Reshaping Distributed Agile and Adaptive Development Environment

Francesco Nocera
Polytechnic University of Bari
Bari, Italy
francesco.nocera@poliba.it

ABSTRACT

Towards the interest of (Collaborative Networked) Organizations in the adoption of emerging technologies to support communication, collaboration and monitoring needs of their Distributed Agile and Adaptive Development Environment (DADE), a tool based on the emerging Liquid Multi-Device Software paradigm is presented.

CCS CONCEPTS

• **Software and its engineering → Collaboration in software development**; *Agile software development*;

KEYWORDS

Agile methodologies, tool, Liquid Software, Collaborative development, Ontology, BPM

ACM Reference Format:
Francesco Nocera. 2018. Reshaping Distributed Agile and Adaptive Development Environment. In *Proceedings of the 26th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '18), November 4–9, 2018, Lake Buena Vista, FL, USA.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3236024.3275435

1 INTRODUCTION

Software has become an essential part of every aspect of the society and of our daily life, as well as the Software Development Process (SDP) phases are becoming among practitioners. A Systems Development Methodology (SDM), defined as a documented collection of policies, processes, and procedures, is commonly used by software development teams to improve the SDP in terms of increased productivity of information technology (IT) personnel and higher quality of the final IT solutions [5]. SDMs are continually evolving to keep up with changing technologies and satisfy new demands from users. The last two decades have witnessed the rise of new SDMs called "Agile" [3, 7] methodologies. Different empirical studies have examined the impact of agile practices, ceremonies and tools on development team performance and clients' satisfaction. Daily meetings valued by agile methodologies seem to enhance communication between team members [8, 20, 23]. They help to control their project towards its goals. Iterative development adds agility to the SDP by providing continuous feedback on the incremental product deliverable. Tools designed to attenuate the above stated challenges should therefore include special features. These include, for example, supporting the interaction of distributed teams by applying communication and collaboration technology, supporting the development of real-world projects, that is, big scale agile projects developed in many sites and process monitoring. The most used and recommended Agile tools are the following: JIRA [1], Monday.com [2], Trello[3], Clarizen[4], etc. For a complete list and features comparison (Project planning, execution, Deployment, Resource Management, etc.), please refer to works [15, 16, 21].
This work addresses the challenges presented before by developing a Distributed Agile and Adaptive Development Environment (DADE) [11] tool based on the emerging Liquid Software [24] paradigm in order to improve agile practice, in particular sequential or simultaneous collaboration on different devices. Another key concept of the proposed solution is the ontology-driven support in order to facilitate the monitoring of the project progress.

2 APPROACH AND UNIQUENESS

This work incorporates the following topics in order to improve existing DADE tools: the liquid software paradigm [10, 14] and the semantic Process mining [1, 4, 13, 19, 25, 26]. Liquid Software is a paradigm in which applications and data can flow from one device or screen to another seamlessly, allowing the users to roam freely from one device to another, no longer worrying about device management, not having their favorite applications or data, or having to remember complicated steps [14, 17, 18]. The essential feature of Liquid Software is to support for hassle-free multi-device experiences, falling into the following categories [12]: *Sequential Screening*, *Simultaneous Screening* and *Collaboration scenario*. Please refer to Liquid Software Manifesto [24] for a detailed description of each category. According to Liquid software architecture, in this first prototype release the tool allows to: *(i)* share code files by placing them in a logical *room*; *(ii)* edit the code simultaneously and sync the changes in real-time; *(iii)* users can interact with each other by seeing themselves in video conference exploiting webRTC technology; *(iv)* compile and run users' project server-side; *(v)* software development processes monitoring.
Figure 1 shows the overall architecture related to the collaborative development module. According to the Liquid approach we design a client-server architecture with multiple masters topology [9].
With respect to the existing tools and related features introduced in the previous section, the proposed Software architecture allows

ESEC/FSE '18, November 4–9, 2018, Lake Buena Vista, FL, USA

ACM ISBN 978-1-4503-5573-5/18/11.
https://doi.org/10.1145/3236024.3275435

[1] https://www.atlassian.com/software/jira
[2] https://monday.com
[3] https://trello.com
[4] https://www.clarizen.com/

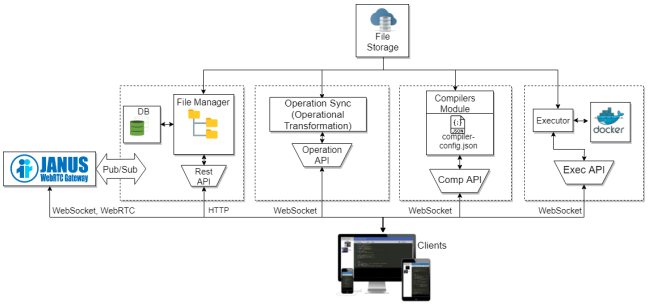

Figure 1: Overview of the proposed tool.

the possibility to run the system from any heterogeneous devices. **Server-side**, the tool is implemented in Node.js[5] since the event-driven nature of javascript allows proper synchronization management between clients. It is composed of five main components. Each component is involved in a particular task, and it can be decentralized since it is independent from the others to guarantee reliability and maintainability and improve performance and scalability requirements. These modules share a repository where users' files are stored. *Janus* [2] is a Selective Forwarding Unit (SFU) server for WebRTC video conference system. The *File manager component* is involved in file management on shared repository. It interacts directly with Janus by its Publish/Subscribe mechanism to organize the users' files in the same rooms.

The component uses a MongoDB database to maintain the property of files (name, programming language, room) handled by the system. The communication with the client is done by HTTP REST API with which client can perform CRUD operations on resources represented by files, such as create or upload a file, retrieval a file, update a file (by changing their property) and finally delete a file.

Real-time operation is an important aspect to be considered in the design of collaborative systems [6]. Users should be able to see the effects of their own actions immediately and those of other users as soon as possible. The *Synchronization component* is in charge of handling editing operations on the file content. The editing operations made by users are sent via WebSocket to Operation API. Each editing operation needs to take into account the conflict that could arise when two or more users are working on the same file. To resolve editing conflict the OT [22] is an appropriate approach for maintaining consistency of the copies of the shared document in real-time collaborative editing systems.

The remote compilation of users' file is an optional requirement of the system, and this task is made by the *compiler component*. The client sends via WebSocket to Comp API some compiling settings to perform compiling task. The compilers module use the compiler-config.json file to select the most appropriate compiler in according to programing language specified from the compiling settings sent by the client.

Based on the idea of adopting different heterogeneous clients, once a file is compiled it can be run on the fly directly on the server in order to execute it in a more performing environment then users' devices. The *Executor component* fulfills this task by receiving from the user the input of the program by Exec API and then returns the output. For security purpose, the executable files are run in isolated spaces. This space is based on a container pattern implemented by Docker[6] containers.

To make the system accessible from any heterogeneous devices a thin **Client** in HTML5, CSS and Javascript using Polymer[7] is developed.

For the processes monitoring purpose the GOJS library[8] version 1.7 is included for the graphical rendering of the software development processes models. The tool is able to discover business processes at runtime from the event log related to the collaborative development module. A suitable level of abstraction is chosen in a preloaded context-aware business ontology. Inspired by [4], the proposed component implements and extends the CNMining [13] PROM plugin for automated discovery of processes (with respect to the alpha algorithm used in [4]), and builds an UML Activity Diagram (AD) abstracting the process resources and the role resources. The assignment of the resources and corresponding roles to the activity are modeled by abstraction on a Domain and Business Ontologies. The Abstraction level is given as input to the tool before generating the output (UML AD). Let us indicate with AL the level of abstraction given as input to the tool with $AL \in \{1, 2, 3\}$. Three levels of abstraction are provided. Suppose that the tool receives as input $AL = 1$, this means that there is total abstraction on activities — resources are not considered with this level — just CNmining algorithm [13] is performed; the tool manages just activities, otherwise, with a medium level of abstraction $AL = 2$, the tool distinguishes activities based on resources by uploading the Domain ontology; even more, with a lower level $AL = 3$, the tool distinguishes activities based on resources and the role of the resources in an Organization, by uploading also a Business ontology context dependent (Business ontology required for a lower level of abstraction $AL = 3$).

3 RESULTS AND CONTRIBUTIONS

This work presents a promising new Collaborative tool for agile software development based on the emerging Liquid Software paradigm that allows people to share, edit and compile documents and knowledge on the Web, in a simple, economical and efficient way on different devices. The proposed tool integrates a process conformance service thank to the implemented ontology-driven support for process monitoring and analysis. We performed a usability study with Ph.D. developers to assess how our collaborative tool is received in practice. We carried out six user evaluation sessions, each having three participants. The evaluation of the prototype showed that the proposed environment is relevant and has improved agile development practices. All in all, the perception of participants towards our approach was very positive, and we consider it as a successful step towards evaluating the prototype using a more complex application in a real-world development scenario.

Future work will include the collaborative modeling and shared editing on the generated code and the reverse process, according to future trends highlighted in the work by Franzago M. et al. [8].

[5] https://nodejs.org/en/

[6] https://www.docker.com/

[7] https://www.polymer-project.org/

[8] https://gojs.net/latest/intro/deployment.html

REFERENCES

[1] Rakesh Agrawal, Dimitrios Gunopulos, and Frank Leymann. 1998. Mining process models from workflow logs. In *International Conference on Extending Database Technology*. Springer, 467–483.

[2] Alessandro Amirante, Tobia Castaldi, Lorenzo Miniero, and Simon Pietro Romano. 2015. Performance analysis of the Janus WebRTC gateway. In *Proceedings of the 1st Workshop on All-Web Real-Time Systems*. ACM, 4.

[3] Kent Beck, Mike Beedle, Arie Van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, et al. 2001. Manifesto for agile software development. (2001).

[4] Stefano Bistarelli, Tommaso Di Noia, Marina Mongiello, and Francesco Nocera. 2017. PrOnto: an Ontology Driven Business Process Mining Tool. *Procedia Computer Science* 112 (2017), 306–315.

[5] Frank KY Chan and James YL Thong. 2009. Acceptance of agile methodologies: A critical review and conceptual framework. *Decision support systems* 46, 4 (2009), 803–814.

[6] Clarence A Ellis and Simon J Gibbs. 1989. Concurrency control in groupware systems. In *Acm Sigmod Record*, Vol. 18. ACM, 399–407.

[7] Martin Fowler and Jim Highsmith. 2001. The agile manifesto. *Software Development* 9, 8 (2001), 28–35.

[8] Mirco Franzago, Davide Di Ruscio, Ivano Malavolta, and Henry Muccini. 2017. Collaborative Model-Driven Software Engineering: a Classification Framework and a Research Map. *IEEE Transactions on Software Engineering* PP, 99 (2017), 1–1. https://doi.org/10.1109/TSE.2017.2755039

[9] Andrea Gallidabino, Cesare Pautasso, V Ilvonen, T Mikkonen, K Systä, JP Voutilainen, and A Taivalsaari. 2017. Architecting liquid software. *Journal of Web Engineering* 16, 5&6 (2017), 433–470.

[10] Andrea Gallidabino, Cesare Pautasso, Ville Ilvonen, Tommi Mikkonen, Kari Systä, Jari-Pekka Voutilainen, and Antero Taivalsaari. 2016. On the architecture of liquid software: technology alternatives and design space. In *Proc. of WICSA*.

[11] Asif Qumer Gill. 2018. Distributed agile development: Applying a coverage analysis approach to the evaluation of a communication technology assessment tool. (2018), 1633–1655.

[12] Google. 2012. The New Multi-screen World: Understanding Cross-platform Consumer Behavior. (2012). http://services.google.com/fh/files/misc/multiscreenworld_final.pdf.

[13] Gianluigi Greco, Antonella Guzzo, Francesco Lupia, and Luigi Pontieri. 2015. Process discovery under precedence constraints. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 9, 4 (2015), 32.

[14] John Hartman, Udi Manber, Larry Peterson, and Todd Proebsting. 1996. *Liquid Software: A new paradigm for networked systems.* Technical Report 96-11. University of Arizona.

[15] Capterra Inc. 2018. Top 10 Agile software. (2018). https://www.capterra.com/sem-compare/...

[16] MultiMedia LLC. [n. d.]. The 12th State of Agile survey, year = 2018, url = https://explore.versionone.com/state-of-agile/versionone-12th-annual-state-of-agile-report, urldate = 2018-06-26. ([n. d.]).

[17] Niko Mäkitalo, Francesco Nocera, Marina Mongiello, and Stefano Bistarelli. 2018. Architecting the Web of Things for the fog computing era. *IET Software* (2018).

[18] F. Nocera. 2018. Student Research Abstract: A liquid software-driven semantic complex event processing-based platform for health monitoring. *Proceedings of the ACM Symposium on Applied Computing* Part F137816, 1464–1465. https://doi.org/10.1145/3167132.3167454 cited By 0.

[19] Vladimir Rubin, Christian W Günther, Wil MP Van Der Aalst, Ekkart Kindler, Boudewijn F Van Dongen, and Wilhelm Schäfer. 2007. Process mining framework for software processes. In *International conference on software process*. Springer, 169–181.

[20] Helen Sharp and Hugh Robinson. 2008. Collaboration and co-ordination in mature eXtreme programming teams. *International Journal of Human-Computer Studies* 66, 7 (2008), 506–518.

[21] Hans Spanjers, Maarten ter Huurne, Bas Graaf, Marco Lormans, Dan Bendas, and Rini Van Solingen. 2006. Tool support for distributed software engineering. In *Global Software Engineering, 2006. ICGSE'06. International Conference on*. IEEE, 187–198.

[22] David Sun, Steven Xia, Chengzheng Sun, and David Chen. 2004. Operational transformation for collaborative word processing. In *Proceedings of the 2004 ACM conference on Computer supported cooperative work*. ACM, 437–446.

[23] Harald Svensson and Martin Höst. 2005. Views from an organization on how agile development affects its collaboration with a software development team. In *International Conference on Product Focused Software Process Improvement*. Springer, 487–501.

[24] Antero Taivalsaari, Tommi Mikkonen, and Kari Systä. 2014. Liquid Software Manifesto: The Era of Multiple Device Ownership and Its Implications for Software Architecture.. In *38th IEEE Computer Software and Applications Conference (COMPSAC)*. 338–343.

[25] Wil Van der Aalst, Ton Weijters, and Laura Maruster. 2004. Workflow mining: Discovering process models from event logs. *IEEE Transactions on Knowledge and Data Engineering* 16, 9 (2004), 1128–1142.

[26] Wil MP van der Aalst. 2016. *Process mining: data science in action*. Springer.