IBM Research Report

Shared Waypoints and Social Tagging to Support Collaboration in Software Development

Margaret-Anne Storey*, Li-Te Cheng, Ian Bull*, Peter Rigby*
IBM Research Division
One Rogers Street
Cambridge, MA 02142

*University of Victoria
British Columbia, Canada

IBM

Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

systems (e.g. CVS) and bug tracking systems (e.g. Bugzilla) are used by developers in their everyday work, and allow users to leave and share comments. Version control systems allow developers to attach comments and specify tags when submitting updated files to a shared repository. Bug tracking systems allow developers to append comments to a bug report. Various studies report how distributed teams of developers leverage this for group awareness and coordination, e.g. Gutwin, Penner, and Schneider [1].

- **Pure Groupware Tools**: Traditional standalone asynchronous groupware such as email, newsgroups, and wikis are used by developers to discuss issues and share snippets of code. Studies like [1] document this usage of groupware tools in addition to formal development tools within software projects.
- **Inline Comments in Source Code**: Collaboration can occur amongst developers by writing comments directly into the source code. Programming languages often support formatting conventions in comments to generate online documentation from source code, such as Java's "Javadoc" set of keywords [8]. Also various programming environments recognize special keywords such as "TODO" which are then highlighted in the editor and aggregated in secondary views. Ying, Wright, and Abrams describe these informal commenting conventions in a project and how this can be leveraged to infer knowledge and coordinate development activities [7].

Formal development tools and groupware tools, such as the ones described above, often focus on higher level concerns around the individual pieces of source code, such as maintaining a consistent set of files in the version control system, enumerating problems in a bug tracking database, and discussing and coordinating work on a mailing list. For the individual developer, these tools help support the central task of creating and maintaining source code.

We are interested in supporting collaborative annotation, navigation, and coordination activities through inline comments. User-created annotations written as comments embedded in the code result in very explicit landmarks for readers to support navigation and coordination. Embedded comments are also portable across different formal development and groupware tools, since the user can simply share the source code as text. In contrast, formal development tools and pure groupware tools, while effective for various collaborative tasks, require developers to copy-paste snippets or make external references to the code being discussed. Jazz, which blends aspects from formal development tools and pure groupware tools into the programming environment, allows users to annotate source code with discussion [2], but it does not support explicit landmarks or the notion of waypoints in the source code for coordination.

The general facility of creating an inline comment is a simple matter of adding some text to the code, but as it is an informal capability, it has some drawbacks. In particular, the unstructured mechanism for adding comments can result in a myriad of conventions and *ad hoc* meta data being used [7]. Moreover, these annotations can become unwieldy and outdated over time. This lack of structure to inline comments also means that more complex information, such as "sequence" or structural information cannot be captured. We begin to address these issues by presenting a lightweight tool that enhances annotations embedded in source code to enable navigation, coordination, and capture of knowledge relevant to the development team.

2. DESIGN AND IMPLEMENTATION

In order to build lightweight tool support for capturing and using collaborative annotations, we have combined two existing concepts. The first, *waypoints*, comes from the discipline of wayfinding in physical spaces. The second, *social tagging*, comes from the world of social bookmarking software.

2.1 Waypoints and Routes

Waypoints are used by geographical positioning systems to save locations of interest [4] that may include checkpoints on a route or a significant ground feature to be avoided. They can be specified by manually entering latitude and longitude coordinates or they can be saved as the user passes close by a point of interest. Waypoints can be referenced according to distance and bearing to a previously saved waypoint, and they are designed to be shared across users and applications. Landmarks are features (e.g. tall buildings) that serve as reference points to guide navigation [10]. In comparison with waypoints, landmarks are typically not the goal of the navigation task. Waypoints are often gathered within *routes* (see Fig.1). A route provides a path from one point to another together with intermediate destinations (a sequence of waypoints).

Fig. 1: A route of GPS waypoints

2.2 Social Tagging

Social tagging, also known as social bookmarking, enables users to create shared bookmarks to online resources with additional metadata beyond the site location. Social tagging websites such as flickr.com and del.icio.us are used to "tag" images and share bookmarks respectively by a large user community. A tag is a one-word term to describe the image or bookmark. The user is not restricted by any preconceived vocabulary, taxonomy or ontology. This bottom-up approach results in semi-structured information spaces that are often referred to as "social classifications" [3]. Tagging is not a new concept to software engineering. Tags have been used for decades for annotating check-in and branching events in software version control systems. This use of tags is for identifying version control transactions rather than for tagging within the source code and documentation.

2.3 The tagSEA Tool

We have combined the concepts of waypoints and social tagging to create "tagSEA" (Tags for Software Engineering Activities), a tool to support collaborative annotations in software development. It has been implemented as a plug-in for the Eclipse Java development environment (www.eclipse.org).

In software, the waypoint analogy corresponds to locations of code model elements (e.g. class, method, package, file), or a location that corresponds to a file name and line number for any type of file. Waypoints are indexed through a set of tags supplied by the programmers. Metadata is also captured or explicitly entered with the waypoint and may include the version of the software file, creation date, author, related bugs etc. Routes are sequences of code or file locations.

Fig. 2 shows a view of tagSEA. tagSEA allows users to associate tags with parts of the source code by using a Javadoc-style keyword.. A tag is created by writing "**@tag**" in a comment block, followed by the actual tag and some descriptive text (see Fig. 2 A). Individual tags are delimited by spaces, or they may consist of multiple words by placing the string in quotes. This Javadoc-style syntax allows for easier adoption of tagSEA with the Java developer community, who is already familiar with similar conventions such as "@author" and "@version".

tagSEA extends Eclipse's enhanced code editor to ease usage of waypoints and routes in source code even further. "@tag" comments are highlighted in the source so that they stand out as clear landmarks or points of interest to the developer. Simply typing "@" inside a comment block will list "@tag" alongside the list of other Javadoc-style annotations. The resulting waypoints (i.e. locations in the code) are automatically associated with the closest enclosing Java model element (e.g. a class). We also support annotation at a coarser level of granularity than lines of code. Waypoints can be scoped to entire files via a context menu in the file navigator. Such waypoints and associated tags are not explicitly added to the code, and hence can be seen as private as they are only in the creator's workspace. The reward for encouraging the use of tagSEA's annotations in code is to provide annotated waypoints for coordination and navigation. This is supported by the **Waypoints Viewer** (see Fig. 2 B). Selecting one or more tags listed on the left reveals the union of the software model elements, their locations, and annotations that have been waypointed in the right side of the viewer. Clicking on the waypoint entries on the right side of the viewer opens the associated file editor, positions the editor at the appropriate location, and highlights the waypointed software model element. Thus, programmers can use the Waypoints Viewer to easily navigate to places of interest.

Tag spaces are often criticized for producing flat structures [3]. However, there are reports of users using their own conventions to encode hierarchical relationships across tags. We believe that programmers may be more comfortable adopting a hierarchical syntax given their experience with formal languages and abstractions. The user can specify the **hierarchical tags** as follows: "@tag bug(performance)". This indicates that there is a "bug" tag, with bug subtypes specified by the parameter in

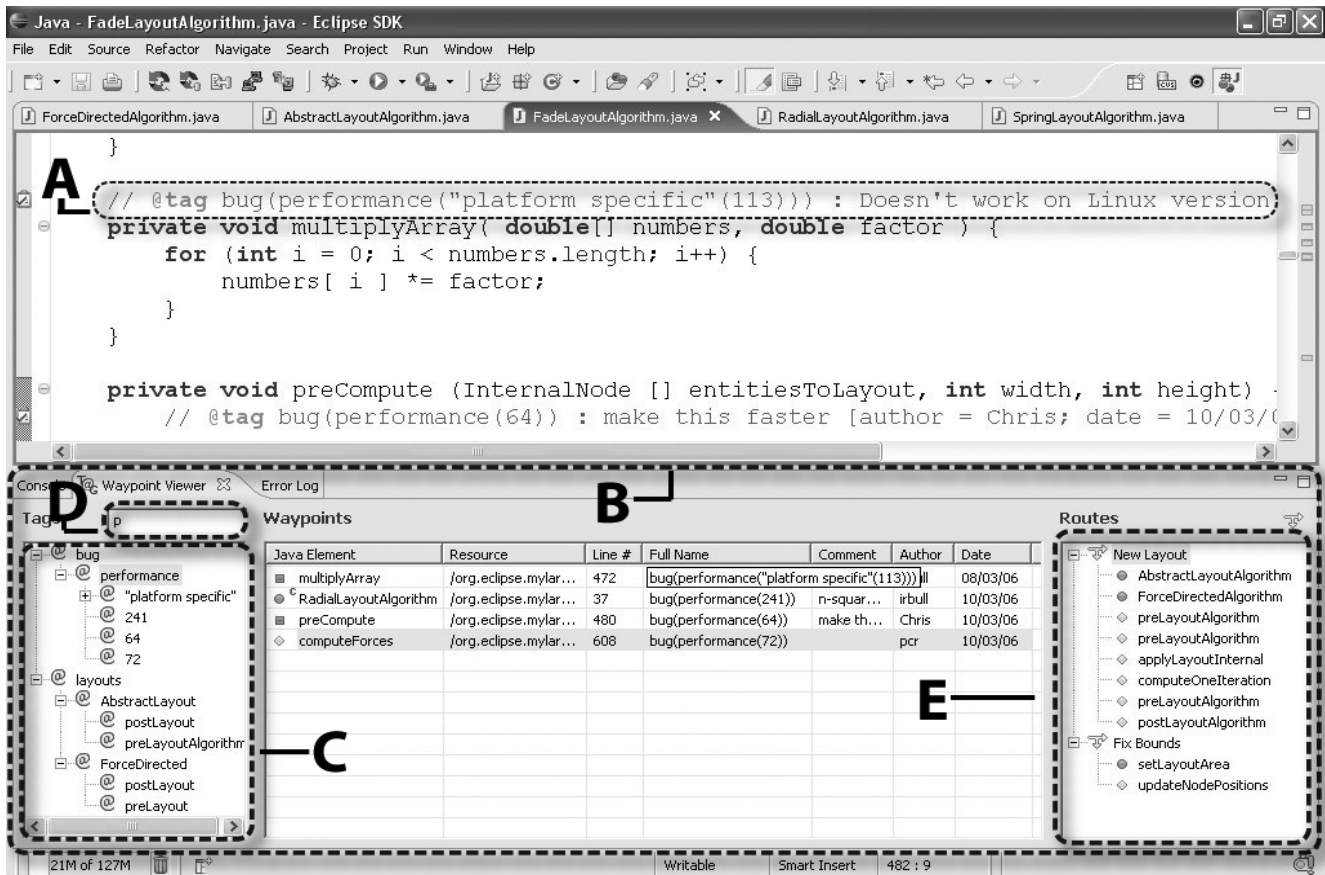brackets. The hierarchy of tags is displayed using a tree at the left of the Waypoint Viewer (Fig. 2 C).

Figure 2: tagSEA Plug-in for Eclipse

One of the challenges in social tagging is using a consistent set of tags over time [3]. tagSEA provides an **automatic tag completion** feature to suggest existing tags based on a partially typed tag. We have also added support for **refactoring** of tags so that they can be easily renamed, reorganized or deleted. For example, renaming a tag in the Tag viewer will result in all instances of that tag in the source code (i.e. the waypoints with that tag) to be updated accordingly. Hierarchical tags can be reorganized by dragging them to other parts of the tag tree. Automatic tag completion and refactoring helps achieve a consensus over tag naming and structure over time.

Managing a growing sea of tags is also a concern for social tagging systems [3]. This may be a greater issue for large software development projects. To address this concern, tagSEA provides some initial support for dynamic **filtering** and **searching** of waypoints. Every keystroke in the filtering text box immediately updates the list of tags that partially match the entered query, allowing a user to condense and explore tag spaces through partial text entry. The user can also **sort** waypoints according to metadata captured such as author or date.

The waypoint metaphor strongly supports the notion of a sequence or **route**. Personalized guided tours have also been suggested for website navigation [6]. Such a mechanism can be used to document a series of steps in a software development workflow, for example, a software inspection code review, or as a step-by-step guide for newcomers to a software library or framework. Since the waypoints and tags are shared, the user need not follow a single path prescribed by one authority. Instead, the user can build a personalized route by combining paths from different experts based on the tag metadata. In tagSEA, the user can create a named Route in an optional panel (see Fig. 2 E). Waypoints may be added through drag and drop on a selected route. Routes are stored externally to the code in the user's metadata, but they can be shared through an export facility and may be checked into CVS. Routes may themselves also be tagged and may be filtered using the tag pane.

3. PRELIMINARY EVALUATION

We have already begun to evaluate tagSEA with a small group of programmers at two sites. Preliminary feedback is encouraging and patterns of usage are already starting to emerge. One developer, Bob[1], reported tagging as he was integrating two unfamiliar systems. The lightweight mechanism allowed him to temporarily tag areas of the code he was changing as he was experimenting with the unfamiliar code. He and another developer, Alice, also reported using tagging to indicate areas that she updated to fix bugs that were in another project "belonging" to Bob. The tagging feature was a useful mechanism to document

---

[1] Names used are fictitious.

3

these changes for future navigation and to support communication between Bob and Alice. The alternative solution would have been to submit the code to a version control system, run a difference tool to list the results, and then navigate to each difference, one at a time by searching through the workspace.

Bob also reported using a set of tags created by Alice as he had to do a similar task. The "route" feature was not available when this user used tagSEA but nevertheless the waypoints as they were entered fulfilled this role, although they lacked the sequence information. Bob described tags as being useful for documenting "common maintenance patterns". Finally, tags were also seen as useful for navigating an unfamiliar code base. Tags were applied on unfamiliar pieces of code for later inspection while the programmers' comprehension of the system evolved.

4. FUTURE WORK

One of the exciting aspects of this research is that it has opened the door to many possible extensions for further investigation. We propose some interesting research questions here.

4.1 Leverage a social bookmarking service

In GPS applications, waypoints are not tied to any one application and are shared across users and applications. Similarly we assume the same benefits could be realized for waypoints in software spaces when the semantics of the referenced locations are likewise shared across applications. Moreover, by using tagging to specify waypoints, we can also investigate how social tagging is used to share and exploit tagging vocabularies and taxonomies, while also increasing awareness of development activities beyond the confines of the IDE. Also, we can begin to infer social networks amongst developers and create filters based on groups using authorship information associated with tags, waypoints and routes. To facilitate this research, we plan to integrate tagSEA with dogear, an enterprise social bookmarking service [6].

4.2 Semi-automate tagging

Creating and maintaining a usable set of tags relies on collective effort. Internet-scale tagging systems such as del.icio.us can draw upon the general Internet population to contribute tagged information. For tagSEA, the pool of contributors can vary from a large open source project to a small in-house development team. This can be especially onerous for small teams maintaining large pieces of legacy software. We are considering semi-automated tagging techniques to ease the burden on smaller groups, such as searching for a keyword in comments and adding a tag based on that search, and automated creation and deletion of tags based on other IDE activities, such as closing or opening of bugs, breakpoint insertion and deletion etc.

4.3 Visualize waypoints and tags

The tagSEA prototype provides a list and tree based interface to manage tag and waypoint information. We are planning to experiment with visualizations such as the "tag clouds" popularized by social tagging sites such as flickr and del.icio.us to provide alternative user interfaces. We also intend to explore how waypoints and routes could be visualized within dependency and architectural views of a software system.

4.4 Evaluate with more users

We plan to provide a broader deployment of an instrumented version of tagSEA to study how tags are created, used and shared over time. Although tagSEA lacks explicit collaborative features, we have been able to determine from our initial users that it is being used to support collaboration through conventions and simple file exchange over a central version control repository. Consequently, it is interesting to observe how collaborative conventions will emerge and how these conventions will be negotiated despite the tool's limited features. Understanding these usage patterns will inform the design of future collaborative tools for software development.

5. CONCLUSION

This paper presents the tagSEA tool for creating shared waypoints and routes through a software space. In addition to the locomotional support they provide, waypoints by way of their tags also provide a lightweight mechanism to share documentation that captures important knowledge about the code while also facilitating the coordination of collaborative activities. Muller *et al.* also discuss how shared landmarks (but not waypoints) can become coordination artifacts [7]. Although our evaluation is preliminary, the feedback we have thus far received indicates that the implicitly captured meta-data combined with the lightweight nature of tagging results in a very promising technique for supporting distributed software development.

6. REFERENCES

[1] Gutwin, C., Penner, R., and Schneider, K. Group awareness in distributed software development. *Proc. CSCW 2004,* ACM Press, New York, NY, 2004, 72-81.

[2] Hupfer, S., L.-T. Cheng, S. Ross and J. Ross, "Introducing collaboration into an application development environment", *Proc. CSCW 2004*, ACM Press, New York, NY, 2004, 21-24.

[3] Hammond, T., T. Hannay, B. Lund, and J. Scott, "Social Bookmarking Tools: A General Review", D-Lib Magazine, Volume 11 Number 4, April 2005.

[4] Larkin, F.J., Basic Coastal Navigation: An Introduction to Piloting, 1999. ISBN 1-57409-052-6

[5] Millen, D., J. Feinberg, and B. Kerr, "Social Bookmarking in the Enterprise", ACM Queue, vol 3, no. 9, November 2005.

[6] Moody, P., WebPath: Sharable Personalized Guided Web Tours, IBM Research (Cambridge), Technical Report 98-09 (1998).

[7] Muller, M.J. et al., "Shared landmarks in complex coordination environments", in CHI '05 extended abstracts on Human factors in computing systems (Portland, Oregon), pp. 1681—1684, 2005.

[8] Sun Microsystems, Javadoc Tool Home Page, http://java.sun.com/j2se/javadoc

[9] Ying, A., Wright, J., and Abrams, S. "Source code that talks: an exploration of Eclipse task comments and their implication to repository mining", In the Intl. Workshop on Mining Software Repositories (MSR '05), St. Louis, 2005, pp. 1-5.

[10] Vinson, N., "Design Guidelines for Landmarks to Support Navigation in Virtual Environments", Proceedings of CHI '99, Pittsburgh, 1999.