WorkItemExplorer: Visualizing Software Development Tasks Using an Interactive Exploration Environment

Christoph Treude, Patrick Gorman, Lars Grammel, Margaret-Anne Storey
Dept. of Computer Science
University of Victoria
Victoria, BC Canada
ctreude@uvic.ca, pgorman@uvic.ca, lars.grammel@gmail.com, mstorey@uvic.ca

***Abstract*—This demo introduces WorkItemExplorer, an interactive environment to visually explore data from software development tasks. WorkItemExplorer enables developers and managers to investigate activity and correlations in their task management system by making data exploration flexible and interactive, and by utilizing multiple coordinated views. Our preliminary evaluation shows that WorkItemExplorer is able to answer questions that developers ask, while also enabling them to gain new insights through the free exploration of data.**

***Keywords*-visualization, coordinated views, task management**

I. Introduction and Motivation

The focus of tool support for software developers has shifted from source code alone towards tools that incorporate the entire software development process. On top of source code editing and compilation, many development platforms, such as IBM's Jazz[1] or Microsoft's Visual Studio[2], now offer explicit support for the management of development tasks. These tasks[3] are important cogs in collaborative software development processes. They need to be carefully aligned with each other, both in what they achieve and in their timing. Since tasks crosscut both the technical aspects of a software project and the social aspects of collaboration and communication, finding the right tasks at the right time is crucial to the success of a project [1].

Task management systems traditionally offered a limited set of properties per task, such as task summary and description, a unique identifier, the creator and owner of the task, as well as priority and severity. Recently, collaboration functionalities inspired by social media, such as comments, feeds, and tags, have been introduced into task management systems for software developers [2]. As a result, tasks have become more complex artifacts with an additional social dimension. For example, individuals manipulating a single task can now be divided into the task creator, the current and past task owners, developers who have commented on the task, developers who have tagged the task, and developers who are subscribed to task updates.

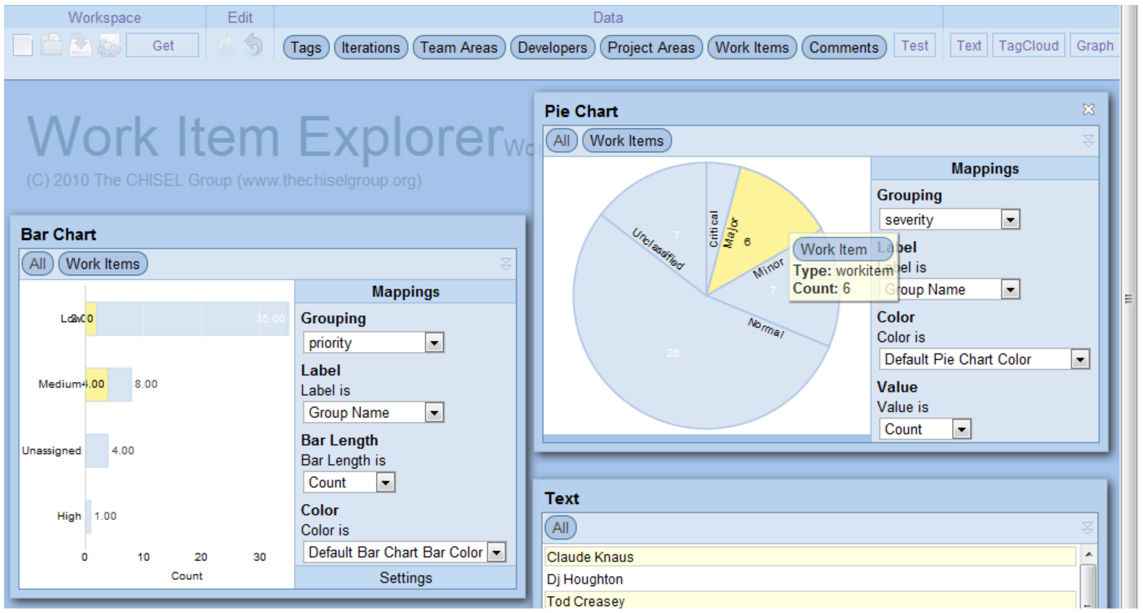
Developers and managers need to maintain an awareness of the abundance of artifacts and the connections between them. This helps them to answer a variety of questions such as: What task should I work on next? Where are the bottlenecks in the process? Current tools that aim at providing an understanding of the state of a task management system, such as developer dashboards, have several shortcomings that we identified in our previous work [3] (e.g., limited interactivity and insufficient visualizations). To better support developers and managers in their understanding of all aspects of their software development tasks, we introduce WorkItemExplorer[4], an interactive visualization environment for the dynamic exploration of data gathered from a task management system (e.g., tasks, iterations, and developers).

WorkItemExplorer leverages multiple coordinated views by allowing users to have multiple different views, such as bar charts or time lines, open at the same time, all displaying the same data in different ways. The coordination comes into play when interacting with the views; highlighting one data element will have a mirrored effect on all other views. This enables the exploration of data relationships as well as the discovery of activities that might otherwise be difficult to see because they span multiple aspects. WorkItemExplorer is a web-based tool built on top of the Choosel framework[5]. We have implemented an adapter for queries against the work item component of IBM's Jazz platform, and we are working on integrating other task management systems.

We conducted a preliminary evaluation of WorkItemExplorer in two ways: we conducted usability studies with four post-doctoral participants to verify the usability of the tool functionality, and we manually verified that WorkItemExplorer is able to answer the questions developers ask as identified by Fritz and Murphy [4]. Our preliminary evaluation yields three main findings:

- WorkItemExplorer can answer questions that developers ask about task management systems,
- WorkItemExplorer enables the acquisition of new insights through the free exploration of data, and

[1]http://jazz.net/
[2]http://www.microsoft.com/visualstudio/en-us
[3]Tasks, work items, and bugs are considered synonyms in this paper.
[4]See http://tinyurl.com/workitemexplorer for video.
[5]http://thechiselgroup.org/choosel



Bar Chart

Grouping: priority
Label is: Group Name
Bar Length is: Count
Color is: Default Bar Chart Bar Color

Low 35.00
Medium 4.00 8.00
Unassigned 4.00
High 1.00

Pie Chart

Grouping: severity
Label is: Group Name
Color is: Default Pie Chart Color
Value is: Count

Work Item
Type: workitem
Count: 6

Text

Claude Knaus
Dj Houghton
Tod Creasey

Figure 1. Screenshot of WorkItemExplorer. The user is exploring the correlation between severity and priority of work items using a bar chart that shows work items grouped by priority, and a pie chart grouped by severity. Since the user is hovering over the pie wedge for major severity, all corresponding items are highlighted in all views. The text view shows the owners of the corresponding work items.

- WorkItemExplorer offers a flexible environment in which different individuals solve the same task in different ways.

II. Background and Related Work

The use of task management systems by software developers has been studied by several researchers. Many of the related studies focus on mining and analyzing quantitative data to reveal information about the evolution of the system [5] or to predict future behaviours [6].

Ellis *et al.* [7] report results from interviews of how developers use Bugzilla, a popular task management system. The motivation for their study was the design of a visualization tool for tasks. One of their key findings was that Bugzilla played a key role in managing the project. Their visualization reveals social and historical patterns in tasks, but it does not focus on exploration activities. FASTDash [8] includes task data in a representation of a shared code base using source code activity as the principal information item. In our previous work, we found that dashboards that report on the state of a task management system can become pivotal to task prioritization in critical project phases and stir competition between developers and teams [3].

Compared to the large number of studies that analyze data available from task management systems, there is little work on user interface design for task management systems. In a study on bug tracking systems, Just *et al.* [9] revealed several hurdles in reporting and resolving bugs. They present several recommendations for task management systems, including contextual assistance, reminders to add information, and assistance to collect and report crucial information to developers. To our knowledge, there is no work on exploration tools for task management systems.

III. WorkItemExplorer

WorkItemExplorer is an interactive tool that allows users to dynamically explore data from a software development task management system. As shown in Figure 1, users can have multiple views open at the same time, all displaying the same data in different ways. In addition, highlighting a data element in one view will have a mirrored effect in all other views.

WorkItemExplorer currently supports seven data elements and the relationships between them: work items, developers, iterations, project areas, team areas, tags, and comments. Using a drag-and-drop interface, these data elements can be moved into seven different views:

- A **text** view with different grouping options (e.g., to see a list of work items grouped by their owner).
- A **tag cloud** for the exploration of work item tags.
- A **graph** for the exploration of relationships between different kinds of artifacts, such as work items and iterations. Expanding a relationship on any node in the graph will add the corresponding nodes to the view.
- A **bar chart** to visualize data with different groupings using bars of different lengths (e.g., to visualize developers by the number of work items they own).
- A **pie chart** to visualize data with different groupings using pie wedges of different sizes (e.g., to show work items by priority).

- A **heat bars** view to visualize work items over time, with an additional grouping option (e.g., to visualize the creation of different work item types, such as defects and enhancements, over time). Heat bars are based on our earlier ConcernLines tool [10].
- A **time line** to analyse data over time. Different time properties, such as creation or modification date, can be chosen (e.g., to visualize team area creation over time).

IV. Example Scenarios

In this section, we describe two scenarios that highlight the functionality of WorkItemExplorer.

A. *Who is working on important work items?*

To identify process bottlenecks or to balance workload, it is crucial to know who works on important tasks. The task "importance" can be defined in many different ways, and an exploratory tool, such as WorkItemExplorer, can be used to understand the implications of different approaches. To explore important work items and their owners, we open up a bar chart and a pie chart in WorkItemExplorer, and then drag all of the work items onto both of them. We then group the bar chart by priority, and the pie chart by severity, and we drag the bars and pie wedges that we are interested in into a third view, such as a text view. For example, we could drag the bars and pie wedges corresponding to *high* priority and *major* as well as *critical* severity. If we now group the text view by work item owner, we get a list of all people working on important work items (see Figure 1), and we can continue to explore their workload in more detail.

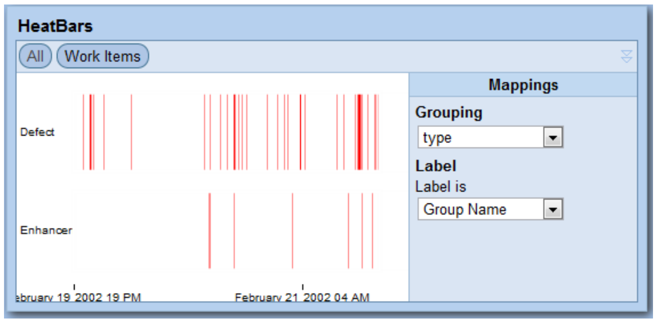
In addition, this configuration allows us to immediately explore the relationship between severity and priority of work items in our data set. As shown in Figure 1, when mousing over *major* severity in the pie chart, through partial highlighting, we can see how work items with major severity are distributed across the different priorities in the bar chart.

B. *What work items are created when?*

Understanding when different kinds of work items, such as defects and enhancements, are being created can be very useful to ensure that a team is correctly following a particular process. To discover this kind of information, WorkItemExplorer features a heat bars view that visualizes work items grouped by one property, such as type or priority, over time. Figure 2 shows the result for a grouping by work item type, and we can see that enhancements were added near the end of the data set, whereas defects were added constantly throughout with a couple of more intense periods. To further explore this phenomenon, we could now drag either of the bars into another view for further analysis.

V. Preliminary Evaluation

We conducted usability studies and we verified that the tool is able to answer a broad range of questions.



Mappings

Grouping: type

Label

Label is: Group Name

Figure 2. Screenshot of the heat bars view showing a heat map of work item creation over time, with work items grouped by their type.

A. *Usability studies*

We recruited four post-doctoral researchers to evaluate the usability of WorkItemExplorer. A data set containing the first 48 work items along with all related artifacts from the Eclipse Bugzilla bug trackerwas used for the study. After a two-minute introduction briefly describing the tool's interface, the participants were given a list of five tasks that are based on the questions that developers ask identified by Fritz and Murphy [4]:

1) Try to find out which work items your team members are working on.[6]
2) Who is the busiest developer at the moment? If we were to reassign some of their work items, who should we give them to?
3) You need to find a new work item to work on. Try to find work items that are of high priority and severity to complete next.
4) You are currently interested in work items with the tags *investigate* and *needinfo*. Try to find the comments on those work items.
5) Please explore the data on your own and let us know of anything interesting that you find.

Table I shows the results of the usability evaluation.[7] The first four tasks were completed (on average) in about two minutes each, and only two participants were unable to solve one of the tasks (denoted in bold). Participants used different views to solve the same tasks. This is an indicator that there are many different ways to gain insights using WorkItemExplorer, allowing for a broad range of insights as well as serendipity. For example, when switching from Task 1 to Task 2, Participant C realized that they had already produced the answer on the screen as part of Task 1.

Feedback was generally positive: *"very usable"* (Participant A), *"the best part is that I can click, select stuff and move it and see what it looks like in another view"* (Participant C), and *"very cool interface"* (Participant D). All

[6]We gave two specific names from our data set as "team members".

[7]We slightly adjusted the wording of Tasks 3 and 4 after Participant A.

	task 1		task 2		task 3		task 4		task 5	
	t	*views*	*t*	*views*	*t*	*views*	*t*	*views*	*t*	*views*
A	1:35	Graph, Text	2:35	2 Bar, Text	3:45	Bar, Text	**6:50**	**7 Text**	3:55	HeatBars, Text, Timeline
B	1:35	Text, Graph	1:50	Graph, Tag, Text	2:10	Bar, 2 Text	1:05	Tag, Graph	0:40	2 Text, Graph
C	1:05	Bar, Text	0:20		1:05	Bar	2:40	Bar, 2 Tag, Text, Graph	7:35	5 Bar, Text, 2 Tag, Graph
D	4:50	Text, 2 Graph	3:15	2 Bar, Pie	2:20	2 Bar, Graph, Text	**4:55**	**3 Text, Bar, 3 Graph**	1:50	Graph

Table I
Summary of the results of our usability study. For each of the four participants, the table shows the time they took to complete each task as well as which views they used during their exploration. Items in bold denote tasks that were not completed successfully.

participants used the opportunity for free data exploration as part of Task 5.

B. *Questions developers ask*

To ensure that WorkItemExplorer can answer questions developers ask in their daily work, we manually inspected the 78 questions that developers ask, but for which support is lacking, as identified by Fritz and Murphy [4]. 17 of these questions are applicable to WorkItemExplorer; the remaining 59 require data beyond what is currently included in WorkItemExplorer, such as builds or source code. However, we did confirm that WorkItemExplorer is able to answer all 17 questions that are focused on task data.

C. *Limitations*

For any specific question, there might be other tools that can provide an answer more quickly than WorkItemExplorer. However, the strength of an exploration tool lies in the free exploration of data when no goal is specified, and in the power to provide serendipitous insights. So far, we have not conducted usability studies with professional developers, but when we showed an earlier version of the tool to an audience of about 30 practitioners using live data from their task management system, the tool created active interest. The audience was eager to see more kinds of artifacts, such as builds, included in the tool, and they particularly liked the increased traceability as well as the opportunity to explore data by *"surfing along different link types"*.

The number of items that WorkItemExplorer can visualize depends on the particular view. Views that group elements (such as the bar chart) can show more elements than a graph, for example. We avoid having to visualize all data in a given system by allowing developers to query their system, and then only visualizing the data contained in the query results.

VI. Conclusions and Future Work

With WorkItemExplorer, we have introduced an interactive environment that enables developers and managers to visually explore data from software development task management systems by leveraging multiple coordinated views. Our preliminary evaluation has shown that WorkItemExplorer is able to answer questions that developers ask, and we have confirmed the usability of the tool.

In our future work, we will add more artifacts to WorkItemExplorer, and we will evaluate the tool in an industry setting to gain further insights into how developers and managers explore data from their task management systems.

References

[1] R. E. Kraut and L. A. Streeter, "Coordination in software development," *Commun. ACM*, vol. 38, no. 3, pp. 69–81, 1995.

[2] M.-A. Storey, C. Treude, A. van Deursen, and L.-T. Cheng, "The impact of social media on software engineering practices and tools," in *Proc. of the workshop on Future of software engineering research*. New York: ACM, 2010, pp. 359–364.

[3] C. Treude and M.-A. Storey, "Awareness 2.0: staying aware of projects, developers and tasks using dashboards and feeds," in *Proc. of the 32nd Intl. Conf. on Software Engineering - Vol. 1*. New York: ACM, 2010, pp. 365–374.

[4] T. Fritz and G. C. Murphy, "Using information fragments to answer the questions developers ask," in *Proc. of the 32nd Intl. Conf. on Software Engineering - Vol. 1*. New York: ACM, 2010, pp. 175–184.

[5] H. Kagdi, J. I. Maletic, and B. Sharif, "Mining software repositories for traceability links," in *ICPC '07: Proc. of the 15th Intl. Conf. on Program Comprehension*. Washington, DC: IEEE, 2007, pp. 145–154.

[6] J. Anvik, L. Hiew, and G. C. Murphy, "Who should fix this bug?" in *ICSE '06: Proc. of the 28th Intl. Conf. on Software Engineering*. New York: ACM, 2006, pp. 361–370.

[7] J. B. Ellis, S. Wahid, C. Danis, and W. A. Kellogg, "Task and social visualization in software development: evaluation of a prototype," in *CHI '07: Proc. of the Conf. on Human factors in computing systems*. New York: ACM, 2007, pp. 577–586.

[8] J. T. Biehl, M. Czerwinski, G. Smith, and G. G. Robertson, "Fastdash: a visual dashboard for fostering awareness in software teams," in *Proc. of the Conf. on Human factors in computing systems*. New York: ACM, 2007, pp. 1313–1322.

[9] S. Just, R. Premraj, and T. Zimmermann, "Towards the next generation of bug tracking systems," in *VLHCC '08: Proc. of the Symp. on Visual Languages and Human-Centric Computing*. Washington, DC: IEEE, 2008, pp. 82–85.

[10] C. Treude and M.-A. Storey, "Concernlines: A timeline view of co-occurring concerns," in *Proc. of the 31st Intl. Conf. on Software Engineering*. Washington, DC: IEEE, 2009, pp. 575–578.