to view or contribute to the file.*

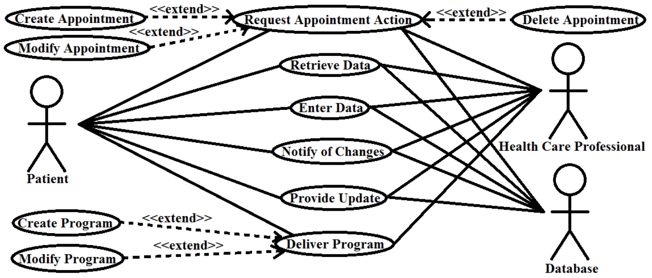

Fig. 4. *Use case diagram of proposed modified system. Here, the patient has more points of entry into the system. The health care professional and patient work together to contribute to the patient's information.*

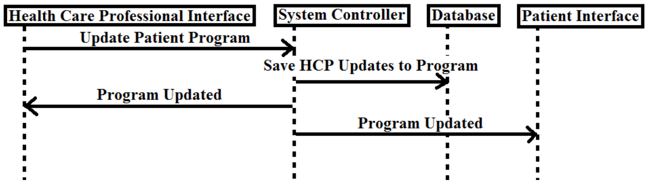

Fig. 5. *Message sequence chart for program update done by both patient and the health care professional. The health care professional updates the patient program. Updates are pushed to both the patient and health care professional interfaces. Contrary to program updates in Figure 3, the patient receives updates/has access to the latest database information.*

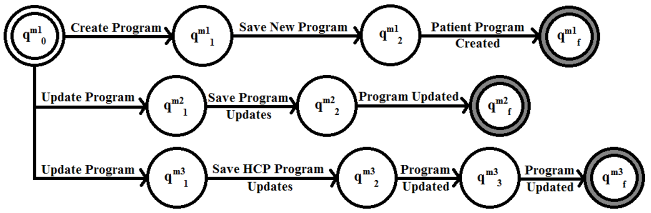

Fig. 6. *Behavior model for the system controller post-modification.*

The use cases in Figure 2 are further detailed by the MSC in Figure 1 which shows the creation of a new patient program. Once the HCP has created the program, they meet with the patient to share the information via paper and verbal explanations. Information is always being routed through the HCP, which leaves room for error. For example, if the HCP is running short on time (a common occurrence), information for patient A may inadvertently be given to patient B.

A second scenario is updating the patient's program (Figure 3). Here, the HCP modifies the patient's program according to their progress. As the HCP receives logs from the patient and enters them into the database, the HCP is aware of the patient’s status (such as any missing logs). Once the program is modified, the HCP meets with the patient to go over updates and changes via paper and verbal interactions. Again, the information is routed through the HCP potentially leading to errors, especially if more than one HCP is involved or time is an issue. Note that Figure 3 highlights limited patient involvement.

B. *Modified System*

Figure 4 shows the use case diagram for a set of proposed modifications to the existing system. Of significant importance is the number of points of entry the patient has into the system and the characteristics of these entry points. Many entry points allow the patient to directly interact with the database rather than dealing with a gatekeeper (the HCP). This allows patients to request and enter information, which may improve data integrity (reduces the number of times information is recorded/number of people entering the same data) and reduce the time the HCP spends on tasks the patients can now execute on their own (such as checking appointment times).

In this scenario, the patient and HCP each have an interface connecting to the system. For example, the patient and HCP are both able to create, modify, and delete appointments. HCPs are able to create and modify programs for patients and patients are immediately notified of changes. Patients are able to enter their logs directly into the system and this data can be immediately seen by the HCP.

Further exploring the use case in Figure 4, Figure 5 illustrates the happy path (sequence of events executed with no exceptions [23]) of updating a program in the modified system. In this case, the HCP views the patient's information and tweaks the program according to the patient's current state. After the changes have been made, the HCP interface is updated, the patient is notified of the changes, and the patient interface is updated. This path assumes the patient information is accurate and up-to-date.

The next section introduces the EBD tool used to verify the MSCs and suggests considerations for system integrity.

III. DESIGN VERIFICATION

Analyzing the design of distributed systems consists of two steps [14]. First, the behavior model with each system component described as scenarios is constructed. Second, the system is analyzed for design flaws such as emergent behavior. In this section, we show how the scenarios from the original system and proposed system can be input into the EBD tool for automated analysis of software design artifacts. The EBD tool performs this analysis by extracting domain knowledge from scenarios and reporting on possible areas of emergent behavior [14, 19].

A. *Behavior Modeling*

The model describing the behavior of each system component is usually called the behavioral model. The process of building the behavioral model from a scenario-based specification is called the synthesis process [15, 24]. A commonly used model for behavioral modeling of individual components is the state machine. There are several reports on the procedure of converting a set of sequence diagrams to a behavioral model expressed by state machines [25-27]. In the synthesis process, one state machine is built for each component. The state machine includes all the interactions of a component based on the messages it receives or sends. Figure 6 depicts the behavior model for the system controller component post-system modification. Theoretically, the behavior of the system can be described by the union (parallel execution) of all the state machines of the individual system components. The detailed mechanism for the synthesis of behavior models has been outlined in [14, 19].

B. *Detection of Emergent Behavior*

At this point, each agent is analyzed for design faults or emergent behavior. This happens when identical states exist in the union of state machines obtained through behavioral

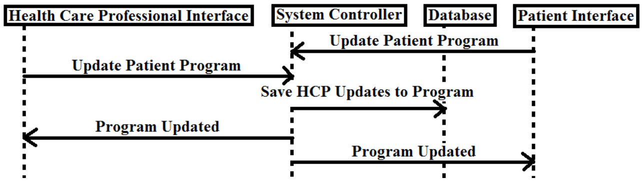

Fig. 7. *Output of the emergent behavior detection tool: possible emergent behavior detected. Here, the patient information in the database is not up to date (such as if the data is not synchronized due to lacking internet connection). The health care professional updates patient program according to out of date information in the database and thus the resulting program may be inaccurate. The patient interface is updated to the new program when the data is synchronized.*

modeling [14, 19]. Identical states are defined and treated differently in various works. For instance, Whittle and Schumann propose the assignment of global variables to the states by the system designer [28, 29]. However, the outcome of this approach is not always consistent as global variables chosen by different domain experts may vary.

This research formally defined the identical states and semantic causality to unify the approaches [14]. Semantic causality is an invariant property of the system and is defined as sequence of messages (events) an agent must keep to perform subsequent operations [14]. This information is extracted from scenarios using an ontology-based approach [30]. Based on semantic causality, an efficient and reliable method to assign values to the states of the state machines has been achieved (details presented in [14]). This methodology has been developed into a software tool to automate the process of emergent behavior detection (Figure 7). This figure accepts scenarios as input and detects and reports emergent behavior.

IV. CONCLUSIONS AND FUTURE WORK

Personal health systems and eHealth systems can provide patients with information about their health while allowing them to contribute additional information. However, creating a distributed system to support this is challenging. By using the EBD software to evaluate the modified system design presented here, we identified an issue preserving system integrity where stale data could affect program changes.

Future work includes integrating the emergent behavior detected into the software design, expanding the clinical workflow, and using the EBD software to detect emergent behavior in a more mature design. In addition, ensuring proper system security is in place, preserving privacy, and considering social factors should be taken into account.

ACKNOWLEDGMENTS

L. Graham would like to thank the Biomedical Engineering Graduate Program at the University of Calgary and Arcurve Inc. of Calgary, Alberta, Canada, for their financial support, N. Culos-Reed and L. Capozzi for information on health systems, and A. Tang for his feedback. M. Moshirpour would like to thank Natural Sciences and Engineering Research Council of Canada (NSERC), Alberta Innovates-Technology Futures (AITF), and the Electrical and Computer Engineering department at the University of Calgary, Alberta, Canada, for their financial support.

REFERENCES

[1] M. J. Ball and J. Lillis, “E-health: transforming the physician/patient relationship,” *Int. J. Med. Inf.*, vol. 61, no. 1, pp. 1–10, Apr. 2001.

[2] C. Showell and C. Nøhr, “How Should We Define eHealth, and Does the Definition Matter?,” in *Quality of Life Through Quality of Information*, 2012, pp. 881–884.

[3] H. Oh, C. Rizo, M. Enkin, A. Jadad, J. Powell, and C. Pagliari, “What Is eHealth: A Systematic Review of Published Definitions,” *J. Med. Internet Res.*, vol. 7, no. 1, Feb. 2005.

[4] A. D. Black, J. Car, C. Pagliari, C. Anandan, K. Cresswell, T. Bokun, B. McKinstry, R. Procter, A. Majeed, and A. Sheikh, “The Impact of eHealth on the Quality and Safety of Health Care: A Systematic Overview,” *Plos Med*, vol. 8, no. 1, p. e1000387, Jan. 2011.

[5] “eHealth Ontario.” [Online]. Available: https://www.ehealthontario.ca/portal/server.pt/community/home/204.

[6] “European Commission eHealth Network.” [Online]. Available: http://ec.europa.eu/health/ehealth/policy/network/index_en.htm.

[7] A. E. Z. M. Faap and G. R. K. M. Faap, “Personal Health Records,” in *Pediatric Informatics*, C. U. L. M. FAAP, G. R. K. M. FAAP, and K. B. J. M. M. FAAP, Eds. Springer New York, 2009, pp. 293–301.

[8] P. C. Tang, J. S. Ash, D. W. Bates, J. M. Overhage, and D. Z. Sands, “Personal Health Records: Definitions, Benefits, and Strategies for Overcoming Barriers to Adoption,” *J. Am. Med. Inform. Assoc.*, vol. 13, no. 2, pp. 121–126, Mar. 2006.

[9] “Fitbit.” [Online]. Available: http://www.fitbit.com/uk. [Accessed: 30-Apr-2013].

[10] “MyFitnessPal.” [Online]. Available: http://www.myfitnesspal.com/. [Accessed: 30-Apr-2013].

[11] N. Archer, U. Fevrier-Thomas, C. Lokker, K. A. McKibbon, and S. E. Straus, “Personal health records: a scoping review,” *J. Am. Med. Inform. Assoc.*, vol. 18, no. 4, pp. 515–522, Jul. 2011.

[12] B. Adsul, M. Mukund, and K. N. Kumar, “Causal closure for MSC languages,” in *FSTTCS 2005, vol. 3821 of LNCS*, 2005, pp. 335–347.

[13] R. Alur, K. Etessami, and M. Yannakakis, “Inference of message sequence charts,” *IEEE Trans. Softw. Eng.*, vol. 29, no. 7, pp. 623–633, 2003.

[14] M. Moshirpour, “Model-Based Detection of Emergent Behavior In Distributed and Multi-Agent Systems from Component Level Perspective,” Master of Science, University of Calgary, Department of Electrical and Computer Engineering, 2011.

[15] A. Mousavi, “Inference of Emergent Behaviours of Scenario-Based Specifications,” PhD Thesis, University of Calgary, Electrical and Computer Engineering, 2009.

[16] H. Muccini, “Detecting implied scenarios analyzing non-local branching choices,” in *Proceedings of the 6th international conference on Fundamental approaches to software engineering*, Berlin, Heidelberg, 2003, pp. 372–386.

[17] S. Uchitel, J. Kramer, and J. Magee, “Negative scenarios for implied scenario elicitation,” *Sigsoft Softw Eng Notes*, vol. 27, no. 6, pp. 109–118, Nov. 2002.

[18] R. F. Goldsmith, *Discovering real business requirements for software project success*. Artech House, 2004.

[19] M. Moshirpour, A. Mousavi, and B. H. Far, “A Technique and a Tool to Detect Emergent Behavior of Distributed Systems Using Scenario-Based Specifications,” in *22nd IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, 2010, vol. 1, pp. 153–159.

[20] “ITU: Message Sequence Charts. Recommendation, International Telecommunication Union.” 1992.

[21] L. Capozzi, “Private Conversation,” 2012.

[22] N. Culos-Reed, “Private Conversation,” 2012.

[23] P. Bollen, “BPMN: A Meta Model for the Happy Path.” [Online]. Available: http://arno.unimaas.nl/show.cgi?fid=18194.

[24] M. Lohrey, *Safe Realizability of High-Level Message Sequence Charts*. 2002.

[25] D. Harel, “From play-in scenarios to code: an achievable dream,” *Computer*, vol. 34, no. 1, pp. 53–60, 2001.

[26] D. Harel and H. Kugler, “Synthesizing State-Based Object Systems from LSC Specifications,” in *Impl. and Appl.of Automata*, S. Yu and A. Păun, Eds. Springer Berlin Heidelberg, 2001, pp. 1–33.

[27] E. Mlikinen and T. Systa, “MAS - an interactive synthesizer to support behavioral modeling in UML,” in *Proceedings of the 23rd International Conf. on Soft. Eng., 2001. ICSE 2001*, 2001, pp. 15–24.

[28] J. Whittle and J. Schumann, “Generating statechart designs from scenarios,” in *Proceedings of the 22nd international conference on Software engineering*, New York, NY, USA, 2000, pp. 314–323.

[29] J. Whittle and J. Schumann, “Scenario-Based Engineering of Multi-Agent Systems,” in in *Agent Technology from a Formal Perspective*, C. A. R. BA MS, P. M. H., MSc, BSc, CMath, CEng, CPEng, CITP, FBCS, FIEE, FIMA, FIEAust SMIEEE, J. R. M. BA, W. T. M. BA, and D. Gordon-Spears, Eds. Springer London, 2006, pp. 159–189.

[30] M. Moshirpour, S. Mireslami, R. Alhajj, and B. H. Far, “Automated ontology construction from scenario based software requirements using clustering techniques,” in *2012 IEEE 13th International Conf. on Information Reuse and Integration (IRI)*, 2012, pp. 541–547.