Software Quality Journal
https://doi.org/10.1007/s11219-024-09696-y

RESEARCH

Towards effective gamification of existing systems: method and experience report

Anderson Uchôa[1] · Rafael de Mello[2] · Jairo Souza[3] · Leopoldo Teixeira[3] · Baldoino Fonseca[4] · Alessandro Garcia[5]

Accepted: 19 August 2024

Abstract

Gamification promotes user engagement with software features through the incorporation of game elements and rules. Gamification is often incorporated a *posteriori* into already existing systems. Success in gamifying an existing system depends on careful planning and the evolution of its previously produced software artifacts. This is particularly true for artifacts produced in the earliest development phases, such as the requirements specification. Incorporating game elements and rules into an existing system is far from trivial. Developers eventually struggle with performing certain development activities, such as evolving existing requirements and selecting game elements and game rules. This paper reports our practical experience in gamifying an existing system. Based on this experience, we introduce Gamify4Fun, a method that aims to assist developers in performing some key activities to gamify existing systems. We built Gamify4Fun based on the experience of 15 developers involved with the gamification of an existing healthcare system. We started by adopting an original method aimed at gamifying systems being built from scratch. As we needed to adapt the original method for supporting the particularities of gamifying a previously developed system, we refined the original method's development activities and their respective phases. We also interviewed the developers to capture their perception of challenging development activities through gamification. The interviews' outcomes guided some further refinements to the original method. Gamify4Fun supports the gamification of existing systems at the earliest development phases: from the preparation of the system gamification to both the gamification and system design. We refined the development activities of the original method as much as needed; we also used or adapted the activities prescribed by other methods from the literature, in the context of gamification from scratch, to fill gaps whenever necessary. By reporting our practical experience and introducing a gamification method, we expect to guide development teams in gamifying their existing systems, as well as shed insights about the current, unaddressed limitations of existing approaches (including ours) to gamifying existing systems.

Keywords Software gamification · Development method · Experience report · Interview-based

Extended author information available on the last page of the article

Published online: 17 September 2024

1 Introduction

Gamification aims at engaging users through the incorporation of game elements and rules in systems (Pedreira et al., 2015; Zichermann & Cunningham, 2011). These game elements and rules may considerably vary in complexity and coverage. For instance, *punctuation* is a game element largely applied in gamified systems. It addresses providing a numerical reward for the system user after accomplishing tasks (Deterding et al., 2011; Zichermann & Cunningham, 2011). The punctuation given to the user may vary according to the rules reported in the points assignment policy of the system. Among others, this policy establishes how many points should be assigned to the user according to the task (Zichermann & Cunningham, 2011).

Gamification can be made *from scratch*, i.e., since the early stages of new systems development. In other words, there are systems originally designed to provide a gamified experience to their users, despite this is not their main purpose. For instance, *Stack Overflow* is a popular platform for sharing knowledge among software developers (Hanlon, 2013). The gaming rules of Stack Overflow include assigning points to developers according to the level of their interaction on the platform. Through Stack Overflow, developers may publish questions, report answers, and make comments. They may also evaluate the quality of the published content, influencing the developers' punctuation. The adoption of policies for assigning and accumulating points through interaction, as well as user ranking, are recognized resources for promoting user engagement in different systems (Bosu et al., 2013; Jin et al., 2015).

Besides the current large availability of systems gamified from scratch, it has been increasing the incidence of gamifying already existing systems (Uchôa et al., 2019). Gamifying existing systems has been a major trend in contexts in which recurrent user interaction is a factor of success (Schacht & Schacht, 2012; Swacha, 2016; Bucchiarone et al., 2020).

Refining existing software artifacts such as requirements and design documentation for introducing game elements and rules may be quite challenging (Uchôa et al., 2019; Fernandes et al., 2019; Dubois & Tamburrelli, 2013) without proper guidance. For instance, developers have to clearly understand the development activities required for picking up the most appropriate game elements and rules for addressing the goals and constraints of the existing system. Otherwise, the effort spent by developers may be unnecessarily increased, resulting in a considerable rework (Fernandes et al., 2019; Uchôa et al., 2019).

We experienced this undesired scenario during the project conducted for gamifying VazaDengue (Sousa et al., 2018), an existing healthcare system launched in 2016 for the prevention and combat of mosquito-borne diseases. VazaDengue requires the engagement of citizens to report issues addressing mosquito-borne diseases. During the VazaDengue project, we perceived that certain development activities became challenging due to the lack of support from a systematic and comprehensive gamification process (Fernandes et al., 2019; Uchôa et al., 2019). For instance, the developers reported considerable difficulty in (i) refining the functional and non-functional requirements previously elicited for the existing system, and (ii) identifying the existing technological and architectural constraints restricting the incorporation of certain game elements and rules.

Although technical literature provides several guidelines (Herzig et al., 2012; Kardan & Arani, 2016) and methods (Morschheuser et al., 2017; Rodrigues et al., 2016) for supporting the building of gamified systems from scratch, it is still scarce software engineering support for gamifying existing systems. More specifically, we could not find at the technical literature a comprehensive approach for systematically supporting the entire gamification process of existing systems. This lack of technical support results in uncertainty regarding the distribution and organization of development efforts. As well as we had struggled with gamifying VazaDengue (Uchôa et al., 2019; Fernandes et al., 2019), other development teams may also struggle with gamifying other existing systems. In this sense, we argue that understanding which development activities are considerably more challenging along the gamification of existing systems is an important step to drive future research efforts for guiding developers on gamifying existing systems. The same rationale is reported by Bucchiarone et al. (2020).

This article is an extension substantially different from our previous works (Uchôa et al., 2019; Fernandes et al., 2019), that are limited to introducing and evaluating the gamified healthcare system developed during our research. Therefore, in this extension version, we present the research activities conducted during our particular experience with gamifying the VazaDengue healthcare system. Whenever we observed the need for performing one or another development activity towards the system gamification, we refined the existing method to address those needs. Thus, our practical experience has strongly impacted the method design (i.e., the activities and execution flow that constitute the method), named Gamify4Fun. Our method was derived from a previous work (Morschheuser et al., 2017), hereinafter called Morschheuser's method, which aimed to gamify systems from scratch.

In this context, we carefully designed and performed two complementary studies aiming at understanding: *What needs to be refined in the Morschheuser's method to support the gamification of existing systems?*. To this purpose, we first report the self-observation study that supported the mapping, organization, and description of Gamify4Fun components. Next, among other activities, we interviewed 15 developers working on the gamification of an existing healthcare system. From these interviews, we mapped 22 challenging activities addressing the gamification of the existing systems, mostly strongly related to software gamification itself. Finally, we introduce all Gamify4Fun's phases and activities. We summarize our key contributions as follows.

- An introduction of a *gamification method*, called Gamify4Fun, aimed to guide development teams in gamifying existing systems. Our method helped us to successfully reuse most of the existing software artifacts while gamifying an existing system (Fernandes et al., 2019; Uchôa et al., 2019). We expect that Gamify4Fun, our actionable contribution, may address the increasing demand for gamifying existing systems.
- We reported a *practical experience* with gamifying of an existing healthcare system. We empirically derived a *conceptual mapping* with 22 development activities that became challenging along the system gamification. Especially, we aim to reveal how challenging the gamification of an existing system might be, from the early requirements elicitation to the system implementation and testing.

In summary, our study shed light on the need for leveraging software engineering practices to support software gamification. Especially, the considerable number of activity refinements applied to the existing method suggests the need for supporting the particularities of gamifying existing systems.

The remainder of this article is organized as follows. Section 2 introduces basic concepts aimed to support the understanding of this paper. Section 3 describes our research methodology. Section 4 presents our self-observational study aimed to incrementally refine the method. Section 5 presents the interview-based study from which we identified challenging development activities along the existing system gamification. We used the developer feedback to refine our method whenever possible. Section 6 introduces the Gamify4Fun method. Section 7 discusses related work. Finally, Section 8 concludes the paper and suggests future work.

2 Background

2.1 Basics of software gamification

Software gamification is employed to promote user engagement with key features of systems through game elements, such as points and rankings, and game rules, such as ranking policy (Deterding et al., 2011; Zichermann & Cunningham, 2011). For instance, gamification has also been successfully employed for boosting people's consumption in market segments (Zichermann & Linder, 2010), engaging students with learning tasks Muntean 2011, and promoting people's well-being and health (Pereira et al., 2014). Gamification has also been applied to support software development tasks (Pedreira et al., 2015) by promoting the completion of eventually repetitive and time-consuming tasks such as writing software test cases (Rojas et al., 2017). We discuss the key elements of any gamified system as follows.

Game elements They represent the basic components of any gamified system. Table 1 list some game elements discussed by previous work (Deterding et al., 2011; Zichermann & Cunningham, 2011). The choice of game elements that help to address the gamification goals will define the possibilities of user interactions with the gamified system (Deterding et al., 2011). For instance, assigning points to users directly rewards user interaction and promotes further interactions in the future. Additionally, it might create a competitive environment among users, to achieve the highest scores in points.

Table 1 Common game elements (Zichermann & Cunningham, 2011; Deterding et al., 2011)

Game elements	Characteristics
Avatar: A visual representation of the system user (Deterding et al., 2011; Zichermann & Cunningham, 2011)	It makes the system user more immerse into the system.
Badge: Special reward earned by users (Deterding et al., 2011; Zichermann & Cunningham, 2011)	It aims at recognizing the user skills as the user interacts with the system.
Challenge: Set of tasks with a common purpose (Deterding et al., 2011; Zichermann & Cunningham, 2011)	It represents a set of actions that the system user should perform to earn more rewards than those provided by a single task.
Level: A control of the user progress (Deterding et al., 2011; Zichermann & Cunningham, 2011)	The level is usually counted in terms of earned points and badges. Levels can control how certain system features are unlocked for the users to interact with.
Ranking: A sorted list of users (Deterding et al., 2011; Zichermann & Cunningham, 2011)	It provides a general view of the system users' progress, which is usually computed based on their levels and points. It aims to promote competition among users.
Point: A unit of reward earned by users (Deterding et al., 2011; Zichermann & Cunningham, 2011)	It usually reflects the user progress in the system. Points are commonly assigned to users after completing tasks.
Task: An atomic user action (Deterding et al., 2011; Zichermann & Cunningham, 2011)	It represents a specific type of user action, where each task should have a clear goal and reward the system users whenever completed successfully.
Team: Group of users with a common purpose (Deterding et al., 2011; Zichermann & Cunningham, 2011)	It aims at promoting a user group engagement with the system. Teams usually promote competition and cooperation among users.

Game rules The game rules aim to define the interactions between people and game elements towards people engagement. There are two types of game rules as follows. The first type of rule defines the *relations between the system and its users* (e.g., points and rankings). Let us exemplify these rule types via points and rankings. For instance, in Stack Overflow the system user earns *points* by performing several tasks, such as asking, answering, commenting, and voting questions. These task awards aim at keeping the user engaged with the system. Thus, the system should acknowledge the user with points that are accumulated in a point count. The second type of rule defines the *relations between game elements*. Based on the previous example, the points assigned to a user who had performed specific tasks are aggregated to establish the user ranking. In this case, there is a relation between points and ranking.

2.2 Existing methods for gamification from scratch

We have found previous work (Kardan & Arani, 2016; Morschheuser et al., 2018; Rodrigues et al., 2016) introducing methods for building gamified systems. Unfortunately, none of these address the particularities involved in gamifying existing systems. Previous studies target the general development phases concerning building gamified systems from scratch.

Kardan and Arani (2016) presented a simplistic method for guiding gamification from scratch. This method is composed of four development activities: *define targets and organization needs*; *define the priority of targets and needs*; *define game rules*, and *determine the user scores and awards for different targets*. These activities target the *preparation for the system gamification*, the *user analysis*, and the *gamification design*. The comprehensiveness of this method is limited since it covers only a few development phases. Thus, we decided to use this method as a starting point for the specific needs of gamifying existing systems.

Rodrigues et al. (2016) introduced a more robust method for gamifying systems from scratch. While the previous method (Kardan & Arani, 2016) prescribed fine-grained development activities, this method prescribes development phases that include one or more development activities. The development phases are five: *business objectives definition*; *game model and characteristics definition*; *methodology and tools of software development*; *game design and software development*; and *gamification quality control and feedback*. This method covers the system gamification from the earliest development phases, from requirements elicitation to the system implementation. Nevertheless, the method targets the particularities of Web development. More critically, the guidelines provided are strongly associated with the e-business software domain. Thus, this method is quite challenging to use this method to gamify other kinds of systems. We reused as much knowledge as possible to build our method.

Morschheuser et al. (2017) described the most comprehensive method for gamifying systems from scratch. This method has eight well-defined development phases: *project preparation*, *context analysis*, *user analysis*, *idealization, design*, *implementation*, *evaluation*, and *monitoring*. Contrasting with previous studies (Kardan & Arani, 2016; Rodrigues et al., 2016), Morschheuser's method is largely documented, having its phases clearly described through development activities. Differently from Rodrigues et al. (2016), this method includes the *user analysis* aimed at profiling users and their needs. However, it does not have a specific phase for selecting game elements and rules for the system, which is a fundamental step in the gamification process.

3 Research methodology

When we decided on the gamification of the existing system, we did not find appropriate methods for supporting us in gamifying an existing software system. Thus, we initially relied on a method to support building gamified systems from scratch (Morschheuser et al., 2017) as the first guidance. We discuss this method in detail in Section 7. After a while, we identified the need to refine this method to address the peculiarities of gamifying existing systems. This resulted in Gamify4fun, a method for supporting the gamification of existing systems. To build Gamify4Fun, we followed the research methodology presented in this section.

3.1 Study goal and research questions

We defined our study goal as follows (Wohlin et al., 2012): *analyze* an existing method for gamifying systems from scratch (Morschheuser et al., 2018), named Morschheuser's method; *for the purpose of* refining this method in order to guide developers in gamifying existing systems; *with respect to* the key development phases and activities for gamifying an existing system; *from the perspective of* software engineering researchers and developers; *in the context of* the gamification of an existing healthcare software system. We list our research questions (RQs) as follows:

RQ$_1$: *Which activities need to be refined to support the gamification of existing systems?* – Through **RQ**$_1$, we aim to identify the activities from Morschheuser's method (Morschheuser et al., 2018) that should be refined to support the gamification of existing systems. We consider four refinement types: discard, full reuse, adaptation, and the addition of an activity not yet supported by the method. Each development *activity* corresponds to part of the software gamification process. For answering **RQ**$_1$, we designed a self-observational study (Emerson et al., 2001), named Study 1, to promote discussions on the activities currently supported by Morschheuser's method and the required refinements (further details are given in Section 4).

RQ$_2$: *What are the development activities perceived as challenging by the developers along the system gamification?* – With **RQ**$_2$. we aim to understand the feasibility of the method. In other words, we are concerned with the extent to which the development teams can follow our method without major issues that hinder the software gamification process. By understanding which software development activities were challenging to perform along the gamification process of the existing system, we can track whether factors associated with the challenging activities are related to the gamification method. *Challenging activities* are those activities considered hard to be successfully performed by the developers. It may be the case that activities tend to become challenging for reasons that extrapolate the followed method, e.g., the lack of supporting tools to perform key activities such as gamification design conformance. For answering **RQ**$_2$, we conducted a second study, performing semi-structured interviews (Wohlin et al., 2012) with developers from the gamification teams (further details in Section 5).

3.2 Study procedures

We conducted two complementary studies for building the first version of Gamify4Fun. Figure 1 shows an overview of how these studies complement one another with respect to their inputs and outputs. The figure also indicates the number of developers involved in each study.

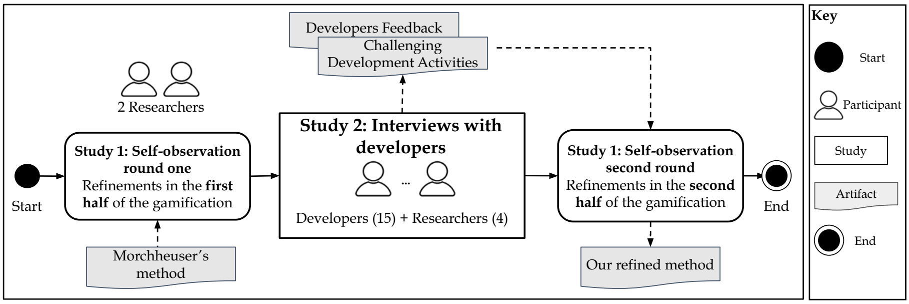

Developers Feedback
Challenging Development Activities
2 Researchers
Study 1: Self-observation round one
Refinements in the first half of the gamification
Start
Morchheuser's method
Study 2: Interviews with developers
Developers (15) + Researchers (4)
Study 1: Self-observation second round
Refinements in the second half of the gamification
End
Our refined method
Key
Start
Participant
Study
Artifact
End

Fig. 1 Overview of the research methodology

The first study consists of a self-observational study (Emerson et al., 2001). In this method, the researcher is both an observer and a participant in some activities. Although well-known and well-used in other disciplines (Emerson et al., 2001), software engineering researchers are only recently taking advantage of this method (Ko, 2017). Such a method has many strengths (e.g., it can provide a unique set of views that could not be perceived from external eyes), however, it also has limitations (e.g., knowledge emerges from a single perspective). To mitigate some of these limitations we also have conducted interviews with developers.

In the first study, we observed how the VazaDengue development team behaved along with the system's gamification and its emerging needs. Based on this knowledge, we identified opportunities for refining Morschheuser's method to overcome the developers' needs. For this purpose, two researchers conducted weekly meetings with the development team to analyze the extent to which Morschheuser's method was appropriate and sufficient for the supporting gamification context, and which refinements could be applied.

The second study consisted of interviewing the VazaDengue developers about the gamification method employed. Four researchers conducted individual interviews with 15 developers. From these, three developers are also collaborating researchers, interviewed by the main researcher. To reduce bias, we opted to not interview the main researcher. The main goal of the interviews is to characterize the development activities that became more challenging along the system gamification. Thus, we could identify opportunities for refining the gamification method towards a more comprehensive and lightweight one. Based on the content gathered from the interviews, we expect to derive a gamification method tailored for introducing gamification in existing systems. The main artifacts involved in our study are available at Uchôa et al. (2024), which includes the participant characterization form, and the interview script.

3.3 Studies context

The development of a gamified version of the VazaDengue system was the goal of an international research project entitled *Leveraging Gamification and Social Networks for Improving Prevention and Control of Zika*. This project was conducted by Software Engineering and Data Analytics researchers from Brazil and the United Kingdom. A total of 25 professionals participated in the project: one project manager; four development team leaders, one per team; 15 software developers, including the team leaders, distributed in two Brazilian cities, each with at least one developer per team, and; seven senior researchers, five from Brazil and two from the UK. The project counted on the active contribution of a dozen Brazilian public health agents, which assisted in many development activities.

Each project member was allocated to one or more of the following teams. *Design Team*, responsible for eliciting the system requirements, conceiving the gamification design, and prototyping both the user interface (UI) and the user experience with the system (UX). *Implementation Team*, responsible for programming the back-end and front-end layers of the web and mobile applications that constitute the gamified system. This team was also responsible for implementing the specific layer that supports the system gamification. *Testing Team*, responsible for programming and running unit test cases, besides conducting interface testing. *Research Team*, responsible for conducting academic research on gamification and data analytics applied to healthcare.

4 Study 1: A self-observation of the refined method

We describe the refinements applied to Morschheuser's method (Morschheuser et al., 2017) as follows. Two researchers with practical experience in Software Engineering (especially in the management of small development teams) have contributed to a retrospective analysis aimed at tracking these refinements along the gamification of the existing system.

4.1 Study steps

Steps 1 to 4 were performed during each development phase of the existing system gamification project. Step 5 was designed to be performed at the end of the system gamification. We describe each step as follows.

Step 1: Reasoning about limitations of the existing method Two researchers reasoned about Morschheuser's method to characterize to what extent each prescribed phase can support the gamification of existing systems. For this purpose, we initially observed the development context of our existing system. As aforementioned, the existing system was gamified by developers organized in a small and distributed development team. Based on our development context, we have discarded certain activities that do not fit the gamification of existing systems. After that, we grouped the activities that could be adapted and reused to support the gamification of existing systems. Finally, we identified which activities were missing and, therefore, should be added to properly support the gamification of existing systems.

Step 2: Collect feedback from developers and researchers We have conducted some meetings, face to face as far as possible, with current and former VazaDengue developers, as well as with researchers involved during the gamification of the existing system. We aimed to capture the developers' needs that could be better addressed by the gamification method.

Step 3: Analyze feedback and refine the method if necessary Based on the collected feedback, we were able to track which activities of Morschheuser's method needed some type of refinement, i.e., adaptation, reuse, or discard. We were also able to identify opportunities for adding key activities to gamify existing systems. For this purpose, two researchers held several meetings aimed at validating the necessary refinements of Morschheuser's method. Thus, for each activity included in the refined method, we minimized biases and reached a consensus with a pair of researchers.

Step 4: Document the applied refinements via technical report We have documented the applied refinements in four technical reports encompassing the main development phases, i.e., design, development, and testing. Each report describes the process and activities performed by developers along the existing system gamification. These reports also describe the main results obtained from each performed activity. Additionally, along with the system gamification, we managed change requests (e.g., the addition of new features and bug fixes) through issues reported via GitHub. These issues contain information about architectural design and interface design decisions.

Step 5: Formalize the evolution history Two researchers have formalized the evolution history of the new gamification method based on milestones (Tausworthe, 1979). The milestones represent a clear sequence of activities or events that incrementally build up until a specific goal is complete (Tausworthe, 1979). We have taken each development phase of the new method as a milestone, since we considered each phase a specific goal. For each phase, we classified the respective activities as adapted (ADP), reused (REU), added (ADD), discarded (DIS), or merged (MER). When pertinent, we preserved the activity names assigned by the Morschheuser's method (Morschheuser et al., 2017). An example of a classified activity is *(ADP) Identify and List Objectives*. With this, we could define a systematic representation of how we incrementally refined the gamification method throughout the development phases.

4.2 Results and discussions

We discuss the main results of **RQ**$_1$ regarding the refinements applied as follows.

System preparation Figure 2 illustrates the refinements applied to the System Preparation Phase. During this phase, we observe the need to adapt two activities of Morschheuser's method. We also rely on the knowledge gained from previous studies (Kardan & Arani, 2016; Rodrigues et al., 2016) concerning the definition of the objectives. Two activities required adaptation: *(ADP) Identify and List Objectives* and *(ADP) Rank Objectives*. The original definition of these activities did not guide the reasoning about gamification goals based on existing system goals. When gamifying systems from scratch, goals are elicited from the expectations of new users. Differently, when gamifying existing systems, it becomes

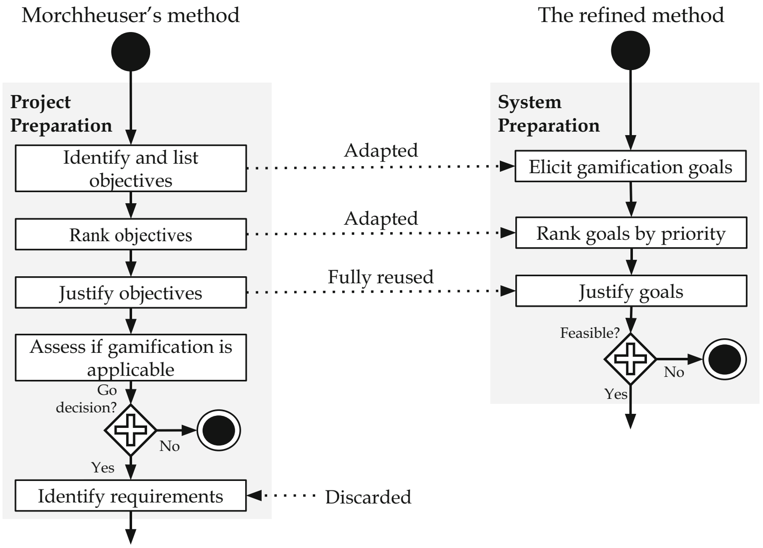


Fig. 2 Refinements applied in system preparation phase

necessary to discuss to what extent the system gamification impacts the existing system goals. By overlooking these goals, gamification may lead existing systems' users to abandon it.

In addition, we decided to split the first activity (*Identify and List Objectives*) into two more specific activities, i.e., elicit gamification goals and rank goals by priority. We reused the *(REU) Justify Objectives* activity. In the earliest gamification phases, we decided not to elicit requirements for the gamifying systems, since we did not consider appropriate eliciting requirements during the systems preparation phase, therefore we discarded *(DIS) Identify Requirements*. Alternatively, we conducted workshops with public health agents to understand their needs and understand how gamification could help address these needs. In addition, we have decided to discuss the profiles of potential system users before eliciting the requirements. Thus, we decided to discard this activity at this point, but we considered reusing this activity in future development phases.

User analysis Figure 3 illustrates the refinements applied to the User Analysis Phase. In this phase, we identified the need to adapt all activities of Morschheuser's method. We also relied on knowledge gathered from a previous study (Kardan & Arani, 2016) on the definition of target users, their needs, and their motivations. We have adapted *(ADP) Identify User Motivations*, *(ADP) Identify User Needs*, *(ADP) Define Target Users*, and *(ADP) Create Personas*. The original definition of these activities did not guide the reasoning about the expectations and needs of existing system users. When gamifying systems from scratch, this reasoning is unnecessary, since there is no existing system to use as a basis, hence no users. Conversely, when gamifying existing systems, it is important to consider the current users of the existing system. Otherwise, these users can eventually leave the system because their needs are not properly addressed anymore. Therefore, we decided to adapt all four activities to include missing reasoning. We highlight that *(ADP) Identify User Motivations* and *(ADP) Identify User Needs* were merged into a single activity named *List User Needs and Motivations*.

Context analysis Figure 4 illustrates the refinements applied to the Context Analysis Phase. We also rely on the knowledge gathered from a previous study (Rodrigues et al., 2016) on methodology and software development tools to be employed in the systems' gamification. During this phase, we decided to fully reuse two activities, add two new ones, and discard

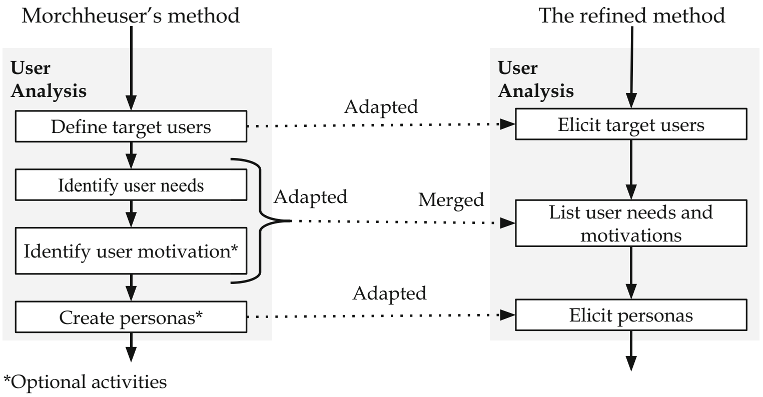


Fig. 3 Refinements applied in user analysis phase

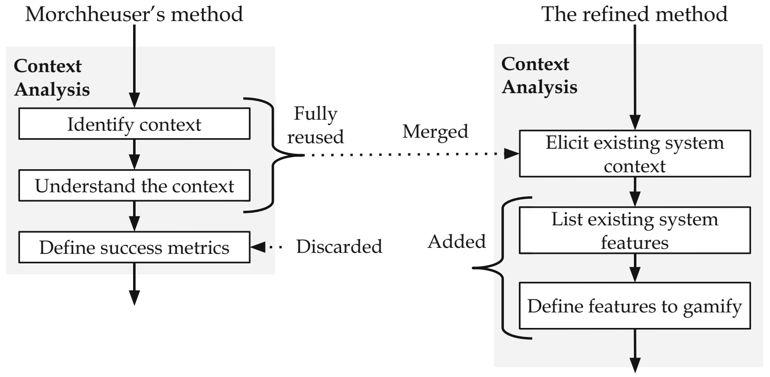


Fig. 4 Refinements applied in context analysis phase

another. We reuse *(REU) Identify the Context* and *(REU) Understand the Context*. Eliciting and understanding context information, e.g., employed technologies and design decisions, is important when performing gamification of a system, whether from scratch or from an existing one. Therefore, we decided to fully reuse both activities. We have been motivated by the need to characterize the limitations of an existing system regarding planned gamification. Gamifying an existing system requires understanding to what extent the incorporation of game elements and rules will affect the current development context. We highlight that we merged both activities into a single one called *Elicit Existing System Context*.

After performing the aforementioned activities, we identified the need to add a new activity to support the elicitation of existing system features, thus we added *(ADD) List Existing System Features*. In our particular case, eliciting the existing system features was necessary, since the system documentation was scarce, and most of the developers involved in the system gamification were not familiar with the existing implementation and system features. In addition, we decided to add a specific activity to support defining which of the existing system features should be gamified, by adding *(ADD) Define Features to Gamify*. We added this activity to discuss the system features that succeeded or not in their purpose. The decision to gamify a system feature strongly depends on the gamification goals defined in the System Preparation phase. Finally, we decided to discard *(DIS) Define Success Metrics*, because we do not consider it a priority. In addition, the system had not been deployed yet, making the computation of such metrics unfeasible.

Requirements elicitation After performing the context analysis phase, we identified the need to revisit the existing system requirements concerning system features that could be gamified. However, Morschheuser's method does not have a specific phase to guide developers to evolve the existing system requirements to incorporate gamification. Along with the conception of this phase, we rely on the knowledge gathered from an empirical study (Rodrigues et al., 2016) on game model definition and systems' characteristics.

This knowledge helps us to specify gamification-specific requirements. We explain the refinements applied as follows. Adapting *(ADP) Identify Requirements* – We adapted this activity from the *System Preparation* phase by transforming it into a new phase to guide the evolution of existing system functional and non-functional requirements to achieve the gamification goals. We called this new phase of *Requirements Elicitation* phase – We have designed three activities to compose this phase – *(ADD) Evolve Functional Requirements*, *(ADD) Evolve Non-Functional Requirements*, and *(ADD) Elicit Gamified Requirements*. In our case, we need to perform these three activities, since the evolution of certain existing functional requirements has affected some of the system's business rules. In addition, some existing non-functional requirements have become more critical with the incorporation of gamification, such as performance and usability.

Gamification design During this phase, we were inspired by discussions provided by the original method on the difficulty of designing a gamified system. We were also inspired by knowledge obtained from a previous study (Kardan & Arani, 2016) concerning the definition of game rules, scores, and awards for the target users. Then, we decided to establish a specific phase for guiding developers in conceiving the gamification design. Due to our lack of expertise in gamification design, we felt the need to systematically establish the game elements and rules that could be incorporated into our existing system. For this purpose, we have added a new phase, called *Gamification Design*. We have designed four activities to compose this new phase, namely *(ADD) Elicit Game Elements Elsewhere*, *(ADD) Pick Useful Game Elements*, *(ADD) Define Game Rules*, and *(ADD) Create the Visual Representation for the Conceptual Model*. Our major goal was guiding the gamification design as a whole, especially in the case of developers not familiar with the definition of game elements, rules, and conceptual models. We provide more details about each added activity in Section 6, while introducing the final version of our gamification method.

Software design Figure 5 illustrates the refinements applied to the Software Design Phase. During this phase, we adapted four activities and discarded two ones. To refine this phase, we used the knowledge gathered from a previous study (Rodrigues et al., 2016) concerning the control of quality and feedback about gamification. For instance, we also defined activity to ensure the control of the quality of the defined prototype through inspection tasks. Adapting

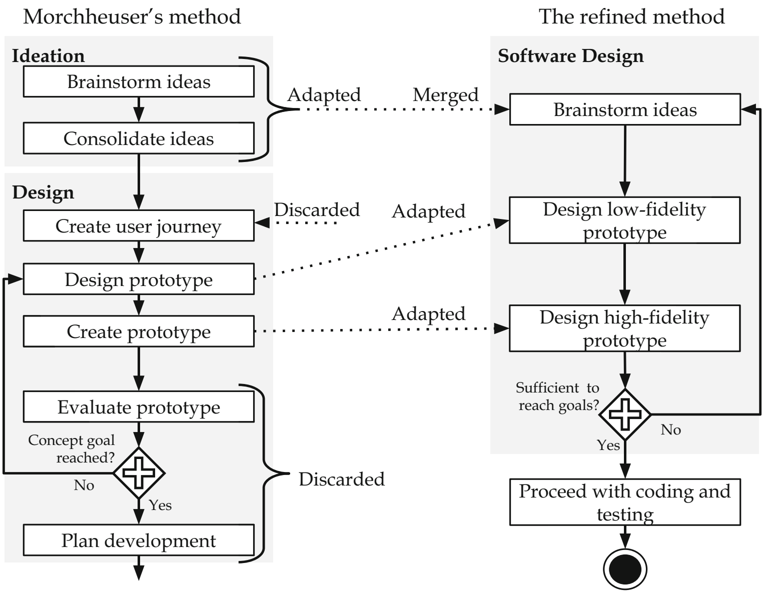


Fig. 5 Refinements applied in software design phase

(ADP) Brainstorming Ideas and *(ADP) Consolidate Ideas*, we changed the original purpose of both activities to focus on discussions regarding the system's visual aesthetics. This phase aims to guide fruitful discussions about (i) visual aspects that worked fine for the existing system and, therefore, one could reuse in the gamified system version; and (ii) what should be modified to leverage the system's attractiveness and enjoyability.

We discarded *(DIS) Create User Journey* since the user journey was informally described during the brainstorming activity. We have adapted *(ADP) Design Prototype* and *(ADP) Create Prototype* to consider the user interface of the existing system, e.g., we have evaluated the visual aesthetics elements that might be reused from the existing system, before creating a new prototype. In addition, we have discarded *(DIS) Evaluating the Prototype*, because we perform evaluations along with the software design. We highlight that *(ADP) Brainstorming Ideas* and *(ADP) Consolidate Ideas* were merged into a single activity called *Brainstorming Ideas*.

In the end, we observed the need to apply much more refinements to the original method than we expected. In fact, during the gamification process of an existing system, most existing artifacts (e.g., requirements documentation) change with the incorporation of game elements and rules. Particularly, the system design required full attention to refinements, intending to reduce the effort spent building a design from scratch. It was surprising that Morschheuser's method (Morschheuser et al., 2017) lacked support for selecting game elements and rules. Both resources are essential to the success of the system gamification, regardless of doing it from scratch or from an existing system.

Summary of RQ$_1$. Many activities of the original method, proposed by Morschheuser et al. (2017), required careful refinement to support the gamification of existing systems. Aimed at comprehensive support, we added additional development activities from insights provided by previous studies (Rodrigues et al., 2016; Kardan & Arani, 2016).

4.3 Threats to validity

Construct validity We have carefully followed the observational research procedures (Emerson et al., 2001) to document the applied refinements via technical reports. We have also conducted biweekly follow-up meetings and workshops with the developers during the gamification of our existing system. Thus, we expect to avoid the lack of observation of the researcher involved in the study. However, some observations may not have been captured during the execution of the activities, being omitted during the meetings. We mitigated this threat through our interview-based study, in which we collected the developers' feedback to refine the gamification method whenever possible.

Internal validity We strictly followed the study procedures to analyze Morschheuser's method, conducting the retrospective analysis to build the evolution history map of the existing method. We performed this activity in pairs to reduce the researcher bias in building the evolution map.

Conclusion validity We carefully performed the analysis and refinement of Morschheuser's method. Two researchers validated together all steps following our methodology. Thus, we expect to avoid providing a wrong understanding of how Morschheuser's method evolved.

External validity We have followed the observational research and counted on the developers' feedback along the existing system gamification to refine Morschheuser's method. Best practices documented by the literature were employed whenever possible to incorporate gamification. Particularly, our method is independent of the implementation and testing practices – its focus on gamification design makes it applicable to organizations that employ varied development practices. Additionally, our method is based on the feedback of real developers and practical experience. Thus, we expect that the refined method applies to similar development contexts – i.e., small to medium-sized development teams with the support of agile practices, which is a development context typically observed in companies and startups (Dybå & Dingsøyr, 2008). This particular threat was also partially mitigated through our interview-based study by capturing as many limitations of our gamification method as possible via the developers' perceptions.

5 Study 2: An interview-based experience report

To answer **RQ**$_2$, we performed an interview-based experience report with developers from each of the development teams of the gamified system, i.e., design, implementation, testing, and research. Our major goal was tracking the development activities that became significantly challenging along the system gamification. In total, we conducted 15 semi-structured interviews (Runeson & Höst, 2009). The interviews were conducted when the system was about 50% completed, and they helped us to refine our method. After interviewing each developer, we applied procedures of Grounded Theory (GT) to further analyze the interviewees' responses.

GT is a qualitative research method that uses a systematic set of procedures to inductively develop a theory about a phenomenon from data (Corbin & Strauss, 2008; Easterbrook et al., 2008). This method often involves unstructured text, such as interview transcripts and field notes. GT is used to understand actions within a substantive area from the perspectives of the actors involved in the phenomenon (Glaser, 1998).

GT contains three coding procedures: *open coding*, *axial coding*, and *selective coding*. Coding refers to the task of data analysis (Corbin & Strauss, 2008; Easterbrook et al., 2008). Open coding involves the breakdown, analysis, comparison, conceptualization, and categorization of the data. Axial coding consists of examining the identified categories to establish conceptual relations between them. In this study, we applied only two procedures: *open coding* for identifying topics of interest from the interview data, and *axial coding* for interrelated grouping topics. We did not apply the selective coding procedure since our goal was not to create a theory.

5.1 Study steps and artifacts

Figure 6 illustrates the phases and artifacts of our second study. We describe each study phase as follows.

Phase 1: Interview design. We have designed an interview with developers from each development team of the gamified system. Interviews are an effective way of understanding, through developers' feelings and perceptions, phenomena that are not easily observable (Runeson & Höst, 2009). We have chosen a semi-structured interview design (Runeson

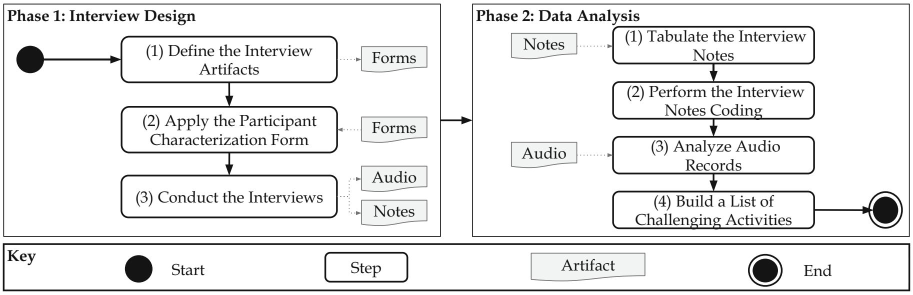

Phase 1: Interview Design
(1) Define the Interview Artifacts
Forms
(2) Apply the Participant Characterization Form
Forms
(3) Conduct the Interviews
Audio
Notes
Phase 2: Data Analysis
Notes
(1) Tabulate the Interview Notes
(2) Perform the Interview Notes Coding
Audio
(3) Analyze Audio Records
(4) Build a List of Challenging Activities
Key
Start
Step
Artifact
End

Fig. 6 Study steps and artifacts of the interview-based study

& Höst, 2009) aimed at allowing high flexibility during the interviews. We explain the three steps conducted to design the interviews as follows.

Step 1 aimed to define and review the following artifacts. The *Participant Characterization Form* aimed to collect the developers' backgrounds. The *Interview Script* follows a funnel structure (Runeson & Höst, 2009) that starts with general questions and ends with specific ones. Our seven-question script has two questions to identify the teams with which the participant collaborated; one question to elicit the perceived challenging activities; and four questions to understand each challenging activity.

Step 2 consisted of inviting all 17 developers from the four gamification teams to fill out this form. We discarded two participants since they contributed to only one development team (design, implementation, testing, and research). Thus, we interviewed 15 developers in the **Step 3**. During the interviews, we strictly followed the script. When possible, we prioritized interviewing the developers through face-to-face meetings in a laboratory environment. In addition to taking notes, we asked for permission to record the interviews (audio). Each interview lasted at most one hour and twenty-five minutes.

Phase 2: Data analysis After interviewing each developer, we analyzed the interview data. Due to the qualitative nature of our study, we have partially relied on the well-known procedures of Grounded Theory (GT) (Corbin & Strauss, 2008; Easterbrook et al., 2008) to analyze the interviewees' responses. Those procedures were useful for both eliciting and understanding each difficulty faced by the developers while gamifying our existing system. We explain the four steps to conduct data analysis as follows.

Step 1 started by tabulating all interviewees' responses into a spreadsheet. Then, we performed pair validation for the tabulation. **Step 2** consisted of coding the interview notes by applying two phases of GT data analysis (Easterbrook et al., 2008) as follows. *Open coding* aimed to elicit the data item about challenging activities. We have identified the data from the tabulated interviews. An example of a challenging activity addressed selecting which game elements will be implemented to engage users. Second, we have labeled the data item with a *code*. Example of code to the aforementioned challenging activity: [GRP03c] Elicit Game Elements aimed to Engage Users. *Axial coding* was applied for labeling groups of challenging activities with *categories*, such as [GRP03] Specify the Gamified Systems.

Step 3 consisted of listening to each audio recording to validate the interview notes, correct inconsistencies, and extract interviewee quotes, which helped to justify why an activity development becomes challenging. In this step, we highlight that the three developers who are also researchers and conducted the other interviews did not listen to their own interview records. This approach was taken to maintain an unbiased perspective and ensure the integrity of the data analysis. These researchers did not listen to their interview records. **Step 4** aimed to build a list of challenging activities. For this purpose, we discarded redundant data items, refined the code and category labels, and validated the two types of relationships mentioned in Step 2. We have built a list of challenging activities representing each type with the corresponding labels. Table 2 introduces the list of challenging activities.

Participant background Before presenting our study results, we discuss the self-assessed interviewed developers' skills collected from the Participant Characterization Form as follows. We observed that the developers have different backgrounds. We discuss the background of the 15 developers in terms of their skills. Our results indicate that the developers have *high* to *very high* skills in software engineering activities like software testing (41%), requirements engineering (59%), and web development (65%). Conversely, the developers have either *low*, *very low*, or *none* skills in gamification (82%) and game design (89%). Although most developers are beginners in gamification, we highlight that the team is composed of experienced software developers in software engineering practices. Additionally, 41.2% of developers hold a Bachelor's degree, and 29.4% hold a Master's degree. In addition, they have an average of six years in terms of development experience involving industrial systems.

5.2 Results and discussions

In this section, we report and discuss the study results addressing **RQ**$_2$. We discuss below the challenging activities perceived by the VazaDengue developers along the system's gamification, and how the interview data supported the refinement of our gamification method.

Challenging activities of gamification We answer our **RQ**$_2$ as follows. Table 2 presents all development activities perceived as challenging by the developers addressing gamification activities. For each group, we provide a description, the list of challenging activities that compose the group, and when each challenging activity emerged along the gamification of the VazaDengue systems. For instance, [GPR01] is a group of challenging activities related to decision-making about gamified systems, e.g., service decomposition. [GPR01] is composed of challenging activities such as [GPR01a] Define the gamified system, which emerged from the project beginning and along the project execution. In total, we have elicited 22 challenging activities, categorized into eight groups of challenging activities.

Table 2 Groups of challenging activities elicited from the interviews

[GRP01] Define the Gamified System	**[GRP02] Conceive the System Art & Design**	**[GRP03] Specify the Gamified System**
Description: Related to decision-making about the gamified systems in terms of source code organization, service decomposition, and data persistence. **Challenging activities:** [GRP01a] Define the Gamified System; [GRP01b] Model the Web Architecture of the gamified system; [GRP01c] Design the Database **Emerged when:** (a) from the project beginning and along the project execution; (b) along the gamification implementation; (c) after the first gamified system specification	**Description:** Related to the creative process of art & design aspects (visual metaphors, colors, etc.) of the gamified system. **Challenging activities:** [GRP02a] Conceive the Systems Art & Design **Emerged when:** (a) from the project beginning of the project, especially after proposing the gamification model	**Description:** Related to the definition of game elements and rules to engage the users through gamification. **Challenging activities:** [GRP03a] Specify the Gamified System; [GRP03b] Ensure the Gamification Design Conformance; [GRP03c] Elicit Game Elements aimed to Engage Users; [GRP03d] Insufficient Stakeholders to Provide Feedback; [GRP03e] Integrate Social Networks into Gamified System **Emerged when:** (c) from project beginning, especially along the system implementation; (d) from project beginning; (e) along the system implementation
[GRP04] Implement the Gamified System	**[GRP05] Understand the Existing System**	**[GRP06] Manage the Development Teams**
Description: Related to the implementation of the gamified system. **Challenging activities:** [GRP04a] iOS Development; [GRP04b] Mobile Development; [GRP04b] Inexperience with Hybrid Mobile Development; [GRP04c] Understand the Gamification Domain; [GRP04c] Manual Systems Deploy; [GRP04d] Synchronize System User Data **Emerged when:** (a) from project beginning; (b) from mobile implementation beginning; (c) from implementation beginning; (d) along the system implementation	**Description:** Related to the difficulties faced by the developers to understand the existing system in terms of context, architectural design, technologies, and source code. **Challenging activities:** [GRP05a] Understand the existing system API; [GRP05b] Understand the Back-end Layer; [GRP05c] Understand the System Domain **Emerged when:** (a) along the system implementation; (b) after a development specialist of the existing system left the project; (c) from the implementation beginning	**Description:** Related to the management of team members' tasks. **Challenging activities:** [GRP06a] Assign Tasks to Team Members **Emerged when:** (a) first task delays and from the project beginning
[GRP07] Communicate the Development Teams		**[GRP08] Test the Gamified System**
Description: Related to the communication among members of different development teams. **Challenging activities:** [GRP07a] Communicate Team Members; [GRP07b] Neglect the Gamified System Specification **Emerged when:** (a) from the definition of members per development team and from the project beginning; (b) from the first implementation outputs		**Description:** Related to the unit test writing and execution. **Challenging activities:** [GRP08a] Write Test Cases **Emerged when:** (a) while changing the implementation of existing system functionalities

Although we have found eight groups of challenging activities in our study, only six are mainly related to the difficulty of gamifying an existing software system: [GRP01, GRP02, GRP03, GRP04, GRP05, GPR08]. The remaining groups are composed of challenging activities that often occur during the development of non-gamified systems. Although [GPR04, GPR08] are related to the difficulty of gamifying an existing system, we do not describe these groups because it depends on the development techniques adopted by the development team of the gamified system. Thus, we discuss only the [GRP01, GRP02, GRP03, GRP05] groups of challenging activities as follows. For each group, we explained how the development activities are challenging.

We explain the four groups of challenging activities as follows. We provide to each group (i) a discussion about the challenging activities that compose the group; (ii) context, i.e, the factors that are associated with these challenging activities along the VazaDengue gamification; and (iii) insight into how certain challenging activities helped us to identify opportunities for refining the gamification method towards a more comprehensive and lightweight one.

[GRP01] define the gamified system was challenging for the developers due to several limitations identified in the existing system. Indeed, the architectural designers of the existing system recognized that the original system was not ready to accommodate gamification features, i.e., game elements and rules. In this sense, it has become evident for the implementation team that they could not address game elements and rules defined by the design team due to the high maintenance costs required [GRP01a]. Moreover, the naturally incremental definition of game elements and rules to be implemented by the architecture has generated additional costs to redesign the database several times [GRP01b], [GRP01c], which has delayed the implementation of game elements that inter-depend, such as team and challenges. We present a developer quote about *[GRP01c] Design the Database* as follows.

> *A major challenge was restructuring the architecture's infrastructure to accommodate gamification.* – Developers 7 about [GRP01c]

Context: Along with the existing system gamification, the unstable definition of game elements and rules has led to several changes in the relational database model. For instance, it led to changes in the database design to optimize the management of game rules and their respective game elements. Besides that, the rotation of certain team members has made it difficult to understand the existing database and identify reuse opportunities for the new architecture's database.

Insight: The challenging activities faced in [GRP01] gave us insights into possible refinements of the Morschheuser's method, e.g., the addition of two new activities in the *Context Analysis* phase: *(ADD) List Existing System features* and *(ADD) Define Features to Gamify*. By complementing the discussions in Section 4.2, the need to add these two activities was due to the excess of rework in architecture changes of the existing system realized without the prior understanding of the technological, functional, and architectural constraints that could be impeded to incorporate game elements and rules into the existing system.

[GRP02] conceive the system art & design was challenging due to the difficulty of designing an attractive user interface that engages our stakeholders, i.e., Brazilian citizens and public health agents. Various VazaDengue developers struggled with defining what visual elements and metaphors (Refaie, 2003) should compose the user interface and how to represent them [GRP02a]. Due to project budget constraints, both design and implementation teams were responsible for evaluating the user interface in terms of attractiveness. However, the lack of feedback from user interface specialists has led to rework in the front-end layer implementation. We present a developer quote about *[GRP02a] Conceive the System Art & Design* as follows.

> *While gamifying the existing system, it was difficult to find a trade-off between the system's serious purpose and the system's attractiveness from the viewpoint of its users. –* Developers 3 about [GRP02a]

Context: The gamified version of VazaDengue, is a healthcare system designed to support health agents in preventing and combating mosquito-borne diseases. At first glance, it was considered a too serious scope for gamification. For instance, the variety and complexity of both game elements and rules made it difficult to design a sufficiently attractive system for its users. In addition, the lack of guidelines for designing gamified systems hindered developers from adopting recommended gamification practices. In general, developers showed difficulty understanding what they needed to do to gamify the existing system.

Insight: The challenging activities faced in [GRP02] gave us insights into refinements of the Morschheuser's method, e.g., the adaptation of three activities that compose the *User Analysis* phase: *(ADP) Identify User Motivations*, *(ADP) Identify User Needs*, *(ADP) Define Target Users* and *(ADP) Create Personas* to considerate the existing system users. As discussed in Section 4.2, this insight became clearer due to the need to understand the different needs of the existing system's users and the new ones, before conceiving the system art & design. Another insight is related to the adaptation of two activities which are part of the *Software Design* phase: *(ADP) Design Prototype*, and *(ADP) Create Prototype* in order to consider the user interface of the existing system, along with the conceiving the system art & design.

[GRP03] specify the gamified system was particularly challenging due to the constant need for validating the conformance between the game elements elicited by the design team and the ones implemented by the implementation team [GRP03b]. Misalignment between teams, mainly due to communication noise among team members, combined with the underlying architectural constraints of the gamified system, made it difficult to reason about what game elements and rules were feasible to implement and reach a satisfactory user engagement [GRP03c]. We present a developer quote about *[GRP03c] Elicit Game Elements aimed to Engage Users* as follows.

> *Elicit functional and non-functional system requirements, besides understanding what requirements should be gamified was found challenging* – Developer 11 about [GRP03c]

Context: Some of the reasons that made [GRP03] a group of challenging activities address general issues of technical knowledge in gamification. This has caused a delayed specification of certain game elements and rules, especially those that inter-depend, such as teams and challenges.

Insight: The challenging activities faced in [GRP03] gave us insights into refinements of the Morschheuser's method, e.g., the addition of three activities that compose the *Requirements Elicitation* phase: *(ADD) Evolve Functional Requirement*, *(ADD) Evolve Non-Functional Requirement*, and *(ADD) Elicit Gamified Requirements*. By complementing the discussions in Section 4.2, this insight became more evident when the addition of game elements and rules, has made the specification of existing system requirements inconsistent. For instance, the requirements for reporting mosquito breeding sites need to be evolved to address certain rules.

Another insight obtained through the interviews is related to the addition of a new phase, called *Gamification Design* which is composed of four activities: *(ADD) Elicit Game Elements Elsewhere*, *(ADD) Pick Useful Game Elements*, *(ADD) Define Game Rules* and *(ADD)*

Create the Visual Representation for the Conceptual Model. This insight became more evident, due to the need to reason more specifically about the game elements that could be incorporated into the system, as well as which types of game rules (system-user or element-element) could be specified to leverage user engagement. Moreover, how these game elements and rules could be documented.

Finally, **[GRP05] Understand the Existing System** became challenging, mostly due to the poor and outdated existing system API document [GRP05a]. In fact, the API has more than 500 methods, which suggests a certain complexity for developers to understand and properly use it, especially without documentation support. This group of challenging activities may depend on the degree of involvement of developers. We present a developer quote about the *[GRP50a] Understand the existing system* API as follows.

> *Make decisions towards the gamification of the existing system and evolve a simple software infrastructure into a more complex one*. – Developer 7 about [GRP05a]

Context: Unfortunately, the documentation available for the existing system was overall poor and outdated. The lack of documentation was a real issue because the existing system' architecture was quite complex, composed of thousands of lines of code that lacked comments and documented rationale. In this context, the group [GRP05a] regards the difficulty of understanding how the existing system was designed through architecture. A key factor to that difficulty was the limited support that old team members were able to provide to new team members in understanding the existing system.

Insight: The challenging activities of [GPR05] gave us insights into the refinement of the Morschheuser's method, e.g., the reuse of the activity of *(REU) Understanding the Context*. By complementing the discussions in Section 4.2, this insight became more evident, given the poor and outdated documentation of our existing systems making it difficult for some members of the development team to understand the existing system from requirements to implementation.

We emphasize that part of the challenging activities, most related to software design, may also exist during maintenance tasks of existing systems without the need for gamification. However, challenges are emerging during the gamification of existing systems that are particularly important to this domain. They are transversal (crosscutting) in many cases and specific warnings and guiding support are important. For instance, even incorporating the simplest game elements and rules into an existing system often requires many changes in system requirements, design, implementation, and test suites. More specifically, high effort is required to: (i) understand the overall impact of implementing certain game elements and rules on the system requirements, especially the non-functional ones such as interoperability and availability; (ii) make decisions regarding a technological modernization (e.g., API replacement) to facilitate the software gamification; and (iii) evolve the systems' architecture to gamify the system, in which certain game elements and rules were postponed and discarded due to the challenging nature of some activities.

Summary of RQ$_2$. We elicited 22 challenging development activities, from which 18 explicitly relate to the systems gamification. Design and development teams have the highest number of challenging activities. Poor design decisions, complex architecture, and lack of software documentation had a negative impact on gamification implementation.

5.3 Threats to validity

Construct validity We designed the Participant Characterization Form aimed to capture the interviewees' expertise in basic software engineering activities from requirements elicitation to testing. To minimize threats related to the sampling variety of interviewees, we selected participants from diverse levels of education, experience in software development, and expertise in key concepts of gamification and software engineering techniques, ensuring a broad representation of perspectives. In addition, we have interviewed only developers engaged with at least two development teams aimed to avoid a poor elicitation of challenging activities. We designed our interview protocol based on the funnel structure aimed to avoid missing relevant information about the challenging activities during the interviews. Thus, we have conducted a more flexible interview starting with general questions and ending with specific ones.

Internal validity We conducted each interview in isolation to make interviewees comfortable with reporting their perceptions about challenging activities. We have also recorded the interview audio to complement the interview notes and avoid missing data. We preferred face-to-face interviews rather than online ones aimed to control the interviewees' attention whenever possible. The interviewers were oriented to strictly follow the Interview Script, avoiding influencing the interviewees' answers.

Conclusion validity We have partially relied on GT (Easterbrook et al., 2008) to define our data analysis protocol. We aimed to reduce the inherent subjectivity of coding interview notes. We analyzed all data in a pair to minimize biases and reach a consensus about the elicited challenging development activities. The main researcher who coded the interviews was not interviewed, so we avoided possible bias. We have analyzed challenges regarding the gamification of both mobile and web applications to make our findings varied.

External validity We interviewed 15 developers from a single project only, but they mostly lacked gamification expertise. This might have boosted the relevance of the elicited challenges, since developers with expertise in gamification may face different challenges from those that have been elicited in the study. From these 15 developers, we have recruited interviewees engaged with two or more development teams: design, implementation, testing, and research. We expect that our challenges encompass varied software engineering activities, from requirements elicitation to testing. Therefore, we elicited challenges of different natures during the gamification process. Most interviewees showed inexperience with gamification. Although it might have biased the reported perceptions of challenging activities, the analysis of the interviewees' backgrounds has helped us characterize the generalization scope of our study results according to the expertise of the developers.

6 Gamify4Fun: A method for guiding the gamification of existing software systems

Our two empirical studies enabled us to derive our method for guiding the gamification of existing systems. We detail our method as follows.

6.1 Method overview

Figure 7 illustrates the phases, activities, and artifacts of the refined method; it also depicts their relationships. The figure relies on the Business Process Model and Notation

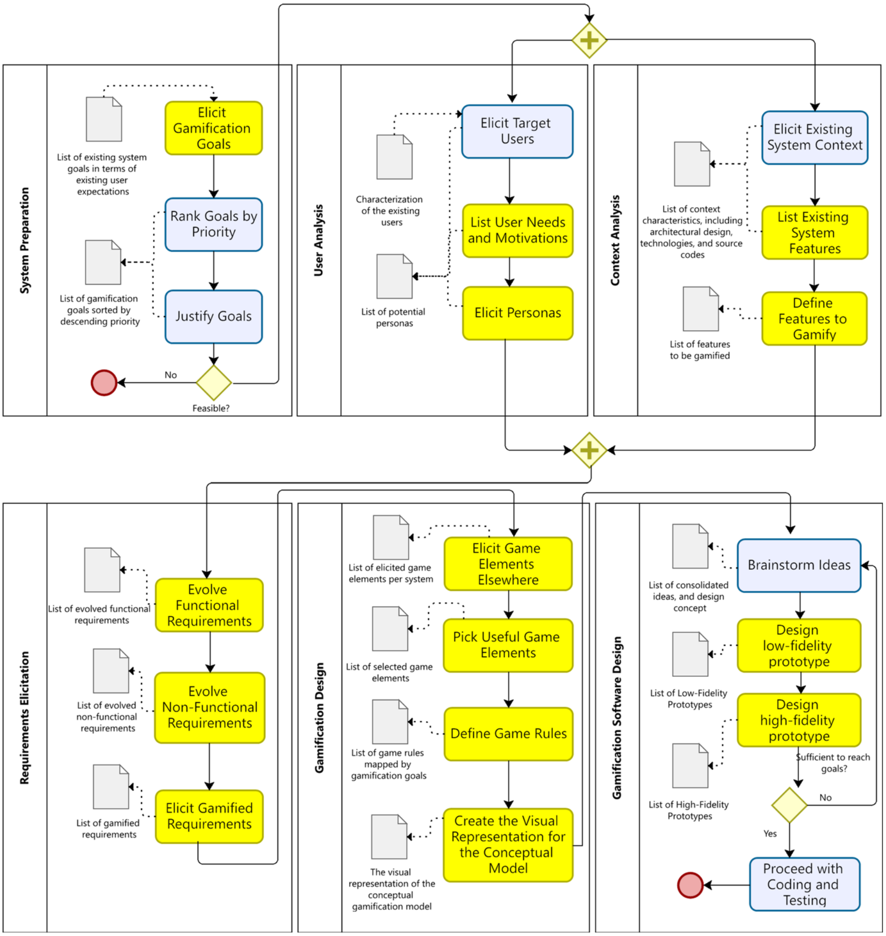

System Preparation
Elicit Gamification Goals
List of existing system goals in terms of existing user expectations
Rank Goals by Priority
List of gamification goals sorted by descending priority
Justify Goals
No
Feasible?
User Analysis
Elicit Target Users
Characterization of the existing users
List User Needs and Motivations
List of potential personas
Elicit Personas
Context Analysis
Elicit Existing System Context
List of context characteristics, including architectural design, technologies, and source codes
List Existing System Features
Define Features to Gamify
List of features to be gamified
Requirements Elicitation
Evolve Functional Requirements
List of evolved functional requirements
Evolve Non-Functional Requirements
List of evolved non-functional requirements
Elicit Gamified Requirements
List of gamified requirements
Gamification Design
Elicit Game Elements Elsewhere
List of elicited game elements per system
Pick Useful Game Elements
List of selected game elements
Define Game Rules
List of game rules mapped by gamification goals
Create the Visual Representation for the Conceptual Model
The visual representation of the conceptual gamification model
Gamification Software Design
Brainstorm Ideas
List of consolidated ideas, and design concept
Design low-fidelity prototype
List of Low-Fidelity Prototypes
Design high-fidelity prototype
List of High-Fidelity Prototypes
Sufficient to reach goals?
No
Yes
Proceed with Coding and Testing

Fig. 7 A method for gamifying existing systems

(BPMN) (OMG, 2011). The figure represents the following six method phases: *1 System Preparation)* consists of preparing the environment for gamifying the existing system; *2 User Analysis)* aims to elicit potential system users based on the existing system users; *3 Context Analysis)* aims to characterize the existing system domain; *4 Requirements Elicitation)* aims to refine existing requirements and defining gamification-specific requirements; *5 Gamification Design)* aims to design the conceptual gamification model; *6 Gamification* Software Design) aims to design and prototype of the gamified system. We do not describe the *Proceed with Coding and Testing* activity because it depends on the development techniques companies and developers adopt. We also highlight the challenging development activities of each phase with yellow color.

In the following subsection, we present Gamify4Fun activities by phase. We describe each activity in terms of its definition, lessons learned, and recommendations gathered from the practical experience.

6.2 System preparation

The first phase aims to promote discussions on the goals developers expect to accomplish when gamifying the existing system. A gamification goal is any concrete need for making the system enjoyable and challenging for its users. In the specific case of VazaDengue gamification, we tried to answer questions such as *How do mosquito-borne diseases spread in economically emerging countries such as Brazil?*, *How do citizens and public health agents contribute to disease prevention?*, and *What tasks are critical to the disease prevention and, therefore, should be constantly performed by the citizens?* This phase consists of three activities:

- **Elicit gamification goals**. ***Definition:*** It consists of eliciting what the existing users and potential new users should expect from the gamified version of the existing system. ***Lesson learned:*** Our experience in gamifying VazaDengue has benefited from meetings with health agents (Fernandes et al., 2019), citizens, and health authorities. ***Recommendation:*** We recommend developers conduct meetings and promote workshops with existing and new system stakeholders, such as (but not limited to) domain experts, institutions, and system users.
- **Rank goals by priority**. ***Definition:*** It consists of ranking the elicited gamification goals by priority. ***Lesson learned:*** In the VazaDengue, we prioritized rewarding users for reporting mosquito breeding sites. This was given the highest priority because public health agents rely on these reports to prevent disease outbreaks. ***Recommendation:*** We recommend that developers should establish a single prioritization criterion measuring the impact of reaching each goal.
- **Justify goals.** ***Definition:*** It consists of documenting the rationale behind each goal. A well-documented rationale can help to understand the enjoyability and the effort required to incorporate gamification into the existing system. ***Lesson learned:*** In the VazaDengue case, we justified each gamification goal together with health authorities, this helped to validate the need to gamify the existing system (Fernandes et al., 2019). ***Recommendation:*** We recommend the continuous support of system stakeholders, as we had from the domain experts. Additionally, once the gamification of the system was sufficiently justified, developers would proceed with the next phase. Otherwise, it may be the case that gamifying the existing system is not the best solution to leverage the user experience.

6.3 User analysis

The second phase aims to characterize the users interacting with the existing system. While gamifying VazaDengue, we aimed to address questions like *What were the VazaDengue users?* This phase also aims to reason about additional users that could be interested in the gamified system. In this case, we tried to answer questions like *Who would be the potential users of the gamified system?*, and *Is there any chance of losing users after the existing system is gamified?* This phase consists of three activities:

- **Elicit target users.** ***Definition:*** It consists of listing the candidate users of the gamified system. ***Lesson learned:*** In the VazaDengue case, we conducted meetings and workshops with the Brazilian public health agents. We aimed at characterizing the existing system users and the potential new ones, e.g., young students and residents of Brazilian communities. ***Recommendation:*** We recommend that developers first identify the current groups of users of the existing system and the new users likely to engage due to gamification.

This will allow developers to consider the potential risks and benefits associated with these users possibly leaving the system after the implementation of gamification.

- **Elicit user needs and motivation.** ***Definition:*** It consists of listing the needs of the gamified system's users. ***Lesson learned:*** In line with the previous activity, we learned that understanding the needs and motivations of potential users directly helps in identifying potential solutions for system gamification and refining the gamification goals. ***Recommendation:*** We recommend developers meet with the stakeholders and discuss the user's needs based on the gamification goals. Additionally, track the motivations behind the current user interaction with the existing system.
- **Elicit personas.** ***Definition:*** It consists of summarizing the lists of target users, needs, and motivation into personas. Personas are mechanisms for abstracting user profiles through human characteristics, such as age, gender, and professional background (Grudin & Pruitt, 2002). ***Lesson learned:*** The data richness documented through the personas elicitation helps to identify new profiles of users potentially estimated due to gamification. ***Recommendation:*** We recommend describing the daily routine of each persona aimed to highlight the context in which they would be engaged with the system features.

6.4 Context analysis

The third phase aims to characterize the context in which the existing system was conceived. Context includes the human resources, design decisions, and technologies employed along with the system's development process. While gamifying VazaDengue, we tried to answer questions like *What development process has guided the VazaDengue development?* and *What technological constraints affected the system development?* Our experience suggests that the clearer the context analysis, the easier is for developers to cope with challenges along with the system gamification. This phase consists of three activities:

- **Elicit existing system context.** ***Definition:*** It consists of documenting any context information, e.g., employed technologies and design decisions. ***Lesson learned:*** While gamifying VazaDengue, the lack of eliciting the existing system design and architecture has led to excessive rework. For instance, poor design decisions led to repeatedly redesigning the system's database and services. ***Recommendation:*** We recommend that developers search by the existing system documentation. In the case of scarce or outdated system documentation, a deep analysis of the system design and architecture should be performed to identify potential constraints for system gamification.
- **Elicit existing system features.** ***Definition:*** It consists of listing the main features of the existing system. ***Lesson learned:*** Initially, we elicited existing features by using the system via the navigation interface, which allowed us to perform a quick elicitation. Next, we analyzed the source code at high and low levels to perform a more accurate elicitation. ***Recommendation:*** We recommend that if the documentation of the existing system is scarce or outdated, feature elicitation should be performed to support the next activity.
- **Define features to gamify.** ***Definition:*** It consists of selecting the existing system features that should be gamified. ***Lesson learned:*** Focus on the main system features and features that have greater interaction with system users. ***Recommendation:*** We recommend discussing with system stakeholders about: (i) system features that succeeded in their purpose without gamification; and (ii) system features that may reach their purpose through gamification. We also recommend paying attention that the system features

to be gamified must be strongly related to the gamification goals defined in the *system preparation* phase.

6.5 Requirements elicitation

The fourth phase addresses the systematic documentation of the gamified system's functional, non-functional, and gamification-specific requirements. This phase consists of three activities:

- **Evolve Functional Requirements (FR).** ***Definition:*** It consists of refining the FR elicited for the existing system. In contrast to eliciting requirements for a non-existing system, there are features that developers should consider before gamifying the system. ***Lesson learned:*** The personas characterization considerably helped to evolve our existing system requirements. By defining personas to the potential system users, we tried to understand the possible interactions of a given user with the gamified system. ***Recommendation:*** We recommend relying on the elicited personas for refining the FR. Additionally, we recommend mapping the FR by gamification goals.
- **Evolve Non-Functional Requirements (NFR).** ***Definition:*** It consists of refining the NFR elicited for the existing system. ***Lesson learned:*** Several NFRs of performance and availability were changed due to gamification. ***Recommendation:*** We recommend considering the technological constraints elicited in the *context analysis* phase to evolve the NFRs. Additionally, a lack of attention to the evolution of NFRs that become critical or more critical can directly affect the success of gamification. We also recommend mapping the NFRs by gamification goals.
- **Elicit gamified requirements.** ***Definition:*** This activity complements the two previous activities by specifying the gamification-specific requirements. These requirements encompass the features that emerged from incorporating game elements and rules into the existing system. ***Lesson learned:*** Elicit the gamification requirements, after evolving the FRs, and NFRs, helps identify incomplete or inefficient evolution of requirements. ***Recommendation:*** Similar to FRs and NRs, we recommend performing the map with the gamification goals. Additionally, we recommend focusing on the features in which the action performed by the user triggers any game elements to elicit the gamification requirements.

6.6 Gamification design

The fifth phase has the main goal of building the gamification conceptual model to be incorporated into the existing system. To build this conceptual model, developers should carefully define the game elements and rules to be implemented. During the VazaDengue gamification, we have debated questions like *What game elements could help us in leveraging the enjoyability and challenge levels of VazaDengue?*, *How should these game elements interact for realizing our gamification goals?*, and *How the system users should interact with these game elements?* This phase consists of four activities:

- **Elicit game elements elsewhere.** ***Definition:*** It consists of searching for game elements to be incorporated into the existing system according to the gamification goals. ***Lesson learned:*** First, we searched for gamified healthcare systems addressing similar gamification goals, i.e., spotting focus and cases of diseases. Since we did not find such systems, we tabulated the game elements from ten popular gamified systems that address similar gamification goals in other domains, e.g., Waze. In Waze, we tabulated game elements addressing the spotting issues on maps. ***Recommendation:*** We recommend this activity for developers with no experience with gamification. We also recommend searching by successful gamified systems that address similar gamification goals, starting from gamified systems in the same knowledge domain. After the search, a list of the most frequent game elements addressing each gamification goal should be made.

- **Pick useful game elements.** ***Definition:*** It consists of picking game elements helpful to achieve the gamification goals. ***Lesson learned:*** To select the appropriate game elements it is necessary to keep an open mind, without forgetting the constraints of the existing system. ***Recommendation:*** We recommend selecting the best game elements by gamification goal. In case of identifying two or more alternatives of game elements fitting the same gamification goal, consider the *constraints* elicited in the *context analysis* phase. The constraint analysis may help to pick up the game elements that will require less effort to gamify the existing system.
- **Define game rules.** ***Definition:*** It consists of defining how the system users and game elements should interact with the system. ***Lesson learned:*** Prioritizing the composition of simple rules helps in defining and introducing gamification into the existing system. ***Recommendation:*** We recommend that a map be made between the game rules and the gamification goals.
- **Create the visual representation for the conceptual model.** ***Definition:*** It aims to draw the conceptual model based on the picked game elements and defined game rules. ***Lesson learned:*** We observed that the design of a detailed gamification model helped us to always track and keep the main game elements and rules consistent across gamified features, avoiding rework while gamifying a system. ***Recommendation:*** We strongly recommend carefully modeling the relationships between the game elements and the game rules.

6.7 Gamification software design

The sixth phase consists of defining the aesthetics of the gamified system. While gamifying VazaDengue, we have tried to answer questions like *What aesthetics elements may be reused from the existing system?* and *What changes should we apply for incorporating game elements into the existing system interface?* This phase consists of three activities:

- **Brainstorming ideas.** ***Definition:*** Consists of promoting discussions regarding gamified system aesthetics. ***Lesson learned:*** Performing brainstorming helped us to think about the best-gamified system aesthetics to address the game elements. In VazaDengue gamification, we have designed different interfaces for mobile and Web environments. ***Recommendation:*** We recommend that the developers list all ideas and ask the stakeholders' opinions about color schemes, layout items, and screen navigation preferences.
- **Design low-fidelity prototypes.** ***Definition:*** Consists of elaborating either manual or tool-supported interface drafts, such as wireframes. ***Lesson learned:*** Keeping low-fidelity prototypes simple, slightly similar to existing interfaces, and oriented towards user interaction with game elements helps in defining intuitive interfaces. ***Recommendation:*** When gamifying a system with many applications, we recommend designing similar drafts that share elements, so that users can easily migrate across applications.
- **Design high-fidelity prototypes.** ***Definition:*** Consists of drawing high-fidelity interface prototypes. ***Lesson learned:*** Continuous inspection activities on the prototypes help to avoid non-conformities, especially if the system to be gamified is cross-platform (our

case). ***Recommendation:*** We recommend validating and refining these prototypes with stakeholders.

6.8 Limitations of the Gamify4Fun method

We are aware of a few limitations of our method. These limitations address development activities we intentionally do not address from the original method (Expected Limitations) and aspects observed through the empirical validation of the proposed method (Observed Limitations). We summarize the major limitations for both types as follows.

Expected limitations In an intentional way, we decided that our method would not guide developers along the system coding and testing. We took this decision, since coding and testing practices may significantly vary across software projects, development teams, and organizations. In the particular case of gamifying VazaDengue, we followed agile practices, for instance, the use of version control systems and bi-weekly follow-up meetings. Thus, although the Morschheuser method that we have refined (Morschheuser et al., 2017) has prescribed activities for writing, validating, and testing code, we have omitted these activities from our method.

Nevertheless, we acknowledge that full support for development teams to gamify their existing systems would require at least a general view of how coding and testing would look. In fact, we observed two groups of challenging development activities strongly related to coding, i.e., [GPR04], and testing, i.e., [GPR06]. One opportunity for improving our method would be to propose a guide to the test and development practices we use in our particular context. Another opportunity addresses proposing a guide composed of different techniques that support the developers in evaluating the prototypes with the stakeholders. For instance, usability testing (Barbosa & Silva, 2010).

Observed limitations Through our empirical studies, we observed that certain activities of Gamify4fun have some limitations. For instance, the technique we recommend to characterize the system users, i.e., *Personas*, may not be sufficient to represent the potential users that would interact with the system. The literature reports other techniques with a similar purpose, e.g., user journey (Deterding et al., 2013) and empathy map (Osterwalder & Pigneur, 2010). One opportunity for improving our method would be to adapt the user analysis phase to include the empathy map technique (Osterwalder & Pigneur, 2010). The empathy map aims to create an empathy degree of developers with the target system users. Thus, this technique could support the developers to better characterize the target users regarding their needs and motivations (Osterwalder & Pigneur, 2010).

Another limitation of our current method addresses the visual representation of the relationships between the system users, game elements, and rules through the gamification model. The gamification model may not be the best way to describe the relationships among the game elements. Due to the existence of other notations better known by developers. For instance, the literature reports other techniques with a similar purpose, e.g., decision trees (Marache-Francisco & Brangier, 2013), storyboard (Hsu et al., 2013), and use-case modeling to support the specification and the gamification design (De Lope & Medina-Medina, 2016; Herzig et al., 2012). One opportunity for improving our method would be to adapt the gamification design phase to provide one of the aforementioned techniques for representing the relationships between the system users, game elements, and rules.

7 Related work

Table 3 compares our method, Gamify4Fun, with the existing methods mentioned in Section 2.2 at the design level. The methods are listed in the rows. The second to seventh columns present the gamification development phases supported by Gamify4Fun. The eighth column informs whether the method is evidence-based, i.e., the method was either derived from empirical studies or empirically assessed. The cells are filled with: *Yes* whether the method covers a phase supported by Gamify4Fun; *Partially* whether the method partially covers a phase; and *No* whether the method does not cover that phase.

System preparation The method proposed by Rodrigues et al. (2016) has an activity, namely *Business Objectives Definition* partially covering the System Preparation phase. This activity concerns the definition of the gamification objectives. The developer is guided to answer questions, such as *What is the purpose of gamification for the business?* and *What product or business application should be gamified?* Similarly, the method proposed by Kardan and Arani (2016) partially covers the System Preparation phase in a phase called *Define Targets and Need of Organization*. Finally, the method by Morschheuser et al. (2018) covers the phase of System Preparation, but this phase required refinements to properly support the particularities of gamifying existing systems (Section 4).

User analysis (Rodrigues et al., 2016) do not formalize a phase aimed at defining and characterizing the target users. Conversely, Kardan and Arani (2016) partially covered the User Analysis phase through the *Define Priority of Targets and Needs* phase. Thus, developers are guided in defining the priority and needs of the target users. However, this phase lacks systematic user profiling. As discussed in Section 6, it is important to characterize those who have been using the existing system so far. Otherwise, these users may eventually leave the system because the gamification did not accomplish their needs. Finally, Morschheuser et al. (2018) supports the User Analysis phase, though refinements were necessary (Section 4).

Context analysis The method proposed by Rodrigues et al. (2016) has two activities: *Methodology Development Software and Tools* and *Game Design & Software Develop*. The first one aims to define the methodology and tools for supporting the system gamification. The developer is guided in answering questions such as *What software development method should be used?* and *What tools should be used in gamification?* The second activity regards questions like *What controls and tests should be implemented?* and *What are the assessment and monitoring processes to be implemented so that the goals of gamification are met?* Both activities partially cover the Context Analysis phase of Gamify4Fun. Concerning the Context Analysis phase, we did not find insights in the Kardan and Arani (2016) work. In this way, we applied some refinements to the phase prescribed by Morschheuser et al. (2018) to cover this phase adequately (Section 4).

Requirement elicitation The method proposed by Rodrigues et al. (2016) has an activity namely *Game Model and Characteristics Definition*. This activity is driven by questions such as *What type of game should be used?* and *What game characteristics are suitable for the business?* Thus, this activity partially covers the Requirement Elicitation phase of our method, especially concerning the elicitation of gamification-related requirements. Again, the method introduced by Kardan and Arani (2016) lacked insights into the Requirement Elicitation phase. Morschheuser et al. (2018) method has a useful activity called *Identify*

Table 3 Study comparison at the design level

Method	System preparation	Development phases					
		User analysis	Context analysis	Requirement elicitation	Gamification design	Software design	Evidence based
Rodrigues et al. (2016)	Partially	No	Partially	Partially	Partially	Partially	No*
Kardan and Arani (2016)	Partially	Partially	No	No	Partially	No	No
Morschheuser et al. (2018)	Yes	Yes	Yes	Yes	No	Yes	Yes
Gamify4Fun	Yes	Yes	Yes	Yes	Yes	Yes	Yes

*Evaluated only with toy examples

Requirements. This activity was converted into a whole development phase dedicated to the requirement elicitation (Section 4).

Gamification design The method proposed by Rodrigues et al. (2016) includes the *Game Model and Characteristics Definition* activity. This activity helps in addressing the Gamification Design phase for the elicitation of game elements and the definition of game rules. In turn, the method of Kardan and Arani (2016) has two activities concerning the definition of game rules and game elements (particularly score and awards). These activities partially cover the Gamification Design phase as well. Conversely, the method introduced by Morschheuser et al. (2018) did not provide insights for accomplishing the Gamification Design phase.

Software design There is an activity in the method of Rodrigues et al. (2016) called *Gamification Quality Control & Feedback*. This activity is responsible for addressing questions such as *What gamification evaluation results should be implemented?* and *What changes should be implemented?* Ultimately, these questions regard the software quality and drive the definition and inspection of system prototypes. In summary, this activity helps partially cover the Software Design phase of Gamify4Fun. Conversely, we did not find activities in the method of Kardan and Arani (2016) that could support the Software Design phase. Finally, the Morschheuser et al. (2018) method covers the phase of Software Design with a few refinements (Section 4).

Evidence-based evaluation It is worth mentioning that only a few of the existing gamification methods were either derived from empirical studies or empirically assessed. Both Kardan and Arani (2016) and Rodrigues et al. (2016) did not build their methods based on empirical evidence. The latter has performed an empirical assessment, but the authors relied on toy examples. Especially when assessing software development methods, the adoption of toy systems rather than real systems may be considered a threat to the actual method's feasibility. Conversely, Morschheuser et al. (2018) have assessed the proposed method with real systems. Nevertheless, Morschheuser's method did not emerge from practical experience, i.e., along with the gamification of a real system. This is a major difference when compared to what we did with Gamify4Fun.

8 Conclusion remarks

In this study, we report our practical experience in dealing with the challenges and complexities of gamifying existing systems, shedding light on the need for robust software engineering practices to support this process. Through our practical experience in gamifying the VazaDengue healthcare system and subsequent research activities, we developed and refined a comprehensive gamification method called Gamify4Fun. This method provides actionable guidance for development teams aiming to gamify existing systems, helping them deal with the complexities of incorporating game elements and rules while leveraging existing software artifacts.

Our method consists of six development phases, from system preparation to software design, each addressing key activities required to refine existing software artifacts and build a gamification conceptual model. We identified and applied activity refinements, guided by a self-observation study and interviews with developers involved in gamifying existing systems. These refinements reflect the specific challenges faced in gamifying existing systems, emphasizing the need for tailored support in this domain.

In conclusion, by providing a systematic approach and addressing the particular challenges inherent in this process, Gamify4Fun aims to support development teams aiming at gamifying existing systems, especially in the case of teams composed of developers inexperienced with gamification and following agile development practices (Schwaber & Beedle, 2002). Future research can further explore and refine gamification methodologies, expanding our understanding and capabilities in this evolving field.

Author Contributions Anderson Uchôa: Conceptualization; Data curation; Investigation; Methodology; Validation; Visualization; Writing - original draft; Writing - review and editing. Rafael de Mello: Conceptualization; Investigation; Methodology; Validation; Visualization; Writing - original draft; Writing - review and editing. Jairo Souza: Investigation; Writing - review, and editing. Leopoldo Teixeira: Investigation; Writing - original draft; Writing - review, and editing. Baldoino Fonseca: Investigation; Writing - original draft; Writing - review, and editing; Project administration, Funding acquisition. Alessandro Garcia: Conceptualization; Methodology; Supervision; Validation; Writing - review, and editing.

Funding This work is partially supported by FAPEAL (60030.0000002725/2022, Institutional Links - Zika Virus and PPGs 14/2016), FUNCAP (BP5-00197-00042.01.00/22), FACEPE (APQ-0399-1.03/17, PRONEX APQ/0388-1.03/14), PPSUS/FAPERJ (E26-102.166/2013), CNPq (423125/2021-4, 315532/2021-1, 465614/2014-0), CAPES (88887.136410/2017-00), the Newton Fund, the Alexander Von Humboldt Foundation, and the INES (www.ines.org.br).

Data Availability The main artifacts involved in our study are available at https://doi.org/10.5281/zenodo.7750456, which includes the participant characterization form, and the interview script.

Declarations

Competing Interests The authors declare that they have no competing interests.

References

Barbosa, S., & Silva, B. (2010). *Interação Humano-Computador* (1st ed.). Elsevier. (In Portuguese).

Bosu, A., Corley, C., Heaton, D., et al. (2013). Building reputation in stackoverflow: An empirical investigation. In: Proceedings of the 10th international conference on Mining Software Repositories (MSR), pp 89–92

Bucchiarone, A., Cicchetti, A., & Marconi, A. (2020). Towards engineering future gameful applications. In: Proceedings of the ACM/IEEE 42nd international conference on software engineering: new ideas and emerging results, pp 105–108

Corbin, J., & Strauss, A. (2008). *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*, 3rd edn. Thousand Oaks

De Lope, R., & Medina-Medina, N. (2016). Using UML to model educational games. In: Proceedings of the 8th international conference on games and virtual worlds for serious applications (VS-Games), pp 1–4

Deterding, S., Dixon, D., Khaled, R. et al. (2011). From game design elements to gamefulness: Defining gamification. In: Proceedings of the 15th international technology conference (MindTrek), pp 9–15

Deterding, S., Björk, S. L., Nacke, L. E., et al. (2013). Designing gamification: Creating gameful and playful experiences. In: Proceedings of the 31st conference on human factors in computing systems (CHI): extended abstracts, pp 3263–3266

Dubois, D. J., & Tamburrelli, G. (2013). Understanding gamification mechanisms for software development. In: Proceedings of the 2013 9th joint meeting on foundations of software engineering, pp 659–662

Dybå, T., & Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. *Information and Software Technology, 50*(9–10), 833–859.

Easterbrook, S., Singer, J., Storey, M. A., et al. (2008). Selecting empirical methods for software engineering research. In: Guide to advanced empirical software engineering. Springer, p 285–311

Emerson, R. M., Fretz, R. I., & Shaw, L. L. (2001). Participant observation and fieldnotes. Handbook of ethnography pp 352–368

Fernandes, E., Uchôa, A., Sousa, L., et al. (2019). VazaZika: A software platform for surveillance and control of mosquito-borne diseases. In: Proceedings of the 16th international conference on Information Technology: New Generations (ITNG), pp 1–4

Glaser, B. (1998). Doing Grounded Theory: Issues and Discussions. Sociology Press, URL https://books.google.com.br/books?id=XStmQgAACAAJ

Grudin, J., & Pruitt, J. (2002). Personas, participatory design and product development: An infrastructure for engagement. In: Proceedings of the 7th Participatory Design Conference (PDC), pp 144–152

Hanlon, J. (2013). Five years ago, stack overflow launched. then, a miracle occurred. Available at Stack Overflow Blog: https://stackoverflow.blog/2013/09/16/five-years-ago-stack-overflow-launched-then-a-miracle-occurred/

Herzig, P., Ameling, M., & Schill, A. (2012). A generic platform for enterprise gamification. In: Proceedings of the 10th Working Conference on Software Architecture (WICSA), pp 219–223

Hsu, S., Chang, J. W., & Lee, C. C. (2013). Designing attractive gamification features for collaborative storytelling websites. *Cyberpsychology, Behavior, and Social Networking, 16*(6), 428–435.

Jin, Y., Yang, X., Kula, R., et al. (2015). Quick trigger on stack overflow: A study of gamification-influenced member tendencies. In: Proceedings of the 12th international conference on Mining Software Repositories (MSR), pp 434–437

Kardan, A., & Arani, A. (2016). A novel gamification-based architecture for web environments. In: Proceedings of the 2nd International Conference on Web Research (ICWR), pp 125–130

Ko, A. J. (2017). A three-year participant observation of software startup software evolution. In: 2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering in Practice Track (ICSE-SEIP), IEEE, pp 3–12

Marache-Francisco, C., & Brangier, E. (2013). Process of gamification: From the consideration of gamification to its practical implementation. In: Proceedings of the 6th international conference on advances in human oriented and personalized mechanisms, technologies, and services (CENTRIC), pp 126–131

Morschheuser, B., Hamari, J., Werder, K., et al. (2017). How to gamify? A method for designing gamification. In: Bui T (ed) 50th Hawaii International Conference on System Sciences, HICSS 2017, Hilton Waikoloa Village, Hawaii, USA, January 4-7, 2017. ScholarSpace / AIS Electronic Library (AISeL), pp 1–10

Morschheuser, B., Hassan, L., Werder, K., et al. (2018). How to design gamification? A method for engineering gamified software. *Information and Software Technology (IST), 95*, 219–237.

Muntean, C. I. (2011). Raising engagement in e-learning through gamification. In: Proceedings of the 6th International Conference on Virtual Learning (ICVL), pp 1–7

OMG. (2011). Business Process Model and Notation (BPMN). Tech. rep., Object Management (OMG), Inc., version 2.0

Osterwalder, A., & Pigneur, Y. (2010). *Business model generation: A handbook for visionaries, game changers, and challengers* (1st ed.). John Wiley & Sons.

Pedreira, O., García, F., Brisaboa, N., et al. (2015). Gamification in software engineering: A systematic mapping. *Information and Software Technology (IST), 57*, 157–168.

Pereira, P., Duarte, E., Rebelo, F., et al. (2014). A review of gamification for health-related contexts. In: Proceedings of the 4th international conference on Design, User Experience and Usability (DUXU), pp 742–753

Refaie, E. E. (2003). Understanding visual metaphor: The example of newspaper cartoons. *Visual Communication, 2*(1), 75–95.

Rodrigues, L. F., Costa, C., & Oliveira, A. (2016). Gamification: A framework for designing software in e-banking. *Computers in Human Behavior, 62*, 620–634.

Rojas, J. M., White, T., Clegg, B., et al. (2017). Code Defenders: Crowdsourcing effective tests and subtle mutants with a mutation testing game. In: Proceedings of the 39th International Conference on Software Engineering (ICSE), pp 677–688

Runeson, P., & Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering (EMSE), 14*(2), 131–164.

Schacht, M., & Schacht, S. (2012). Start the game: Increasing user experience of enterprise systems following a gamification mechanism. In: Software for people. Springer, p 181–199

Schwaber, K., & Beedle, M. (2002). *Agile software development with Scrum*. Prentice Hall.

Sousa, L., de Mello, R., Cedrim, D., et al. (2018). VazaDengue: An information system for preventing and combating mosquito-borne diseases with social networks. *Information Systems (IS), 75*, 26–42.

Swacha, J. (2016). Gamification in enterprise information systems: What, why and how. In: Proceedings of the 11th Federated Conference on Computer Science and Information Systems (FedCSIS), pp 1229–1233

Tausworthe, R. (1979). The work breakdown structure in software project management. *Journal of Systems and Software (JSS), 1*, 181–186.

Uchôa, A., Fernandes, E., Fonseca, B., et al. (2019). On gamifying an existing healthcare system: Method, conceptual model and evaluation. In: Proceedings of the 1st international workshop on Software Engineering for Healthcare (SEH), co-located with the 41st International Conference on Software Engineering (ICSE), pp 1–8

Uchôa, A., de Mello, R., Souza, J., et al. (2024). Replication package for the paper: "Towards Effective Gamification of Existing Systems: Method and Experience Report". https://doi.org/10.5281/zenodo.7750456

Wohlin C, Runeson P, Höst M, et al (2012) Experimentation in Software Engineering, 1st edn. Springer Science & Business Media

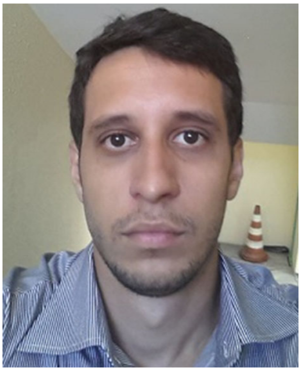
Zichermann G, Cunningham C (2011) Gamification by Design: Implementing Game Mechanics in Web and Mobile Apps. O'Reilly Media

Zichermann, G., & Linder, J. (2010). *Game-based Marketing: Inspire Customer Loyalty through Rewards, Challenges, and Contests*. John Wiley & Sons.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Anderson Uchôa is an assistant professor at the Federal University of Ceará (UFC). His research interests include code review, software maintenance and evolution, program comprehension, and human and social aspects in software engineering. He holds a PhD in Informatics from PUC-Rio. Contact him at andersonuchoa@ufc.br.

Rafael de Mello is an assistant professor at UFRJ, heading the ITER research group. His research interests include human and social aspects of software engineering, software verification, and empirical software engineering. He holds a DSc. in Systems Engineering and Computing from the Federal University of Rio de Janeiro (COPPE/UFRJ). He collaborated as a Postdoctoral researcher at PUC-Rio and worked in the industry for 19 years. Contact him at rafaelmello@ic.ufrj.br.



Jairo Souza is a PhD student at the Informatics Center (CIn) of the Federal University of Pernambuco. His main research interests include artificial intelligence for software engineering, program analysis, data mining, and software evolution.

Leopoldo Teixeira is an associate professor at the Informatics Center (CIn) of the Federal University of Pernambuco, where he leads the Software Testing and Analysis Research group. His main research interests involve the following topics and their integration: highly configurable software systems, software evolution, formal methods, software testing, and program analysis.

Baldoino Fonseca is an associate professor at the Federal University of Alagoas (UFAL), heading Engineering and Systems Group (EASY). His main research interests include software maintenance and evolution and machine learning. Contact him at baldoino@ic.ufal.br

Alessandro Garcia is an associate professor of Software Engineering at PUC-Rio, heading the OPUS Research Group. His main research interests include software maintenance and evolution. He holds a PhD degree in informatics from PUC-Rio. Contact him at afgarcia@inf.puc-rio.br

Authors and Affiliations

Anderson Uchôa[1] · Rafael de Mello[2] · Jairo Souza[3] · Leopoldo Teixeira[3] · Baldoino Fonseca[4] · Alessandro Garcia[5]

✉ Anderson Uchôa
andersonuchoa@ufc.br

Rafael de Mello
rafaelmello@ic.ufrj.br

Jairo Souza
jrcms@cin.ufpe.br

Leopoldo Teixeira
lmt@cin.ufpe.br

Baldoino Fonseca
baldoino@ic.ufal.br

Alessandro Garcia
afgarcia@inf.puc-rio.br

[1] Federal University of Ceará (UFC), Itapajé, Ceará, Brazil

[2] Federal University of Rio de Janeiro (UFRJ), Rio de Janeiro, Rio de Janeiro, Brazil

[3] Federal University of Pernambuco (UFPE), Recife, Pernambuco, Brazil

[4] Federal University of Alagoas (UFAL), Maceió, Alagoas, Brazil

[5] Pontifical Catholic University of Rio de Janeiro (PUC-Rio), Rio de Janeiro, Rio de Janeiro, Brazil