Position Paper: Collaborative Gamification Design for Scientific Software

Francisco Queiroz
Tecgraf Institute and Arts & Design Dept.
PUC-Rio
Rio de Janeiro, Brazil

Rejane Spitz
Arts & Design Dept.
PUC-Rio
Rio de Janeiro, Brazil

***Abstract*— Gamification, a design trend that is extensively applied to education and citizen science, is regarded as a means to improve scientific software usability. However, development and use of scientific software have special needs and characteristics that might present design challenges. Our position is that gamification and usability design for scientific software should be facilitated by an open, collaborative design process supported by conversational media. We believe this approach is compatible with qualities often attributed to computational science community regarding openness and collaboration between members of varied professional backgrounds. Through an illustrative scenario, we exemplify the use of conversational media for collaborative design. We expect the synergy between collaborators to result in better usability, greater user acceptance, and adequacy to requirements, obtaining optimal design solutions in a sustainable way.**

***Index Terms*— Scientific software, gamification, usability, open design, collaboration.**

I. Introduction

Wolff [49] has proposed that gamification could improve scientific software usability – an aspect that many researchers consider neglected in that type of software [22], [1], [40]. Despite gamification being a relatively recent trend, the notion of bringing video game-like interfaces to scientific software is not exactly novel: in the early 2000's, Houstis and Rice have predicted that, by this current decade, Problem Solving Environments (PSEs) would resemble video games [16]. In fact, over two decades ago, PSEs were already applying 3D and multimedia technologies in data visualization [2]. Moreover, young scientists have expressed the desire for game-like capabilities in scientific software over ten years ago [17].

Recently, gamification of STEM software seems to be advancing in engineering applications – especially CAD and BIM (Building Information Modeling) software – which take advantage of video games' technologies, mechanics and aesthetics [23]. Gamified engineering applications can provide more compelling experiences [4] and ease of learning through playing, [24] or even by making games [28]. Gamification has also been proposed as a mean to improve software engineering practices such as requirement elicitation [13] and, in the specific case of scientific software, community building [18].

More recently, gamified support forums have been used to foster the exchange of knowledge among scientific software users [51], [52]. Forums like these typically make use of reputation systems to encourage user participation.

Gamification is also extensively, and very successfully, applied to science education [30] and citizen science [41], where it is used to engage the general public into collecting and analyzing scientific data for research purposes. Scientific software, however, can be a very particular, idiosyncratic field regarding development and use. It seems important, then, that gamification initiatives and methods are adequate to scientific software development environment, culture, and particular challenges. This study reflects the current state of an ongoing doctoral research and is primarily concerned about proposing adequate tools and methods to assist with scientific software usability and gamification design. We believe that open and collaborative design can address those challenges by providing opportunities for all stakeholders to take action and have a voice in that discussion. In fact, collaborative design appropriately reflects a spirit that has been strongly present in the computational science community for over half a century [12].

This paper is structured as follows: *Background* presents information on research topics and, also, preceding studies conducted by the authors, that inform and build up to the present discussion. *Agenda* presents our position on collaborative design for scientific software gamification. *The Design Board* proposes and exemplifies the use of conversational media for approaching scientific software usability design. *Conclusion* summarizes the paper, also presenting opportunities for future research and a call to action.

II. Background

In this section, we present background information on gamification and contextualize this study by presenting findings from previous stages of its underlying research.

A. Gamification

Gamification has been defined as "the use of game design elements in non-gaming contexts" [9]. In some cases, that means reconstructing an activity as a game, often by employing points, levels, scoreboards, winning conditions, goals, and so forth as motivational affordances [14]. This approach is often called *gameful design* [9]. On the other hand, gamification might also denote the use of game-like aesthetics and interactivity, and privilege the transposition of the highly interactive quality of games into usability design through the use of cross-media references [32]. The present research primarily aligns itself with this approach for two main reasons: first, video games have been, for over 40 years, a testbed for pleasurable interactive experiences – not only because of their inherent challenge provided by gameful elements, but largely for presenting interesting and compelling interfaces between player and game. Second, often justified as means to engaging users, design elements dedicated to performance evaluation – usually associated with gameful design – such as points and scoreboards, can be, in fact, demotivating [33], and less engaging than visual *stimuli* and interesting aesthetics [19]. Furthermore, scientific software users have been described as a highly motivated group [20] – in which case gameful motivational affordances might be, at times, unnecessary, or even counterproductive.

However, that does not mean that gameful design should be discarded, as it can also be used to reinforce an activity's structure and translate game design into user-centric interaction [8]. Work performed with scientific software can have its own structure, motivations, preferable outcomes, time pressure, distinct phases, recognition, and so forth. In this case, gameful elements are best when used as feedback for the actual structure and progress of scientific work – and not as a second arbitrary structure overriding it

B. Using Video Games as Inspiration for Scientific Software

As a first experiment in scientific software gamification within our doctoral research, we have conducted a four-month-long action research within a development group for a software dedicated to visualization and simulation of oil & gas production fields [35]. Throughout that study, we have looked for design elements that could be ported from video games into the software in question, in order to improve usability and interactivity. We have searched for desirable features in games depicting similar activities (e.g., tycoon and building simulators), games featuring desirable functionalities (e.g., time manipulation), and games supporting similar devices (in our particular case, Nintendo's *Wiimote* controller).

In some cases, we had to adapt selected game design elements to scientific software specific needs. The *equipment positioning tool*, for instance, was inspired by building placement tools found in titles such as *FarmVille* and the *Sim City* series. However, whereas researched games restricted building placement to fixed slots within a grid, the software's placement tool should allow for greater levels of precision. For that reason, in addition to numerical input fields, a new mechanic was designed: upon a directional key press, the increment to the object's position would be inversely proportional to its closeness to the user's point of view. In this case, zooming the camera into the object would allow for more accurate positioning.

Then, we have showcased new functionalities through an interactive prototype. However, despite perceived improvement in functionality and usability, prototyped features were not built into the actual software, as difficulty and cost for implementation were considered too high given development team size and software production tools. In this case, although that study was fruitful in terms of exploring solutions inspired by video games, it failed to acknowledge and embrace scientific software development characteristics such as software complexity, difficulties in requirement elicitation [37], and incremental changes [48]. Overall, that study's outcome suggested the need for additional research on (a) scientific software characteristics and (b) on methods for better collaboration and communication between designers, developers, and other stakeholders.

C. A Design Lens for Scientific Software

In an effort to identify challenges and opportunities for scientific software design, we have conducted a literature review on scientific software usability, development, and gamification. Based on its findings, we have proposed *a Design Lens* for scientific software. Design Lenses were first elaborated by game designer Jesse Schell [38] as a series of principles for designers to have in mind in when planning their games from a particular perspective. Lenses are typically formatted as a statement on a particular topic, followed by questions for the designer to reflect upon. Table 1 shows the Lens of the Lab, generated through our literature review [34], which approaches issues related to development, use, professional context, gamification practices, science, and academia.

The Lens of the Lab is meant to be a design aid to designers and developers envisioning interface functionalities and evaluating its feasibility, as it should serve as a reminder of which aspects and stakeholders to consider when designing interfaces for scientific software, gamified or otherwise.

TABLE I. THE LENS OF THE LAB

The Lens of the Lab
Scientific software should augment insight, productivity, and knowledge. It should facilitate and integrate supported stages of scientific work (modeling, simulation and result analysis), and generate output for publication, sharing, or further research. When designing for scientific software, consider the questions:
• How can the interface represent the scientific matter, reinforce the way it works and support the theory behind it? How can it present and explore complex data at high levels of precision? How can it prevent and fix errors? • Is the user interface intuitive, consistent and uncluttered? Is it flexible enough to allow for incremental expansion and customization? Is it adequate to the platforms it was designed for, and to other software it should be integrated to? • How do scientists work? How is the work environment, culture, ethics, conventions, current practices and best practices? What do users need and expect? How can design embrace different levels of scientific specialization, computer literacy, and programming skills? How can it promote and attract collaboration or community building? • How can games inform and inspire the software aesthetics and interactivity? Which game design elements could provide structure, goals, feedback, guidance, progression, flow, fun and experimentation? Would competition and point-based systems motivate or demotivate? • Is implementation feasible regarding scope, planning, timescale, technologies, human resources, and software lifecycle?

III. Agenda

Having identified main characteristics of scientific software development and use – and their potential impact on usability design – we were left with the need for identifying and generating methods for collaboration and communication between developers, users, designers and other members of software's development community. We have found in collaborative, participative, and open design, the potential for conducting informed and sustainable design practices.

A. Collaborative and Open Design in Scientific Software

Scientific software is usually developed for a very specialized group of users. Understanding their needs and how they work is a major concern for those in charge of usability [31], [43], [6]. Bringing users into the design process can be a way of gathering requirements, validating ideas and, overall, ensuring the design is adequate. There are numerous documented examples of successful collaborative efforts in scientific software design. The *Project Community*, for instance, is a platform for all project stakeholders, users, and even external members to contribute and discuss requirements and other usability issues of *OMERO* software [26]. Community-based solutions and discussion tools were also contemplated by the European Middleware Initiative (EMI) when envisioning an open source community for scientific applications [10]. The development process for *STAR* software shows how a collaborative effort supports iterative design [11].

More recently, researchers from the University of Illinois have promoted a *hackathon* for taxonomists, developers and information scientists to co-design the interface of a taxonomy software [46]. Initiatives like these are successful in responding to series of challenges in scientific software development such as: responding to the needs of an actual user-base and allowing them to add value to the software [7]; balancing and integrating information from different disciplines [5], different visions for the project [42] and individual notions of authorship [47]; dealing with scientists' busy schedules [21], [11].

In some cases, collaboration can be extended from design to programming. That is the case for *biok*, a programmable software for biologists, which was co-designed and co-developed by its users [25]. This approach seems to integrate, in a consistent and coordinated way, two concepts: (1) the philosophy behind open design and (2) the spirit of "professional end user developer". Open design celebrates the democratization of design, a "do-it-yourself" mentality, the cult of the *amateur*, and shared knowledge between amateurs and professional designers [3]. Furthermore, it promotes innovation in thinking and making design in a way that mirrors scientific thinking [36]. It is worth noting that open design takes inspiration from the open source movement which, in its turn, is rooted in computational science, a pioneering field for open source which has been fostering collaboration for over 60 years [12]. As in open design, scientific software development often carries a "do-it-yourself" mindset, which can be exemplified by the figure of the "professional end user developer": a domain-expert who is apt to develop software for himself or his community, but is not formally trained in software engineering, hence adopting an informal development process [39].

It seems clear that scientific software development can benefit from formal software engineering practices [15], [29]. However, it should be acknowledged that professional end user developers have, indeed, explored creative solutions when designing and developing software that is adequate to scientific inquiry. Collaborative and open design could be, then, a channel and a laboratory for their ideas, allowing them to contribute to the design process from the perspective of their specialized knowledge and skills.

B. Open Gamification Design

We propose that scientific software usability issues could be adequately addressed by open gamification design: open access, for all stakeholders, to the conceptualization and planning of gamified functionalities in a collaborative manner, in order to make the design phase as informed as possible. We are not proposing, however, that the design process itself should be gamified – although we would not have any objections to that.

In the context of open design, the role of the designer would shift from single-handedly conceiving the end product to creating, researching, organizing and facilitating the design process [45]. The designer could also apply his professional skills and design literacy in refining, researching, and generating content for user interfaces – a role that others might regard as "the last thing a scientist wants to deal with" [27, 497]. For those reasons, we believe that the design process should ideally include a professional from that area, or at least a team member from another specialization willing to play that role.

Regarding the relationship between game design and open design, it is worth noting that game design elements and interactivity patterns are commonly sampled, borrowed and adapted from one game to another. That applies in particular to user interface design, which can take advantage from copying (and adapting) user interfaces of successful titles, with "the benefit of being a familiar interface to your users" [38, 274]. That also happens with *game design patterns* commonly found in many titles and genres that have become design conventions (e.g., Boss Monsters, Levels, Time Limit, Bluffing, and so forth), which can help generating, structuring, refining and communicating design ideas [44]. This appropriation and modification of preexisting models and patterns strongly relate to open design culture.

IV. The Design Board

In this section, we propose the use of conversational media as a tool for collaborative gamification design of scientific software.

A. Overview

Conversational media allows users to exchange messages, documents, images and, in some cases, audio and video clips. Examples of conversational media include online services such as blogs and online forums; development-dedicated platforms such as *Confluence*, *Trello* and *Slack*; code hosting services such as *GitHub*; digital whiteboards such as *Realtimeboard* and

Stormboard; and even their analogic counterparts, such as whiteboards and noticeboards. Conversational media could be a platform for a *design board*: a channel for communication between developers, users, and other members of scientific software development community dedicated to discussing the software's usability and interface design. It should serve as a place for presenting usability needs and issues, proposing ideas, and discussing their feasibility. It should be a venue for dialogue between all stakeholders and the search for optimal solutions. It should be a place for the exchange of ideas, information, and the construction of a design vocabulary. It should be used as a repository for ideas on usability, design, and interactivity, informed by the specialized knowledge of every participant involved. While it could be a place for imaginative, experimental and speculative ideas, it should also be dedicated to evaluation, validation, and assessment of the viability of those ideas. Furthermore, although envisioned to assist with gamification processes, it could be applied to general usability and UI design.

B. Adequacy to Scientific Software Development

We propose that the design board should be used in conjunction with the Lens of the Lab, which provides a set of design issues and opportunities that are particular to scientific software. In that case, those issues and opportunities could be adequately addressed by specialists on the topics at hand, and also discussed from the perspectives of professionals from diverse backgrounds. Moreover, we believe the design board is suited to iterative development, small increments and emergent requirements that are typical of scientific software development.

C. Sustainability

Usability is an important aspect of sustainability [50]. The design board should foster more efficient, effective and satisfactory ways of user interaction. It is worth noting that the Lens of the Lab reinforces other sustainable aspects such as scalability and interoperability. Moreover, developers could potentially incorporate the design board into whatever preexisting conversational media is in use by the software community, reducing the need for additional infrastructure.

D. Design Board Features

Conversational media is a very inclusive term, which can describe a broad range of products and services. We believe the idea of a design board should not be exclusive or restricted to individual products. Instead, it could be adapted to any platform that allows participants to:

- Initiate and join public discussions.
- Publish and access supportive material (e.g., text documents, images).
- Search and/or browse past discussions.
- Access to appropriate design guidelines (i.e., the Lens of the Lab)

Those are general terms to describe popular functionalities, commonly found across platforms mentioned in the previous subsection. However, each platform might offer those functionalities in a particular way: initiating a discussion in *Trello*, for instance, requires the user to add a new card to a list. *GitHub*, on the other hand, would require the user to add a new issue to the project. Likewise, whereas the Lens of the Lab could be displayed in a dedicated list in *Trello*, it could be posted as a Wiki page on *GitHub*. Both platforms, however, should be apt to support the design board - even if presenting the necessary capabilities in different ways.

E. Illustrative Scenario

Through the next subsections, we present the working dynamics of a design board through a fictional case inspired by a usability issue reported by a user of an engineering software made by a university-based scientific software development team.

1) Issue / Request:

User A works with the engineering software 'X' to categorize hundreds of 3D geometry pieces from a CAD model. In order to do that, he must follow these steps: (1) turn on the 'picking' tool by clicking on the appropriate toolbar icon UI; (2) navigate the viewport for the 3D scene; (3) keep the CTRL key down to activate selection of multiple pieces; (4) click on 3D geometry pieces from a common category, thus highlighting the names of those pieces on a hierarchical tree view window; (5) Drag highlighted items from the tree view into a folder, named after the category items belong to, in a second hierarchical tree view window. He posts the following usability issue in the Design Board:

"I am having problems with the current workflow for categorizing geometry. First, having to keep the CTRL key down all the time feels tiring and unnecessary. Second, whenever I accidently release the key, it makes me unselect everything, and I have to start all over. Third, dragging names between hierarchical tree views often causes me to unselect everything".

2) Design Suggestion:

Designer B reads User A's issue. After reflecting on the Lens of the Lab (*How can games inform and inspire the software aesthetics and interactivity?*), he looks for inspiration in games that make use picking mechanics and, later, replies:

"After looking into games featuring mechanics for picking objects, I've found that Popcap's casual game Bookworm (Figure 1) *had an interesting mechanic: you could select pieces by clicking on them and double-click the last one to submit the selection, triggering an animation effect portraying selected pieces being directed to outside the board. You can try it at* www.mousebreaker.com/game/bookworm.

P*erhaps we could do the same, but instead of animating pieces, we could highlight the folder named after the selected category".*

3) A Developer's Perspective:

Developer C has been following the discussion. He also reflects on the Lens of the Lab (*Which game design elements could provide feedback and progression? Is implementation feasible?*). He decides to contribute to the discussion:

"I like your suggestions, but highlighting a layer on the hierarchy tree view might be unfeasible given our UI development framework. Maybe we could display an on-screen message such as '100 objects have been added to Category A layer, and also add an object counter beside that layer's name in the tree view. Also, we could add a counter for uncategorized items".

Fig. 1. Popcap's Bookworm game. Source: www.popcap.com

4) *Mockups, Diagrams, and Prototypes:*

After all participants have expressed their satisfaction with the idea proposed, Designer B starts producing ways of demonstrating the proposed solution – which could be done through UML use-case diagrams, low or hi-fidelity prototypes, illustrative mockup screens, and so forth. Later, Designer B posts an illustration of the solution designed (Figure 2).

5) *Implementation:*

Once approved by all discussion participants (and other community members, if required), the feature can have its implementation planned according to the methodology used by the development team.

F. Potential Gains

We expect design boards to improve software usability and development process in a number of ways.

1) *Better Usability:*

The discussion of ideas supported by the design board could lead to better products informed by participants' remarks on innovative practices, industry standards, case studies, and professional experience. Combined knowledge and availability of design references could lead to insights on usability and interactivity.

2) *Better Compliance and User Acceptance:*

Close involvement of all stakeholders should facilitate the gathering of all sorts of requirements, special needs and specifications, mitigating the risk of impediments and the need for redesign.

3) *Less time from design to implementation:*

We believe the design board should foster active collaboration between participants, accelerating the design process.

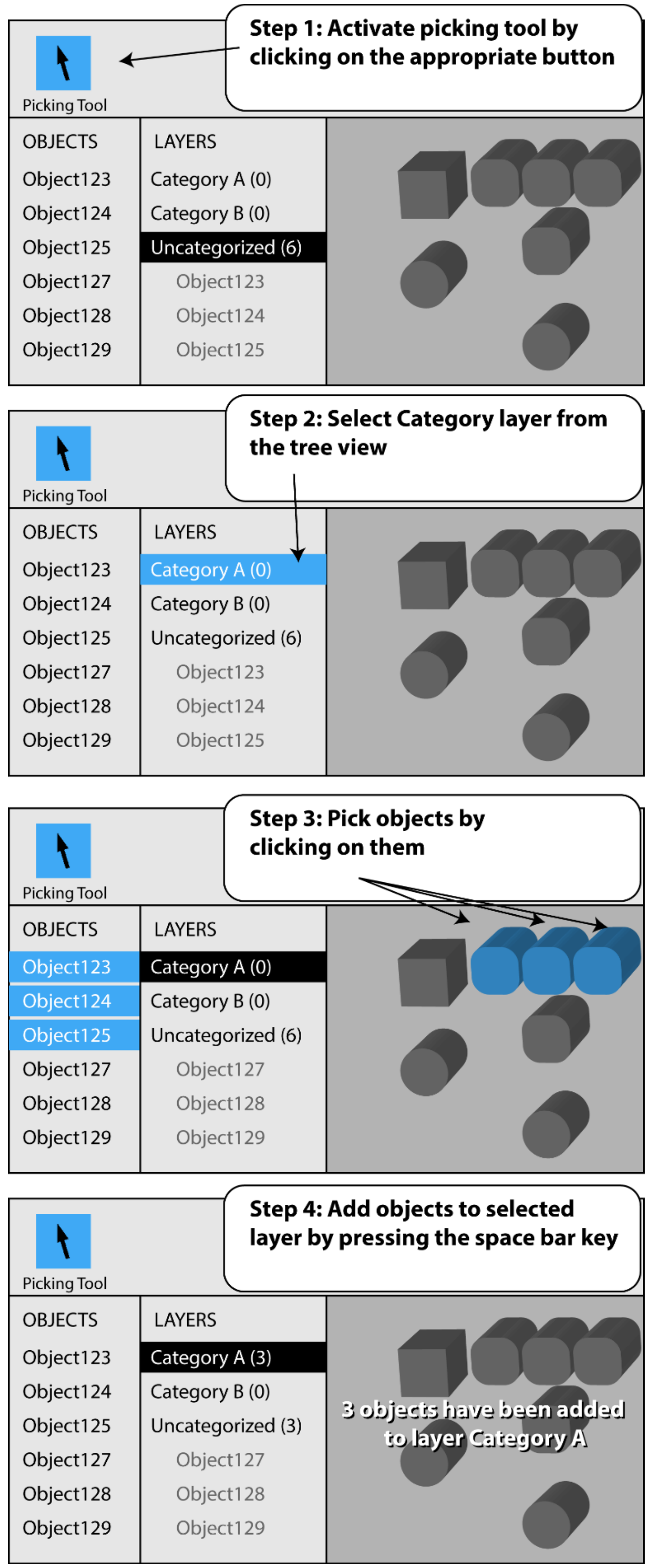

Fig. 2. Illustrative mockup screens

V. CONCLUSION

Gamification has the potential to improve scientific software usability. However, that requires attention to characteristics of that type of software regarding both use and development, also demanding clear communication between users, developers, and other software community members. Having those challenges, opportunities and obstacles in mind, we have positioned ourselves in favor of gamification design supported by open and collaborative design processes. Encouraged by a number of successful initiatives, we have argued for an inclusive, open design process that, in consonance with scientific software development culture, values participation and input from developers, specialists, and users alike. In addition, we have positioned ourselves for gamification processes primarily concerned about improving usability through video-game inspired interactivity – an approach that is also in consonance with open design. We have, then, proposed and exemplified ways for conversational media to support that process, based on dialogue and the articulation of knowledge imparted by all participants. As a result, we expect a well-informed design phase capable of delivering satisfying results by respecting iterative design, small increments, system complexity and other characteristics of computational science.

As the underlying research moves forward, further research will focus on testing both the Lens of the Lab and the design board in scientific software development and also on finding ways of reinforcing the structure of scientific work with gameful design elements. Moreover, we would like to encourage all members of the scientific software community to try using design boards and the Lens of the Lab in their projects and give us their feedback, suggestions, and comments on whether and how those tools might have helped to improve their software's usability in a sustainable manner.

ACKNOWLEDGMENT

We would like to thank the anonymous reviewers for their feedback and suggestions on how to improve the quality of this paper. Also, we would like to thank WSSSPE4 Organizing Committee and The Gordon and Betty Moore Foundation for their travel funding support. Finally, we thank PUC-Rio and Tecgraf Institute for making this research possible.

REFERENCES

[1] Z. Ahmed, S. Zeeshan, and T. Dandekar, "Developing sustainable software solutions for bioinformatics by the 'Butterfly' paradigm," F1000Research, vol. 3, 2014.

[2] V. Anupam and C. L. Bajaj, "Collaborative multimedia scientific design in SHASTRA," in Proceedings of the first ACM international conference on Multimedia, 1993, pp. 447–456.

[3] P. Atkinson, "Orchestral manoeuvres in design," in Open Design Now, why design cannot remain exclusive. Amsterdam, Netherlands: BIS Publishers, 2011.

[4] S. Boeykens, "Using 3D Design software, BIM and game engines for architectural historical reconstruction," in CAAD Futures, Liege, Belgium, 2011.

[5] C. Chen, J. Zhang, and M. S. Vogeley, "Reflections on the Interdisciplinary Collaborative Design of Mapping the Universe," in Human-Computer Interaction. Interacting in Various Application Domains, Springer, 2009, pp. 693–702.

[6] P. De Matos, J. A. Cham, H. Cao, R. Alcántara, F. Rowland, R. Lopez, and C. Steinbeck, "The Enzyme Portal: a case study in applying user-centred design methods in bioinformatics," BMC bioinformatics, vol. 14, no. 1, p. 103, 2013.

[7] D. De Roure and C. Goble, "Software design for empowering scientists," Software, IEEE, vol. 26, no. 1, pp. 88–95, 2009.

[8] S. Deterding, "The lens of intrinsic skill atoms: A method for gameful design," Human–Computer Interaction, vol. 30, no. 3–4, pp. 294–335, 2015.

[9] S. Deterding, D. Dixon, R. Khaled, and L. Nacke, "From game design elements to gamefulness: defining gamification," in Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments, 2011, pp. 9–15.

[10] A. Di Meglio, F. Estrella, and M. Riedel, "On realizing the concept study ScienceSoft of the European Middleware Initiative: Open Software for Open Science," in E-Science (e-Science), 2012 IEEE 8th International Conference on, 2012, pp. 1–8.

[11] G. O. Domik and K. D. Mickus-Miceli, "Software design and development in a scientific environment: lessons learned during the development of STAR, an astrophysical analysis and visualization package," in Astronomical Data Analysis Software and Systems I, 1992, vol. 25, p. 95.

[12] J. Dongarra, G. H. Golub, E. Grosse, C. Moler, and K. Moore, "Netlib and NA-Net: Building a scientific computing community," IEEE Annals of the History of Computing, vol. 1, no. 2, pp. 30–41, 2008.

[13] J. Fernandes, D. Duarte, C. Ribeiro, C. Farinha, J. M. Pereira, and M. M. da Silva, "iThink: A Game-Based Approach Towards Improving Collaboration and Participation in Requirement Elicitation," Procedia Computer Science, vol. 15, pp. 66–77, 2012.

[14] J. Hamari, J. Koivisto, and H. Sarsa, "Does gamification work?– a literature review of empirical studies on gamification," in System Sciences (HICSS), 2014 47th Hawaii International Conference on, 2014, pp. 3025–3034.

[15] D. Heaton and J. C. Carver, "Claims about the use of software engineering practices in science: A systematic literature review," Information and Software Technology, vol. 67, pp. 207–219, 2015.

[16] E. N. Houstis and R. John, "Rice, On the Future of Problem Solving Environments," Computational Science, Mathematics, and Software, 2002.

[17] H. Javahery, A. Seffah, and T. Radhakrishnan, "Beyond power: making bioinformatics tools user-centered," Communications of the ACM, vol. 47, no. 11, pp. 58–63, 2004.

[18] D. S. Katz, "Building Scientific Software Communities", HEP Software Foundation Workshop, Menlo Park, CA, 2015.

[19] D. L. Kappen, J. Johannsmeier, and L. E. Nacke, "Deconstructing'gamified'task-management applications," in Proceedings of the First International Conference on Gameful Design, Research, and Applications, 2013, pp. 139–142.

[20] D. F. Keefe, "Integrating visualization and interaction research to improve scientific workflows," Computer Graphics and Applications, IEEE, vol. 30, no. 2, pp. 8–13, 2010.

[21] D. Kelly and S. Smith, "3rd CASCON Workshop on Software Engineering for Science," in Proceedings of the 2010 Conference of the Center for Advanced Studies on Collaborative Research, 2010, pp. 420–422.

[22] R. P. Kendall, D. E. Post, J. C. Carver, D. B. Henderson, and D. A. Fisher, “A proposed taxonomy for software development risks for high-performance computing (HPC) scientific/engineering applications,” 2007.

[23] Z. Kosmadoudi, T. Lim, J. Ritchie, S. Louchart, Y. Liu, and R. Sung, “Engineering design using game-enhanced CAD: The potential to augment the user experience with game elements,” Computer-Aided Design, vol. 45, no. 3, pp. 777–795, 2013.

[24] W. Li, T. Grossman, and G. Fitzmaurice, “Gamicad: a gamified tutorial system for first time autocad users,” in Proceedings of the 25th annual ACM symposium on User interface software and technology, 2012, pp. 103–112.

[25] C. Letondal and W. E. Mackay, “Participatory programming and the scope of mutual responsibility: balancing scientific, design and software commitment,” in Proceedings of the eighth conference on Participatory design: Artful integration: interweaving media, materials and practices-Volume 1, 2004, pp. 31–41.

[26] S. Loynton, J. M. Burel, D. Sloan, and C. Macaulay, “The ‘Project Community Approach’ to Academic Scientific Software Development.”. 2009.

[27] M. Lundstrom and G. Klimeck, “The NCN: science, simulation, and cyber services,” in Emerging Technologies-Nanoelectronics, 2006 IEEE Conference on, 2006, pp. 496–500.

[28] M. E. A. Marasco, L. Behjat, and W. Rosehart, “Integration of Gamification and Creativity in Engineering Design.”. 2015.

[29] A. F. Mohammad, “A new perspective in Scientific Software Development,” in Innovations and Advances in Computer Sciences and Engineering, Springer, 2010, pp. 129–134.

[30] B. J. Morris, S. Croker, C. Zimmerman, D. Gill, and C. Romig, “Gaming science: the ‘Gamification’ of scientific thinking,” Frontiers in Psychology, vol. 4, 2013 [Online]. Available: http://journal.frontiersin.org/article/10.3389/fpsyg.2013.00607/abstract. [Accessed: 12 – Aug – 2016]

[31] C. M. Pancake, “‘Improving the Usability of Numerical Software through User-Centered Design,” Quality of Numerical Software: Assessment and Enhancement, ed. B. Ford and J. Rice, Chapman & Hall, 1996.

[32] Popa, D. “Gamification of Productivity Software: A User-Centered Design Approach,” presented at the CHI 2013 Workshop: Designing Gamification: Creating Gameful and Playful experiences, Paris, France, 2013.

[33] C. Preist, E. Massung, and D. Coyle, “Competing or aiming to be average?: normification as a means of engaging digital volunteers,” in Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing, 2014, pp. 1222–1233.

[34] F. Queiroz, R. Spitz, “The Lens of the Lab: Design Challenges in Scientific Software”, The International Journal of Design Management and Professional Practice, vol. 10, no. 3, pp. 17-35, 2016.

[35] F. Queiroz, R. Spitz, P. Elias, R. Pinheiro, T. Azevedo, V. Rodrigues, L. Reis, A. Raposo, “Video games as inspiration for scientific software,” unpublished.

[36] M. Ratt, “Critical Making”, in Open Design Now, why design cannot remain exclusive. Amsterdam, Netherlands: BIS Publishers, 2011.

[37] R. Sanders, “The development and use of scientific software,” Masters thesis, Queen’s University, Kingston, Ontario, Canada, 2008.

[38] J. Schell, The art of game design: a book of lenses, Second edition. Boca Raton: CRC Press, 2015.

[39] J. Segal, "Some Problems of Professional End User Developers", in IEEE Symposium on Visual Languages and Human-Centric Computing, 2007, pp. 111–118.

[40] J. Segal and C. Morris, “Developing software for a scientific community: some challenges and solutions,” Handbook of Research on Computational Science and Engineering: Theory and Practice, pp. 177–196, 2011.

[41] B. Simmons, C. Lintott, K. Masters, A. Greenhill, G. Graham, and K. Holmes, “Defining and Measuring Success in Online Citizen Science: A Case Study,” Computing in Science & Engineering, vol. 17, p. 28, 2015.

[42] M. Spencer, “Brittleness and Bureaucracy: Software as a Material for Science,” Perspectives on Science, no. 4, 2015, pp. 466-484.

[43] R. R. Springmeyer, “Applying observations of work activity in designing prototype data analysis tools,” in Visualization, 1993. Visualization’93, Proceedings., IEEE Conference on, 1993, pp. 228–235.

[44] S. Bjork and J. Holopainen, Patterns in game design, 1st ed. Hingham, Mass: Charles River Media, 2005.

[45] P. J. Stappers, F. S. Visser, and S. Kistemaker, ”, in Open Design Now, why design cannot remain exclusive. Amsterdam, Netherlands: BIS Publishers, 2011.

[46] A. K. Thomer, M. B. Twidale, J. Guo, and M. J. Yoder, “Co-designing Scientific Software: Hackathons for Participatory Interface Design,” in Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems, 2016, pp. 3219–3226.

[47] M. Turk, “Fostering Collaborative Computational Science,” Computing in Science & Engineering, vol. 16, no. 2, pp. 68–71, 2014.

[48] G. Wilson, D. A. Aruliah, C. T. Brown, N. P. Chue Hong, M. Davis, R. T. Guy, S. H. D. Haddock, K. D. Huff, I. M. Mitchell, M. D. Plumbley, B. Waugh, E. P. White, and P. Wilson, “Best Practices for Scientific Computing,” PLoS Biology, vol. 12, no. 1, p. e1001745, 2014 [Online]. Available: http://dx.plos.org/10.1371/journal.pbio.1001745. [Accessed: 12 – Aug – 2016]

[49] C. Wolff, “The Case for Teaching ‘Tool Science,’” in 2015 IEEE Global Engineering Education Conference, Tallinn, Estonia , 2015, pp. 932–938.

[50] C. Venters, L. Lau, M. Griffiths, V. Holmes, R. Ward, C. Jay, C. Dibsdale, and J. Xu, “The blind men and the elephant: Towards an empirical evaluation framework for software sustainability,” Journal of Open Research Software, vol. 2, no. 1, 2014, pp 1-6.

[51] E. Afgan, D. Baker, M. van den Beek, D. Blankenberg, D. Bouvier, M. \vCech, J. Chilton, D. Clements, N. Coraor, C. Eberhard, and others, “The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2016 update,” Nucleic acids research, doi: 10.1093/narlgkw343.

[52] D. Blankenberg, J. Taylor, and A. Nekrutenko, “Online resources for genomic analysis using high-throughput sequencing,” Cold Spring Harbor Protocols, 2015, doi: 10.1101/pdb.top083667.