

Software Developer's Journey

A Story-Driven Approach to Support Software Practitioners

Murat Yilmaz^{1,2}, Berke Atasoy³, Rory V. O'Connor^{4,5}, Jean-Bernard Martens³
and Paul Clarke^{4,5}

¹ Çankaya University, Virtual Reality Laboratory, Ankara, Turkey

² Çankaya University, Department of Computer Engineering, Turkey
myilmaz@cankaya.edu.tr

³ Technische Universiteit Eindhoven, Eindhoven, Holland
berkeatasoy@gmail.com
j.b.o.s.martens@tue.nl

⁴ Dublin City University, Dublin, Ireland

⁵ Lero, the Irish Software Engineering Research Center
Rory.OConnor@dcu.ie
Paul.M.Clarke@dcu.ie

Abstract. Agile development requires a highly iterative and collaborative design process, which relies on the successful interpretation of software development activities amongst team members throughout the overall process. However, contemporary methods and tools that support agile efforts provide little help in addressing context-specific tacit knowledge, which is difficult to externalize without a shared method of interpretation. Without a continuously updated interpretation of the project vision, it is difficult to claim a shared mental model, while this is actually vital for the success of an agile process. In this paper, we address this issue and seek guidance in an approach that is commonly used in film storycraft. Film production has ample experience with externalizing experiences with the help of visual planning tools and related techniques to orchestrate the creative efforts of vast interdisciplinary production teams. We therefore propose that methods and tools from visual storycrafting can be adapted to assist software developers, not only with externalizing and discussing context-specific tacit knowledge but also to keep them creatively engaged in the development process.

Keywords: Agile Software Development, Story-driven software development, Storycrafting, Software Development Process.

1 Introduction

While agile methodologies have recently been proposed as a way to overcome challenges in the software development process, integrating agile approaches successfully into an existing company culture is actually a very challenging task.

Agile software development consists of a set of iterative software methodologies. Team-based production is advanced through collaborations of team members on the requirements that are collected from stakeholders [1]. It can be characterized by incremental development cycles, lightweight documentation, and face-to-face communication between team members. It requires teams that are capable of organizing themselves around cross-functional development activities by means of effective *tacit knowledge sharing sessions*. The basic assumption of agile methodologies is that effective software development relies on *context-specific tacit knowledge* that is at times hard to make explicit and share. The success of sharing such knowledge is affected by practitioners viewpoints, beliefs and value systems [2], as well as by their personalities [3]. Many development methodologies aim to support the conversion of tacit knowledge into formal and explicit system requirements that can in turn be converted into technical software specifications [4].

Many proponents of agile software development argue that it is more effective when a *shared mental model* [5] is developed by a team of practitioners who also have the technical skills to subsequently convert the shared objectives into technical specifications and implementations. Recent evidence suggests that this process of formulating and sharing (i.e. externalizing) tacit knowledge is actually much more difficult than the reverse transformation of explicit knowledge into tacit understanding (i.e. internalization). This latter method of knowledge transfer is known as hands on experience or learning by doing [6].

In storycraft there is a well-established tradition in representing and sharing tacit knowledge about human experiences by capturing such experiences in both textual and visual form (e.g., using storyboards). This tradition builds on the ability of most people to quickly sympathize, i.e., develop an emotional relationship, with the context and characters in a story, even in cases where this story has only been sketched in rudimentary form and has not been worked out in detail. Stories do not only convey functional information, such as the features of a new software program, but also propose contexts and users that can give meaning to such features. Stories can also assist in orchestrating team efforts, as team members develop an understanding of how they can contribute with their specific skills to the overall team objective, and possibly provide end-user value [7].

A quick, inexpensive and flexible parallel process to continuously translate tacit knowledge into explicit conversation can assist in capturing the essence of the motivation, and in understanding the impact of the actions taken by individual team members. The ability to think in terms of stories can help with externalizing context specific tacit knowledge and with coordinating the software developers' creative efforts.

From a software development point of view, storycrafting may assist the agile approach in several ways: (1) providing insight into the progress of the strategy behind the development process by visualizing not only actions and accomplishments but also the underlying reasoning and mindset, (2) providing engagement by visually capturing an explicit representation of the strategic moves between

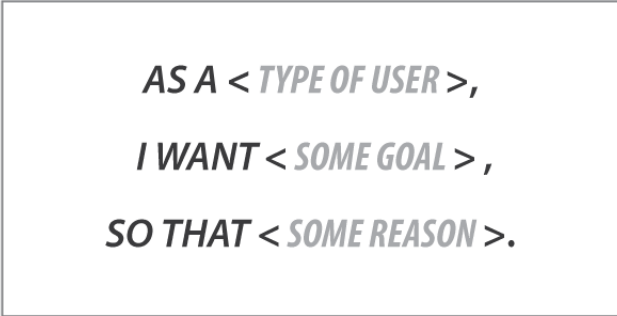
actions, (3) motivating and engaging team members into the overall process not just as *task responsables* but also as *creative contributors*, (4) providing a common awareness of the impact of individual contributions throughout the whole process, (5) establishing a visual structure that engages practitioners not only practically but also emotionally in the task at hand.

The aim of the exploratory study that we report here was to assess the usefulness of storycrafting for improving the communication during an agile development processes. The more specific research question was whether or not we could improve an agile software development process by enhancing the visual communication process using story crafting techniques.

The remainder of the paper is organized in the following manner. Section 2 reviews relevant literature. Section 3 describes the details of our proposed approach. In the final section, we discuss the potentials and benefits of this approach.

2 Background

Agile development proposes the notion of *user stories* which describe planned features of the software artifact within a context, so that an understanding can be developed of the resulting customer experience (See Figure 1). it has been empirically observed that this is an approach that is quite effective for capturing software requirements [8]. An important reason for this is that stories are an effective way of communicating a demand and hence an efficient way to exchange tacit knowledge [9].



**AS A < TYPE OF USER >,
I WANT < SOME GOAL > ,
SO THAT < SOME REASON >.**

Fig. 1. User Story Cards.

While the introducing *user stories* into an agile process has proven to be valuable, the full potentials of narrative imagining has not yet been accomplished. We are convinced that visual storycraft has much more to offer to an agile development process than is currently the case.

Storytelling is an explanatory activity that aims at providing a symbolic representation of a range of interactions and consequent emotions that an individual may experience with the software service under development. on one or more subjects. The contextual information within a story is important tacit knowledge that assist in developing a more explicit awareness of consumer value and the relevance of specific software requirements. Kalid and Saifullah [10] for instance proposed to use narratives as a way of capturing the tacit knowledge in a project, while Linde [11] argues that social knowledge can be transmitted effectively by using a narratives. Narratives provides a way to share tacit knowledge on socially relevant aspects such as lessons learned from previous projects, alternative ways of executing specific tasks, ways of resolving conflicts, etc.

Imagining in the form of stories is a rational thinking tool for looking into the future, predict, plan, and explain situations that do not yet exist [12]. All of the above are cognitive efforts and they require a common ground and a shared process to explicitly discuss, assess and decide. From a storycrafting perspective, agile efforts do not only promote interdisciplinary communication and process management of software development but also help to coordinate co-creative efforts towards innovative software design.

2.1 Storycrafting

Storycrafting takes various forms such as oral, written, theater and film [13]. Crafting a story does not entirely rely on the creative intuition of artistic minds. It relies on a well-established tradition in translating tacit knowledge into everyday thinking through structural frameworks and pre-defined guidelines and techniques. Aristotle maps the structure of actions in a story as a unified plot with a beginning, middle and end [14]. Propps Morphology of the Folktale establishes 31 commonly occurring themes that are called *story functions* which are classifiable actions that a protagonist can take in a story [15]. Campbell points out a common structure called Hero's Journey in every great story that has resonated through history [16]. Freytag describes a five-act structure to build a story that mostly applies to ancient Greek and Shakespearean drama [17]. Field's Paradigm establishes a three-act structure in which definitive moments called plot points control the change in direction of a story from the beginning till the end [16].

Certain forms of storycraft such as feature film-making have a proven record in managing and orchestrating highly complex production processes by interdisciplinary teams that passionately work and share an extremely complex process. Their process requires an interpretation of the tacit knowledge embedded in the scripts to be shared with a vast crowd of interdisciplinary workers. Visual planning such as story, e.g. through continuity boards, provides an efficient interface to explore options, reflect, discuss and detect problems and make changes quickly, easily and inexpensively [18].

2.2 Stories as developer experiences

Agile development is an exploratory process, which starts with a prioritization of aspects such as required collaboration and interaction in the software development process. Consequently, recent development and design challenges in software development require more human-centered approaches and the software industry is shifting its demands on software practitioners towards soft skills. In order to achieve sustainability in software decision making, practitioners should understand their dependence on their teammates, respect skills and approaches that they do not master themselves, and be able to cope with personality differences.

In this paper, we suggest that we should more rigorously organize software development experiences in terms of stories that can be used for communicating our development experience to other software practitioners. Furthermore, we suggest that agile software development demands a concrete but flexible way with a quick and inexpensive interface to externalize and capture the fleeting moments of discussion and decision-making in order to improve the quality of the design and development processes.

3 Software Developer’s Journey

We propose a novel approach to assist teams in coordinating software development tasks by using a visual externalization process. The goal is to reveal important aspects of the software development progress in a visual environment with the intention to promote transfer of tacit knowledge. Software practitioners’ experiences can be in the form of code snippets that solve a known issue, or propose an alternative approach towards resolving previously encountered problems.

Firstly, based on the notion of a shared metaphor that is one of the twelve practices in agile development [19], the second step in the process is to interpret the project tasks jointly with team members by exploring the potential statements in which such a metaphor is invoked. Here, we interpret software development as a theory building activity, which aims to bridge the gap between tacit knowledge and software project documentation by effectively capturing the design knowledge from an experience-focused perspective.

Secondly, a software team assesses the collected software development experiences into story elements such as *actors*, *domains* and *artifacts* in order to utilize them in the software design process. The tacit knowledge consists of a set of rules and procedures that can assist in reaching decisions on specific development tasks [20], and hence guides the overall software development process.

Thirdly, a software team identifies the key stakeholder(s), i.e., relevant personas and contexts as well as the intended experience. The demographics of the persona and his relationship to the software development tasks are essential for understanding what defines successful use of the software. Identifying stakeholder conflicts at an early stage can provide a software team with an understanding of

the emotional consequences that completing or failing software tasks may have. There is evidence to suggest that such an understanding can have influence the success of a software project [21]

Fourthly, the projected experience can be contextualized through a visual representation of all actions that can shared, e.g., by means of an agile task board¹, which may in turn help to coordinate the software development process (see Figure 2).

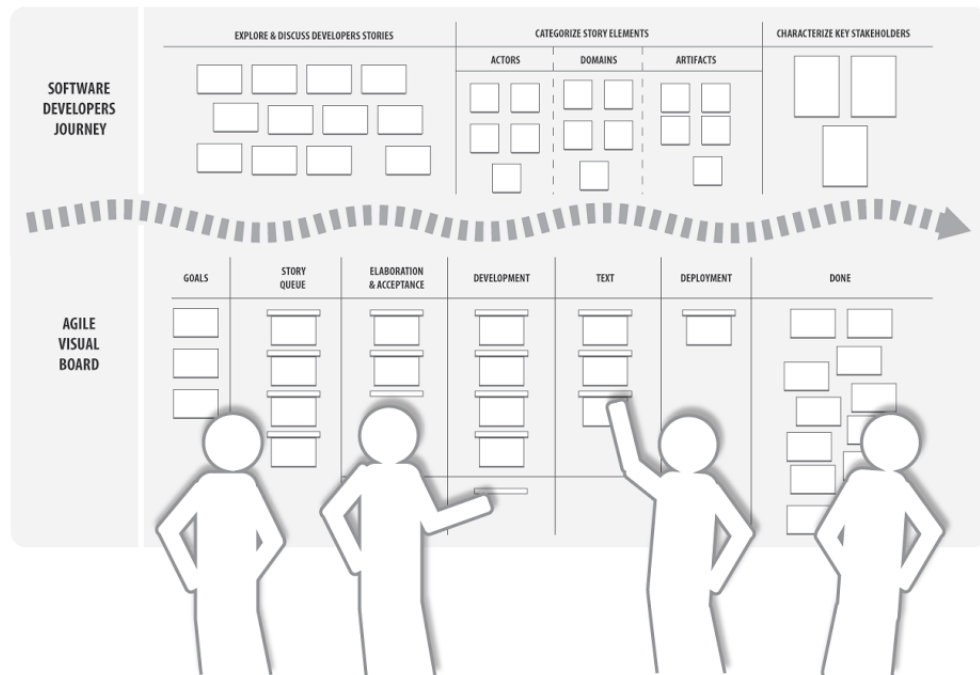


Fig. 2. The enhanced agile task-board.

To explore the feasibility of our approach, we initially subscribed to the guidance of a method called Storyply. This method explores the application of storycraft in user experience design in order to improve the focus of interdisciplinary design teams on the targeted end-user experience [22]. We appropriated certain stages of Storyply Method in order to test some of our claims on a senior graduation project conducted by a team of three novice researchers at Cankaya University Computer Engineering Department (see Figure 3). The project group was developing a VR-based application for creating a virtual storytelling experience about GobekliTepe, which is a pre-historic site about 12000 years old at

¹ The task board is a visual display of the software development progress.

SanliUrfa, Turkey. The project lasted for a year and it was supervised by two of the authors.

The team used Scrumban, which is a combination of Kanban and Scrum, in their development process. We added two phases from the Backstage of Storyply, i.e., Insporation and Categorizing Story Elements, as well as a condensed version of the On-Stage phase. We were interested in observing the effects of the Storyply activities on the Scrumban methodology and on the team process and outcomes. Our preliminary results indicate the effectiveness of the proposed approach in several respects. Firstly, it helped the development team to build a shared metaphor. Secondly, the visual progress in the development phase was visible to all team members so that it could be used as a reference in the discussions. Thirdly, this approach helped the team to reveal already at an early stage the potential problems that needed to be resolved in the software development process.



Fig. 3. Testing how Storycraft resonates with software developers while conducting a Scrumban process with support from Storyply.

4 Discussions

A story is a structural artifact which can effectively be used in software development. Software practitioners should focus on software development tasks where stories can act as a narrative asset that may improve visual planning of software development and to keep team motivation alive. Stories can guide software teams to give priority to the experiences of targeted customers already in the early stages of the design process.

Stories guide us to create a personal interpretation of a system metaphor, which is valuable for self-reflection of the software practitioners. A shared metaphor

can help a software team explore the diversity of different perspectives and formalize a shared vision in very early stages of the development. This is inline with Beck [23] who already suggested that a team should simplify a software artifact to fit a a single metaphor. Software practitioners might agree that the design of a particular software service is similar to an assembly or product line or it may look like a coffee shop with service personnel and chiefs and customers. As it follows a storified life-cycle, the proposed approach also promotes that software teams may more easily claim ownership of the software artifacts.

An important goal of this preliminary study is to demonstrate the benefits of a story-driven software development. In fact it is important to capture and store the excessively valuable tacit knowledge using a rich story-based approach. Consequently, we claim the need for an emerging skill called *creatively engaged software practitioners* who also master the process of capturing software development experiences. Software practitioners will hopefully recognize the importance of such an approach and the potential benefits that it can bring.

Although this research envisions a preliminary framework that utilizes stories in an agile development, more qualitative and quantitative data should be collected on the proposed approach. In particular, this approach needs further testing with software teams conducting agile processes and experts should iterate on the approach based on such observations. Most importantly, a future cross-national study involving various software development teams from different countries is essential. There is, therefore, a definite need for a complementary study that could also assess the long-term effects of using stories in agile software development.

References

1. Penkar, S.: From Projects to Programs: A Project Manager's Journey. CRC Press (2013)
2. O'Connor, R.V., Yilmaz, M.: Exploring the belief systems of software development professionals. *Cybernetics and Systems* **46** (2015) 528–542
3. Yilmaz, M., O'Connor, R.V., Clarke, P.: An exploration of individual personality types in software development. In: *Systems, Software and Services Process Improvement*. Springer (2014) 111–122
4. Nonaka, I., Takeuchi, H.: *The knowledge-creating company: How Japanese companies create the dynamics of innovation*. Oxford university press (1995)
5. Larman, C., Vodde, B.: *Practices for scaling lean & Agile development: large, multisite, and offshore product development with large-scale scrum*. Pearson Education (2010)
6. Ryan, S., O'Connor, R.V.: Acquiring and sharing tacit knowledge in software development teams: An empirical study. *Information and Software Technology* **55** (2013) 1614–1624
7. Patton, J., Economy, P.: *User Story Mapping: Discover the Whole Story, Build the Right Product*. " O'Reilly Media, Inc." (2014)
8. Cohn, M.: *User stories applied: For agile software development*. Addison-Wesley Professional (2004)
9. Groff, T., Jones, T.: *Introduction to knowledge management*. Routledge (2012)

10. Kalid, K.S., Saifullah, M.S.: Project story capturing system: The use of storytelling to capture tacit. *Building a Competitive Public Sector with Knowledge Management Strategy* (2013) 315
11. Linde, C.: Narrative and social tacit knowledge. *Journal of knowledge management* **5** (2001) 160–171
12. Turner, M.: *The literary mind: The origins of thought and language*. Oxford University Press (1998)
13. McClean, S.T.: *Digital storytelling: The narrative power of visual effects in film*. Mit Press (2007)
14. Hiltunen, A.: *Aristotle in Hollywood: The anatomy of successful storytelling*. Intellect Books (2002)
15. Hammond, S.P.: *Childrens story authoring with propps morphology*. (2011)
16. Duarte, N.: *Resonate: Present visual stories that transform audiences*. John Wiley & Sons (2013)
17. Freytag, G.: *Freytag’s technique of the drama: an exposition of dramatic composition and art*. Scholarly Press (1896)
18. Glebas, F.: *Directing the story: professional storytelling and storyboarding techniques for live action and animation*. Taylor & Francis (2009)
19. Cockburn, A.: *Agile Software Development*. Addison-Wesley Professional (2001)
20. Reber, A.S., Lewis, S.: Implicit learning: An analysis of the form and structure of a body of tacit knowledge. *Cognition* **5** (1977) 333–361
21. Boehm, B.: Value-based software engineering: reinventing. *ACM SIGSOFT Software Engineering Notes* **28** (2003) 3
22. Markopoulos, P., Martens, J.B., Malins, J., Coninx, K., Liapis, A., eds.: *Collaboration in Creative Design*. Springer (2016)
23. Beck, K.: *Extreme programming explained*. Addison-Wesley (2000)