Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 97 (2016) 73 – 83

Procedia Computer Science

CLOUD FORWARD: From Distributed to Complete Computing, CF2016, 18-20 October 2016, Madrid, Spain

User Involvement in Software Development Processes

Iosif Alvertis[a], Sotiris Koussouris[a,*], Dimitris Papaspyros[a], Evangelos Arvanitakis[a], Spiros Mouzakitis[a], Sebastian Franken[b], Sabine Kolvenbach[b], Wolgang Prinz[b]

[a]*Decision Support Systems Laboratory, National Technical University of Athens, 9 Iroon Polytechneiou Str, 15780, Greece*
[b]*Fraunhofer-Institut für Angewandte Informationstechnik FIT, Schloss Birlinghoven, 53754 Sankt Augustin, Germany*

Abstract

Costs of software development and deployment are decreasing due to numerous open source projects and novel Cloud-based services (IaaS, PaaS, SaaS), but competition increases due to lowering entry barriers. The need to bring developers closer to their customer becomes vital for success, especially involving users into the very early stages of software development. This allows detecting flaws of conceptual and design nature, minimize unnecessary development costs, and warrant relevance for customers. This paper presents the CloudTeams methodology and platform that aim to bridge this gap, based on an existing groupware system supporting the notion of collaborative software development.

Peer-review under responsibility of organizing committee of the international conference on cloud forward: From Distributed to Complete Computing

Keywords: Cloud Platform; Collaboration; Software Development; User Engagement; Ideation; Crowdsourcing; Cloud Computing

1. Introduction

Software Development requires a large number of technical resources as well as stakeholders. Those are software developers on the technical side, and potential users on the market side. The success factor of software heavily depends on whether the software solution is able to fulfill the expectations of the addressed users. Software design experience teaches us that it gets more and more expensive to remove conceptual flaws the later the process of software development gets. Some authors postulate an exponential connection between the number of different

* Corresponding author. Tel.: +30-210-7723514; fax: +30-210-7723550.
E-mail address: skous@epu.ntua.gr

Peer-review under responsibility of organizing committee of the international conference on cloud forward: From Distributed to Complete Computing
doi:10.1016/j.procs.2016.08.282

phases of software development and the costs for removing unresolved errors in these phases[14]. It is therefore crucial to as early as possible align software with the expectations of stakeholders in the software development process to avoid unnecessary costs and to speed up the development process.

The paper at hand showcases how the social collaborative development platform CloudTeams helps towards this direction, as an approach to involve end users in every phase of the software development process. The aim is to align software development with market expectations throughout the software development cycle. CloudTeams provides the methodology and the necessary tools, as interface points of the methodology, that developers demand for project management and team collaboration, software development, testing, deployment, and of course for interaction with perspective end users. The platform itself consists of several technical components to offer the required functionality to all stakeholders without re-implementing existing approaches. An existing groupware system forms the basis for the collaboration platform for software development.

In this paper, the CloudTeams methodology and the technical platform are presented, with details on its customisation to facilitate software developers and promote end user participation. In the next chapter, we present a landscape analysis of related work for all relevant aspects of CloudTeams, while the third chapter introduces the CloudTeams methodology. The fourth chapter describes the architecture and the current state of the CloudTeams technical collaboration platform, while the fifth chapter provides an overview on the concept's assessment methodology based on diverse pilot installations. Finally, the paper concludes with an outlook.

2. Landscape Analysis

Bringing closer potential users and software developers during software development projects requires a methodology and a technical implementation that will dictate how such interactions can take place, in order to maximize the benefits of such communications, minimizing the risks of information overload and out-of-scope feedback generation. This is undisputedly of major importance for any kind of software, however interaction with large user bases and the ability to integrate the extracted knowledge is mostly concerning small and agile working software teams that deliver consumer software products (such as mobile Apps) as development and success evaluation are executed in very short cycles, in order to continuously improve the product and satisfy as many customers as possible. Our analysis of related work identifies a range of systems that can contribute to the overall process of software engineering with user participation. Such systems include social platforms, groupware systems, software development models, local development environments, code repositories and code analysis, customer participation, and social requirements engineering. We could derive deep insights into the set of required features by conducting a semi-structured series of interviews with stakeholders of the platform[2].

When it comes to social platforms, in professional business contexts these are systems that help enterprises strengthen their social interaction and informal communication for both internal and external use[5]. Popular social media platforms such as Yammer (www.yammer.com), Facebook (www.facebook.com), or Twitter (www.twitter.com) are the ones that paved the way and outline the basic features that enterprises should use. In opposition to this, social media platforms often offer just rudimentary groupware functions, e.g. they do not provide advanced group and access management functions or sophisticated document management functions.

Classic groupware systems dominate the intra- and inter-company cooperation, while in some application domains the use of groupware is not yet common[1]. Those are often complex tools with a broad palette of functions for project and task management, and document management. In implementations such as Microsoft SharePoint (http://office.microsoft.com/de-de/sharepoint/), one might notice the weak integration of lightweight social networking and communication functionalities. The software classes of groupware and social software both offer possibilities that support professional exchange as well as cooperation between locally distributed cooperation partners. Each class of systems, groupware and social software, on its own cannot offer exhaustive support for the full range of requirements of cooperative processes. Although groupware research has yielded a number of productive and successful systems, it appears that social media is somewhat ruling out the traditional cooperation systems such as cooperative document management or team room systems. Even if recent implementations aspire to

blend groupware systems and social media[3,9], there is no support for software development and end user involvement.

When it comes to the technical support of code development processes, local integrated development environments (IDE's) are as important as their cloud-based pendants. Supporting code development with local means involves systems such as Eclipse, Komodo, NetBeans, Visual Studio, among many others. But modern software engineering in geographically distributed teams cannot be imagined without collaboration among the individual developers. As most modern software development processes are iterative and incremental, following the agile development methods and seeking constantly for evolving user requirements, constant changes or unforeseeable issues might be raised and have to be addressed by the development team. From the very early stages of the development process towards software deployment and maintenance the developers need to communicate within the team to share the knowledge and experiences, discuss potential solutions, agree on changes, and stay informed on the on-going process. The communication tends to be rather asynchronous due to unavailability (over the supported communication channels) of distributed team members or due to a significant difference in the time zones between the working environments locations. Although the synchronous communication tools are in place, the coordination of audio or visual calls can be time consuming. The distributed nature of such teams impacts the product management work at the most, as the product vision communication, product design techniques and related activities impact the time-to-market indicator if practiced asynchronously (thus, iteratively).

Modernisation of production is pushing towards work distribution, and the same applies for software development. In this context, the above mentioned challenges are tackled by distributed teams through the utilization of novel tools and platform such GitHub (www.github.com), which combines source code management and distributed version control features. Parallel approaches are BitBucket (https://bitbucket.org/) or Kiln (www.fogcreek.com/kiln/). Some of these platforms also allow team management or issue tracking, as well as code analysis. Other platforms exclusively care about code analysis, such as SonarQube (www.sonarqube.org/) or Squale (www.squale.org/). This multitude of systems exclusively for software development makes it hard to properly interact with end users in software development processes. Furthermore, the software development process can be aligned with several software development models, reaching from classical waterfall models towards more cyclic and issue-driven SCRUM process models[12].

At the same time with the above mentioned approaches for bringing development team members closer towards increasing productivity and software quality, similar approaches are surfacing regarding customer integration. Social media are being used as main communication vessels for bringing customers closer to intra-project developments. Crowdsourcing approaches help gathering work results from interested customers. Platforms like Amazon Mechanical Turk (www.mturk.com/mturk/welcome) offer the possibility to outsource tasks to users for micropayments. Projects from various areas make use of this possibility, may it be for image classification, bird voice recognition, or others. However, these approaches only work by extrinsic motivation aspects, as people work for the money and not so much because they are interested in the project itself. On platforms like Kickstarter (www.kickstarter.com/) or Indiegogo (www.indiegogo.com), users can explore projects they are explicitly interested in, and back them with financial resources. In these cases, customers pay in advance for having the chance to get one of the future products. The customer involvement in the development process stays moderate. A further approach is social requirements engineering. This approach brings together the groups of end-users and developers for negotiating requirements in mutual interest. Requirements Bazaar is a web-based solution for social requirements engineering[10,11]. However, also this approach builds on a known base of users that has to be acquired in advance.

From a business perspective, crowdsourcing has been characterised an "Open Source CRM"[†], using "wisdom of crowds to collect customers' feedback, analyse their needs and identify market gaps that need to be fulfilled. Social media have been lately another indirect crowdsourcing platform; social media are used to collect specific

[†] Crowdsourcing is an open source CRM. http://zurmo.org/features/open-source-crm-crowdsourcing

information as gender, age, geography or favourites activities of users in order to understand customers' need and better target them. Such information can be extracted using the API provided by each service, like *Facebook Graph API* or *Foursquare API*. Users may authorise an application to retrieve information from their profiles, while storing and processing data are addressed in the developers' terms of use for the API. Some of the most popular services that are based on the analysis of social media accounts and the storage of social activities to profile, categorise and evaluate users are *Klout* and *AddThis*; *Klout* is connected with various accounts, like Facebook, Twitter, Google+ etc. to track users' activity and social feedback, while *AddThis* offers a widget to share content from various web pages to social media, and collects the activities to profile its users and target advertising. While social activities can be measured through the usage of specific social media accounts, there are means to catalogue and track users' habits even outside the web, using specific sensors or applications in devices that may document daily habits of their users, implicitly or explicitly. *FitBit*, *Nike+*, *Nike Fuelband*, *Fitocracy*, *Runkeeper* or *Expensify* are such solutions that can track users' behaviour and allow them to insert specific characteristics of their lifestyle habits. Habits patterns are extremely important factors for the success of a product, while it is difficult for the users themselves to identify such patterns. Crowdsourcing platforms are not sufficient; results generated by them cannot be generalised as they provide only incentives and need quantitative methods to be confirmed. But existing crowdsourcing platforms do not try to establish solid relationships with the community; they act more like content aggregators.

The feedback process and generally the social collaboration in software life cycle should be explored in more depth. In that direction, the IEEE 12207 standard for Systems & Software Life Cycle Standards[‡] has issued the relevant categories of Software Life Cycle; Three main categories are in the interest of social collaboration in software engineering: (a) systems context (i.e. stakeholders requirements definition, acceptance support), (b) software development (i.e. software requirements analysis, software quality testing), and (c) software support (i.e. software quality assurance, software validation, software problem resolution). These categories may be summarized in two phases, where social interactions with external interested parties (e.g. customers or clients) may be established: (i) during the software development or maintenance to improve the analysis of software requirements, as well as (ii) after the software is developed to ensure quality. No matter the methodology every software team may choose, such social interactions exist both in rigid methods (i.e. Waterfall-Model, V-Model, Incremental Build Model, Rational Unified Process, Spiral-Model) and in agile ones (i.e. eXtreme Programming, Scrum, Feature Driven Development); what changes is the phase and the frequency such interactions take place.

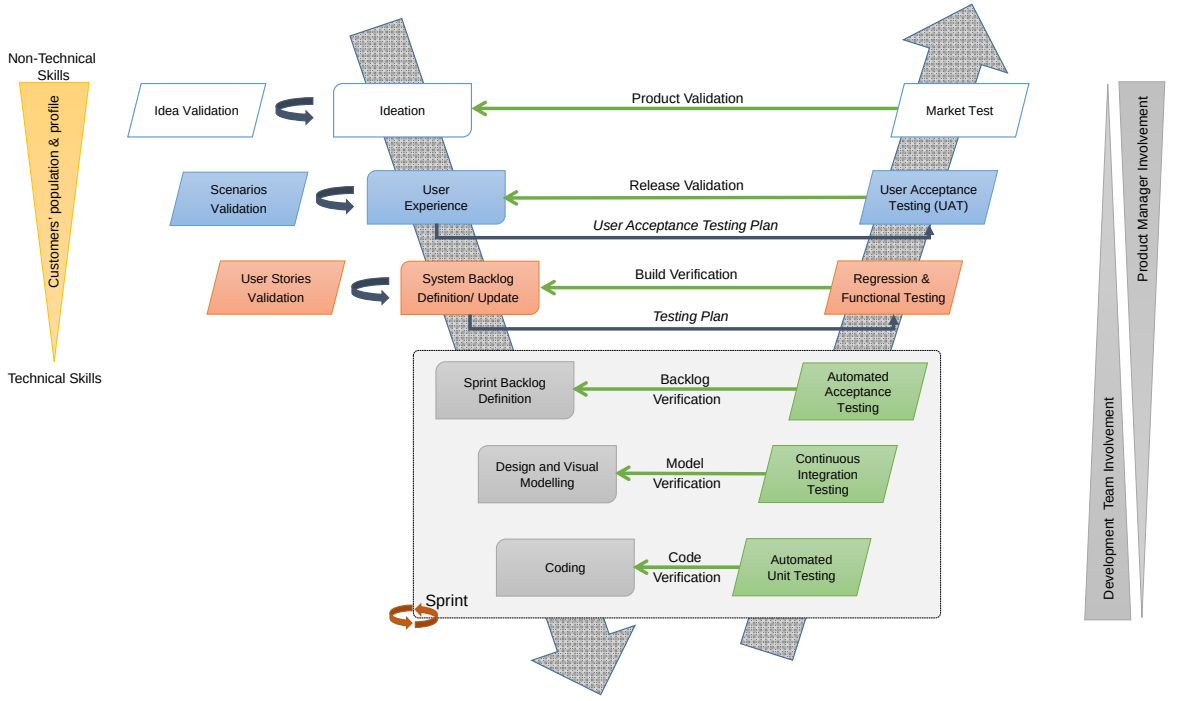
The main focus of CloudTeams is on the development of workflows that may increase end-users' input (crowdsourcing in software engineering), in order to validate and improve quality and relevance of software. In this quest, CloudTeams mainly targets consumer applications developed mostly by startups and small software teams and that are addressing customers as a user group, thus are not being tailored to the needs and preferences of each customer. Such teams do not possess the necessary resources to conduct extensive market analysis activities, not do they possess a large user base and therefore the identification of likings and preferences of their target audience as a whole is a critical success factor for their products, as well as the proper coordination, monitoring of software output performance and rapid reaction to user feedback from the teams perspective.. However, during the project, also other types of software are tested with CloudTeams, such as B2B software with the most notable example being a ERP implementation that is quite popular amongst small enterprises. Various attempts have recently worked on that direction: Stolee et al[13], used Amazon Mechanical Turk as a crowdfunding platform for software engineering study; Lim et al[7] introduced StakeSource to crowdsource stakeholder analysis and reduce the effort from experts; Murray-Rust et al[8] described a conceptual model for combining process models with crowd-sourced teams to create software artefacts in support of dynamic communities, via a Social Compute Unit (SCU) combining human workers with social agents who order and prioritize requirements. This formalisation paves the way for increased intelligence to be brought into crowd-sourced software development, creating a responsive, community-centred process.

[‡] https://standards.ieee.org/findstds/standard/12207-2008.html

3. Engaging Users in the Software Development Lifecycle – The CloudTeams methodology

CloudTeams provides an environment to support the whole lifecycle of software development with social features, starting from the ideation phase and ending at the final market release of a software product. The CloudTeams methodology provides a path to the development lifecycle and defines the points and ways of interaction among the software teams and their prospective customers, as well as a set of recommended verification and validation activities. It is not an obligatory process, but a recommended methodology to increase the effectiveness of the tools of the CloudTeams platform, by following specific steps.

The goal of the methodology is to make the social interaction and feedback with customers easier and part of the software development process. The CloudTeams methodology, which is depicted in Figure 1, is based on two existing models: (i) the V-model[§], a modified version of the waterfall method incorporating verification and validation processes, and (ii) the FITMAN V&V model[6], which supports V&V in distributed software development environments. In general, such a model is not the preference of software development teams that deliver software products that are in the focus of the CloudTeams project, as small teams work in an agile manner, following LEAN principles and do not usually follow strict validation and verification steps. As such, they are able to deliver results fast, however failure is greatly attributed in many cases to the lack of proper structures for validation mostly (as code verification is addressed adequately), where projects fail simply because they do not satisfy the needs of the target audience. In this context, the authors believe that such a V-model inspired methodology which incorporates many agile working principles, is in the position to change the mindscape of smaller development teams and alert them about the importance of validation of outputs with customers, and therefore educate them on a systematized manner to produce software.



[§] http://www.waterfall-model.com/v-model-waterfall-model/

Fig. 1. Elements and steps of the CloudTeams methodology.

The CloudTeams methodology goes far beyond these two models as it aims to provide an inclusive, yet flexible, easy to understand procedure that can be used by software teams to deliver better products and services. In this context, the methodology emphasises on the ideation phase and customers' opinions utilisation, to drive collaboration between Customers and Software Teams, but also internally among Software Team members. Regarding the utilisation in software products development, the CloudTeams methodology is agile-oriented**,** supporting popular agile-driven software methodologies, like *Scrum, Kanban, Scrum-ban, etc*. On the other hand, this does not exclude teams that wish to follow the *waterfall model*, as long as the methodological steps in the grey background are amended respectively. A high-level diagram of the methodology is available in Figure 1.

The methodology depicted above tries to specify the CloudTeams concept and the platform's operation in a standardized manner. It is based on the V-model and the FITMAN V&V framework, and works towards extending it in specific, software-specific sections. The adoption of the CloudTeams methodology by the Software Teams should not be obligatory, but it suggests better results if used properly with the CloudTeams toolkit; following the suggested methodology will verify to teams and customers that the produced software is an outcome of a collaborative, user-tested and validated process. In detail, the CloudTeams methodology provides an iterative process of the following steps (Figure 1):

- The ideation phase of the methodology and ideation takes place via the CloudTeams ideation modules. This step aims at connecting development teams to potential customersand at supporting the validation of given ideas (e.g. through interviews or market analysis) on the basis of the customers' opinions provided in a number of alternative ways.
- The second step focuses on user (customer) experience, in terms of collecting feedback from targeted customer groups, represented by different personas in CloudTeams tools, leading to the specification of the complete customer requirements list of the software to be developed. This process takes place initially by developing and validating *usage scenarios*, together with *mock-ups* or *videos* showcasing, and later by collecting customers' feedback on early releases of the product. A complementary outcome of this step is the definition of the *User Acceptance Testing* (UAT) plan, to be used for validating the software before its release.
- The third step targets at the definition of the *functional* and *non-functional* requirements of the software as user-stories. An initial user story list is compiled and gets validated both from the development team, towards specific criteria, and from the customers as far as it concerns the value of each user story from an end-user perspective. With the responsibility of the Product Manager, the user-stories list is being filtered according to the validation results and the stories are being classified in the System Backlog, considered as a registry of all approved requirements with given priorities. After the finalisation of the System Backlog, a series of tests need to be developed in order to support functional and regression testing of the different software builds.
- The next steps of the methodology refer to core development of the software (grey box in the bottom of Figure 1) and cover the sprints/iteration cycles required for the release of the software. CloudTeams adopts the FITMAN V&V Method[6] according to which testing is incorporated in each iteration in a standardised way, so that it can be tracked and certified later. The exact flow within the sprints can be customized according to the exact development methodology that a software development team follows, as long as (semi-)automated testing processes – both in unit and in sprint output level – are incorporated to the overall process and monitored in each cycle with the help of the CloudTeams platform.
- After the completion of each build of the software, the CloudTeams methodology supposes functional/ regression testing of the release, taking advantage of the testing plan delivered in the third step of the methodology. To assure the quality of a product, automated tests incorporated in each sprint should be accompanied by a formal verification process. The main reason for that is that even alterations to an application's source code that seem insignificant can ripple outward in surprising ways breaking functions that seem completely unrelated to the new modification. Given that, the establishment of a regular testing process, according to a standard series of tests

which (re-)run after the completion of each build, is considered an important step of the CloudTeams methodological approach.

- Following the verification of each software build, the suggested methodology supposes the establishment of a validation process, according to the User Acceptance Testing plan (UAT) delivered in the second step. Only if the UAT completes successfully the software can be released to the market; else appropriate measures have to be decided, getting back to the previous steps of the process.
- The last step of the CloudTeams approach refers to identifying the Market Acceptance of each release in practice. Monitoring and analysing redeems, mentions and comments regarding the software not only can reveal how the market has welcomed the release, but can also provide useful insights for fine-tuning the software or even the idea itself.

Considering the CloudTeams approach as a whole, going down through the steps of the methodology requires more technical skills both regarding the customers and the software teams. Respectively, the role of the product manager gets smaller as the process goes down while the role of the development team becomes more crucial. Going up, following the verification and validation steps of the methodology, reverses the involvement of the product manager and the developer team respectively, and at the same time allows more customers to participate in the process even if they have limited technical skills or IT experience.

4. The CloudTeams Platform – Operational Scenario and System Architecture

4.1. How CloudTeams Works – Simple Operational Scenario

The CloudTeams platform comes as a tool that provides software teams with all tools for team collaboration, software developing, testing and deployment, and also facilitates their interaction with customers, while enabling them to follow the CloudTeams methodology. In a simple scenario of operation, Software developers set up their software project (which is mostly a consumer application) and are provided with the necessary tools for project management and collaboration within the software team and for the interaction with customers during the whole software development process. In order for the platform to deliver its full potential, the presence of end-users (e.g. customers) is necessary. In this context, perspective customers upload, synchronize, and manage their personal data in order to analyze and visualize their daily activities, while they can give permissions to software teams to select them to co-develop better software solutions under several rewards. The platform integrates and aggregates the most important services and information relevant to project management and team collaboration, software development, testing and deployment, while it also enables end users to connect their cloud-services, browse ongoing and follow interesting software projects, and get notified of project invitations.

In such a scenario, a customer (named Joe) comes into the platform and connects some of the cloud-based services for fitness tracking that he uses (e.g. Fitbit) to the platform, letting the platform collect data for this kind of activity which is transformed into an appropriate activity ontology that matches those with data from similar services. The customer is then also suggested with some projects that are already registered into the platform that have to do with fitness tracking and health, which he visits and declares his interest. At the same time, a software team (TeamY) that is working on a novel health monitoring application (HealthyApp) is seeking for users to evaluate some new features that they are delivering. The team executes a query into the platform seeking for young people that live in big cities, possess an iPhone and are running a lot, and is able to find users with the specific characterizes; however, the platform presents these users with fake usernames (Joe is shown as userx43Y) and with obfuscated activity data so no direct link to the real individuals is possible. The team then selects some of these users and sends them a questionnaire, which they can answer, accompanied by a link for A/B testing that has been auto-generated through the CloudTeams platform by using the appropriate connectors that allow to automatically deploy sandboxed instances of the project's code in cloud based infrastructures for performing testing. Joe gets a notification that TeamY which delivers an application called HealthyApp is asking him to fill in a questionnaire and perform an A/B testing; in return he will get a small reward by the team (3 free months of their App usage) in case he is interested. He logs into the platform where he is presented with more information about the HealthyApp

project, as well as about TeamY, which, according to the information provided by the platform, has been since long a team present in CloudTeams and is revealed to constantly take into serious consideration users feedback as the platform indicated the number of interactions this team has initiated and the number of different deployments of its project it has uploaded. Joe, thrilled by this opportunity and answers the questionnaire and at the end is able to collect his award, which is provided in this case as a voucher send to him in a private message from the CloudTeams platform. At the same time, the project manager of TeamY who is handling the account of the company in the CloudTeams platform collects the anticipated feedback by a prospective customer (userx43Y) that he has selected in the previous steps and most probably, based on his activity analytics, matches the profile of his company's targeted users and is able to extract new issues and feature requests, notifying his team of the new developments that emerged through interaction with a prospective – anonymous- client.

4.2. The CloudTeams Platform Architecture

The overall CloudTeams platform is constructed by developing the platform components as extensions to existing systems, and integrating the components as well as best practices from software development and user participation under a unified platform using REST and XMLRPC APIs.

CloudTeams addresses the developer community and the end user community, it provides each community with a tailored platform component and implements a central anonymization component as a broker between them. The workflows for customers are implemented by the customer platform component. The workflows for development teams are implemented by the team platform component. The persona builder component allows developers to access customer data in a trustful and abstract way only. For scalability reasons most components are implemented stateless and depend on input and output data. In CloudTeams business- and user-related data are stored in databases. CloudTeams uses two main data storage facilities, one for customers and one for developers. Using this strong separation allows us to better protect the customers' privacy and keep project data protected.

Data exchange happens between the team platform and the customer platform regarding project profile data which has actively been published by the developers. Customer interactions with the software projects, such as following a project, posting an idea or answering a questionnaire is mirrored from the customer platform to the team platform. CloudTeams connects with many different cloud-based services. Customers may connect to different social or activity tracking services (e.g. Runkeeper, Facebook, Fitbit, Twitter, YouTube etc.), while developers may use services for software deployment, code management, or quality reviews.

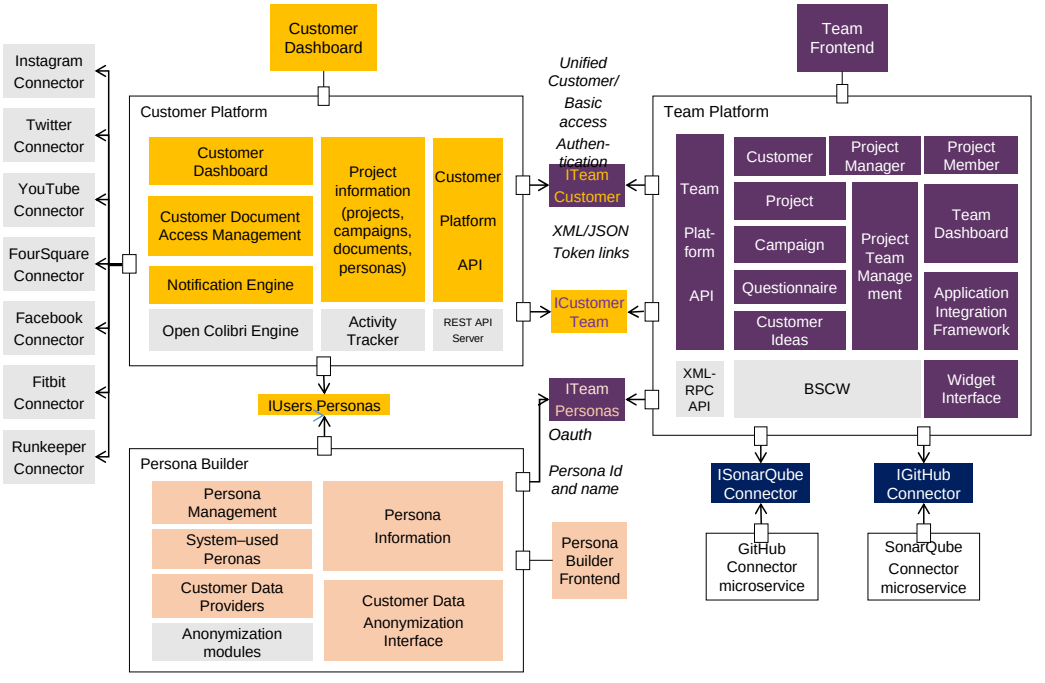


Fig. 2. CloudTeams Architecture.

4.3. Customer platform

As the customer platform we refer to the platform which is accessible by end-users and acts as a showroom of the development teams' work. Users are able to browse ongoing projects, read project news in the project blog, and find running campaigns, which software developers have published to the customer platform. Further interaction with software projects and the participation in campaigns requires the users to register for a CloudTeams customer account. This account offers the possibility to reveal some information about oneself, for example the region of origin, favored devices or companies, and connections to social media services. This data is analyzed and anonymized by CloudTeams and gets published to the software developers by the persona builder, the tool for developers to invite end users matching the target audience.

Once customers discover an interesting project, they can decide to follow the project, post ideas to the project teams or comment on the software team blog posts. Once following a project, users get notified about ongoing campaigns, in which software teams ask for user participation. If users are willing to offer help to software developer teams, they can participate in the campaigns, and can e.g. give feedback on shared documents or participate in the questionnaires the software team has created for the campaign. Software teams have the possibility to reward participating users for their work, e.g. by early prototypes or free licenses.

4.4. Team platform

Software developers register for an account on the team platform to create and manage their software projects or may get invited into an existing project. In their personal dashboard, software developers find an overview of their ongoing projects which they have created themselves or which they were invited in. The team dashboard reveals whether a project has been published to the CloudTeams customer platform and shows notifications about the most recent activities in the projects. Furthermore, it allows setting up a developer profile, searching for projects, or filtering the project list by their categories. In general developers own in a project either the role of a project manager or project member. The project manager has advanced access right, e.g. inviting team members to a project, publishing a project and starting a campaign.

The detailed project page reveals all relevant information for a single software project. It includes information about statistical numbers of the software project, such as the numbers of campaigns or team members, and information about the visibility of the project and the user's role within the project. It shows the project logo, gives an overview of the project's draft, open, and closed campaigns, and shows the project developer team. Moreover, the developer team is enabled to manage and share project documents. In order to provide the developer team with customer feedback, customer ideas posted to the project are listed in the project page as well. The campaigns are the major areas for the interaction with end users. With the persona builder developers define and invite the target user group (persona) for a campaign. Then developers can share information (e.g. tutorials, videos etc.) and develop and run questionnaires with customers that either follow a project or match the persona. As the CloudTeams methodology combines software engineering and collaboration features in one unified platform, the team platform supports retrieving information through API calls from different third-party services (such as GitHub for code collaboration management and issue tracking and SonarQube for code quality assurance). This enriches the software engineering and collaboration features needed for using and evaluating the suggested methodology through a single, unified platform. These service connectors are implemented and integrated as widgets to the team platform.

4.5. Persona builder

Data that users are willing to share about themselves is anonymized and passed to the persona builder. This is a component located between the customer and the software team platform and manages the users' data. The persona builder collects information about user characteristics that may be interesting for the software developers. Instead of waiting for users to follow a project, software teams are able to define a persona by configuring a set of desirable user characteristics. The persona builder matches the user-defined criteria with the user base and reveals how many CloudTeams users match these criteria. When the software developers are satisfied with the amount of possible users, they can invite the persona to their campaign. Connecting a persona to a campaign means anonymously inviting users with the desired characteristics to the campaign. Therefore, personas are a means for acquiring a user crowd which is potentially interested in the development and is there-fore likely to help the software developers.

5. Evaluation through diverse real-life pilots

The evaluation of the platform is done by a number of startup software companies using CloudTeams for their projects, while being guided by project partners. These are provided with the task to incorporate the project's results, including the proposed methodology into their current production and business processes extending its use to other business areas or products. The effectiveness of CloudTeams will be confirmed though the comprehensive testing and validation by users that represent the pilot partners and also external users that can represent customers. The pilots will benefit by the usage of CloudTeams platform and methodology with the possible improvement of the development process and their software products.

As the platform offers two entry points (one for customers and one for software teams), it is important to validate it from the viewpoint of both stakeholders. From the software development team viewpoint feedback must be provided by developers, project managers and members of a software team. From the customer viewpoint feedback should be provided by external users who are current of potential customers of the actual pilots' applications. The pilot's feedback will raise new requirements to even further target the platform towards its customers.

6. Outlook and Conclusions

This contribution presents the results of the first year of the EU-funded CloudTeams project. Building upon the results of a first round of interviews with stakeholders[2], we developed the CloudTeams methodology and realized the three main components of CloudTeams which seamlessly work together for integrating end-users into software development projects. The next release will provide extended functionality such as further connectors to third-party services, gamification, and rewarding end users for their participation in campaigns. Moreover we expect feedback from our pilots that helps refining the existing functionality to extend the platform with new features. The current version of CloudTeams is released under http://www.cloudteams.eu.

Acknowledgements

CloudTeams received funding from the EU's H2020 programme under grant agreement No. 644617.

References

1. Franken S, Jeners N. Challenges of Social Software in Clinical Environments. *Supplementary Proceedings of the 12th European Conference on Computer Supported Cooperative Work.* Aarhus; 2011.
2. Franken S, Kolvenbach S, Prinz W, Alvertis I, Koussouris S. CloudTeams: Bridging the Gap between Developers and Customers during Software Development Processes. *Procedia Computer Science* 2015;68. p. 188-195.
3. Franken S, Kolvenbach S, Prinz W. Social Media Integrated into Groupware. *Adjunct proceedings of the 13th European Conference on Computer Supported Cooperative Work*. Paphos; 2011.

4. Franken S, Prinz W, Jeners N. Making groupware Social – The Case of a Cooperative Platform for Surgeons. *Proceedings of the 6th International Conference on Intelligent Networks and Collaboratibve Systems* 2014. p. 49-56.
5. Koch M, Richter A. *Enterprise 2.0 – Planung, Einführung underfolgreicher Einsatz von Social Software in Unternehmen*. Oldenbourg Publishers; 2007.
6. Lampathaki F, Panopoulos D, Kokkinakos P, Bompa C, Koussouris S, Askounis D. Infusing Verification and Validation in ICT Solutions in Manufacturing: The FITMAN V&V Method. In: Mertins K, Bénaben F, Poler R, Bourrières JP, editors. *Enterprise Interoperability VI*. Springer International Publishing 2014;7. p. 307-317.
7. Lim SL, Quercia D, Finkelstein A. StakeSource: harnessing the power of crowdsourcing and social networks in stakeholder analysis. *ICSE '10 Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering* 2010;2. p. 239-242.
8. Murray-Rust D, Scekic O, Truong HL, Robertson D, Dustdar S. A collaboration model for community-based Software Development with social machines. *10th IEEE/EAI International Con-ference on Collaborative Computing* 2014
9. Prinz W, Kolvenbach S. From Groupware to Social Media – Extending an Existing Shared Workplace System with Social media Features. *Information Technology* 2012;54. p. 228-234.
10. Renzel D , Klamma R, Jarke M. Requirements Bazaar: Experiences, Added-Value and Acceptance of Requirements Negotiation between End-Users and Open Source Software Developers. *Software Engineering and Management* 2015. p. 122-123.
11. Renzel D, Behrendt M, Klamma R, Jarke M. Requirements Bazaar: Social Requirements Engineering for Community-Driven Innovation. *Proceedings of the 21st IEEE International Requirements Engineering Conference*. Rio de Janeiro 2013. p. 326-327.
12. Schwaber K, Beedle M. *Agile Software Development with SCRUM*. Prentice Hall; 2001.
13. Stolee K, Elbaum S. Exploring the Use of Crowdsourcing to Support Empirical Studies in Software Engineering. *Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*; 2010
14. Westland JC. The cost of errors in software development: Evidence from industry. *The Journal of Systems and Software* 2002;62. p.1-9.