Scandinavian Journal of Information Systems

Volume 32 | Issue 1 Article 5

6-30-2020

Theorizing Affordance Actualization in Digital Innovation from a Socio-Technical Perspective: The case of the video game industry

Hrafnhildur Jónasdóttir
Kristiania University College, hjonasdottir@gmail.com

Sune Dueholm Müller
Aarhus University, Department of Management, sdm@mgmt.au.dk

Theorizing Affordance Actualization in Digital Innovation from a Socio-Technical Perspective: The case of the video game industry

Cover Page Footnote

We would like to thank Bendik Bygstad for clarification and valuable insight into the affordance framework. We would also like to extend our gratitude to the team at Snowcastle for their participation in the study and especially Bendik Stang, CTO at the company for allowing us to conduct the interviews and observations over an extended period of time. We also thank the reviewers whose valuable and insightful comments and suggestions helped clarify and improve this article.

© Scandinavian Journal of Information Systems, 2020 32(1), 147–180

Theorizing Affordance Actualization in Digital Innovation from a Socio-Technical Perspective.

The case of the video game industry

Hrafnhildur Jónasdóttir
Kristiania University College, Department of Technology
Hrafnhildur.Jonasdottir@kristiania.no

Sune Dueholm Müller
Aarhus University, Department of Management
sdm@mgmt.au.dk

Abstract. Digital technology provides opportunities for new product development and innovation through affordances. However, in digital innovation, human actors are constrained by the design of technology and its interaction with different aspects of the socio-technical context. In this article, we investigate the relationship between digital technology and developers in a video game development company and its role in supporting and hindering digital innovation. We build on theory of affordances and constraints in answering the research question: How does the actualization of affordances in video game development influence the innovation process and outcome? Based on empirical analysis, we identify four affordances: Tool development, prototyping, user testing, and patching. We theorize affordance actualization and distinguish between innovation outcome and process innovation affordances. Furthermore, we theorize the dependencies between human actors, the organization, and technology in the affordance actualization process and mechanism.

Key words: digital innovation, affordance actualization, game development, socio-technical.

Accepting editor: Helle Zinner Henriksen

1 Introduction

In the age of digital revolution, organizations have access to powerful technologies and almost unlimited information resources. Everyday technologies are being connected via the Internet and integrated with other digital technologies (Yoo 2012). This revolution gives rise to an era of Internet of Things (IoT) where human actors and technologies are linked in highly distributed networks, creating new markets and products, and changing the way we communicate, work, and learn (Xia et al. 2012).

Innovation within and across organizations is influenced by the spread of digital technology capable of processing, storing, and communicating different types of information. Such development trends are fueled by the unification of previously distinct products through digital technology (Tilson et al. 2010; Yoffie 1996). New possibilities also emerge through market disruption and digital innovation (Yoffie 1996), i.e., "a product, process, or business model that is perceived as new, requires some significant changes on the part of adopters, and is embodied in or enabled by IT" (Fichman et al. 2014, p. 330). There is, however, a growing need to develop theories explaining the role of digital technology and human actors in product development in organizational contexts, and how each component influences the innovation outcome and process (Orlikowski and Scott 2008; Strong et al. 2014).

In this paper we use the concept of affordances to examine how digital technology is used in an organizational context. In extant literature, affordances are conceptualized as how the functionality of technology "favors, shapes, or invites, and at the same time constrains, a set of specific uses" within an organizational context (Zammuto et al. 2007, p. 752). The main purpose of this paper is to examine the affordances of technology and associated actualization processes in the development and innovation process that leads to novel products and processes. To that end, we understand an affordance—similarly to (Zammuto et al. 2007)—as a potential for action by a goal-directed human actor which is actualized in a social context (Volkoff and Strong 2013; Volkoff and Strong 2017). First, we identify affordances in video game development. Second, we develop four models showing the actualization of affordances resulting in either an innovation outcome or process innovation. Using the concept of affordances provides us with a powerful analytical lens in investigating how human actors' application of digital technology enables or constrains digital innovation. This necessitates an understanding of the technology itself (Orlikowski and Iacono 2001). Leonardi and Barley (2008) hypothesize that features of technology act as both affordances and constraints on user behavior. Furthermore, they explicate the importance of understanding the socio-technical context of work practices and organizational changes in which social and technical aspects are viewed as both separate and interdependent (Volkoff and

Strong 2017). It is, for example, important to understand and cope with the persistent problems and practices within information systems development documented by Kautz et al. (2007). Technology is both a product and mediator of human action which is further shaped by the organizational context. The actions of humans interacting with technology affect structural properties of organizations (i.e., social and technological), either by reinforcing or transforming them (Orlikowski 1992). Thus, "affordances for organizing depend not only on the functionality characterizing the information technology, but also on the expertise, organizational processes and procedures, controls, boundary-spanning approaches, and other social capacities present in the organization, implying their essentially sociomaterial nature" (Sarker et al. 2013, p. 14). Anderson and Robey (2017) have credited Strong et al. (2014) with identifying three factors (abilities and preferences of the individual, features of the system, and characteristics of the work environment) that both support and constrain an individual's affordance actualization. They encourage future studies to examine how the relationship between human actors, the organization, and technology influences change through affordance actualization. This is in line with Sarker et al. (2019) who encourage future research to embrace the socio-technical perspective as an axis of cohesion for the IS discipline.

To address this knowledge gap, we study how affordances and constraints of digital technology influence the innovation process and outcome through human interaction with technology, and we investigate how human actors actualize the affordances within their organizational context for the purpose of expressing their creativity in video game development. This leads us to our research question: *How does the actualization of affordances in video game development influence the innovation process and outcome?*

We set out to answer this through a case study of video game development, which in itself is uncharted IS territory. In IS research, there are few empirical studies of the innovation process within the video game development industry (Murphy-Hill et al. 2014; Nandhakumar et al. 2013; Stacey and Nandhakumar 2009). The video game industry is relevant to the study of innovation processes and affordance actualization as it is driven by digital technologies in a highly competitive market. According to Kanode and Haddad (2009), video game development is, compared to traditional software development, "unique in that it combines the work of teams covering multiple disciplines (art, music, acting, programming, etc.), and that engaging game play is sought after through the use of prototypes and iterations". Whereas the main objective of software development is to enable users to "execute a set of tasks, determined by a clear functional objective, in a predetermined context" (Sánchez et al. 2009, p. 66), in video game development the goal is for "users to achieve concrete objectives in varying degrees of effectiveness, efficiency and satisfaction, within a specific context of use" through usability (p. 66). Computer games are in principle closed worlds in which requirements can be defined arbitrarily by developers. This stands in contrast to other types of software development in which requirements are elicited, defined, and managed in collaboration with customers and other stakeholders. Video game requirements can be defined top-down by the parent company or through engagement with users through prototypes of game mechanics and gameplay (Parnas 1986). Video games are more diverse interactive systems for fun and entertainment. They do not support task completion in the manner of, e.g.,word processors or spreadsheets, but serves a different yet very specific purpose: *to make the player feel good when playing it* (Sánchez et al. 2009). This is closely related to Stoneman's (2010) concept of soft innovation, which concerns changes in products that primarily impact on sensory or intellectual perception and aesthetic appeal rather than functional performance. The role of the user in video game development is becoming increasingly important in providing insight into the gameplay and how games are experienced. User requirements influence video game development and even drive technological development in the form of tools and platforms, which are also influenced by developers' needs and ambitions (Schreier 2017).

In a large empirical study, Murphy-Hill et al. (2014) compare traditional software engineering to video game development and identify clear differences related to the development process, and product of video games. In summary, the key differences are: (1) video game developers are confronted with more ambiguous requirements, (2) video game developers tend to use agile methods more frequently, (3) creativity is more important in video game development, (4) the ability to communicate with non-technical personnel is of greater value in video game development, (5) video game development requires diverse set of skills, and (6) being a video game developer is considered more prestigious. Hence, due to rapid technological development and needs for both soft and hard innovation of products and processes, video game development is an ideal object of study when it comes to investigating technological development and innovation.

At a general level, our case study contributes to a greater understanding of video game development as an innovation process, and how the use and development of digital technology drive innovation. Specifically, this paper contributes to understanding and theorizing the actualization of affordances across contexts and technologies and how the nature of the environment and resources influence the process (Majchrzak 2012; Strong 2014; Volkoff 2017). By looking at affordances in video game development from a socio-technical perspective, we contribute to an understanding of the dynamic relationship between people, technology, and organizational contexts when pursuing digital product and process innovation. We accomplish these goals by applying Bygstad et al.'s (2016) framework for critical realist data analysis.

2 Background

The concept of affordances is familiar to psychology, industrial design, and human-computer interaction, and was first introduced by the psychologist Gibson (Zittrain 2008; Gibson 1979). Majchrzak and Markus (2012) argue that affordances and constraints are neither properties of human actors nor technology, but should be viewed as potential interactions between actors and technology. Thus, affordances emerge through the interaction between technology and human actors and are not features of a technology. Affordances are properties that independently of perception make particular actions possible for an actor "equipped to act in certain ways" (Gaver 1991, p. 80). Affordances are action possibilities "associated with achieving an immediate concrete outcome and arising from the relation between an object (e.g., an IT artefact) and a goal-oriented actor or actors" (Bygstad et al. 2016). Affordances exist as a relationship between a knowledgeable human actor and a technology, and it may be interpreted as the coming together of needs (human actor) and capabilities (technology) (Bygstad et al. 2016). When a human actor takes advantage of one or more affordances—through technology use—to achieve an immediate, concrete outcome in support of, e.g., organizational goals, the action possibilities are actualized (Strong et al. 2014). As such, affordances are action possibilities, and actualization is the process of realizing one or more possibilities, resulting in particular outcomes. "Affordance Theory takes a socio-technical perspective that lets us be specific about the technology while simultaneously incorporating social and contextual elements." (Volkoff and Strong 2017). Whether actors recognize the action possibilities of affordances is determined by their culture, social setting, experience, and intentions (Gaver 1991). From a socio-technical perspective the social subsystem includes structures and human actors, whereas the technical subsystem includes technology and tasks. The relationship between the subsystems are particularly important in understanding the influence and impact of the larger socio-technical system (Bostrom and Heinen 1977) and how the subsystems moderate affordance actualization (Sarker et al. 2013). As stated by Strong et al. (2014), it is assumed that human actors have the requisite abilities and means to effectively actualize available affordances, although this is not always the case. Users may not possess the knowledge to engage in the process and the technology may be faulty. Such factors affect whether the affordance is actualized fully, partially, or not at all (Volkoff and Strong 2017). The socio-technical context enables or constrains the actualization of the affordance (Bygstad et al. 2016). According to Arnold (2003), affordance actualization does not follow linear cause-and-effect patterns. Rather, there are multiple possible outcomes of affordance actualization—some of which are positive while others are negative. Affordances are consciously recognized in the sense that human actors are aware of them, i.e., that they can take certain actions to effect changes. Hidden affordances are those that are neither actualized nor recognized by human actors (Van Vugt et al. 2006). Technological constraints refer to restrictions on human action, i.e., the ways in which human actors are restricted or prevented from accomplishing particular goals when using a technology (Majchrzak and Markus 2012).

In digital innovation, human actors make decisions—both consciously and subconsciously—about how to respond to the tension between affordances and constraints (Henfridsson and Yoo 2013). In response, human actors may take advantage of the properties of digital technologies. For example, digital technology affords human actors the possibility of extending the functionality of a physical product by "entangling it with software-based digital capabilities" (Yoo et al. 2010, p. 725). See also (Zammuto et al. 2007). If the functionality is found wanting, it can be further developed to meet unfulfilled needs (Ghazawneh and Henfridsson 2013). Thus, digital products become platforms for future development, allowing developers to extend and add to the original product enabling complementary innovations (Boudreau 2010; Eaton et al. 2015). As human actors interact with the technology, they become aware of the affordances and constraints that support and hinder them in reaching their goals (Van Vugt et al. 2006). Digital products sometimes evolve beyond the anticipation of their creators, which paradoxically increases their likelihood of success (Eaton et al. 2015). Furthermore, when new features are added, they may influence the existing functionality, which opens the possibility for future modifications and the continuous evolution of the digital product (Um and Yoo 2016).

3 Research method

This paper is based on a single case study of the video game development company Snowcastle. It is an independent, medium-sized company based in Oslo, Norway. It was established in 2009 and released its first game in 2011. The company was chosen for our study as it has had success with previous games, is well-established in the industry, and was developing "Earthlock: Festival of Magic" during our investigation, which provided an excellent opportunity to study how the developers take advantage of the affordances available to them during their innovation efforts. The game has been released on four different platforms; Xbox One, Steam for PC, PS4, and Wii U. Snowcastle develops its own tools based on standardized third-party software for internal use. This gives us the opportunity to observe and gain insight into the process of video game development and to study the actualization of affordances in digital innovation.

The company consists of 12 people: a COO, a CEO, programmers 2D and 3D artists, designers, graphic designers, a composer, and a sound engineer. The composer and sound engineer are freelancers located in Canada and USA who have developed music for the game. Though the employees in Norway are hired to fill specific roles, most of them are involved in tasks related to testing, marketing, and development that are not part of their official job description. This can for example be seen in the case of the concept artist gradually taking on the tasks of a game designer as well. Even the CEO works with design and marketing related tasks. As the organizational structure is flat and all employees share an open office environment in close proximity to each other, communication and collaboration within the team is an integral part of every workday. The COO and the CEO share many tasks as well, and the rest of the team is largely self-managing. Their work practices are flexible with little structure and few scheduled meetings. Formal meetings are held once a week for the purpose of keeping one another updated and sharing ideas. They use a large portfolio of software systems and digital technologies ranging from communications and organizational tools, such as Asana and Slack, to design tools, including Photoshop and Maya. The employees communicate with players mainly through Kickstarter, social media, and e-mail. Unity—a cross-platform game engine—is their core development environment. Snowcastle chose Unity as it is compatible with all platforms for computer, consoles and mobile, enabling them to focus on creative aspects of game development and be certain that the game would be compatible with major platforms at time of release.

3.1 Data collection

Data were collected over a period of three years, from 2013 to 2016, at Snowcastle. A qualitative case study approach was chosen as a research strategy for understanding the dynamics of the phenomenon in a single setting (Yin 2011). A combination of semi-structured interviews, observations, and documents constitute the empirical data basis for our investigation. The interviews were recorded and transcribed verbatim. All employees at the Oslo office at Snowcastle were interviewed (Table 1), some more than once, which resulted in 17 transcribed interviews ranging from 30-90 minutes in length. Follow-up questions were asked in later interviews and e-mail correspondence, which served to clarify and broaden our understanding of work tasks, technology use, and interactions among employees. For validation purposes, the informants were asked to provide feedback on our interpretations and findings. Once the data had been collected and transcribed, it was imported into NVivo for qualitative data analysis purposes.

Code	*Role*	*Description*
CEO	Chief Executive Officer	Management, strategic business planning, financial management, QA
COO	Chief Operating Officer	Strategic business planning and financial management
SD	Senior Designer	Design, testing, QA, programming
JD1	Junior Designer	QA, bug fixing, graphical design.
JD2	Junior Designer	Marketing, QA, platform optimization, bug fixing
JD3	Junior Designer	Marketing, QA, bug fixing, art
P	Programmer	Technical design, programming, scripts
M	PR and Marketing Manager	Product management, press management, QA
A	3D Artist	Environment design, 3D modeling
CAD	Concept Artist/Designer	Concept and character development, environment design

Table 1. Informants and their roles

3.2 Data analysis

We used the stepwise framework for critical realist data analysis shown in Table 2 by Bygstad et al. (2016). This framework consists of 6 main steps with 4 substeps that aid the researcher in identifying affordances and contribute to theorizing generative mechanisms through a process of retroduction, which seeks to explain events by hypothesizing causal mechanisms.

Step	*Description*
Description of events and issues	Identification of key events; coding interviews.
Identification of key entities	Identification of entities and events, including individuals, organizational units, technology, and the relationships between them.
Theoretical re-description (abduction)	Identification of relevant theories to observe, describe, interpret, and explain the events within the frame of a new context.
Retroduction	Substep 1: Identification of immediate, concrete outcomes. Substep 2: Analysis of interplay between technology and human entities. Substep 3: Identification of candidate affordances. Substep 4: Identification of stimulating and releasing conditions.
Analysis of affordances	Analysis of dependencies between affordances and categorization of affordances.
Assessment of explanatory power	Search for mechanisms with the strongest explanatory power related to the empirical evidence.

Table 2. Framework for critical realist data analysis by Bygstad et al. (2016)

According to Bygstad et al. (2016, p. 93), "starting with the identification of affordances makes the analysis process easier. While a generative mechanism is usually a relatively abstract (and unobservable) chain of causality, an affordance is more concrete, arising from the relationship between a purposeful actor and an IT artefact." We illustrate the relationship between structures, affordances, and outcomes in Figure 1, which has been adapted from (Sayer 1992). In our case, structure refers to the combinations of social and technical elements, which enable or create potentials for action (i.e., affordances). The outcome of actualizing an affordance feeds back into the structure (Bygstad et al.

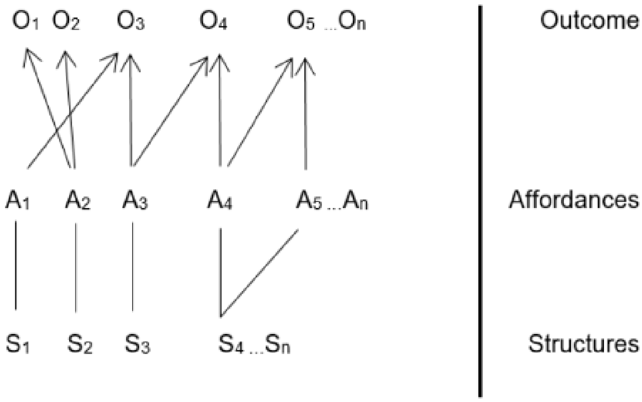

Figure 1. Structures, affordances, and outcomes adapted from Sayer (1992)

2016). This framework aided us in uncovering structures and affordances through deduction by first identifying events and outcomes in the case. In Sayer's (1992) original figure, affordances are substituted for mechanisms. Mechanisms are identified through the explanation of events, and they establish a link between structures and outcomes of events. The outcome of a generative mechanism will differ depending on the context which means that mechanisms are mainly used to explain rather than predict phenomena. The collection of different social and technical components and the relationships between them reveal the generative mechanisms (Bygstad et al. 2016). Affordances provide a steppingstone in the process of identifying mechanisms.

As a first step, we coded the data in Nvivo using preselected codes based on key concepts from the literature. These include affordances, creativity, innovation, obstacles, events, and structure as well as technology and different roles of the employees. We were then able to identify key events and outcomes by focusing on the innovation process of video game development. These events and issues were identified partly by the informants and partly by the researchers. Some were revealed through interviews to understand employees' use of technology and the influence on video game development and the innovation process as well as its outcomes. Some outcomes were innovations in themselves. These related to the game as well as the Unity engine and add-ons. One example is a proprietary toolbox, which has been built into Unity, developed to serve the needs of the development team and the process of developing RPGs. Another example is a side story in the game which introduces a new way of acquiring ammunition for the characters not previously seen in other games.

As a second step, we identified key entities related to the events, including individuals, organizational units, technology, and the relationships between them, and we used data displays to visualize the data (Miles and Huberman 1994).

As a third step, we analyzed the events through abduction to understand the nature of the case through different theoretical perspectives and explanations (Bygstad et al. 2016). This was an ongoing process throughout the entire study in which concepts and theories were discussed in trying to answer our research question through the process of retroduction.

As a fourth step, which includes substeps 1-4, we focused on immediate, concrete outcomes of the video game development and innovation process in order to identify candidate affordances. These outcomes relate to events that, on the one hand, require the use of digital technology and, on the other hand, are key to the realization of human actors' goals (Bygstad et al. 2016). We identified seven such outcomes related to the developers, the users, and the video game platform distributors. For each outcome, we analyzed the interplay between the human actors and the technologies involved. Based on this interplay, we were able to identify candidate affordances. Initially we identified six affordances, some of which lead to the same outcome, such as the affordance *gameplay*. According to Volkoff and Strong (2017), it is important to "understand the role of technology and user actions, i.e., the mechanisms involved that provide the explanatory power that is a core contribution of using Affordance Theory. To accomplish this focus, it helps to be careful in our naming conventions for affordances by using a verb participle, such as 'sitting' or 'communicating', reserving noun forms for the immediate concrete outcome that results" (p. 5). Through an iterative analytical process, these affordances were reduced to four: *Tool development, prototyping*, *user testing*, and *patching*. Once identified, we investigated how each affordance is actualized, which depends on the contextual conditions. As part of this effort, we uncovered stimulating conditions that are organizational arrangements that facilitate action and releasing conditions that are specific decisions motivating action (Bygstad et al. 2016). These are presented in the findings section.

The two remaining steps, step five and six, correspond to an analysis of the set of affordances and associated mechanisms and an assessment of their explanatory power. In this article, we focus on defining affordances, describing the process through which they are actualized, and theorizing the influence of human actors, the organization, and technology on the actualization process. Next, we analyzed patterns across the affordance actualization processes to theorize the relationship between human actors, the organization, and technology in video game development with the aim of arriving at an overall affordance actualization mechanism (Figure 8). This was accomplished through the visualization of the affordance actualization (Figures 3-6) and synthesizing them with the identified trifecta (Figure 7).

4 Findings

By following the analysis steps described in section 3, we identified four affordances: *Tool development*, *prototyping*, *user testing*, and *patching*. Figure 2 illustrates how these affordances relate to outcomes and structures. They are derived from the outcomes identified in our analysis and emerge out of the relationship between technology in use

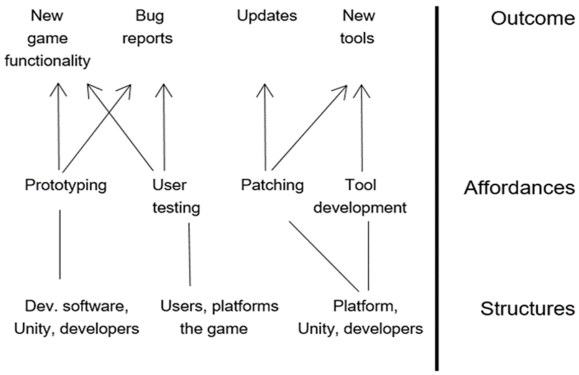

Figure 2. Case specific relationships between structures, affordances, and outcome in video game development

and actions of developers engaged in digital innovation. Each affordance has stimulating and releasing conditions that influence whether or not it is actualized, leading to particular outcomes. The affordances are described below in relation to the case.

4.1 Affordance 1: Prototyping

The prototyping affordance derives from the outcome *new game functionality.* The process of developing new game functionality begins with ideas being discussed in meetings between developers and management. These ideas relate to the game in terms of story, gameplay, mechanics, design, and the visual expression of the game. The meeting in which ideas are presented and discussed is the stimulating condition, and it relates to the organizational structure and environment that enables individual developers to voice their opinions and present their ideas. The meetings are not always formal meetings as the development teams work in an open landscape which facilitates communication between each team member. Throughout the development process ideas surface during day-to-day interactions or meetings between developers, and everybody is encouraged to participate in discussing and deciding whether the ideas should be further developed or not. As stated by the CEO:

> It is so important to talk and discuss and agree to know what to do. It's golden information that I'm very dependent on. I overhear it while two [people] next to me talk together, so I have the impression of what's going on even though I'm not a part of the discussion. It is the advantage of open-plan offices.

The ideas that—through mutual agreement—are found suitable and in alignment with the vision for the game are developed and tested iteratively through prototypes either internally or externally with players. The mutual agreement between developers and management is the releasing condition. The knowledge and abilities of the developers and the features of the technology available to them determine whether ideas are feasible, hence they function as possibilities or constraints. Developers have certain knowledge and abilities that influence what they observe as technologically possible during the ideation and development process.

> The team has evolved and everything is going faster now with more knowledge. We have also brought in new experienced people who make [the game] better and of higher quality. We then start to look at what we have done earlier and want to improve this, but then it is quick to get into an 'infinite loop' and make 'Duke Nukem' forever, so we have taken some things and redesigned, e.g., home bases. At the same time, it has improved design and increased flexibility. (CEO)

Constraints can present themselves during the prototyping process and affect the outcome through unforeseen technical challenges or the lack of knowledge to solve a particular problem. Once ideas have been agreed upon, prototypes are developed in Unity that can then be tested in terms of functionality, suitability, and general "feel" regarding how the game plays and is experienced.

> Everyone on the team must play the game, so we test ourselves and comment on each other's implementation. There are things that are constantly changing in an

> iterative process, because it doesn't work well enough, or you thought of a better way of doing it. (CEO)

Once the test criteria are met, the prototype is implemented in the game. If the outcome of prototyping is not satisfactory, the prototype goes through another iteration where changes are tested, or the prototype is simply scrapped. Prototyping is therefore an essential part of the development and innovation process. It affords the possibility of testing creative ideas and the "fun factor" of the gameplay, which have innovation potential but cannot be fully evaluated until they are combined with other elements of the game and tested by real players.

> It is important to focus on innovation that can generate a lot of money and that is in 'soft innovation'. (CEO)

Hence, game developers need to focus on user experience and gameplay, making the game fun and enjoyable.

> As far as gameplay is concerned, you are testing a prototype until it feels good to play. We've also taken things out of the game again because it didn't feel good. It was a great idea on paper, but it is boring or repetitive in the wrong way. It has to be removed or [decided] how can we make it cool. (CAD)

An example from the Earthlock game is the side story in the game in which ingredients needed to create ammunition for weapons are grown from plants. Snowcastle initially developed a prototype in which a game character harvested the ammunition ingredients from plants on behalf of the player. The feedback from players indicated that the entire activity was perceived as superfluous as the players had no influence on the growth and harvesting actions. The developers then changed it to an activity in which players have to sow their own seeds in a farm-like environment and cultivate the plants so they grow, which in turn provides players with the opportunity to acquire stronger ammunition at higher levels of gameplay. When this quest was tested, players would spend a lot of time tending to these plants as it was perceived as a meditative activity though it required more involvement than earlier. Once the game launched, a comment from a user was posted on social media stating: "The game is so RPG that even the plants level up!"

> But 'game feel' is an entirely subjective thing, and it's not necessarily something you can put your finger on. I feel like 'Earthlock: Festival of Magic' was a love

> letter to old RPGs. We're playing on a lot of nostalgia, trying to get people into that old mindset, and bring them back into what they thought was fun. They might not necessarily be there, but it might spark that same emotion. It's hard to put your finger on it. (JD2)

4.2 Affordance 2: User testing

The user testing affordance derives from the outcome *bug reports.* The simulating condition for user testing is the need for distributed problem solving. When developing a game and testing code, features such as mechanics, game play, graphics and the visual experience of the of the game are an integral part of the development process from the beginning. Once a working version of the game or part of the game has been developed, it needs to be tested thoroughly. In larger development companies, a QA team is available to carry out testing in-house. In smaller companies, developers often do not have access to a QA team and have to rely on each team member.

> Since it's a small company, QA is done by lots of people, but it tends to fall to [CEO], and [Lead artist] does the visual QA and we do sort of the game flow, game play issues. We'd address those but it's pretty tough cause we only have five of us and suddenly our game goes out to 1,2 million downloads and 600,000 users, and within an hour they have done 600,000 hours of playtesting and there's only 5 of you, so off course we have had lots of bugs. But that's my job so it helps us to design and balance the game. (SD)

Hence, to get different functionalities tested on a larger user base they rely on user testing. As one developer (JD2) stated regarding the value of users identifying bugs not discovered by the developers:

> It is great QA testing, it really is. But it feels like we're not living up to their expectations, so being a developer is kind of hard. Then seeing maybe that you're failing the target audience — like in any game, that doesn't necessarily only apply to video games — it's kind of a weird thing. Definitely from a marketing perspective, we have a lot to take in there.

Different testing environments are used which provide different inputs and observations. These can be digital or physical environments hosted by the development company. Commonly, in a digital environment the feedback is written in a report while in a physical environment the feedback can be visual, verbal, or written. In the case of Snowcastle, user testing was performed across several different environments. Social media platforms, such as YouTube and Kickstarter, were used in innovative ways to gather information about bugs and opinions from the users during development, before and after launch. Early versions of the game were accessible to select users who were provided with online access to beta versions and allowed to playtest the game. Kickstarter was used for uploading images, videos, and other information about the game, which sparked a flood of feedback from a large user base.

> Now we have had beta testing, invited every Kickstarter backer to test. And there are several who commented that they think we have done a very good job compared to what we announced on Kickstarter. After all, it is natural that you change some things and that things disappear from the plans and the like, but it seems that they are happy. (CEO)

Once the game was released, playthroughs were uploaded by users to YouTube, revealing bugs and other issues that needed to be addressed. This information was then used to optimize the game. Providing the users with early access to the game is the releasing condition for getting feedback from users through testing. When bugs are identified, a bug report is produced which feeds back into production of the game and new features through user testing. User testing is done in different phases of production mainly as alpha and beta tests, and testing stops when the game is perceived to be stable and bug-free. When the game is developed for more than one platform such as console and PC, it can reveal new challenges and bugs which put constraints on development. The developers need, for example, to go through testing on several platforms which is both time consuming and costly.

> Now we aim to get a Steam build out at the same time as the Xbox, to maximize the PR value that the Xbox provides. But it requires us to create a PC interface with mouse and keyboard and remapping keys and slightly different things. And a menu so that you can use the mouse to press buttons or menu items. Now everything is just for PlayStation and Xbox controller. And it is a task that is quite large, which we must try to get ready for launch. And if we don't, it will be after launch. (CEO)

4.3 Affordance 3: Patching

The patching affordance derives from the outcome *updates*. The distributor of the development platform updates the software regularly through patches. The developers using the software, in this case Unity, have to install these patches to ensure that the software is up to date. The distribution of the patch is the releasing condition. The patches contain fixes to security vulnerabilities and other bugs as well as new features. As Unity is used to port the game to all consoles, some updates influence both the technology standards of the game engine and the consoles.

> In Unity where we make our game there is a separate Unity version for all the consoles, and they are not necessarily the same version. So, if there are upgrades to version 5 in Unity to PC, then I have to run version 4.5 to Wii U, and to the XBox I may run the new version, but it's not compatible. So, it's a little fussy, but it's much easier than in the old days, where you had to have a whole tech-team working on each. (CEO)

As the development engine Unity is modular, allowing the development and connection of third party add-ons, these updates containing fixes and new features sometimes create problems by causing previous functions in the software to break. The developers are then forced to either respond by fixing the issues, find workarounds, or wait for a new update from either Unity or the third-party developer.

> We've got like a 4GB project and moving up to some new revisions that require you to reimport your entire assets and that can take 17-18 hours. All day just gone waiting for your computer to do some work. That's what we've had a lot of. The early days with the engines, the lighting doesn't work when we keep going to new versions of Unity and we had to patch it and it was a frustration. A lot of bugs with lighting that caused a lot of problems. It would work on this version and then we would update, and it would break it all and then it would take 14 hours to bake a light in a scene. We wasted many, many days just for technological issues with patches for unity. (SD)

Because the updates to the development engine are released frequently, they disrupt the development and innovation process. As the example in the quote above shows, an update broke the lighting in all scenes, which resulted in the need to re-render the scenes. This cost the company one week of development effort. Responding to the constraints imposed by the patches, the company has institutionalized a routine by which the programmer downloads, installs, and tests updates to determine if anything breaks.

> Mostly, [the programmer] just patches Unity on his computer and he tests if our game is compatible, and if it is, we all update. So of course, that takes some time. (JD1)

This routine allows the company to evaluate and manage the outcome of patching by solving any issue and avoiding major setbacks in the development and innovation process. During the testing phase, the stability of the patch is revealed. If the outcome is that the patch breaks parts of the software, the information about what broke and why feeds back as input to a new iteration of tests. Once a new patch is released, testing is initiated again with the information from the previous test. Establishing this test procedure is the stimulating condition—a process that makes patching safer and easier.

4.4 Affordance 4: Tool development

The tool development affordance derives from the outcome *new tools*. Sometimes a developer needs a functionality that is not available through existing tools, which triggers the acquisition or development of a tool that addresses the specific need. The technology has to be flexible and modular enough to allow for external add-ons and individual configurations.

> Unity that we use has the asset store with a whole bunch of 3rd party off-the-shelf items that you can buy, and we have tested a lot of it ourselves, but then over time we build more of it ourselves and we try to look forward, that we plan to have more games. So that we have a toolbox to build on I think will help very much with the speed of production going forward. Unity has lots of things in it from the get-go, but this has been a collaboration between programming and design all the time. If a designer wants something and [does a task] over and over again, and it is very difficult, we can create a tool that allows [them] to get to the end result faster. So, I think we quite early on found out that we should have a database that contained much of the data for the game since it is a role-playing game that is driven by a lot of numbers lying on the backend, so [tool development] is smart for us and felt necessary. (CAD)

The need for tools that make game development easier is the releasing condition that animates programmers to develop new tools. All employees sit in an open office environment which makes it easier to identify and resolve difficulties in the development and innovation process, and it accommodates the communication needs of developers. The open office environment is the stimulating condition as the communication barriers are low, and it becomes transparent what each employee is working on or having difficulty with. If team members were sitting in their own offices, or if they were placed physically according to roles, which is common in larger game development companies, the challenges of working with these cutting-edge digital technologies would not be detected as easily unless explicitly communicated by developers or other employees. Hence, the open office environment encourages developers to discuss their problems and needs for tools in support of productivity and task accomplishment. Once a need is identified, a search is initiated for an existing tool that solves the problem. If such a tool is not available, the programmer is given the task of developing it.

> There are two types of programs for scripts I write. One, I write tools for the designers to use in Unity — sort of like an extension of Unity itself, adding more functionality to our game engine — and the other one is writing functionality for the game itself, depending on what that could be. It could be the character walking around, or damage calculations when characters attack each other, or basically anything happening endgame. So those are the two things — endgame functionality and editor-extended functionality. — P

If he has the necessary abilities, he develops the tool. Otherwise they have to make do without the tool or find other ways of achieving the desired outcome. As the company is small, they only have one main programmer. Therefore, all other development efforts are put on standby while the programmer works on the tool. This is a constraint on game development. The company has developed a toolbox that addresses common needs for different members of the team and makes the game development process more effective.

> Our programmer has written an integrated data system, so I work a lot with that. All the data in our game, all the characters, all the abilities, all the effects, everything is in the database. So, we can tweak everything and that's all done in Unity. A lot of the technology we use we either buy from the asset store in Unity, if someone has already done it before, or we write it ourselves. The database is

> one big system he has written and its pretty good for keeping track of everything. So that's an inhouse tool. (SD)

Also, collaborating with other companies and sharing technology and knowledge are strategies that are pursued in certain situations to compensate for the disadvantages of being a small company.

> If there are anyone who wants to look at the gameplay mechanisms or the tools we make, then we are really open to share. We were away at [another game company] and saw how they made tools for dialogue. When we started working on our dialogue system, the system programmer designed on paper an idea for a system; how to have language, dialogue, versions, page texts, options, and all sorts of things in a database. And then, before he started programming, we went to [the other game company] and saw that they had done it the same way. But they were then finished making a tool, so we could then be satisfied that we at least had had sensible thoughts. They were happy with the system and it worked for them. But it is so special that we could not only use the system they have, so we must make our own, unfortunately. (CEO)

5 Implications for practice and theoretical contribution

Our study responds to calls for research into the role of affordances and how human actors actualize affordances in digital innovation (Henfridsson et al. 2009; Henfridsson and Yoo 2013; Leonardi and Barley 2008). We take into account the three factors (abilities and knowledge of human actors, features of the technology, and characteristics of the organization) which support and constrain affordance actualization, addressing the need for future studies to examine how the relationship between technology, human actors, and the organization influences change through affordance actualization (Strong et al. 2014 (cited in (Anderson and Robey 2017)). The affordances illustrate the possibilities that the technology provides its users, and the role it plays in enabling and constraining human actors in carrying out tasks related to digital innovation. We elaborate below on how our findings contribute through theorizing to new insights into affordance actualization.

5.1 Innovation outcome affordances

In our specific context of video game development, actualization of two of the affordances—prototyping and tool development—led to innovation outcomes in the form of new game functionality or new tools.

Prototyping affords developers the opportunity to test product ideas both internally and externally. Prototyping minimizes the risk of implementing ideas and features in the game that do not function properly or ruin the gameplay experience. As new

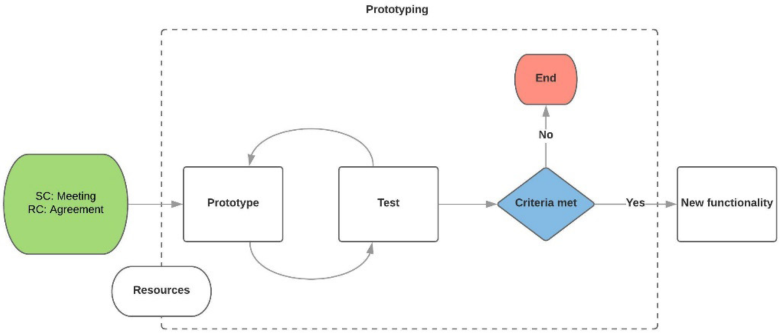

Figure 3. Actualizing the prototyping affordance: Innovation outcome

insights are generated through the ideation phase, innovative solutions emerge. Furthermore, ideas can be tested in combination. The harvesting example presented in the findings section shows how the developers—by testing the prototype—discovered that the original idea was untenable, which led to changes and further testing and resulted in an innovative solution that increased the quality of gameplay and differentiated the game from existing RPGs.

The actualization of the prototyping affordance is presented in the model below (Figure 3). The stimulating condition, meeting, motivates the developers to share prototype ideas that are discussed freely, and each individual's knowledge and abilities are reflected in their voiced opinions. This relates to the organizational context of the company, particularly their culture and physical work space. The releasing condition, agreement, initializes the development of a prototype as developers and management decide which ideas to further mature. The stimulating and releasing conditions jointly trigger the actualization of the prototyping affordance, which is moderated by the knowledge and abilities of individuals, features of the technology, and the organizational culture and physical work space encouraging participation and creativity (Strong et al. 2014). Prototyping is an iterative process that ends when the exit criteria are met, or the prototype is no longer considered viable. Testing the soft and hard elements of the prototype is moderated by the abilities and knowledge of the involved human actors in terms of how to bring about the intended emotional response (excitement, feeling challenged etc.) from users and ensure the entertainment value and playability of the game. If the criteria are met, the actualization of the affordance results in an innovation outcome in the form of new game functionality.

Tool development affords developers the opportunity to acquire tools customized to their specific needs, giving them an advantage during development such as significantly shortening the time in executing tasks. When the need for digital tools, which are not available on the market, arises, developers have the option of developing the tools themselves. This is in line with Ghazawneh and Henfridsson (2013) who conclude that limitations in the functionality of digital technology trigger users to develop new functionality to meet their needs. When such needs arise, they can be communicated in two different ways: Directly to other team members (verbally) or indirectly through (physical) signs of struggle with whatever they are working on at that particular moment. What stimulates or hinders such communication is the physical work space. Working in an open landscape increases the propensity of knowledge sharing, and needs are easily captured as these struggles become more transparent. In more closed work environments, verbal and physical expressions of needs are less frequent or more easily overlooked as there are communication barriers between employees. The identification of needs for technological solutions, e.g.,digital tools, drives digital innovation through the development of new tools—provided that the underlying technological features and standards as well as individuals' abilities support this development. This shows how the relationship between human actors and technology, their needs and abilities, and

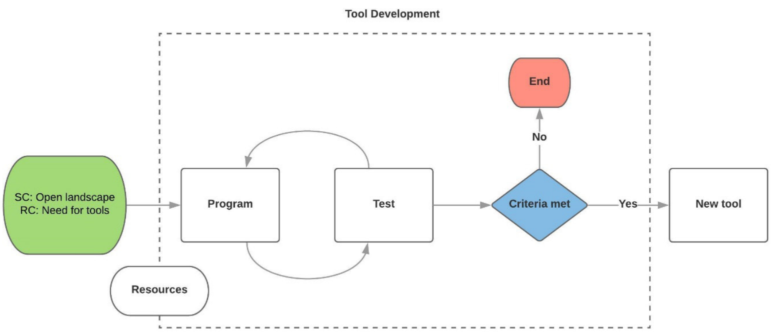

Figure 4. Actualizing the tool development affordance: Innovation outcome

the organizational environment (i.e, the context) plays an important role in facilitating digital innovation (Bygstad et al. 2016). This also goes toward showing how digital technologies afford developers opportunities to expand on the existing technology by "entangling it with software-based digital capabilities" (Yoo et al. 2010; Yoo et al. 2012).

The actualization of the tool development affordance is presented in the model in figure 4 below. Here we see the stimulating condition, *open landscape,* which provides transparency among the developers and their work processes, and the releasing condition*, need for tools,* which stems from the preferences and abilities of the developers, initialize the development of a tool. The stimulating and the releasing conditions jointly trigger the tool development affordance. If the necessary resources are in place, the actualization process is initiated. In the case of tool development, affordance actualization is an iterative process which ends once the exit criteria are met or the tool does not meet the requirements of the developers which results in termination of the process. If the criteria are met, the affordance is actualized, and the end result is an innovation outcome in the form of a new tool.

5.2 Process innovation affordance

Actualization of two of the identified affordances—patching and user testing—resulted in process innovations related to ensuring stable and updated software.

User testing affords developers the opportunity to gather useful information during and after product launch. When players are involved in testing early versions of a game, they encounter resistance in the form of bugs, and their perception of the game is influenced by personal preferences and previous experiences from playing other games. This affects the type of bugs they notice and therefore both the kind of feedback they provide and the type of changes they suggest. With the option of allowing players to test software and games online, e.g.,the Earthlock game, by means of digital distribution and access, the reach in terms of feedback is greater compared to for instance in-house testing or using focus groups. This saves time and is cost-efficient as internal resources can be dedicated to other parts of the development and innovation process, which is especially important for small development companies as they often do not have quality assurance teams dedicated to playtesting and bug reporting. The information gathered through user testing is processed by the developers, bugs are fixed, and suggestions are evaluated. This introduces new possibilities, which may lead to, e.g., suggestions for new game functionality that can be prototyped and further tested.

The actualization of the user testing affordance is presented in Figure 5 in the model below. It shows the stimulating condition, *need for distributed problem solving,* where

© Scandinavian Journal of Information Systems, 2020 32(1), 147–180

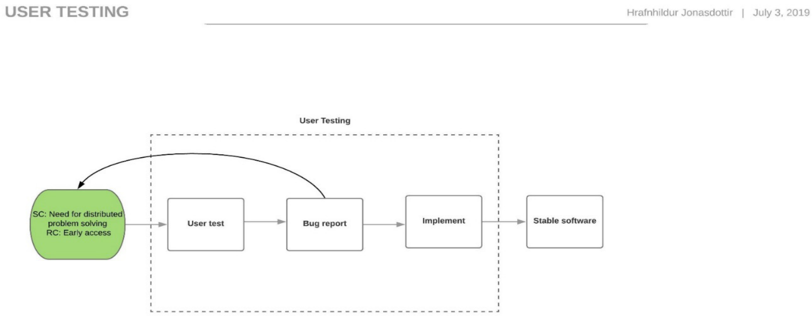

Figure 5. Actualizing the user testing affordance actualization: Process innovation

a large user base is needed to properly test the software because of the small size of the company. The releasing condition, *early access*, is a decision made by management and provides users with access to test the product before release. The stimulating and the releasing conditions jointly trigger the user testing affordance. The actualization process is initialized by a user test that results in a bug report. The information provided in the report forms the basis for implementing bug fixes to the game which result in temporarily stable software. In the case of user testing, the actualization process is linear but with a feedback loop that provides information as input to another process. The end result is a process innovation through which digital technology is deployed to effect a change—an innovation in the process of ensuring stable software and establishing an environment that allows for distributed problem solving.

Patching affords developers the opportunity to securely test software updates. When using third-party software for development, there is a dependency on the vendors to keep the software updated and working. As software updates and patches lead to changes in the architecture and the affordances of the technology, users need to adapt to these changes. This corresponds to the case study by Murphy-Hill et al. (2014), which illustrates the importance of code review to avoid developers installing patches without a preceding review.

In the case of Unity, the architecture is designed with a mind to the flexibility and possibility of adding new functionality in the form of add-ons. As the add-ons are maintained by third-party developers and not Unity, broken functionality is seen in these add-ons as a result of updates to Unity. As technology evolves rapidly, updates are provided regularly and may cause disruptions in the workflow of users. Managing the uncertainty that each update creates is important in providing stability for the users. As Snowcastle has institutionalized an update installation process, they are able to address potential problems with Unity patches early and avoid major setbacks in their devel-

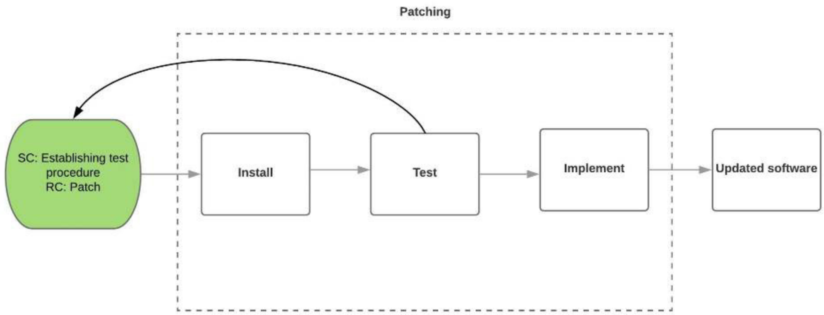

Figure 6. Actualizing the patching affordance: Process innovation

opment when implementing the updates on all employees' computers. This patching actualization process constitutes a process innovation as it saves the company precious time and costs by ensuring the stability of running software at all times. It shows that employees actively and creatively exploit the patching affordance in order to identify alternative courses of action and work around obstacles (Henfridsson et al. 2009).

The actualization of the patching affordance is presented in figure 6 in the model below. It shows the stimulating condition, establishing test procedure, which creates a safe environment for patching as the routines are changed in the company, and the releasing condition, patch, which is the availability of a software patch for installation made by the third-party distributor. The stimulating condition and the releasing condition jointly trigger the patching affordance. The process is initialized by the decision to install the patch which is then tested and eventually implemented once possible issues have been resolved. In the case of patching, the process is linear but with a feedback loop that provides information as input to the next process. Patching constitutes a process innovation through which the company is able to test patches and address unintended consequences (in the form of, e.g., wicked problems) in a safe environment.

5.3 Theorizing affordance actualization

All the identified affordances are action possibilities that influence video game development through the use of digital technology (i.e., software in this case) resulting in either process innovation (in terms of the development process) or product innovation (with regard to the game or the development software). All affordances influence each other to some degree. The patching affordance influences the tool development affordance as patching might lead to broken features, which in turn animate developers to create workarounds through for example new tools. Implementing new tools changes the way the developers work with the technology during the development and innovation process as they, e.g., make certain tasks easier and faster to complete. Prototyping influences user testing as ideas, functionality, and early versions of the game are prototyped and tested both in-house and externally with players. Prototyping also contributes to bugs being discovered and fixed, increasing the stability and playability of the game. Prototyping is also sometimes used in connection with Tool development, for example when it comes to testing the functionality and ensuring stability. User testing promotes feedback from users which provides the developers with a better understanding of what works and what does not, but it also triggers new ideas which lead to new prototypes. This in turn strengthens the stability of the game, increases the "fun factor" of the game, and nurtures positive relationships with players.

The four models show technology as an enabler of both process and product innovation. Consequently, the action potential of technology (affordances) is both about enabling innovative solutions (outcome) and the activities through which such solutions are generated (process). Commonalities were found across affordances. The two affordances, patching and user testing, share important features. In the one case it is about an environment that allows for safely testing unintended consequences (patching). In the other case it is about establishing an environment through technology use that allows for distributed problem solving (user testing). They are both linear processes with feedback loops providing information as input to new processes. What sets these two models apart from those found in extant literature, such as Strong et al. (2014), is that the resources such as features of the available technology or the abilities and knowledge of the developers had little influence on whether the affordance was actualized or not. Patches are easily tested and need to be implemented at some point, and user testing meets with little resistance in terms of the human actors' abilities, knowledge or the technology available to them. Extant literature does not describe let alone theorize the actualization of affordances. Nor does it show the importance of the actualization outcome, in this case innovation outcome or process innovation, in how the process is executed. We find that the outcome gives an indication of how the process unfolds and how resources (e.g., developers' abilities or technology features) influence the process through support or constraints. Hence, lack of such resources can constrain affordance actualization or inspire them to creatively work around this obstacle in search for a solution. The process innovation affordances do not share this dependency on internal resources.

Figure 7 illustrates the affordance actualization trifecta.

Looking for patterns across the affordance actualization processes reveals an interdependence between three components of the actualization context; human actors, the

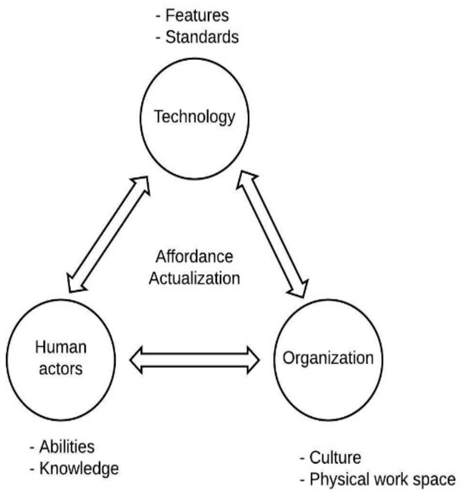

Figure 7. Affordance actualization trifecta

organization, and technology forming a trifecta. The human actors involved possess knowledge and abilities which form the basis for their opinions and actions. They act upon the releasing conditions of affordance actualization. The organization provides the environment for affordance actualization through organizational culture and the physical work space. It is therefore directly related to the stimulating conditions. The human actors and organization components are mediated by features and standards of technology. This can be illustrated through the case example of prototyping affordance actualization. At Snowcastle, developers voice their individual ideas and opinions in the context of team work, the open office environment, and the organizational culture that nurtures knowledge sharing. The knowledge and abilities of developers influence the feasibility assessment of various solutions (i.e., prototypes) which in turn influences the affordance actualization process.

Moving from an empirical to a theoretical level, it is possible to theorize the affordance actualization mechanism (Figure 8). Affordance actualization depends on the stimulating and releasing conditions related to the human actors and organization components of the trifecta (Figure 7), which is part of the social subsystem, and the technology component that is part of the technical subsystem. This builds on the socio-technical perspective of relational subsystems by Bostrom and Heinen (1977). Affordance actualization implies human action which is related to the three components of the trifecta (Figure 7) and dependent on the social and technical subsystems (Figure

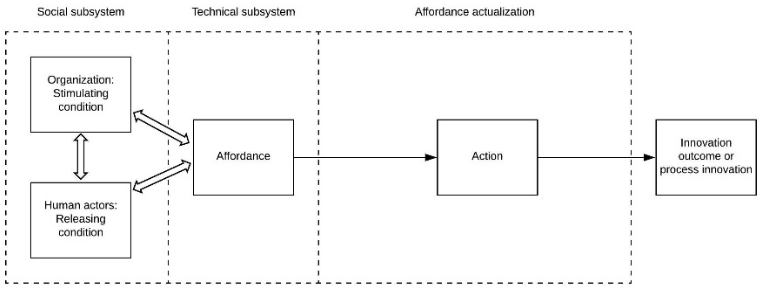

Figure 8. Affordance actualization mechanism

8). This underscores the continued relevance of the socio-technical perspective in IS research (Sarker et al. 2019). Furthermore, this research provides theoretical and empirical insights into the persistent problems and practices within information systems development by applying a socio-technical perspective on human action, the environment, and structures as documented by Kautz et al. (2007).

Through our theorizing efforts, we are able to show how video game development is influenced by the organizational context through stimulating conditions, and how human actors' decisions to act (cf., releasing conditions) trigger processes of affordance actualization. Depending on the type of outcome (innovation outcome or process innovation), the affordance actualization is mediated by resources in the form of developers' knowledge and abilities, organizational culture and physical work space, and technology features and standards. The presented models show that constraints not only hinder affordance actualization but may become stimulating conditions themselves, triggering affordance actualization and as a result lead to process innovation. The trifecta (Figure 7) and Figure 8 further our understanding of affordances from a socio-technical perspective by applying affordance analysis to a video game development company and by providing an empirical account of the actions and relations of the developers. It also shows how significant the trifecta, as a representation of the socio-technical system, is to our understanding of the stimulating and releasing conditions in a system of actors, technology, and an organization. This adds to state-of-the-art knowledge of affordance actualization by theorizing and providing new insights into the way human actors, the organization, and technology influence change and result in digital innovation through affordance actualization (Leonardi and Barley 2008; Strong et al. 2014). In doing so, we provide practical insights to managers attempting to effect change by illustrating the important dependencies between human actors, the organization, and technology in understanding affordances and actualizing them.

6 Conclusion

In this paper, we present a case study of a Norwegian video game development company and describe how developers engage in digital innovation through the use of digital technology. We identify four affordances—*tool development*, *prototyping*, *user testing*, and *patching*—and theorize the associated actualization processes through four separate models. These models directly respond to our research question: *How does the actualization of affordances in video game development influence the innovation process and outcome?*

By identifying the four affordances, describing the actualization processes, and theorizing the affordance actualization mechanism, we provide insights into the possibilities of developers to use and develop digital technology for innovation purposes. We highlight the important relationship between human actors, the organization, and technology, and discuss how affordances are actualized and constraints leveraged, resulting in digital innovation both in terms of outcome (product innovation) and process (process innovation) of video game development.

The presented models provide guidance to practitioners engaged in digital innovation by identifying affordances and theorizing their actualization, bringing awareness to the stimulating and releasing conditions of affordance actualization. The trifecta (Figure 7) and affordance actualization mechanism (Figure 8) highlight the importance of and interdependency between human actors, the organization, and technology, and both illustrate that technology mediates the conditions stimulating and releasing affordance actualization. Practitioners face a plethora of challenges associated with a competitive environment and innovation processes that are constantly being pushed by technological development as well as ever-growing expectations from users and platform distributors. The contributions of this paper help practitioners face these challenges and use their personal, organizational, and technological resources to their advantage in the pursuit of digital innovation. This study is limited by virtue of being a single case study of a small video game development company. Despite its limitations, we have generalized from our empirical account to theoretical statements. We encourage, however, other researchers to follow in our footsteps and investigate the extent to which these and other affordances influence digital innovation efforts in other contexts, identifying and comparing processes of affordance actualization for the purpose of theorizing universally valid mechanisms. We also recommend further research on how relationships between organizational structures (flat, hierarchy etc.) and technology as "generative digital artefacts" influence innovation outcomes.

References

Anderson, C., and Robey, D., (2017). Affordance potency: Explaining the actualization of technology affordances. *Information and Organization,* (27:2): 100-115.

Arnold, M., (2003). On the Phenomenology of Technology: The "Janus-Faces" of Mobile Phones. *Information and Organization*, (13:4): 231-256.

Bostrom, R., and Heinen, J., (1977). MIS Problems and Failures: A Socio-Technical Perspective. Part I: The Causes. *MIS Quarterly*, (1:3): 17-32.

Boudreau, K., (2010). Open platform strategies and innovation: Granting access vs. devolving control. *Management Science,* (56:10): 1849-1872.

Bygstad, B., Munkvold, E. B., and Volkoff, O., (2016). Identifying generative mechanisms through affordances: a framework for critical realist data analysis. *Journal of Information Technology,* (31:1)**:** 83-96.

Eaton, B., Elaluf-Calderwood, S., Sorensen, C., and Yoo, Y., (2015). Distributed tuning of boundary resources: the case of Apple's iOS service system. *Mis Quarterly,* (39:1)**:** 217-243.

Fichman, R. G., Dos Santos, B. L., and Zheng, Z., (2014). Digital Innovation as a Fundamental and Powerful Concept in the Information Systems Curriculum. *Mis Quarterly,* (38:2): 329-343.

Gaver, W. W., (1991). Technology affordances. In: *Proceedings of the SIGCHI conference on Human factors in computing systems,* pp. 79-84.

Ghazawneh, A., and Henfridsson, O., (2013). Balancing platform control and external contribution in third-party development: the boundary resources model. *Information Systems Journal,* (23:2): 173-192.

Gibson, J. J., (1979). *The ecological approach to visual perception*, Houghton Mifflin.

Henfridsson, O., and Yoo, Y., (2013). The liminality of trajectory shifts in institutional entrepreneurship. *Organization Science,* (25:3): 932-950.

Henfridsson, O., Yoo, Y., and Svahn, F., (2009). Path creation in digital innovation: A multi-layered dialectics perspective. *Association for Information Systems,* (9:20).

Kanode, C. M., and Haddad, H. M., (2009). Software Engineering Challenges in Game Development. In: *Proceedings of the 2009 Sixth International Conference on Information Technology: New Generations.* IEEE Computer Society, pp. 260-265.

Kautz, K., Madsen, S., and Nørbjerg, J., (2007). Persistent problems and practices in information systems development. *Information Systems Journal*, (17:3): 217-239.

Leonardi, P. M., and Barley, S. R., (2008). Materiality and change: Challenges to building better theory about technology and organizing. *Information and organization,* (18:3): 159-176.

Majchrzak, A., and Markus, L., (2012). *Technology affordances and constraint theory of MIS*. Sage, Thousand Oaks, CA.

Miles, M. B., and Huberman, A. M., (1994). *Qualitative data analysis: an expanded sourcebook,* Thousand Oaks, Calif.; London, Sage.

Murphy-Hill, E., Zimmermann, T., and Nagappan, N., (2014). Cowboys, ankle sprains, and keepers of quality: How is video game development different from software development? In: *Proceedings of the 36th International Conference on Software Engineering*. ACM, pp. 1-11.

Nandhakumar, J., Panourgias, N. S., and Scarbrough, H., (2013). From Knowing It to "Getting It": Envisioning Practices in Computer Games Development. *Information Systems Research,* (24:4): 933-955.

Orlikowski, W. J., (1992). The duality of technology: Rethinking the concept of technology in organizations. *Organization science*, (3:3): 398-427.

Orlikowski, W. J., and Iacono, C. S., (2001). Research Commentary: Desperately Seeking the "IT" in IT Research--A Call to Theorizing the IT Artifact. *Information Systems Research,* (12:2): 121-134.

Orlikowski, W. J., and Scott, S. V., (2008). 10 sociomateriality: challenging the separation of technology, work and organization. *The academy of management annals,* (2:1): 433-474.

Parnas, D. L., and Clements, P. C., (1986). A rational design process: How and why to fake it. *IEEE transactions on software engineering*, (12:2): 251-257.

Sánchez, J. L. G., Zea, N. P., and Gutiérrez, F. L., (2009) From usability to playability: Introduction to player-centred video game development process. *International Conference on Human Centered Design.* Springer, pp. 65-74.

Sarker, S., Chatterjee, S., and Xiao, X., (2013). How "Sociotechnical" is our IS research? An assessment and possible ways forward. In: *ICIS 2013: International Conference on Information Systems*, *Reshaping Society Through Information Systems Design*, pp. 1812-1835.

Sarker, S., Chatterjee, S., Xiao, X., and Elbanna, A., (2019). The sociotechnical axis of cohesion for the IS discipline: Its historical legacy and its continued relevance. *MIS Quarterly*, (43:3): 695-720.

Sayer, A., (1992). *Method in Social Science: A Realist Approach*, Routledge.

Schreier, J., (2017). *Blood, Sweat, and Pixels: The Triumphant, Turbulent Stories Behind How Video Games Are Made*, HarperCollins.

Stacey, P., and Nandhakumar, J., (2009). A temporal perspective of the computer game development process. *Information Systems Journal,* (19:5): 479-497.

Stoneman, P., (2010). *Soft Innovation: Economics, Product Aesthetics, and the Creative Industries*, OUP Oxford.

Strong, D. M., Johnson, S. A., Tulu, B., Trudel, J., Volkoff, O., Pelletier, L. R., Bar-On, I., and Garber, L., (2014). A theory of organization-EHR affordance actualization. *Journal of the Association for Information Systems,* (15:2): Article 2.

Tilson, D., Lyytinen, K., and Sørensen, C., (2010). Research commentary-digital infrastructures: the missing IS research agenda. *Information systems research,* (21:4): 748-759.

Um, S., and Yoo, Y., (2016). The Co-Evolution of Digital Ecosystems. In: *ICIS Preceedings*. 11, pp. 1-15. AIS eLibrary.

Van Vugt, H. C., Hoorn, J. F., Konijn, E. A., and De Bie Dimitriadou, A., (2006). Affective affordances: improving interface character engagement through interaction. *International Journal of Human-Computer Studies,* (64:9): 874-888.

Volkoff, O., and Strong, D. M., (2013). Critical Realism and Affordances: Theorizing IT-associated Organizational Change Processes. *Mis Quarterly,* (37:3): 819-834.

Volkoff, O., and Strong, D. M., (2017). Affordance theory and how to use it in IS research. *The Routledge Companion to Management Information System.* New York: Routledge.

Xia, F., Yang, L. T., Wang, L., and Vinel, A., (2012). Internet of things. *International Journal of Communication Systems,* (25:9): 1101-1241.

Yin, R. K., (2011). *Applications of case study research*, Sage.

Yoffie, D. B., (1996). Competing in the age of digital convergence. *California Management Review,* (38:4): 31-53.

Yoo, Y., (2012). The Tables Have Turned: How Can the Information Systems Field Contribute to Technology and Innovation Management Research? *Journal of the Association for Information Systems,* (14:5).

Yoo, Y., Boland Jr, R. J., Lyytinen, K., and Majchrzak, A., (2012). Organizing for innovation in the digitized world. *Organization Science,* (23:5): 1398-1408.

Yoo, Y., Henfridsson, O., and Lyytinen, K., (2010). Research commentary-The new organizing logic of digital innovation: An agenda for information systems research. *Information systems research,* (21:4): 724-735.

Zammuto, R. F., Griffith, T. L., Majchrzak, A., Dougherty, D. J., and Faraj, S., (2007). Information technology and the changing fabric of organization. *Organization science,* (18:5): 749-762.

Zittrain, J., (2008). *The future of the internet--and how to stop it*, Yale University Press.