Engaging Clinicians in the Visualization Design Process – Is It Possible?

Kostas Pantazos

IT-University of Copenhagen

ABSTRACT

Creating and customizing visualization for electronic health record data requires a close collaboration with clinicians, to understand their tasks, needs and mental model. This process can develop into an infinite process. Taking into consideration the existence of clinicians with advanced IT knowledge, but not programmers, we focus on engaging them to create their own visualizations. This paper presents how clinicians can use uVis Studio to create three visualizations by dragging and dropping controls into the design panel, and specifying formulas for each control in the property grid.

KEYWORDS: Visualization Tool, Spreadsheet Formulas, Development Environment, Design Process, Health Care.

INDEX TERMS: H.5.2. [Information Interfaces & Presentation]: User Interfaces – Graphical User Interfaces (GUI)

1 INTRODUCTION

Healthcare systems provide a huge amount of data and the challenge of presenting these data is present. Clinicians need easy and intuitive presentations that fulfill their tasks and needs based on their experience and knowledge [7]. Most of EHR systems use more table or text based presentation rather than visualization techniques. Innovative visualizations like LifeLines [9], TimeLine [3], etc. provide a better presentation. These visualizations have been developed in close collaboration between developers and clinicians who have the domain knowledge. Creating and customizing advanced visualizations need programming skills and considerable time.

Although several visualizations have been developed for clinical data, there is a need for more novel and customizable visualizations [3]. Clinicians need presentations which are easy to understand and to access the right information [3]. Furthermore, the visualization has to match the mental model of the clinician. To overcome this challenge, it is recommended that clinicians are involved during the development process of a user interface or visualization [7]. Applying user-centered design may resolve these issues, but still questions like: "What about the clinicians that did not participate in the design process? Are the representatives a good sample, to conclude to the right visualization?". Furthermore, is the same visualization sufficient for the same department but in different hospitals ? Answering these questions raises several challenges which are also closely related with the available time, budget and resources used.

Using user-centered design does not solve the problem of customizability; adjusting an existing visualization to clinician needs. For instance, different departments or different hospitals have different needs. Different clinicians perform the same tasks in different ways, because of different experiences, knowledge and so forth. The same visualizations can be integrated in different departments or used by different clinicians, but to achieve better user satisfaction some changes may be needed. Furthermore, there is a need for more customizable visualizations to fulfill users' needs [3], and more tools which can support this customizability.

Nowadays, some clinicians have gained advanced IT skills, starting from simple browsing through web-applications to more advanced applications, such as MS Excel or MS Access. For instance at Bispebjerg hospital in Copenhagen, Denmark, a department uses a system developed in MS Access by one of the clinicians. We believe that in the healthcare environment there are a considerable number of such clinicians with advanced IT knowledge. So, with proper training, engaging clinicians in the process of developing their own visualizations using a specialized development environment will increase even more the possibility of developing successful visualizations for clinical data.

We present uVis, a formula-based visualization tool for clinicians. This tool provides clinicians with a development environment (uVis Studio) to design their visualizations. Clinicians with advanced spreadsheet level knowledge and familiar with basic database concepts can design visualization by dragging and dropping controls into the design panel. Next, specifying simple and advanced formulas in the property grid, they can bind controls to data and specify controls properties such as color, height, width, etc. The uVis Studio provides the basic features a development environment has, and more specialized ones such as data related intellisense and a design panel which shows visualizations as it would look to the end-users, described in the next sections. Finally, clinicians without IT experience can collaborate with IT experienced clinicians to create visualizations and use them as well.

2 RELATED WORK

2.1 Visualizations in healthcare

One of the most well-known visualizations in healthcare is the LifeLines [9]. It presents the history of a patient's medical record and it was designed in close collaboration with clinicians initially, and later with a cardiologist. This presentation uses the timeline metaphor, data presented in facets, color coding and size coding. The evaluation showed that the Lifelines was more understandable and that clinicians responded faster than the traditional presentations. This visualization was developed in Java, and customizing it requires advanced programming skills. The TimeLine system by Bui et al. visualizes problem-centric patient data [3]. Their study showed that clinicians need more flexible visualizations which fulfill their needs and tasks. A need for more flexible visualization and customizable by clinicians is raised by An et al. [1]. An integrated viewer for EHR was developed with basic visualization techniques, where clinicians were able to hide and show visualizations but not customize them to their needs.

Although, several previous research projects have concluded that there is a need for more customizable visualizations in healthcare, to our knowledge there is no previous research addressing this problem or engaging clinicians directly in the development process.

2.2 Visualization tools

We investigated some popular tools in the market for non-programmers mainly used in the business area. MS Excel [8] provides a user-friendly interface where built-in visualizations can

be created with few steps. However, this tool provides a limited number of visualizations which are not fully customizable. For instance, graph colors cannot reflect data values. Furthermore, users cannot create new visualization types and integrate them into MS Excel. Finally, due to the amount and structure of data in an EHR system, clinicians may encounter difficulties in creating meaningful visualizations with MS Excel. Other visualization tools such as Spotfire [10] and Tableau [11] are more specialized in data visualization and provide a larger variety of visualizations. Nevertheless, these tools do not support users to create and customize advanced visualizations, such as LifeLines. User creativity is restricted to the pre-designed views. Furthermore, creating appropriate visualizations with Tableau or Spotfire needs some advanced knowledge on how to create visualizations.

In academia, there are several visualization toolkits [2, 5, 6] for programmers. Programmers can create and customize visualizations by means of programming. Unfortunately, this approach is too complex for users with advanced spreadsheet-like knowledge, such as clinicians. Most of these toolkits miss an integrated development environment. Usually, they can be integrated in general-purpose integrated development environments (IDE) such as Visual Studio, Eclipse, etc., but still is not enough for non-programmers. A specialized IDE should support users in creating and customizing visualizations by means of simple actions such as drag-and-drop.

3 SOLUTION

Previous research [1, 3] has been using user-centric design where clinicians had a close collaboration with the developer. We propose a different approach on developing visualizations for healthcare data: allow clinicians with advanced IT knowledge to create and customize their own visualizations using uVis.

uVis Studio (figure 2) is the development environment of uVis and contains six work areas. *Toolbox* lists the available controls, and supports drag-and-drop. *Design Panel* shows the visualization as it would look to the end-user. This panel is updated every time a control is dragged-dropped or a control property is changed. Hence, the user sees exactly the same screen in development mode as well as in end-user mode. *Property-Grid* is the area where a user can type the formulas. We integrated the intellisense feature in the Property-Grid to reduce typing errors and misunderstandings. Furthermore, the intellisense assists clinicians with suggestion related to control properties, tables and table fields. *Solution Explorer* is the area where project files are listed. The clinician can create a new project by adding a visualization mapping document (.vism) and a visualization file (.vis). Vism files contain information regarding the database the user is using, the tables, etc. The Vis file contains the visualization specifications. *Design Modes* allows the user to choose the mode for viewing and interacting with visualizations in the design panel. For instance, the user can select the mode *InteractionMode*, which deactivates event handlers attached to the visualization in the development environment. *Data Map*, currently under development, provides a visual overview of tables, fields and relationships in the database the user is using. It resembles an entity relationship (ER) diagram.

In the remainder of this section, we present three scenarios, three visualizations and elaborate on how they were created by the author.

3.1 Scenario 1: Simple LabResults visualization

In one of the clinics at Copenhagen Hospital, clinicians use the VistA EHR system. For each patient that comes in the clinic, they

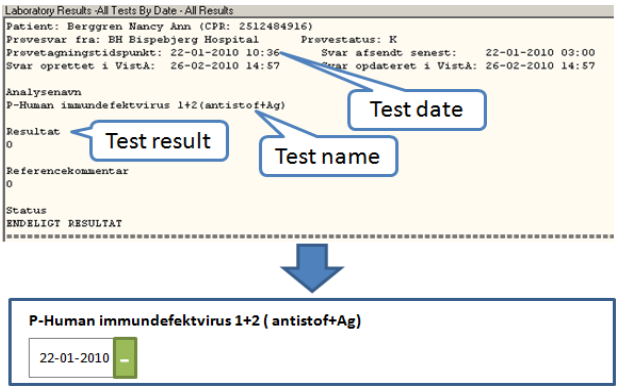

Figure 1. a) Current presentation at the clinic and b) a potential solution for presenting patient Lab Results.

have to check the lab results of the patient. Figure 1.a presents a screenshot of the presentation of a patient lab result that clinicians use, and our simple solution using uVis in figure 1.b. Clinicians have to go through all the cumbersome texts for more than one lab test and find the important information for the patient. The lab test has a positive or negative result. A simple overview of the current state of the patient is missing. In the early phase of our research, we collaborated with clinicians who identified three important variables (date, result and lab name) in the texts, which are used in our visualization created using uVis Studio. Our approach is trying to minimize this collaboration and empower the clinicians to create their visualization.

Preconditions: uVis can visualize only relational data at the moment, for instance data in MS Access. The Vism file has to be created the first time by the database manager, unless the clinician who will use uVis studio has good database knowledge. Furthermore, an introduction of how the studio works and how to use formulas is necessary for clinicians.

3.1.1 Using uVis Studio

Figure 2 shows a screenshot of the studio, containing a simple visualization for the lab results, and some of the steps clinicians have to follow. The clinician opens the uVis Studio and selects the Vism file using the explorer. The default Vis file is opened in the design panel. In our case it will be an empty form.

Clinicians can drag and drop controls (e.g. panel, label, textbox, etc.) in the design panel. Furthermore, they can resize the controls and move them around the design panel. For each control they specify simple and advanced formulas for control properties in the property grid. Every change done in the property grid reflects on real-time on the design panel. Unlike other development environment, uVis Studio shows the form exactly as it will be shown at the end-user outside Studio. Clinicians use the property grid to specify the formulas. Intellisense feature helps them to write the correct formulas. For instance, clinician starts typing "cli" in the DataSource property and a list of suggestions will pop-up with name of tables, table fields, controls and control properties that contain "cli".

3.1.2 Key Principles of uVis Kernel

In this section we present some of the key principles of uVis Kernel which are used in creating the LabResult visualization, Figure 2.

Controls: Visualizations are created by combining .Net controls, simple shapes (e.g. triangle) and several special uVis controls (e.g. timescale). A control can be bound to data that

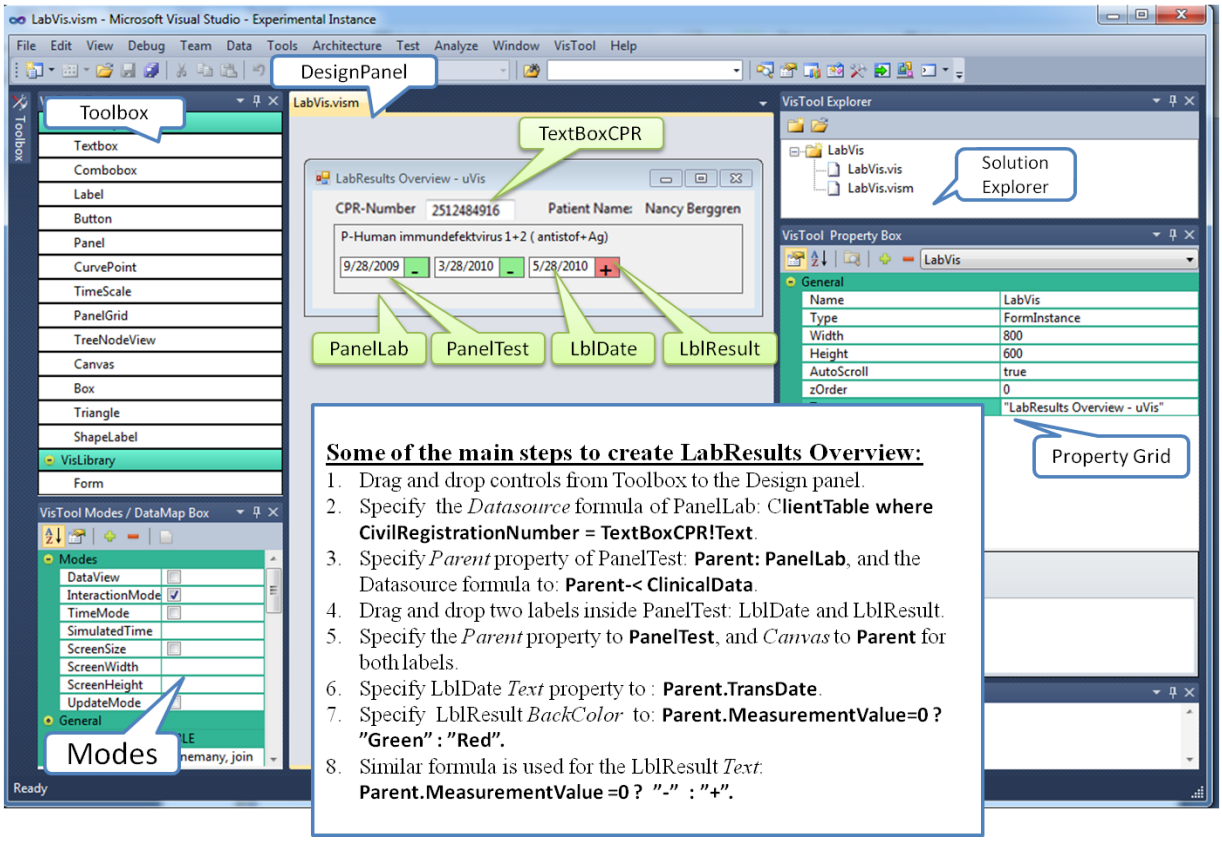

Figure 2. Creating LabResults overview with uVis Studio.

makes it repeat itself. A control has a number of properties that specify its appearance and its behavior.

Formulas: Control properties can be specified by spreadsheet-like formulas. The formula specifies how to compute a property value for a control. A formula can refer to data in the database, control properties. uVis kernel computes the formulas for each control, and sets the property values accordingly.

Bind control to data: Each control may have a data source that binds it to data rows. To define the data source, in this case the clinician specifies the DataSource, the uVis property of the control. The clinician writes a formula which represents an SQL statement. uVis kernel translates the DataSource formula into an SQL statement, retrieves data from the database and generates the corresponding record set. Next, the control creates one control for each row in the record set. Each control is bound to a row in the record set.

To create the visualization showed in figure 2, we used only two tables from our EHR database: ClientTable and ClinicalData. Each patient may have one or more clinical data. For instance in Figure 2, the patient is tested three times for P-Human immundefektvirus 1+2.

The clinician specifies the DataSource of panel PanelLab as follows:

ClientTable where CivilRegistrationNumber = TextBoxCPR!Text

ClientTable refers to a table in the data model and CivilRegistrationNumber is a field in table ClientTable. The dot (.) operator allows the clinician to access a table field. TextBoxCPR is the control of type TextBox that shows the patient civil registration number (CPR). The operator ! allows the user to access a control property. Thus, TextBoxCPR!Text is the current patient's CPR. As a result, the data source of PanelLab is the patient record whose civil registration number is specified in TextBoxCPR. As a result, uVis kernel creates one PanelLab control.

To show the lab tests of a patient, the user drags and drops a panel (PanelTest) inside PanelLab and specifies the DataSource of PanelTest as follows:

Parent -< ClinicalData.

Parent means the data parent of PanelTest, in this case PanelLab. The operator -< allows us to navigate from one row to multiple rows. Therefore, we navigate from the parent row (the ClientTable row) to the related ClinicalData rows. This allows us to access the lab tests of the patient. uVis kernel automatically detects the tables and table fields used in the formulas. Next, uVis Kernel translates the formula to an SQL statement, which is executed and a record set is created. In this case the record set contains three rows. Clinicians are not involved in this process, apart from the fact that they need to specify the correct formula in the property grid.

3.2 Scenario 2: Advanced LabResults visualization

We present in addition lab tests with numerical value as results. Instead of going through the text, clinicians can create or customize the first version of Lab Results Overview and present numerical lab tests as shown in Figure 3.

Following the same principles presented before, the clinician can bind controls to data. PanelTestScale presents visually the lowest and highest value this test may have in theory. However, in this case one of the test result was higher than 10. In this presentation, the clinician can spot it out easily, compared to the text based presentation. To align LblResultLine to PanelTestScale clinician specifies the Left property to:

PanelTestScale!Left

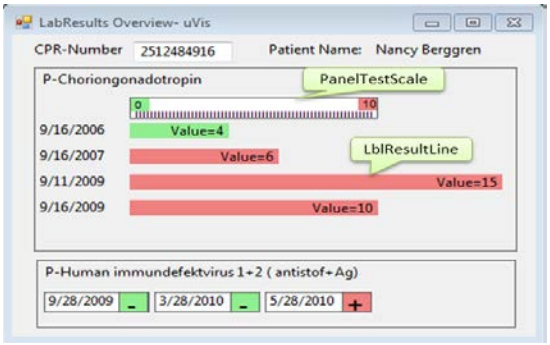

Figure 3. LabResults Overview using uVis Studio

To calculate the width of the LblResultLine, the clinician specifies this formula for the width property:

PanelTestScale!Width * Me.MeasurementValue / (Me.ValueHigh - Me.ValueLow).

Me is used to refer to the current instance, which is bound to a row. Using the dot operator we can navigate to a specific field of this row (MeasuremnetValue, ValueHigh and ValueLow in our case).

3.3 Scenario 3: LabResults using LifeLines

In the last scenario, the clinician creates a simple LifeLines visualization for some of the lab tests, shown in figure 4.

The clinician follows the same steps as before to bind controls to data. The difference in this case is the Timescale control, which is a uVis control. The clinician defines the period shown in the timescale by specifying the BorderValues to:

#2011-08-01#, #2011-09-01#

Clinicians can interact with the TimeScale control, moving the date backwards or forwards. To align the LblLabResult the clinician specifies the left position to:

TimeScaleLab!HPos (me.TransDate).

HPos is a special function in the timescale which translate date to pixels.

4 Discussion

Nowadays, computers are part of our daily and working life. More and more users are using computers to facilitate their working process. Starting from simple usage (such as checking emails, browsing web application), users, especially the new generation, are moving towards a better and broader understanding of how to utilize computers in daily work. The real case in Copenhagen Hospital, where a clinician developed an application in MS Access, confirms this tendency. Although several visualization tools exist, there is a need for new tools which provide a development environment for clinicians with advanced IT knowledge, but not programmers. Such a tool will facilitate the development process, allowing clinicians to create and customize their own visualization based on the department needs or their mental model.

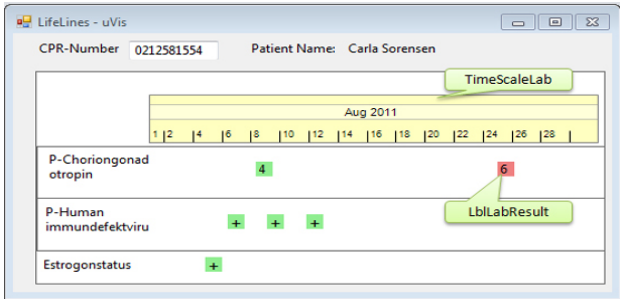

Figure 4. Simple LifeLines visualization using uVis Studio

In this paper, we present an on-going research project, which focuses on engaging clinicians in developing simple and advanced visualization using spreadsheet-like formulas. The spreadsheet formulas have proven to be successful approach among users and programmers [4]. Furthermore, by means of the development environment, clinicians can customize their visualization and adjust them to fulfill their needs.

The abovementioned visualizations were created by the author who has a good understanding of uVis Studio and formula principles, but is not a clinician. A more in depth evaluation with real clinicians is needed, and we are planning to conduct it in the future. The evaluation will show if our approach is adequate and if it is possible to engage clinicians in the visualization design process.

Now, we are focusing on making uVis Studio more stable. Data Map is being developed and simpler and advanced controls are being developed. A more specialized error messaging system for clinicians is being developed.

5 Conclusion

In this paper we presented a new visualization tool for clinicians. Clinicians can create and customize visualizations by means of iteratively dragging and dropping controls and specifying spreadsheet-like formulas. Although, three visualizations for lab results were developed, we plan to conduct an evaluation with real clinicians. To conclude, in this paper we present a first attempt to engage clinicians more and allow them to visualize the data in their own way.

References

[1] J. An, Z. Wu, H. Chen, X. Lu, H. Duan: Level of detail navigation and visualization of electronic health records, Proceedings of Biomedical Engineering and Informatics (BMEI), 2010.

[2] M. Bostock and J. Heer. Protovis: A graphical toolkit for visualization. IEEE Trans. Vis. and Comp. Graphics, 15(6):1121–1128, 2009

[3] A.A. Bui, D.R. Aberle, and H. Kangarloo, "TimeLine: Visualizing Integrated Patient Records", IEEE transactions on information technology in medicine, vol. 11, no. 4, 2007.

[4] M. Burnett, John Atwood, Rebecca Walpole Djang, James Reichwein, Herkimer Gottfried, and Sherry Yang. 2001. Forms/3: A first-order visual language to explore the boundaries of the spreadsheet paradigm. J. Funct. Program. 11, 2 , 155-206, March 2001.

[5] Flare. http://flare.prefuse.org, February 2011.

[6] J. Heer, S. K. Card, and J. A. Landay. “prefuse: a toolkit for interactive information visualization”. *In Proc. ACM CHI*, pages 421–430, 2005

[7] C. M. Johnson, T. R. Johnson, J. Zhang. 2005. A user-centered framework for redesigning health care interfaces. J. of Biomedical Informatics 38, 1, 75-87, February 2005.

[8] Microsoft Excel. http://office.microsoft.com/en-us/excel/, February 2011.

[9] C. Plaisant, R. Mushlin, A. Snyder, J. Li, D. Heller, and B. Shneiderman, “LifeLines: Using visualization to enhance navigation and analysis of patient records,” *In Proc. American Medical Informatic Association Annu. Fall Symp.*, Orlando, FL, , pp. 76-80, November 1998.

[10] Spotfire. http://spotfire.tibco.com/, February 2011.

[11] Tableau. http://www.tableausoftware.com/, February 2011.