Ensemble: a Recommendation Tool for Promoting Communication in Software Teams

Pei Feng Xiang[■] , Annie T.T. Ying[◆] , Ping Cheng[■], Ya Bin Dang[■], Kate Ehrlich[◆],
Mary E. Helander[◆], Paul M. Matchen[◆], Andrew Sempere[◆],
Peri L. Tarr[◆], Clay Williams[◆], Shun Xiang Yang[■],

[■]IBM China Research Center
{pfx,chping,dangyb,yangsx}@cn.ibm.com

[◆]IBM Watson Research Center
{aying,katee,helandm,matchen,asemper,tarr,clayw}@us.ibm.com

ABSTRACT
Successful software development requires effective coordination among developers. In this paper, we propose Ensemble, an approach and a set of tools that aim to help developers better coordinate their work. Built on IBM Rational Team Concert, Ensemble helps developers select the right people to collaborate with, the right times to collaborate with them, and to stay coordinated with them over time.

1. INTRODUCTION
Effective coordination among software developers is crucial in successful software development, a highly collaborative activity. Developers who work on related tasks or related parts of a software system must coordinate with one another to reduce the possibility of costly inconsistencies, incompatibilities, errors, and delays. Previous studies have shown that a significant proportion of errors in software projects are caused by lack of communication between developers [4]. In a distributed project, maintaining the right level of coordination is even more difficult, as developers have limited opportunity for spontaneous communication and reduced visibility into current work [11]. Developers do not always know who is the right coordination partner and end up either coordinating with the same people and missing those who they really need to coordinate with, or coordinating with too many people which wastes time.

Researchers have identified three approaches to improve coordination: collaboration features, expertise recommendation systems, and awareness tools. However, each of these approaches have some limitations. Collaboration features, such as extended instance messaging capabilities [12] are supported by a few IDEs which provide developers with ready access to communication and awareness. However, these features do not fully support the dynamic changes in coordination partners. Expertise recommendation systems (e.g., Expertise Browser [8], Expertise Recommender [6], bug triage tool [1], and EEL [7]) recommend developers based on heuristics mined from software repositories, such as the source code change repositories and bug tracking systems. However, these systems require that users recognize the need to find a coordination partner. Awareness tools (e.g., Palantir [8], FASHDash [2], [5]) help developers stay connected by highlighting the relevant activities of other team members. However, these tools are not constantly available and so may not provide an on-going solution.

In this paper, we introduce Ensemble, an approach that is designed to help developers select the *right people* to collaborate with, the *right times* to collaborate with them, and ways to *stay coordinated* with them over time. Ensemble is a set of lightweight tools built on IBM Rational Team Concert, an Eclipse-based client, part of the IBM Jazz [1] collaborative development platform.

2. ENSEMBLE
Ensemble analyzes a developer's current work to recommend other people who are working on related artifacts – source code, work items [2] and change sets [3] – for communication. As a developer moves between work items or source code, Ensemble automatically updates recommendations to show those related to the current artifact. Developers can use Ensemble to maintain a small Watch List of other people. Ensemble notifies the developer when one of the people on the Watch List has updated a particular artifact.

Ensemble currently consists of two tools: Recommender, which helps developers select the *right people* to collaborate with; and Watch List, which helps developers realize the *right times* to collaborate and to *stay coordinated* over time. In the rest of this section, we describe these two tools.

2.1 Recommender
Ensemble Recommender relies on previous work on gap analysis [3] to identify people working on many related artifacts, with whom the user is not currently communicating, as a coordination partner. This approach is similar to EEL with one important exception. Where EEL recommends people based on congruence, we recommend people based on gaps.

Based on the previous work on gap analysis [10], we generate the recommendations as follows. We use two heuristics to suggest that two developers should communicate: arc mirroring and node ties. The arc mirroring heuristic is based on the intuition that if developer P_1 works on artifact A_1, developer P_2 works on artifact A_2, and A_1 depends on A_2, then P_1 should communicate with P_2. The node tie heuristic is based on the intuition if two developers work on a same artifact, they should communicate with each other. We compute a gap between a two developers P_i and P_j as follows. If m_{ij} denote the number of arc mirroring instances, t_{ij} the number of node tie instances, and c_{ij} the number of communication activities between P_i and P_j, the gap g_{ij} between the two developers is given by $g_{ij} = (m_{ij} - c_{ij}) + (t_{ij} - c_{ij})$. We produce a ranked list of communication partners by selecting and sorting the developers with greatest gap. We are currently

[1] http://jazz.net
[2] Work items are Bugzilla-like tasks in Jazz.
[3] Change sets are semantically coherent set of changes to files in Jazz.

applying this algorithm on artifacts in Jazz, more specifically, work items and source code.

We have implemented the Recommender interface as a view in Eclipse, as shown in Figure 1. The Ensemble Recommender viewFigure 1. The left half of the Figure shows a ranked list of three people with greatest communication gaps to the current developer. The right half of the Figure shows context of why a recommendation is generated, after the user expands the triangle under the recommended person's name, as indicated by the arrow (in red). The context includes artifacts modified by both the recommended person and the user, such as the first artifact `Money.java` highlighted by the box (in green) (the node ties heuristics). The context also includes artifacts modified by the recommended person that are related to the artifacts modified by the current user (the arc mirroring heuristics). This context on the related artifacts helps a developer decide whether and for what artifacts to communicate with the recommended person. To stay coordinated with a recommended person, the user can check the "watchlist" box, highlighted by a circle (in blue) in Figure 1, to add the recommended person to the Watch List, which we describe in Section 2.2.

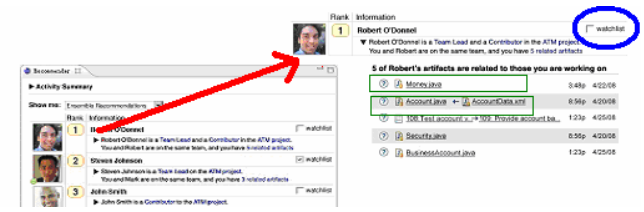

Figure 1. The Ensemble Recommender view

2.2 Watch List

The second tool in Ensemble is intended to help people know when to communicate with someone and stay coordinated by implementing local recommendations and a Watch List-like feature.

The local recommendations provide awareness into the activities of people working on related artifacts. To help developers realize the right times to collaborate, Ensemble automatically updates recommendations to show those related to the current artifact, as a developer moves between work items or source code. To help developer stay coordinated over time without being intrusive, the Watch List is intended to sit at the side of developer's screen. The view consists of photo icons of two kinds of people: people *recommended* by Ensemble based on the context and people *declared to be of interest* by the user. Each photo icon not only provides a metaphor to the associated person, but also equips with some useful features, such as a link to the instance messaging capabilities and a hover box containing contact information and the context of the recommendation (the left half of Figure 2).

Ensemble helps users stay coordinated by allowing them to “pin” people they are interested in. Ensemble notifies the user when one of these pinned people has changed a selected artifact, by generating a notification (as shown in the bottom right of Figure 2) and displaying an indicator on the photo icons (top right of Figure 2). Some activities that Ensemble make explicit on the Watch List are events such as the change in a work item status and change-set delivery.

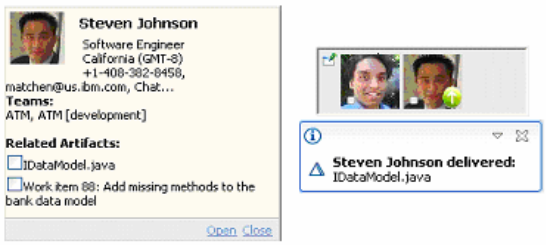

Figure 2. Watch List features

3. CONCLUSION

In this paper, we have presented a brief introduction to Ensemble, which provides a new approach to recommendations that identifies who to coordinate with, when to coordinate, and how to stay coordinated. Next steps on this project include testing these tools with distributed development teams over an extended period of time.

4. REFERENCES

[1] Anvik, J., Hiew, L., and Murphy, G. C. Who should fix this bug? In Proc. of ICSE, 2006, 361-370.

[2] Biehl, J.T., Czerwinski, M., Smith, G., Robertson, G.G., and Bailey, B. FASTDash: A Visual Dashboard for Fostering Awareness in Software Teams. In Proc. of CHI, 2007, 1313-1322.

[3] Cataldo, M., Wangstrom, P. A., Herbsleb, J. D., and Carley, K. M. Identification of Coordination Requirements: Implications for the Design of Collaboration and Awareness Tools. In Proc. of the CSCW, 2006, 353-362.

[4] Curtis, B., Krasner, H., and Iscoe, N. A Field Study of the Software Design Process for Large Systems. CACM, 31, 11 (1988), 1268-1287.

[5] Holmes, R. and Walker, R. Promoting Developer-Specific Awareness. In Proc. of the international workshop on Cooperative and Human Aspects of Software Engineering, 2008.

[6] Mcdonald, D. W. and Ackerman, M. S. Expertise recommender: a flexible recommendation system and architecture. In Proc. of CSCW, 2000, 231-240.

[7] Minto, S. and Murphy, G. C. Recommending Emergent Teams. In Proc. of the International Workshop on Mining Software Repositories, 2007.

[8] Mockus, A. and Herbsleb, J. D. Expertise browser: a quantitative approach to identifying expertise. In Proc. of ICSE, 2002, 503-512.

[9] Sarma, A., Noroozi, Z., and Hoek, A.v.d. Palantír: Raising Awareness among Configuration Management Workspaces. In Proc. of ICSE, 2003, 444-454.

[10] Valetto, G., Helander, M., Ehrlich, K., Chulani, S., Wegman, M., and Williams, C. Using Software Repositories to Investigate Socio-technical Congruence in Development Projects. In Proc. of the Int'l Workshop on Mining Software Repositories, 2007.

[11] Herbsleb, J. D., Mockus, A., et al. Distance, Dependencies, and Delay in a Global Collaboration. In Proc. of ICSE, 2000, 319-328.

[12] Cheng, L.-T., De Souza, C. R. B., et al. Building Collaboration into IDEs. In ACM Queue, vol. 1, 2003, 40-50.