MARBLS: A Visual Environment for Building Clinical Alert Rules

Dave Krebs, Alexander Conrad, Milos Hauskrecht, Jingtao Wang
Department of Computer Science
University of Pittsburgh
210 South Bouquet Street, Pittsburgh, PA 15260, U.S.A.
{djk37, conrada, milos, jingtaow}@cs.pitt.edu

ABSTRACT
Physicians and nurses usually rely on hospital information systems (HIS) for detecting a variety of adverse clinical conditions and reminding repetitive treatments. However, the acquisition of alert rules expected by HIS from experts remains a challenging, error-prone, and time-consuming process. In this work, we present MARBLS (Medical Alert Rule BuiLding System) - a visual environment to facilitate the design and definition of clinical alert rules. MARBLS enables a two-way, synchronized visual rule workspace and visual query explorer. Monitoring rules can be built by manipulating block components in the rule workspace, by querying and generalizing region of interests in the visual query explorer via direct manipulations, or a combination of both. Informal testing with doctors has shown positive feedback.

ACM Classification: H5.2 [Information interfaces and presentation]: User Interfaces. Graphical user interfaces.

General terms: Design, Human Factors

Keywords: visual language, end user programming, health care, clinical monitoring rules

INTRODUCTION
Modern hospitals often rely on various clinical decision support systems and tools [2] aimed to help physicians and nurses to monitor patients, alert on the occurrences of various adverse events, or send reminders assuring the adherence to various clinical guidelines. In majority of existing systems, the knowledge about what to detect and when to alert is expressed in terms of if-then (or condition-action) rules. As an example, consider a rule for detecting a patient who is at risk of the heparin-induced thrombocytopenia (HIT) [11], a life threatening condition:

***If** patient is on heparin, dalteparin, or enoxaparin and the most recent platelet count is low (<100k) or the platelet count has dropped over 50% during last seven days, **then** send an email alert to the attending clinical pharmacist.*

According to our interviews with physicians, the number of these rules in a monitoring subsystem is typically in hundreds. These rules are usually written in a special rule-definition language, such as Cerner CCL [3]. If rules are simple, the task of writing them for an expert is often very easy. However, if rules and conditions they detect are more complex, writing them accurately in the given language can become a very challenging process. In fact, it is not uncommon that the physicians have to work closely with programmers to accurately encode them in the system.

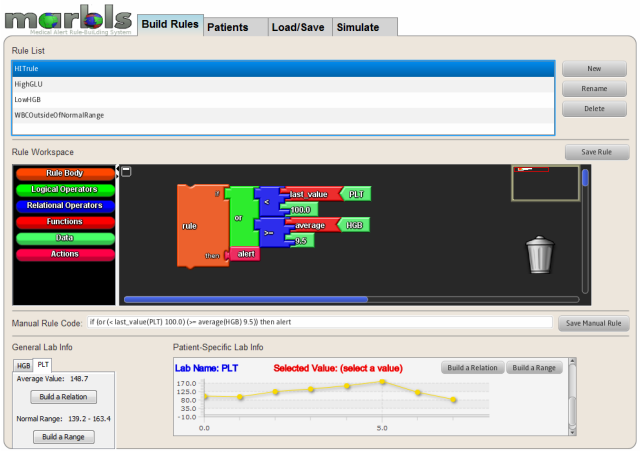

Figure 1: The primary interface of MARBLS. Top: commonly used alert rules. Middle: the interactive *rule workspace*. Bottom: the *visual query explorer*.

THE DESIGN OF MARBLS
In this paper, we present MARBLS (Medical Alert Rule BuiLding System), a visual programming environment for defining clinical alert rules interactively by non-programmers (Figure 1).

Similar to end-user programming platforms such as KidSim [4], Scratch [9] and Google App Inventor [5], our system supports a visual environment (*rule workspace*) to build alert rules by dragging-and-dropping puzzle-piece style rule components. Like TimeSearcher [7], our system also provides a *visual explorer* to specify and update alert rules by selecting and generalizing rules via direct manipulations. Most importantly, MARBLS is unique in that it maintains two-way, synchronized connections between the *rule workspace* and the *visual query explorer* – e.g. operations applied in one view are able to generate and update properties in the other view, and vice-versa. This property

UIST'11, October 16–19, 2011, Santa Barbara, California, USA.
ACM 978-1-4503-1014-7/11/10.

can be considered a generalized brushing [1] operation between data charts and a visual programming environment.

MARBLS supports four types of blocks at this moment; they are body blocks, operator blocks (logical and relational), function blocks, lab observational data (both generic and patient specific) blocks (e.g. platelets level, a.k.a. PLT; glucose level, a.k.a. GLU), and action blocks. Users directly manipulate blocks, dragging them onto the workspace and connecting them together to form rules. The particular outlines of the docking sites on each block piece ensure that blocks can only be connected to others in such a way that will yield syntactically valid rules. Such interlocking puzzle-piece based approaches have been demonstrated to be effective by non-technical users to complete programming tasks [8].

MARBLS also supports building and updating rules in the *rule workspace* by conducting selecting and lassoing operations in the *visual query explorer*. For example, a user may want to build a rule that checks to see if a user's glucose (GLU) level is in the normal range. If the user adds a glucose observation block to the *rule workspace*, glucose data will be presented automatically, including the normal range of values in the *visual query explorer*. Then by pressing a button in the *visual query explorer*, the user can add the condition which evaluates to *true* if the patient's glucose level falls within the normal range.

IMPLEMENTATION

MARBLS is implemented in Java. JavaFX and OpenBlocks [10] have been adopted for charting and block manipulation support. MARBLS is an open source project. The source code is available at https://github.com/omgpotatoes/marbls.

DISCUSSIONS AND FUTURE WORK

MARBLS is still under development. To test our ideas and verify the viability of the approach, we have conducted an informal study with two physicians, who tested MARBLS while attempting to implement a rule for one of the adverse clinical conditions.

In summary, we received highly positive, enthusiastic feedback during the informal study. Both subjects enjoyed using the graphical rule-building interface of MARBLS and found the rule building features easy to understand and intuitive to use.

At the same time, we also received multiple suggestions for further improvements. First, the slope (i.e. moving trend) of certain lab observational data is important in many decision making tasks. Although MARBLS supports slope operations via function blocks, it will be better to support more generic slope based range query [7] and relaxations [6] in the *visual query explorer*. Second, the users expected that MARBLS could provide normal range values of healthy people when lab observational data are being accessed in either the *rule workspace* or the *visual query explorer*.

To the best of our knowledge, MARBLS is the first attempt to support the construction of clinical alert rules by combining a two-way, synchronized *visual rule workspace* and with the *visual query explorer*. We believe such an approach may be beneficial also for other rule-based applications in temporal environments.

The puzzle-piece style programming paradigm and interactions are chosen not only for providing visual constraints of rule grammars, but also for supporting touch screens and touch operations. We are in the process of porting MARBLS to Android Tablets. Based on informal feedback, the portability and affordance of such internet tablets is ideal for health care environments. Future work also includes adding medication (treatment related tasks) to the system and developing a better interface for the rule testing/debugging.

CONCLUSION

We have presented MARBLS (Medical Alert Rule BuiLding System), an end-user programming environment for defining clinical alert rules. MARBLS leverages a combination of puzzle-piece based visual editor and visual query explorer to lower the threshold of clinical alert rule construction for non programmers. Informal testing with doctors has shown promising initial results.

REFERENCES

1. Becker, R., Cleveland, W., Brushing Scatterplots. *Technometrics*, 29(2):127-142. 1987.
2. Berner, E., ed. Clinical Decision Support Systems: Theory and Practice. New York, NY: Springer, 2007.
3. Cerner CCL, http://en.wikipedia.org/wiki/Cerner_CCL
4. Cypher, A., Smith, D., KidSim: End User Programming of Simulations. In *Proc of CHI 1995*, pp. 27-34.
5. Google App Inventor for Android. http://appinventor.googlelabs.com/about/
6. Heer, J., Agrawala, M., et al, Generalized Selection via Interactive Query Relaxation, In *Proc of CHI 2008*, pp. 959–968.
7. Hochheiser, H., Shneiderman, B. Dynamic query tools for time-series data sets: Timebox widgets for interactive exploration. *Information Visualization*, 3:1-18. 2004.
8. Horn, M., Solovey, E., et al. Comparing the use of tangible and graphical programming languages for informal science education. In *Proc of CHI 2009*, pp. 975–984.
9. Resnick, M., Maloney, J., et al. Scratch: Programming for All. In *Comm of the ACM*, November 2009.
10. Roque, R., OpenBlocks : an extendable framework for graphical block programming systems, *Master thesis, Dept. of EECS, MIT*, 2007.
11. Warkentin T., Heparin-induced thrombocytopenia: pathogenesis and management. British Journal of Haematology; 121:535-555, 2003