Regular Paper

Human Software Interaction in Software Development Community

Xin Yang, Kar-Long Chan, Papon Yongpisanpop, Hideaki Hata, Hajimu Iida and Kenichi Matsumoto[†1]

1. Introduction

Human Computer Interaction (HCI) is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them[1)]. HCI is a big research fields, and has long histories focusing on the interaction between people and computers. Although HCI targets interactions, we believe that what we need to study is the different interactions, that is, interaction with software data and interaction between people in software ecosystem. We also believe that such software-related interactions should be needed to study and its not limited research scope. Therefore, we call this software-related interaction research field as **Human Software Interaction**, and try to develop techniques and frameworks of studies. Intuitively, HSI begins with the combination with MSR/SA and HCI, and tries to design the interaction between human and software data, and the interaction between human and human in software development. We also hope to generalize the usage of HSI into both industries and Open Source Softwares. In the remainder of the paper, we will present the challenges we are expecting in designing such interaction, and also our visions of future software development, in which we will introduce a conceptual framework called guild.

2. Current Social Issues

Difficulties in knowledge sharing and volatilities of communications: Collaboration is one of the main feature in software since people are now connected with the internet. But the question is how are we going to enhance the human-software interaction and human-human interaction in collaboration. He et al.[2)] studied about interaction while doing collaboration between human to human and human to computer in CAD software system. They found out that adding collaborative capabilities to single-user CAD application is helpful but also difficult. Another problem is when people are working together, only technologies cannot be thought of as just another independent tool. People need to have a strong comprehension about collaboration. Lack of collaboration comprehension can also lead to these problems such as unclear roles, information (over)sharing and not enough doing. In HSI, we are not only research on just technology integration but we also study on how well people are integrated into the practices of those who work together.

Low stickiness: Since human power is an essential resource, the population of contributors in a software development community is one of the health indicators of OSS project. Onoue et al. studied the communities' population by proposing *software population pyramids*, graphical illustrations of the distribution of various experience groups in a software development community[3)]. From the empirical study of the status of the population distributions and their transitions with OSS projects in GitHub, it revealed that contributors leave easily even if in attractive OSS projects.

Social Network in Software Engineering: We studies the social aspect of software engineering in order to know how developers work together and communicate with others. There are some related works have been done using techniques from social perspective to solve software engineering problems. Bird et al. extracted potential structure for latent sub-communities and studied the relationship among them in OSS projects[4)]. Wolf et al. use social networks to search the development network structure and they predict the potential failure[5)] Work by Damien[6)] looked at the social networks of global teams communication and collaboration.

3. Our vision of the future

In the future research, by adopting gamifica-

†1 Nara Institute of Science and Technology (NAIST)

tion, we hope to construct a framework named as Guild to effectively connect people with one another for collaboration in software development. The original term **Guild** comes from medieval, which means an association of artisans or merchants who control the practice of their craft. As for the modern acknowledgement, the term of guild is more famous as a social mechanism functioning in MMORPG (Massively Multiplayer Online Role-Playing Game), which is a major type of online game. A MMORPG connects game players all over the world and each of them is required to create his/her avatar in a virtual world. Usually each avatar poses with different skills, representing different roles in the virtual world. Based on avatar's skills, player can accomplish varied missions individually or by cooperating with other players. Guild in MMORPG is a social approach to group different players into one community, aiming to ease the heterogeneity between players, causing less confrontation, and building up better cooperation to achieve certain goal. In fact, the mechanism of guild brings a lot of profits for players including avatar levelling, or efficiently challenging some of the most difficult missions. Unlike Software Craftsmanship, a movement of emphasizing the importance of coding skill in software projects, the guild is based on promoting the collaboration among contributors and encourage contributions.

In the next two parts, we will introduce how we want to implement the mechanism of guild into software development.

- **Mission** Software development activities such as Coding, Reviewing, Testing, Building can be treated as different type of missions and they will be ranked based on the difficulties. The difficulty of the mission is judged by high-ranking developers in the guild. Based on the difficulty of the mission, high-rank member can assign it to certain low rank member based on his skills. By completing certain mission, the member can be promoted to higher level, which grants him/her the chance to work on more challenging mission. Status of different missions can be easily tracked with a designated interface, which should be built upon versioning system such as Git or SVN. Such tracking interface should be very user-friendly, by showing the most essential information.
- **Guild** In order to form a guild in software development, usually the project leader and other senior engineers should be assigned as high-rank member in the guild. When a certain programmer/developer wants to join the guild for developing the software, he/she is required to fill in his skills and experiences. In addition, high-ranking members could also assign some tasks as entrance exam to test his/her ability. His/Her performance in the test and profile information will treated as input to quantify his/her status point in software development. The status point should be an important index for high-ranking members to base on in order to assign missions. Furthermore, in the environment of guild, beginner has better chance to improve his/her skill in software development since it is easy for he/she to team up with high-rank members to accomplish certain tasks, while at the same time gaining valuable experience.

References

1) "ACM SIGCHI curricula for human-computer interaction," New York, NY, USA, Tech. Rep., 1992.
2) F.He and S.Han, "A method and tool for humanhuman interaction and instant collaboration in cscw-based {CAD}," *Computers in Industry*, vol.57, no. 89, pp. 740 – 751, 2006.
3) S.Onoue, H.Hata, and K.Matsumoto, "Software population pyramids: The current and the future of oss development communities," in *Proc.of 8th ACM-IEEE Int. Symp. on Empirical Softw. Eng. and Measurement*, ser. ESEM '14. New York, NY, USA: ACM, 2014, pp. 34:1–34:4.
4) C.Bird, D.Pattison, R.D'Souza, V.Filkov, and P.Devanbu, "Latent social structure in open source projects," in *Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering*. ACM, 2008, pp. 24–35.
5) T. Wolf, A. Schroter, D. Damian, and T. Nguyen, "Predicting build failures using social network analysis on developer communication," in *Proceedings of the 31st International Conference on Software Engineering*. IEEE Computer Society, 2009, pp. 1–11.
6) D.E. Damian and D.Zowghi, "An insight into the interplay between culture, conflict and distance in globally distributed requirements negotiations," in *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*. IEEE, 2003, pp. 10–pp.