

User Tailorable Microcontexts

Wendy A. Kellogg, Rachel Bellamy, Thomas Erickson, John Richards, John C. Thomas,
Jonathan Brezin, Laretta Jones, Jason Ellis

IBM T.J. Watson Research Center
Yorktown Heights, NY 10598

{ wkellogg | rachel | snowfall | ajtr | jctomas | brezin | jason@us.ibm.com }

ABSTRACT

We build on the work of Erickson and colleagues [1] to explore how end users might create visual representations based on streaming (and other) data. Such “microcontexts” might range from extremely simple alerting mechanisms to content-filtered customized feeds, to more elaborate situated applications that assist users in coordinating distributed work in particular contexts. We describe the characteristics of tailorable microcontexts and briefly sketch one example – a compliance process monitor – that we have implemented.

INTRODUCTION

The web is emerging as a premier end-user collaboration platform, with a growing number of composed applications, a phenomenon encompassing ‘mashups’ and ‘amateur integration.’ Yet creating even simple mashups through the variety of APIs available is still too hard for most non-programmers.

In our work in social computing, we have been interested in how *social proxies* – simple, abstract visual representations of people and their activity – can serve as resources for people to coordinate their behavior, particularly when they are collaborating remotely. Our approach is based on social translucence [2]: the idea that making people and their activities visible through perceptual cues leads to awareness and accountability. Creating such resources creates common ground on which people can more easily coordinate their behavior, particularly in online, distributed settings.

Erickson, Huang, Danis, and Kellogg (2004) described a “task proxy” in which people are displayed as hexagons arranged in terms of their place in an organizational structure (Figure 1). The color of a hexagon represents the status of that person with respect to a task (e.g., whether or not they have completed it). This kind of representation is useful for getting a quick overview of the state of a task across the organization. If the task proxy is made visible to participants, social dynamics emerge (e.g., peer pressure or proactively assisting those who are lagging behind). While the task proxy is a kind of social proxy because it represents people and activity, the task is the focus and person information is backgrounded in the visualization; what is seen at a glance is the state of the task.

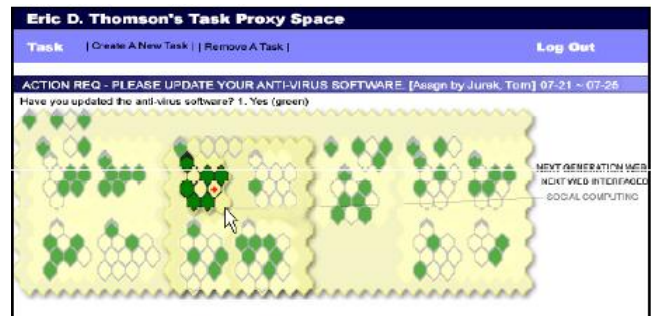


Figure 1. The task proxy. In this representation, people are hexagons laid out by their place in the organization. Hexagons turn green when the task has been completed. The proxy allows state-specific communication (e.g., to all who haven't yet completed the task). Making it visible to all the participants creates social dynamics.

In the same vein, we are interested in exploring whether mashups and the web 2.0 paradigm can be leveraged to allow end users to create their own task or information proxies. Such “microcontexts” would allow digital resources such as feeds or data in other forms to be filtered and rendered in ways meaningful to a user's context. We believe microcontexts can also serve as resources for collaboration, much as we have observed in the social and task proxies in our previous work.

We have identified several different ways in which microcontexts can serve as resources for collaborative activities:

1. Microcontexts representing the state of a process involving people can help people keep track of the process.
2. Microcontexts representing the work product of people (perhaps otherwise uncoordinated), can help people understand how their individual actions are related to a more holistic view.
3. Microcontexts representing (more directly) the activity, behavior or location of people, can provide a basis for reasoning about the people and about social activities.
4. Microcontexts directly supporting collaborative activities such as e-meetings and collaborative decision making.

5. The creation of new microcontexts by adapting microcontexts created by other people is itself a collaborative activity.
6. The entire ecosystem of end-user programming expressed in parts kits, templates, et cetera, is only successful as a collaborative venture when it involves users with different levels of programming and domain expertise.

MICROCONTEXTS

A user tailorable microcontext is a simple, purposeful digital representation whose essential characteristics are that it is:

- **Visual.** Microcontexts concentrate information into a compact, glanceable representation.
- **Shared.** Microcontexts provide a common ground for collaborators, and enable them to both act and communicate with respect to one particular context.
- **Specific.** Microcontexts are circumscribed and concrete, expressed in the language of the domain they address.
- **Symbiotic.** Microcontexts leverage stuff in the world (e.g., feeds, applications, visualization engineers, other microcontexts – i.e., digital resources).
- **Tailorable.** Microcontexts are simple enough that end users can create and tailor them to suit their particular and immediate needs.

There is a clear evolution on the web towards greater end-user *appropriation*: that is, the adaptation and tailoring of information to specific user needs. This has always involved a leading edge of programmers (professional and now amateur), and a widening window of less expert users. Today, for example, any blog or news site can generate a feed, but the feeds are monoliths and feed readers are barely customizable. Microcontexts, however, require that end users are able to discover and filter feeds in a way suitable for their own purposes.

A SIMPLE MICROCONTEXT: COMPLIANCE PROCESS MONITOR

Figure 2 shows a simple example of a microcontext that we have implemented in Squeak [3], and are currently re-implementing in JavaScript. The microcontext allows end users to create custom visualizations of the IBM SOX compliance process. Let's walk through its components at a high level. At the top is a gray header that contains the legend and interface-widgets for configuring the content of the visualization. Below the header, the screen is divided into three columns. The first column (left side of screen) contains a scrollable list of 142 processes ordered alphabetically; each colored shape represents a control for that process. The second column (middle of screen) provides a more detailed view of a single process: it shows the controls for a selected process broken down by country. The third column (right side of screen) provides a view of the details of a single control. Thus, the visualization shows, from left to right, a view of all processes, a view of all controls of a single process broken down by country, and the details of a single control within a process.

Now let's look take a look at the visualization in Figure 2 in a little more detail. First, we'll look at the list of all processes in the leftmost column. The name of each process is followed by a block of shapes, each shape corresponding to one of the controls associated with that process. The shape of a control represents its type: circles signify financial controls; squares signify operational controls. Shading indicates whether the control had a defect the previous quarter, and its color indicates its status. There are currently five colors corresponding to the following statuses:

- dark orange: unremediated defects for the current quarter
- light orange: defects have been remediated in the current quarter
- yellow: no defects current quarter, but under-tested
- green: no defects current quarter, fully tested
- grey: missing data

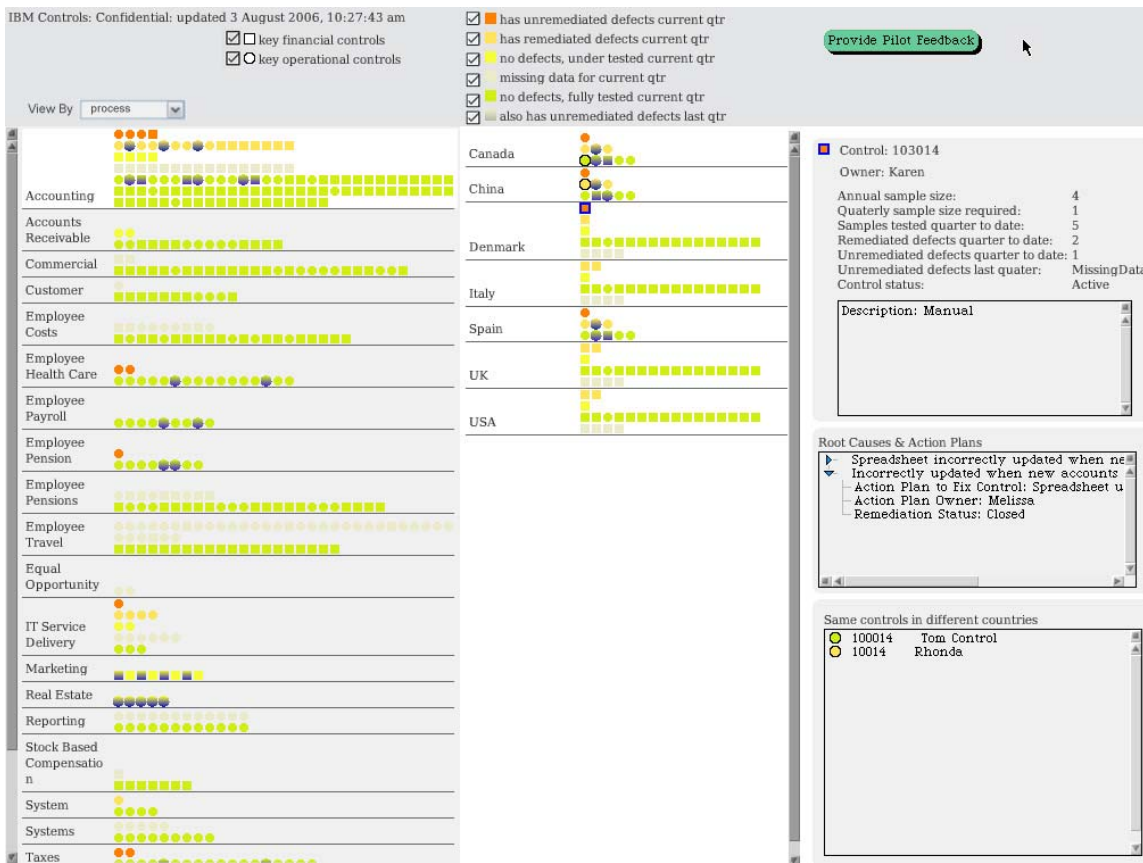


Figure 2. The deployed SOX visualization. (Note: Data shown are fictitious).

An executive viewing the processes in the first column of Figure 2 might wish to explore the topmost process (“Accounting”), noting that it has four orange controls (i.e., controls with defects), and a number of other controls that are under-tested or have no data at all. Selecting this process would provide an expanded view of it in the middle column (as shown), with its controls grouped by country. Thus the executive could quickly see that while the controls run in Italy, UK and USA are in fairly good shape, Canada, China, Denmark and Spain give more reason for concern.

When the executive clicks on a particular control in the middle column, the details for that control appear in the right column. These details include a description of the process, and the root causes and action plans for any defects. The selected control (under “Denmark”), whose details are shown on the right, is highlighted by a wide blue border in the middle column. Some other controls in the column (under “Canada” and “China”) are also highlighted by a narrow black border. These represent the same control, but run in different countries. These controls are also listed at the bottom of the right column.

As stated previously, the gray header contains several interface-widgets that the user can use to configure or filter the visualization. The drop-down menu on the left allows the user to have the controls in the first column grouped by either the business processes, as shown in the figure, or by countries. When the left-column view is ‘by country,’ the middle column becomes a breakdown of the controls for the country by process. The check boxes allow the user to turn on and off the display of controls of a certain status and/or type. So, for example the user might choose not to show controls that are fully tested, and when they uncheck that option, the green squares would not be shown in the visualization.

The overarching rationale for the design was to provide at a ‘glance’ an overview of control status, and provide a common resource for the various people involved in the process. Early in the design process, we did consider creating a simpler, static visualization of the data. However, given the number of processes, controls, and countries, we soon realized that a static visualization could not accommodate all the information. As a result, the visualization needed organizing principles that would present a shared picture, but at the same time enable users to explore those aspects that were of particular relevance to them. The end result is an interactive visualization that shows aggregate status at one point in time with the ability to selectively explore and drill down on particular elements.

A consequence of this approach is that users will have to negotiate a learning curve in order to use the visualization effectively. Without some training and experience users may find it difficult to understand the significance of what is shown, how to navigate the data, or how to best configure the visualization to serve particular needs. Our judgment was that this tradeoff was worth making because of the potential usefulness of providing a single visualization to which all users could refer.

CUSTOMIZING THE MICROCONTEXT

While there are times when individual controllers need to see the ‘larger’ picture, often their responsibilities focus on a sub-set of business processes and on coordination of the controls activities for those processes. Because the visualization is a microcontext, individual controllers can customize it to show just those processes, control types and status that are of particular interest to them. They can also customize the horizontal ordering of controls by control number, or date. We have created a customization panel that surfaces these to the end-user and allows them to quickly and easily create their customized view.

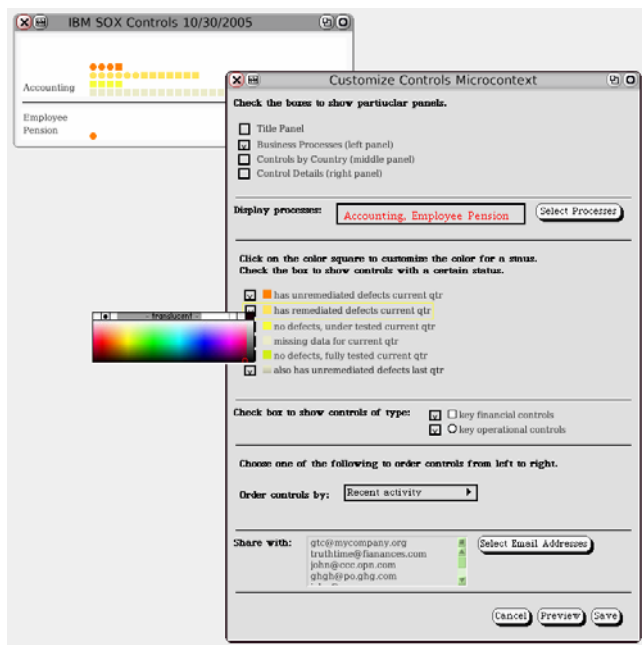


Figure 3. A customization panel for the IBM SOX microcontext. The panel allows the end-user to quickly and easily create a custom view of the ‘larger’ visualization shown in Figure 2.

The scenario is that the user (a controller) wishes to monitor two business processes (‘Accounting’ and ‘Employee Pension’) to ensure that several controls with defects are receiving the attention they need. The user begins from the visualization shown in Figure 2 and chooses ‘Customize’ from the menu. The Customization panel shown in Figure 3 is used to create a custom view of the visualization and share it with other users. The user first checks a box to see only the Business Processes panel; they then select the two Businesses Processes in which they are interested. They further customize the view by choosing not to view controls that are fully tested and have no defects (i.e., the green controls). They then choose to order controls by ‘recent activity.’ As shown in the figure, we also enable them to change the colors associated with the various control statuses. They are almost done when they identify the people with whom they want to share this custom view. They can preview their selections at any time, and when they save their customization, a new custom visualization is created and an email alert containing a link to this custom microcontext is sent to those who have been given access.

CONCLUSION

Putting the ability to create simple visual representations tied to customized feed data in the hands of end users could create a breakthrough in collaboration rivaling that seen in “desktop publishing” that was seen in the 1990s. However, to make such a capability feasible, web data must be more easily manipulated by non-programmers and tied into sharable representations. Just as APIs have emerged in a shareable form to enable programmers to leverage each others’ work as a resource, we hope to create the ability for non-programmers to do the same.

REFERENCES

1. Erickson, T., Huang, W., Danis, C., and Kellogg, W.A. A social proxy for distributed tasks: Design and evaluation of a prototype. *Proc. CHI 2004*, ACM Press (2004), 559-566.
2. Erickson, T. and Kellogg, W.A. Social translucence: An approach to designing systems that support social processes. *ACM Transactions on Computer-Human Interaction*, 7(1), 2000, 59-83. New York: ACM Press.
3. See <http://www.squeak.org/>