

Science as a game: conceptual model and application in scientific software design

Francisco Queiroz, Maria Lonsdale & Rejane Spitz

To cite this article: Francisco Queiroz, Maria Lonsdale & Rejane Spitz (2022) Science as a game: conceptual model and application in scientific software design, International Journal of Design Creativity and Innovation, 10:4, 222-246, DOI: [10.1080/21650349.2022.2088623](https://doi.org/10.1080/21650349.2022.2088623)

To link to this article: <https://doi.org/10.1080/21650349.2022.2088623>



© 2022 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.



Published online: 28 Jun 2022.



Submit your article to this journal [↗](#)



Article views: 634






View related articles [↗](#)



View Crossmark data [↗](#)



Science as a game: conceptual model and application in scientific software design

Francisco Queiroz ^a, Maria Lonsdale ^a and Rejane Spitz ^b

^aSchool of Design, University of Leeds, UK; ^bDepartment of Arts & Design, Pontificia Universidade Católica do Rio de Janeiro, Brazil

ABSTRACT

Scientific inquiry is often described as, and compared to, a game. This paper expands on that analogy to propose a conceptual model of scientific practice built upon Jesper Juul's game definition, and informed by parallels between the two activities collected from selected works from history and philosophy of science. Moreover, the paper presents a design method, based on the model described, for fostering creative solutions in scientific software user interface design. Results from pilot case studies suggest both model and method are helpful, allowing participants to describe requirements and ideate solutions, as well providing a framework for the exploration of the game-science analogy within the context of scientific research conducted through computational resources.

ARTICLE HISTORY

Received 19 April 2021
Accepted 26 May 2022

KEYWORDS

Scientific software;
philosophy of science;
conceptual model; user
interface design;
gamification

1. Introduction

Models are created across scientific fields to represent functional and structural aspects of researched phenomena, allowing scientists to elaborate innovative theories and experiments. Often initiated from cross-domain analogies, models can be represented as diagrams, visual representations, and/or mathematical formulae (Nersessian, 2008), and can also be used in design innovation and problem-solving (Bila-Deroussy et al., 2017). In this paper, we explore analogies between games and science to generate a conceptual model of scientific work – a task undertaken with the help of selected texts from history and philosophy of science, through which we identify systemic and ontological similarities between games and scientific practice. Furthermore, we describe a pilot study on the application of that model into workshops for designing solutions in scientific software, motivated by one of the investigators' previous experience with scientific software, and also by the perception of that type of software as particularly challenging regarding usability – including installation, data input, poor documentation and lack of intuitiveness (List, Ebert, & Albrecht, 2017) – as well as the potential of gamification as a way of improving it.

This study aims at two objectives. First, we aim at moving toward the validation of the model. To do so, we analyze the outcomes of design workshops informed by the model itself. The following question is then put forward: Are workshop participants' accounts of their work within computational science – and the solutions they propose – consistent with the concepts and relationships presented by the model and theoretical framework of reference?

Second, our objective is to investigate the usefulness of the game-science analogy by presenting a novel method for communicating issues, identifying needs, and generating ideas in scientific software user experience. In this case, the question raised is as follows: Is the method useful for participants to explore issues and conceptualize solutions?

This paper is structured as follows: *Section 2* discusses previous uses of games as model and inspiration for scientific practice and software; *Section 3* presents the construction of the conceptual model and its theoretical framework of reference; *Section 4* describes the rationale, methodology, and results of four pilot case studies; *Section 5* reflects on case study findings; *Section 6* presents a summary of the paper, limitations, and suggestions for future work.

2. Background and related work

2.1. Games as a model for science

Conceptual models can be applied to evaluate and improve the quality of scientific practice. Such is the case, for instance, of the *concept model of research* (Mårtensson et al., 2016), which could potentially be applied to scientific software design. The present investigation, however, aims at exploring game-like qualities of science, in which case the conceptual model should make more evident game-like elements and dynamics. Numerous analogies between science and games have been previously established, some of which could form the basis of a conceptual model. A well-known exemplar would be Kuhn's depiction of normal science as puzzle-solving in *The Structure of Scientific Revolutions* (Kuhn, 2012), which extensively draws parallels between the two activities. According to Kuhn, scientists have in science a compelling, challenging problem-solving activity governed by rules and possible solutions established by scientific paradigms. Similar analogies were drawn in McCain and Segal's *The Game of Science* (1988), aimed at introducing science to the general public, describing it as a game through concepts such as rules, players, activities, motivation, and results – respectively, representing the method; scientists; idea generation, theories and experiments; challenge, delight, and curiosity; and consequences to society, technology, and funding. Latour and Woolgar's *Laboratory Life*, on the other hand, highlights game-like strategy and competitiveness experienced by scientists (Latour & Woolgar, 1986), particularly regarding the production of papers. It is Cunningham's *Getting the game right: Some plain words on the identity and invention of science* (Cunningham, 1988), however, that explicitly proposes a game-inspired model for science through which to determine historical practices as genuine scientific practices, characterizing science as a structured activity governed by rules and procedures that should be obeyed by scientists aiming at achieving their goals. Cunningham's work sets itself apart for two reasons: first, for proposing – to the best of our knowledge – the only game-inspired model for science; second, for proposing an application for it. Cunningham's model, however, was not based on a formal, rigorous definition of games, but on game elements and characteristics identified by its author, a significant limitation shared by previous models and analogies which, despite indisputable merits (Kuhn's and Latour and Woolgar's analysis providing important contributions to the present study), describe structural and functional elements of games and their scientific counterparts in a partial manner, often omitting fundamental aspects of games, such as player's emotional attachment to the game's outcomes (in the case of Cunningham's model) or a separation between games and reality (in the case of McCain and Segal's).

2.2. Game-related approaches to scientific software

Similarities between scientific software and games, particularly electronic ones, have been pointed out for at least 20 years: Houstis and Rice (2000) predicted that, around the year 2020, the interactivity of certain types of scientific software would increasingly resemble video games. On the one hand, similarities rely on technical aspects such as the advanced use of computer graphics.

Indeed, scientific software employs, at least since the 1990s, resources that are common to games, such as multimedia, multi-user support, and 3D interaction (Anupam & Bajaj, 1993). On the other hand, it is the possibility of establishing and playing with internal rules that makes the comparison between games and scientific software valid. In that sense, descriptions of scientific software as playboxes (Feibush et al., 2000) and playgrounds (Larkin et al. 2009) reflect the degree of freedom found in both.

Conscious efforts in bringing scientific software closer to games could be divided into two approaches. First, one that focuses on tools, technologies, and interactivity patterns – a prolific approach that inspired solutions such as 3D virtual geovisual analysis of Mars (Wang et al., 2012) and immersive environments for entomology and medical imaging (Bergmann et al., 2017) and fetal medicine (Dos Santos et al., 2016). A second approach would be the reframing of scientific software through structural game design elements (e.g., points, levels, rankings) – a design practice known as *gamification* (Blohm & Leimeister, 2013), whose effectiveness can vary according to area of application and user characteristics (Hamari et al., 2014). Despite suggestions that gamification could improve scientific software, literature seems to be scarce on fully realized examples, despite providing case studies of conceptualization of redesigned tools featuring gamified elements (Hutson et al., 2016; Queiroz et al., 2017). This scarcity could reflect technical challenges and complexity involved in software development (Dubois & Tamburrelli, 2013) – particularly scientific software – as well as potential shortcomings of gamification practices which, despite achieving popularity and recognition, have been criticized as potentially exploitative and demotivating (Bogost, 2014; Kim, 2015). The maturing of gamification, it has been observed, requires greater understanding of target activities and attention to systemic quality of games, and not the mere application of game design elements.

2.3. Summary

On the one hand, game-inspired models for science proposed thus far seem limited, given the lack of a formal definition to support them. On the other hand, previous attempts at bringing scientific software closer to games focus on adapting game technologies, esthetics, and structure without necessarily identifying inherently game-like aspects of scientific practice. Through the model proposed in the next section, we aim at addressing those limitations by establishing a model that could foster alternatives to more traditional gamification practices. For a more objective approach, our model will be based on an established definition of games, as proposed by Juul (2005).

3. From games definition to a game-based model of scientific practice

To establish our game-based model of scientific practice, detailed throughout this section, we have (1) created a diagrammatic representation of Juul's game definition; (2) identified analogies and relationships between game defining elements and scientific practice, as informed by literature on history and philosophy of science; and (3) created additional diagrammatic representation to reflect the analogies and relationships identified in both scientific practice and scientific software (Figure 1).

Games, according to Juul, present six fundamental traits: '[a] game is a rule-based formal system with a variable and quantifiable outcome, where different outcomes are assigned different values, the player exerts effort in order to influence the outcome, the player feels attached to the outcome, and the consequences of the activity are optional and negotiable' (Juul, 2005, p. 36). From such definition, we could establish a systematic view (Figure 1 -A) through which to visualize defining traits and the relationships between them: players put effort into a rule-based system of optional consequences to real life; the system's rules constrain their effort and determine outcomes, which is valued in accordance with rules, and influence future players' efforts.

Our model relies on analogies between game-defining traits and their scientific counterparts. If, on the one hand, the concepts of 'game' and 'players' could be immediately associated with science and scientists, concepts such as 'effort,' 'rules,' 'outcomes,' and 'negotiable consequences' were, on

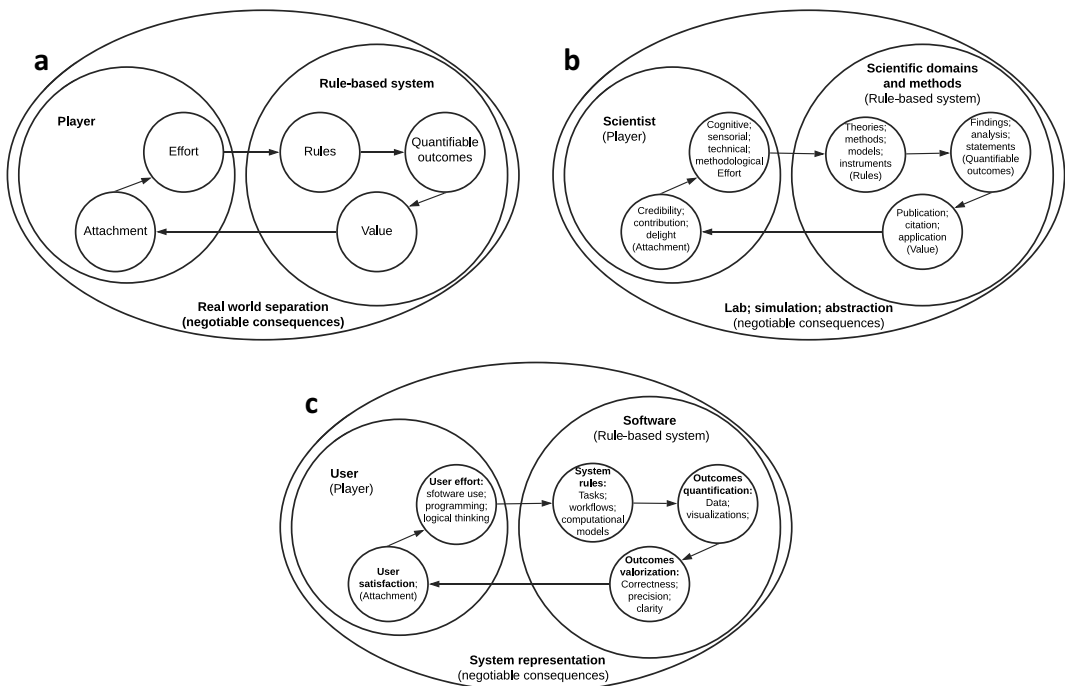


Figure 1. a: Systematic view of games, based on Juul's definition; b: Game-inspired model of scientific work; c: Game-inspired model of use of scientific software.

the other hand, investigated through bibliographic research on areas dealing with ontological, epistemological, and social aspects of science. Throughout the next subsections, we present those analogies, to establish a stronger comparison between science and games. The non-systematic bibliographic research was conducted by one of the authors. Aiming for diversity and historical perspective, we have selected works originally published between 1902 and 2014 by authors from science studies, psychology, history, and philosophy of science (Table 1). We have then conducted a directed content analysis (Hsieh & Shannon, 2005, p. 1281), using as initial coding categories the defining traits from Juul's definition. Finally, we have then related those aspects to characteristics of scientific software development and use, to support the application of the model in that domain.

3.1. Rule-based systems

Rules are sets of instructions that establish the structure of a game, how it should function, and how players' actions should be constrained. On the one hand, that is very similar to Kuhn's description of scientific paradigms and theories, in the way they limit 'acceptable solutions and the steps by which they are to be obtained' (Kuhn, 2012, p. 36). Still, according to Kuhn (2012, p. 41) – and Bachelard (1984, p. 13) before him – those rules could also be embedded into and enforced through scientific instruments. Theories on phenomena and systems can also be presented as models that illustrate the system's interaction (Nersessian, 2008, p. 12). In scientific software, those theories and models can be implemented as computational models (Daniluk, 2012), built upon a knowledge-base where information about those systems and underlying theories are stored (Keller & Rimón, 1992). The constant changes and evolution of scientific concepts (and how to best implement them) characterize emerging requirements (Segal & Morris, 2008), a trait of scientific software that makes it particularly challenging to develop.

Table 1. Sources included in bibliographic research.

Author	Source and original publication date
Henri Poincaré	Science and Hypothesis [1902] The Value of Science [1905] Science and Method [1908]
Gastón Bachelard	The New Scientific Spirit [1934] The Formation of the Scientific Mind [1938] The Philosophy of No: A Philosophy of the New Scientific Mind [1940]
Karl Popper	The Logic of Scientific Discovery [1934] Conjectures and Refutations: The Growth of Scientific Knowledge [1963] All Life is Problem Solving [1994] 2017
Thomas S. Kuhn	The Structure of Scientific Revolutions [1962] 2012 The Essential Tension: Selected Studies in Scientific Tradition and Change [1977] The Road Since Structure: Philosophical Essays, 1970–1993 [2000]
James D. Watson	The Double Helix: A Personal Account of the Discovery of the Structure of DNA [1968]
Paul Feyerabend	Against Method [1975] Science in a Free Society [1978] Farewell to Reason [1987]
Bruno Latour and Steve Woolgar	Laboratory Life: The Construction of Scientific Facts [1979]
Bruno Latour	Science in Action: How to Follow Scientists and Engineers Through Society [1987] Cogitamus: Six Lettres sur les Humanités Scientifiques [2010]
Karin Knorr-Cetina	The Manufacture of Knowledge: An Essay on the Constructivist and Contextual Nature of Science [1981] Epistemic Cultures: How the Sciences Make Knowledge [1999]
Nancy Cartwright	How the Laws of Physics Lie [1983]
Ian Hacking	Representing and Intervening, Introductory Topics in Philosophy of Natural Science [1983]
Evelyn Fox Keller	Reflections on Gender and Science [1985] Making Sense of Life: Explaining Biological Development with Models, Metaphors, and Machines [2003]
Peter Galison	How Experiments End [1987]
Sherry Turkle	The Second Self: Computers and the Human Spirit [1995] Simulation and its Discontents (with William J. Clancey, Stefan Helmreich, Yanni Alexander Loukissas and Natasha Myers) [2003]
Mihaly Csikszentmihalyi	Flow: The Psychology of Optimal Experience [1990] Creativity: Flow and the Psychology of Discovery and Invention [1996]
Howard Gardner, Mihaly Csikszentmihalyi and William Damon	Good Work – When Excellence and Ethics Meet [2001]
Lorraine Daston and Peter Galison	Objectivity [2007]
Nancy J. Nersessian	Creating Scientific Concepts [2008]
Catelijne Coopmans, Janet Vertesi, Michael Lynch and Steve Woolgar (eds.)	Representation in Scientific Practice Revisited [2014]

Rules, on the other hand, manifest themselves in scientific domains as methods, procedures, and methodologies followed by scientists. As described by Popper, 'Methodological rules are here regarded as *conventions*. They might be described as the rules of the game of empirical science' (Popper, 2005, p. 32). Computational science relies on tasks, often organized, automated, and diagrammed as workflows (Woollard et al., 2008) – which can improve the reproducibility of experiments (Sochat et al., 2017). Henceforth, we refer to both methodological rules and theories enforced by scientific software as *system rules*.

3.2. Negotiable consequences

The term 'negotiable consequences' denotes a separation between the activity and the real world. In the case of games, this notion was conveyed by Huizinga's concept of a 'magic circle' where games took place away from ordinary life (Huizinga, 1949). In that sense, Bachelard observed that modern science distanced itself from real-world phenomena, into mathematical abstraction (Bachelard, 1984, p. 60). Also, separate from the real world are laboratorial conditions and equipment that allows for phenomena to be produced. In the words of Latour and Woolgar, 'It is not simply that phenomena *depend on* certain material instrumentation; rather, the phenomena are *thoroughly constituted* by the material setting of the laboratory' (1986, p. 12). Scientific phenomena is, then, presented and represented through theories and laboratorial practices. Such instrumentation now includes computational resources, capable of generating simulations that might even prescind from real-world manifestation. As put by Knorr-Cetina, scientific domains might '(...) operate within and processes this artifactual reality, it moves within a medium of simulations and material "fictions" according to its own designations. Yet (...) truth effects can be derived (through an alternate epistemology), technologies can be put into effect, and universes can be "understood"' (2009, p. 249). Despite the separation between certain aspects of scientific practice and the real world, it is worth remembering that those universes might influence each other, either by the application of science and technology in the world (Hacking, 1983, p. 31), or by the influence of institutions, policies, industries, and society at large might exert over scientific practice (Latour, 1987, p. 162). Moreover, the performance of the scientist might affect their career.

Scientific software is, in its turn, a manifestation of science conducted in isolation to the real world. It runs as a computational simulation (Woollard et al., 2008), through the programming of digital and virtual environments and labs (Wang et al., 2010). Henceforth, we refer to both the transposition of theories into code and the ways in which they can be visualized and interacted with in scientific software as *system representation*.

3.3. Variable and quantifiable outcomes

Outcomes are the end result of players' actions within the game. Experiment results, analysis, and scientific statements are arguably their scientific counterparts. Indeed, Popper described 'the *creation of satisfactory theories* as the *goal of science*' (2013, pp.14–15). Latour and Woolgar would seem to agree, stating that 'the name of the scientific game is to push a statement (...) as far as possible toward fact-like status' (1986, p. 181). Scientific statements, however, are usually supported or initiated by data obtained from successful experiments, now promoted to evidence or demonstration (Galison, 1987, p. 1). The role of scientific software in the generation of quantifiable outcomes includes the output of data, data visualization, and result analysis (Milewicz et al., 2019). We will, therefore, use the term *outcomes' quantification* to describe the role of scientific software in calculating, communicating, and/or visualizing those outcomes.

3.4. Valued outcomes

The value of scientific outcomes is usually measured in publications but also in real-life application. On the one hand, several authors emphasize how important it is, for scientists, to have their work published and cited. In academic life, according to Knorr-Cetina, 'Success in making things work is a much more mundane pursuit than that of truth, and one which is constantly turned into credits in scientific everyday life via publication' (1981, p. 4). That success, Latour would argue, depends on citations and references from subsequent articles: 'To survive or to be turned into fact, a statement needs the next generation of papers' (Latour, 1987, p. 38). The value (and validity) of a scientist's

work must be confirmed by their peers (Kuhn, 2012, p. 163) – after the scientists themselves have assessed it: ‘it is we [scientists] ourselves who, after severe scrutiny, decide upon the answer to the question which we put to nature’ (Popper, 2005, p. 280).

On the other hand, as Hacking points out, scientific discovery might be valued for its application in real life: ‘Theories try to say how the world is. Experiment and subsequent technology change the world’ (Hacking, 1983, p. 38). In scientific software, though, *outcomes’ valorization* might depend on aspects such as correctness, which is a high priority in scientific software development (Heaton & Carver, 2015, p. 2), precision (Hatton & Roberts, 1994), and validation (Segal & Morris, 2008), as well as its contribution to insight.

3.5. Effort

In search of better outcomes, scientists might put effort into their practice in many different ways. Arguably, it starts with familiarizing themselves and learning about their scientific domain, its practices, methods, theories, traditions, and skills (Latour & Woolgar, 1986, p. 54). Based on that knowledge, scientists will put forward their own questions and design their experiments (Popper, 2005, p. 280), cultivating the sense of objectivity that is central to scientific practice (Daston & Galison, 2007, p. 40). Through those experiments, they will collect data, which should be later visualized, analyzed, and presented. In that sense, understanding ‘data’ could involve outcomes as distinct as captured images of the scientific phenomena or interactive charts and graphs – both being ‘associated with the capacity to reveal what has hitherto been hidden’ (Coopmans, 2014, p. 37).

As far as the game analogy is concerned, it is an interesting peculiarity that a significant amount of scientists’ *effort* is in constructing, themselves, methodological and theoretical rules that shape their activities. By doing so, they (and their peers) help defining which and how outcomes should be quantified and valued – occasionally redefining the rules of the game. As put by Kuhn, ‘The scientist requires a thoroughgoing commitment to the tradition with which, if he is fully successful, he will break’ (1977, p. 234).

Experiments might also involve effort in developing and applying other skills, including sensorial, technical, and instrumental ones. They might involve, as put by Hacking, ‘speculation, calculation, model building, invention and technology in numerous ways’ (1983, p. xiii). Latour makes a similar point: ‘We are now in a puddle of blood and viscera (. . .) Now, we realize that many other manual abilities are required in order to write a convincing paper later on’ (1987, p. 66).

Scientists must also put effort into transforming all of that knowledge, methods, data, and experiments into insight, which demands problem-solving skills, such as the identification of patterns and anomalies, requiring ‘the solution of all sorts of complex instrumental, conceptual, and mathematical puzzles’ (Kuhn, 2012, p. 36). On the one hand, that might be achieved through reasoning and logical thinking – ‘the instrument of demonstration’ (Poincaré, 1921, p. 219) – but also from analogical thinking for identifying similarities between unrelated phenomena (Knorr-Cetina, 1981, p. 50), intuition and speculation (Popper, 2005, p. 280), as well as playful experimentation (Feyerabend, 1993, p. 17).

The scientific process demands a lot of writing – Latour and Woolgar described the lab as a ‘system of literary inscription’ (1986, p. 52), where scientists ‘spend the greatest part of their day coding, marking, altering, correcting, reading, and writing’ (1986, p. 49); and although the scientific paper, which requires persuasiveness, can be seen as the end-product of those efforts, they also include the note-taking that follows academic research. Moreover, in order to succeed, scientists might also need to put effort into managing resources (Latour & Woolgar, 1986, p. 229); time; and professional networks (Knorr-Cetina, 2009, p. 216).

Scientific software, in its turn, demands *users’ effort* that reflects many aforementioned skills – including the use of software itself as part of technical skills – and, indeed, is often designed to support those by providing access to knowledge-bases of the scientific domain (Badal et al., 2019);

visual programming and domain-specific languages for easier coding (Gu et al., 2021); workflows for configuring the sequencing of tasks (Atkinson et al., 2017); tools for data visualization and analysis (Coopmans, 2014); functionalities for error prevention, precision, annotation (Keefe, 2010); and collaboration (Da Rocha B Pinto et al., 2002).

3.6. Attachment to outcome

Juul (2005) describes the player's attachment to the game's outcome in terms of their satisfaction in winning (or dissatisfaction in losing). In that sense, the idea of gratification can also be found in science's intrinsic and extrinsic rewards. The former, generated from scientists' 'desire to be useful, the excitement of exploring new territory, the hope of finding order, and the drive to test established knowledge' (Kuhn, 2012, pp. 37–38) or – as in the case of math, from 'delights analogous to those given by painting and music' (Poincaré, 1921, p. 280). The latter, expressed in 'credibility [that] makes possible the conversion between money, data, prestige, credentials, problem areas, argument, papers, and so on' (Latour & Woolgar, 1986, p. 200).

Scientists' satisfaction (and arguably, game players' as well) derives not just from valued outcomes but from scientific work itself, which seems to elicit a sense of flow, feeling 'entranced by the thing they were doing; (...) they [forget] themselves and their immediate environment' (Knorr-Cetina, 2009, p. 170). Indeed, a sense of fulfillment could be derived from unexpected or negative outcomes – possibly from learning more about the problem at hand (Popper, 2013, p. 13), and occasionally for the possibility of encountering anomalies that might lead to a breakthrough (Kuhn, 1977, p. 173).

In scientific software, we argue intrinsic and extrinsic rewards (from either satisfactory results or pleasure from use) would be better described as *user satisfaction*.

3.7. Conceptual model

Based on the initial model and analogies that were established, we propose Figure 1 -B as a diagrammatic representation of science as a game, which serves as a conceptual model of scientific work. It also serves as an intermediate model between the systematic view of Juul's definition (Figure 1 -A) and a game-based model of scientific software use (Figure 1-c).

3.8. From conceptual model to design method

Concepts from the model described throughout previous subsections were transposed to the proposed design method through their incorporation in a toolkit devised for workshops. The toolkit consists of 77 cards. Three *stage cards* represent the typical stages of software-based research: modeling, simulation, and result analysis. Six *reflection cards* (Figure 2) feature questions for participants to reflect on. Those are related to, and named after, the defining elements of game-based scientific practice, transposed to scientific software (as outlined throughout the previous subsections: user effort; system rules; system representation; outcomes quantification; outcomes valorization; and user satisfaction).

Finally, 68 *probe cards* feature keywords and illustrations representing concepts associated with scientific practice, including interaction elements typical of scientific software such as workflow diagrams (Oinn et al., 2006), as well as traditional concepts from gamification design (see Figure 3 for a sample, and Table 3 for a complete list). Additional materials required for the activities include sticky notes, paper (A4 and A3 formats), ruler, pen, pencil, markers, glue, and scissors.

As it will be clearer in the next section, the toolkit, as the design method it supports, does not require participants to refer to the model in its diagrammatic illustration, rather relying on textual descriptions of the system elements it represents.

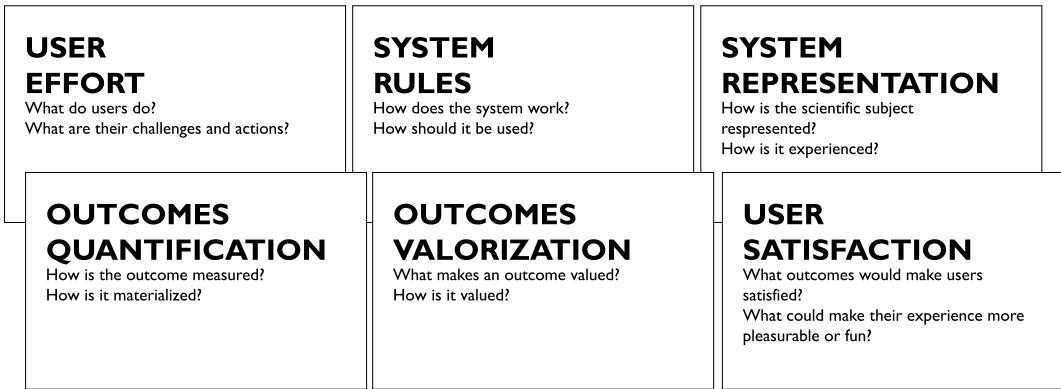


Figure 2. Reflection cards.

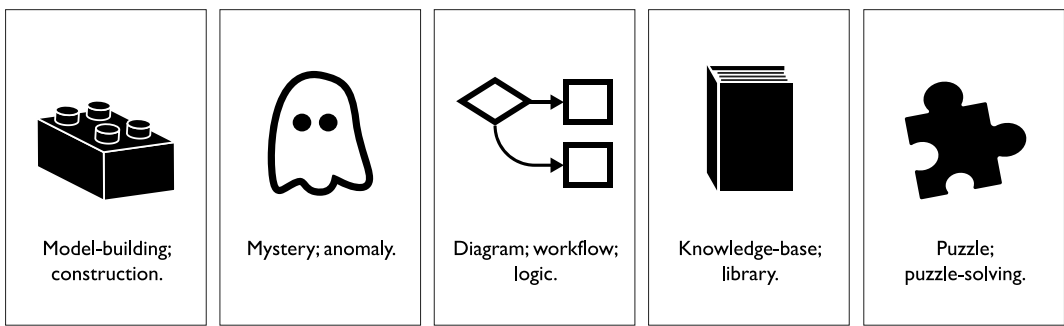


Figure 3. Probe cards (sample).

4. Methodology

4.1. Introduction

Usability and user interface design have been traditionally described as neglected aspects of scientific software (Ahmed et al., 2014; Lundstrom & Klimeck, 2006; MacLeod et al., 1992). Arguably, one of the reasons is the lack of perceived importance of usability in software quality from scientific software developers, who are often 'end users developers,' using code to develop their own tools, or to use third-party software (Johanson & Hasselbring, 2018). Users, then, are often developers, or at least programmers, for the tools they use, developing or extending software functionality. As put by List et al., 'software development as part of scientific research is usually carried out by individuals or small teams with no more than two or three members' (2017, p. 1). Given that particularity of scientific software use and development, we have designed a study that is qualitative and exploratory in nature, inviting users and/or developers of scientific software conducting research in diverse fields to one-to-one workshop sessions through which participants could reflect on scientific work and conceptualize better user experiences for the software they were involved with, based on the conceptual model proposed.

The small number of participants in this study reflects its focus on in-depth, qualitative data on highly specialized, individual experiences with their respective expert systems in different scientific domains. As such, this study relies on purposeful sampling, in which case participants were not chosen randomly, but based on the quality of data, they might provide, helped by their proximity to the topic of investigation (Leavy, 2017, p. 79). A similar approach was taken by Damen and Toh (2021), who investigated how designers structure and represent information through individual design sessions with a similar number of participants (four designers of similar areas of expertise).

However, our approach differs from Damen and Toh's in some ways: first, our study relies on heterogeneity sampling, aiming at maximum variation within the central theme of investigation (Patton, 2002, p. 235). In this case, despite reduced number of participants, those are diverse in terms of areas of research, degrees of specialization, and use of scientific software. Second, this study involves non-designers in the conceptualization of design solutions – an approach that is similar to Tran Luciani, Löwgren, and Lundberg's study on air traffic control interaction involving four one-to-one co-design sessions with air traffic controllers (2020). Unlike Tran Luciani et al., however, the present study does not aim at designing a single solution that can be extended to all participants, nor the evaluation of prototyped solutions.

Participants were recruited among users and/or developers of scientific software conducting research in diverse fields of study. The recruitment process involved (1) a call to participation sent to various Schools and Faculties across the University of Leeds; and (2) direct e-mail sent to individual researchers. Participants collaborated on a purely voluntary basis. Two sessions were conducted by a researcher and an assistant, and one session was conducted by a researcher only. Sessions lasted between 90 and 120 minutes, and were registered through notes and photographs. Subsequent individual meetings were arranged for further discussion and prototype iteration. Digital prototypes, when applicable, were developed by a researcher with professional experience in digital and interactive design.

Collected data and outcomes generated through the workshops were analyzed and compared to the model and theoretical framework. As in the case of the bibliographic research conducted to build the conceptual model, directed content analysis was employed to identify, within data collected (including sketches, cards used, verbal and written comments), themes and topics related to games' defining traits used as coding categories.

Throughout the next subsections, we present the workshop activities and generated outcomes.

4.2. Activity 1: Introducing participants and their contexts

4.2.1. Protocol

After introducing themselves and the study, researcher asks the following questions to participants:

- What is your current role/position?
- What is your line of research/work?
- How do you use scientific software for that work?
- What are aspects of scientific work you think are enjoyable/fun?

4.2.2. Outcomes

Despite a small number of participants, areas of expertise, levels of involvement with software development, roles, and specialization were diverse. Participant 1 (P1), a Postdoctoral Research Fellow at the Condensed Matter Physics Group, uses scientific software OriginPro 2017 (OriginLab corporation, 2017) for result analysis. P1 had been working in that research group for over 3 years at the time of this study. Participant 2 (P2), a Postgraduate Research Fellow, develops a process tomography software (Wang et al., 2017), used in industry, dedicated to visualization and analysis of flow inside reactors and pipelines. At the time of this study, P2 had been developing that software for over 4 years, with over 14 years of experience in software development. Participant 3 (P3), a Mathematical Physics PhD student, uses scientific frameworks in Python when elaborating algorithms for solving partial differential equations (PDEs) and generating geometrical visualization of the studied phenomena. P3 had 12 years of experience teaching undergraduate students several topics, including programming in the language used in his

research. Participant 4 (P4) is a PhD candidate in Social Psychology (in his fourth year) who champions the Research Computing group at the University, using High-Performance Computing (HPC) resources to analyze social media data using Deep Learning models that relate emotions to online activism and social movements. At the time of this study, P4 had been using programming for research for approximately 8 years. The four participants, therefore, are varied in terms of use and development of computational science resources, but can also represent natural sciences (P1), engineering (P2), math (P3), and social sciences (P4).

Participants expressed, from different perspectives, a sense of fun in finding answers: the feeling of being absorbed navigating and analyzing large amounts of data (P1); the speed through which one could go through data analysis (P2); the uncertainty in looking for answers – particularly when one is wrong (P3); and to ‘write code and see how everything works,’ besides ‘playing around’ with computers and supercomputers, and finding out the right way of doing the analysis (P4).

4.3. Activity 2: Describing issues and features in software

4.3.1. Protocol

Researcher asks participant to write down, on sticky notes, issues they might be currently having with software, or functionalities they would like to propose. Participants are given the *stage* cards to indicate phase (modeling, simulation, or result analysis), if necessary.

4.3.2. Outcomes

Most issues described were related to data visualization (Table 2), ranging from the need to generate multiple static graphs quicker (P1), finding ways to calculate and generate new graphs (P2), and interactive simulation and visualization in 3D (P3). P4, on the other hand, described issues related to the modeling phase: (a) time consumed using command line interface (CLI) to transfer files from an online repository to the university’s supercomputer where those files would help fine-tuning the model; and (b), difficulties accessing documentation and learning resources. P4 mentioned two additional issues: institutional approaches to open science, and inspection for Deep Learning models – both discarded given their scale and scope.

4.4. Activity 3: Exploring the problem

4.4.1. Protocol

Researcher provides all 68 probe cards to participant, placing them on a surface. Participant is asked to freely explore them and pick those related to the issue and potential solutions. Participants are free to interpret or subvert the meaning of cards as they wish. Participant is then asked to describe the relationship between picked cards and the issue on a sticky note.

Table 2. Issues/features described by participants.

Participant	Issue/feature	Stage
P1	‘Plotting multi panel graphs with shared axes/Generally changing parts of a plot quickly – setting scale etc.’	Results analysis
P2	‘implement a function for calculating and displaying a graph of the flow’s velocity’	Simulation/Results analysis
P3	‘How to simulate three-dimensional vector fields related to a PDE/Interaction with parameters.’	Simulation/Results analysis
P4	‘Moving data with GUI/Better documentation/Better promotion for training/Better training in containers’	Modeling

Table 3. Probe cards list.

Illustration	Keywords	Inspiration
Brain	Imagination; thought; reasoning	Science
Building	Institutions	Science
Ghost	Mystery; anomaly	Science
Light bulb	Idea; enlightenment	Science
Microscope	Scrutiny; augmentation; focus	Science
Quotation marks	Quotation; reference; statements	Science
Bar chart	Data analysis; data visualization	Science; software
Blackboard	Formula; theories; equations	Science; software
Book	Knowledge-base; library	Science; software
Box	Black box; self-contained; opacity	Science; software
Diagram	Diagram; workflow; logic	Science; software
Document file icon	Document; file	Science; software
Dot, line, square, cube	Dimensions; simplicity	Science; software
Eye	Vision; visuals	Science; software
Finger cursor	Interaction; manipulation	Science; software
Graph	Network; graph	Science; software
Image file icon	Image; photograph	Science; software
Lab	Work environment; lab	Science; software
Magnifying glass	Search; inspect	Science; software
Pencil	Writing; annotation; drawing; editing	Science; software
Question mark	Question; doubt; help	Science; software
Ruler	Measurement; compliance	Science; software
Swiss army knife	Versatility; improvisation; tools	Science; software
Target aim	Precision; accuracy	Science; software
Checklist	Tasks; checklist	Science; software; gamification
Circling arrows	Cycle; stages; repetition; automation	Science; software; gamification
Conversation bubbles	Conversation; dialog	Science; software; gamification
Gauge	Measurement; indicators	Science; software; gamification
Gavel	judgment; validation	Science; software; gamification
Globe	Global; community	Science; software; gamification
Group of people	Collectivity; collaboration; users	Science; software; gamification
Hurdle	Obstacle; impediment; setback	Science; software; gamification
Lego brick	Model-building; construction	Science; software; gamification
Waves	Flow; immersion	Science; software; gamification
Battery	Energy; resources	Science; gamification
Coins	Investment; resources	Science; gamification
Counter	Counter; score	Science; gamification
Crossed swords	Conflict; contest	Science; gamification
Dice	Probability; chance; randomness	Science; gamification
Hourglass	Time constraints; schedule; deadlines	Science; gamification
Inventory	Inventory; collection; taxonomy	Science; gamification
Jigsaw puzzle	Puzzle; puzzle-solving	Science; gamification
Military rank badge	Rank; achievement; hierarchy	Science; gamification
Ninja	Mastery; agility	Science; gamification
Numbered list	List; rankings	Science; gamification
Stairs	Progress; steps	Science; gamification
Trophy	Prize; reward	Science; gamification
Brackets; Programming	languages	Software
Programming display	Software; coding	Software
Ear	Listening; sound	Software

(Continued)

Table 3. (Continued).

Illustration	Keywords	Inspiration
Media player controls	Playback; control	Software
Two-way arrows	Portability; exchange; conversion	Software
Undo icon	Recovery; reversal	Software
Warning sign	Warning	Software
Wrench and screwdriver	Customization; maintenance; tools	Software
Fist	Strength; brute force	Software; gamification
Ladder	Support; shortcut; extension	Software; gamification
Map	Guidance; instructions	Software; gamification
Rocket	Speed; travel	Software; gamification
VR headset	Virtuality; technologies; avatar	Software; gamification
Shield	Protection; shielding	Software; gamification
Bomb	Disruption; destruction	Gamification
Checkered flag	Finish line; goal	Gamification
Flag	Milestone; flag	Gamification
Infinity symbol	Infinity; eternity	Gamification
Jack-in-a-box	Surprise	Gamification
Magician's top hat	Magic; spell	Gamification
Power-up mushroom	Power-up; boost	Gamification

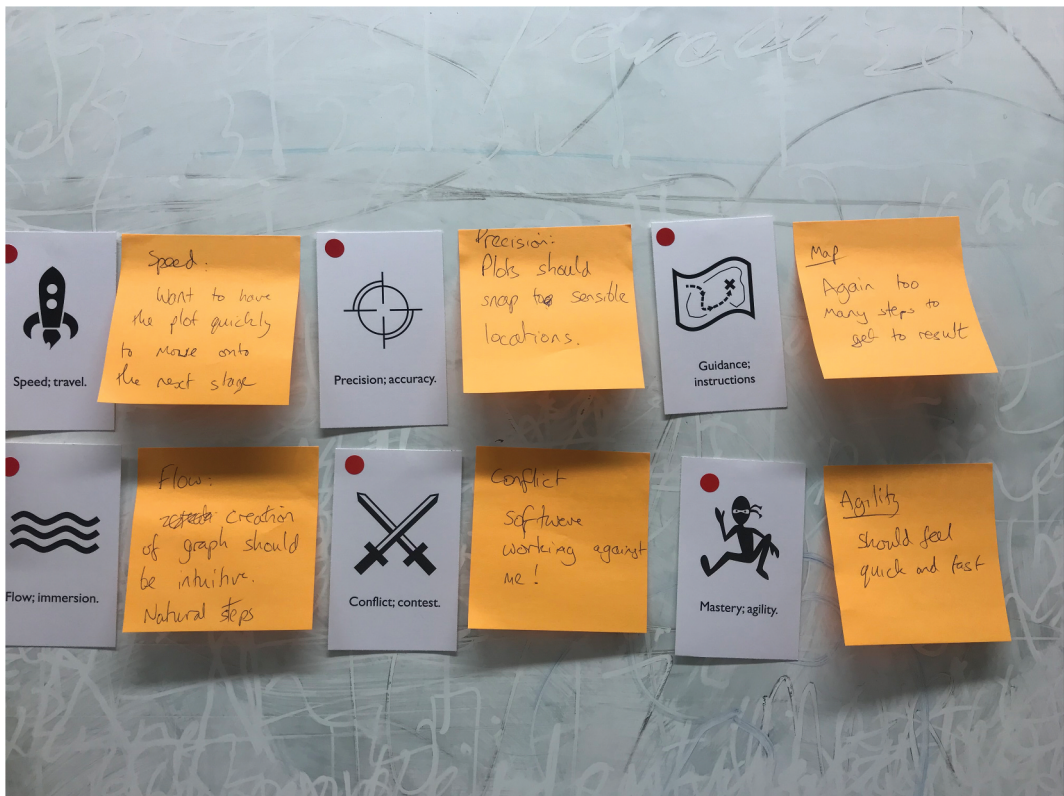


Figure 4. P1's chosen cards and notes.

4.4.2. Outcomes

Participants' approaches varied. P1 quickly summarized specific issues regarding the functionality he wished to improve (Figure 4), describing his desire for more agility ('Should feel quick and fast' – he wrote by the 'Mastery; agility' card), as well as frustration with the current interface ('Software working against me,' 'Conflict, contest' card). P2, on the other hand, described relevant aspects and features of his software, such as the way it simplifies multi-dimensional data, and how it allows for users to watch an animation of results ('Need to control the display of the results,' 'Playback; control' card). P3 provided an overview and reflections on his work, from the need to go through literature ('Knowledge-base; library' card), and solving problems through imagination and reasoning: 'specially in pure maths, the interesting problems are abstract problems so to research on must thought and imagine what someone else discovered solely with the paper.' Finally, whilst verbally communicating the issue (the time he feels is wasted doing and re-learning certain tasks), P4 provided short descriptions of the desired solutions ('GUI for data management,' 'HPC library inventory'). For a complete list of comments and selected cards, please refer to Table 4.

Table 4. Participants' probe cards and descriptions.

Part. Probe cards	Description
P1 [Mastery; agility] [Conflict; contest] [Speed; travel]	'Should feel quick and fast' 'Software working against me!' 'Want to have the plot quickly, to move onto the next stage.'
[Guidance; instructions] [Precision; accuracy] [Flow; immersion]	'Again, too many steps to get results' 'Plots should snap to sensible locations' 'Creation of graph should be intuitive. Natural Steps'
P2[Dimensions; simplicity] [Precision; accuracy]	'Multi-dimensional data fusion; decrease dimensions for better displaying' 'The results have to be precise to reflect multiphase scan dynamics'
[Model-building; construction] [Data analysis; data visualization]	'The software is built upon models.' 'The aim is to visualize the fused data and retrieve more information from the data'
[Black-box; self-contained; opacity] [Playback; control] [Image; photograph]	'The software is wrapped as a black box for users' 'Need to control the display of the results' 'The results are presented in images and compared to the images from the high-speed camera'
[Measurement; indicator]	'Flow measurements; criteria of what is a good result'
P3[Writing; annotation; drawing; editing] [Imagination; thought; reasoning]	'This is a loop in my day-to-day job: to write partial results; to annotate new findings; to draw something meaningful in the problem; to correct mistakes' 'Specially in pure maths, the interesting problems are abstract problems, so to research one must thought and imagine what someone else discovered solely with the papers'
[Knowledge-base; library] [Formula; theories; equations] [Vision; visuals]	'I read a lot of papers, looking for clues to solve small steps in a bigger problem' 'Everything in the literature is expressed with equations (PDEs and ODEs mostly)' 'My research problem is geometrical. It comes from physics and is an abstraction of the real 3D world'
[Finish line; goals] [Software; coding]	'It is the motivation to research (to achieve the goal)' 'To visualize the findings – or – to seek for guidance with respect of which direction to follow'
[Document; file]	'Documents store the information I need. I read a lot of them, or look at plots of physical or geometrical meaning'
[Puzzle; Problem-solving]	'In mathematics, research is like solving a puzzle. I have clues and a break the problem in pieces, afterwards I must figure out how to fit them together'
[Data analysis; data visualization]	'Because it is a geometric problem in a infinite-dimensional space reduced by symmetries.'
P4[Interaction; manipulation] [Knowledge-base; library] [Inventory; collection; taxonomy]	'GUI for data management' 'HPC training' 'HPC library inventory.'

4.5. Activity 4: Sketching solutions

4.5.1. Protocol

Researcher asks participant to sketch an interface that addresses issues and potential solutions expressed in the probe cards.

4.5.2. Outcomes

Participants' sketches varied in scope, complexity, and feasibility of implementation. P1 sketched a user interface intended to be a custom-made extension for Origin 2017 (Figure 5). P2 reproduced his software's interface, adding a new graph for displaying flow velocity – a functionality of high cost of implementation (Figure 6). P3 envisioned a completely new interface, including simulation playback controls, fields for manipulating parameters, viewports for interactive 3D visualization, annotations, animation, and image capture for publication manuscripts (Figure 7) P4 sketched two solutions: (a) a drag-and-drop interface to transfer multiple large dataset files between remote file storage system *OneDrive* and the university's HPC resource, and (b) a library of short HPC training videos (Figure 8).

4.6. Activity 5: reflecting on the solution

4.6.1. Protocol

Researcher directs participant to *reflection* cards, requesting them to write on succinct answers to each question on sticky notes. Participants are welcome to incorporate new details to their sketches.

4.6.2. Outcomes

Answers to reflection cards reflected each participant's approach to previous activities. P1 wrote specific, straightforward notes that reflect a single functionality and its *system rules*: 'Ask user for plot info (...) Type of presentation, [and] create empty plot'; as well as *user satisfaction*: 'Plot closer to expectations. Fewer button presses. Plot looks better.' P2 described the user experience of the software he develops, including how outcomes should be *quantified* ('phase distribution and phase decomposition') and *valued* ('compared to high-speed video logger; compared to references'); P3 provided the overview for a novel, complex visualization framework, which is reflected in his description of *system representation* ('(...) represented with graphics and experienced with a visualization interface of both the statics and dynamics of the system,' and *user effort*:

Users set up parameters, then run a mathematical simulation, and generate both the static configuration of the system, and the initial position of the vortex-antivortex pair. The users then run a simulation of the dynamics of the system. Challenge: to fine-tune the parameters.

Finally, P4 described easy-to-use drag-and-drop functionalities (taken for granted in most user interfaces but not HPC ones) and instructional videos, emphasizing the speed of learning and use. As his focus was on the modeling phase, when there is still no output data to be analyzed, less time spent on conducting it were considered the outcomes to be quantified and valued. Recurring themes included entering and manipulating parameters (user effort); calculation of results based on user input (system rules); graphs, images and statistics (system representation); data and plots of results (outcome quantification); validity of results (outcome valorization); ease of use, output quality, speed, and accuracy (user satisfaction). For a complete list of answers, please refer to Table 5.

4.7. Post-workshop follow-up

From participants' sketches and descriptions, one investigator produced digital interactive prototypes in *Adobe Animate* to simulate desired functionalities (Figure 9). Subsequent short meetings were scheduled on a case-by-case basis for further discussion and new iterations. Upon being

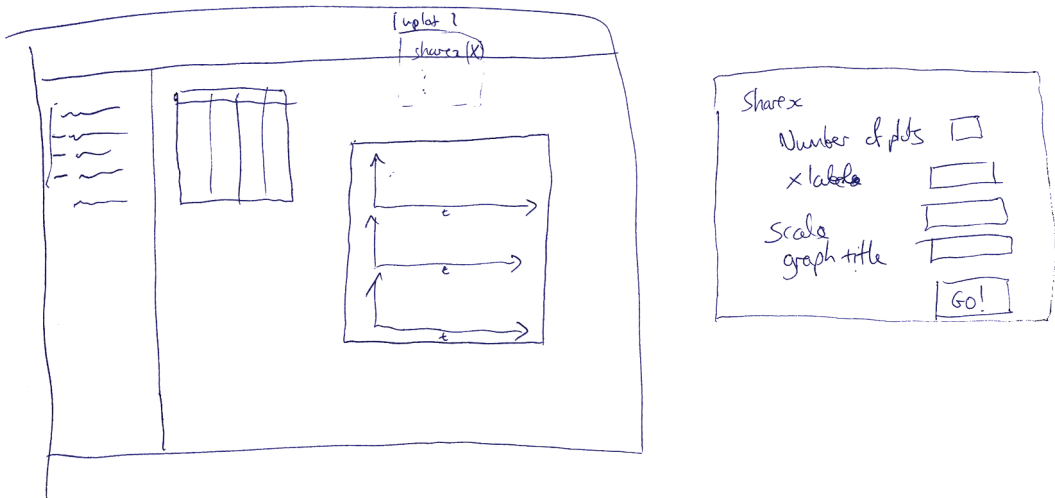


Figure 5. P1 user interface sketch.

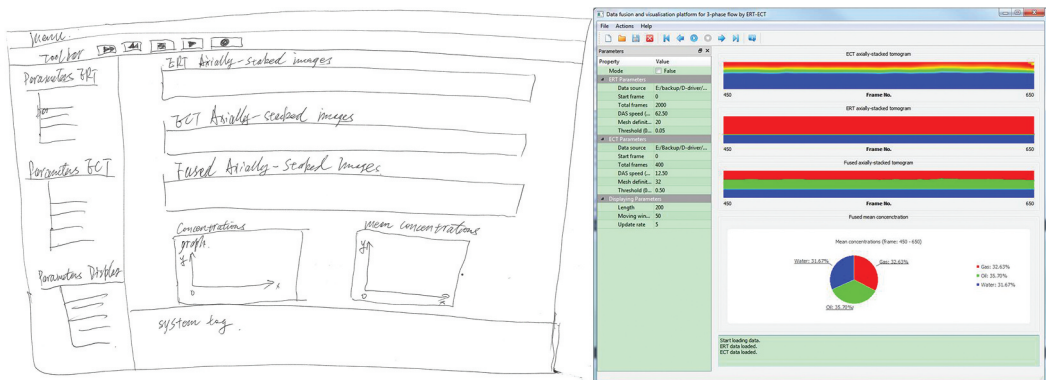


Figure 6. P2 sketch (left) and current interface from Wang et al. (2017)(right).

presented with an interactive version, participants would often come up with observations and ideas (P3, for instance, suggested that users should be able to drag particles on a graph). In one case (P2), suggestions included changes to the user interface's visual style (Figure 10). In the case of P4, the impossibility of using drag-and-drop tools (due to security reasons) inspired an online tool, prototyped in HTML and JavaScript, that generates programmatic command lines from files dragged into its interface, along with instructional video to explain its use (Figure 11).

5. Discussion

Overall, case studies corroborated similarities between games and science identified through our bibliographic research, also shedding light on the role of scientific software within that context.

Regarding participants' sense of effort, we observed an emphasis on a puzzle-like, problem-solving quality described by Kuhn (2012). This theme pervaded all case studies, including accounts of literature review and development of algorithms and code, but especially regarding result analysis conducted through data and scientific visualization. Perhaps more importantly, some users put effort into creating their own data displays for insight and publication. Indeed, most case studies

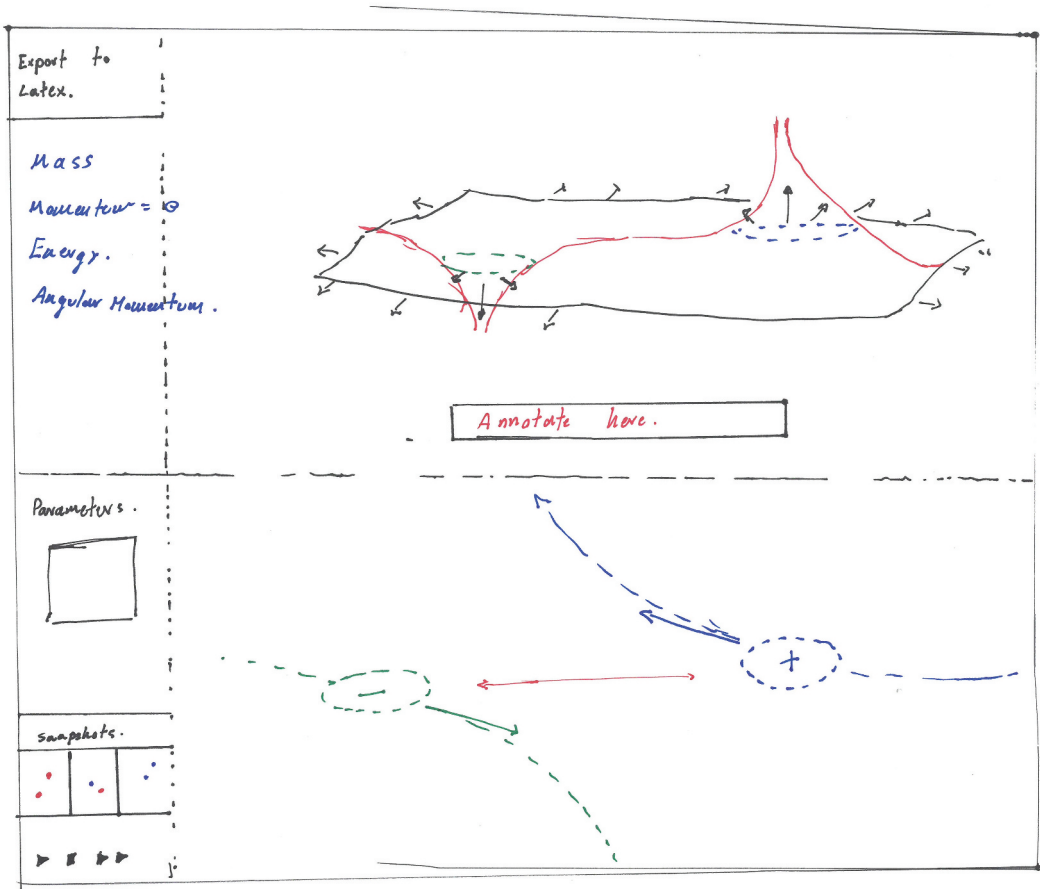


Figure 7. P3 user interface sketch.

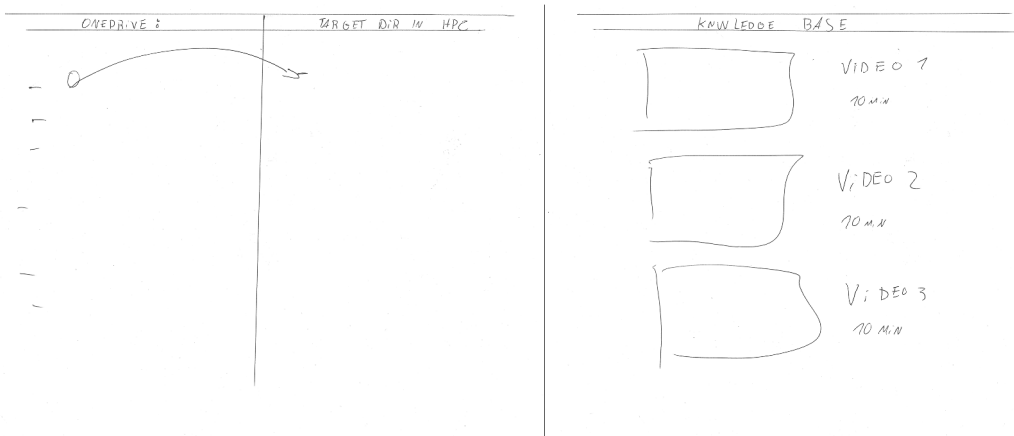


Figure 8. P4 sketches: drag-and-drop solution for transferring large datasets to HPC resources (left); Collection of explanatory videos for HPC users (right).

Table 5. Participants' answers to reflection cards.

	P1	P2	P3	P4 (issues A and B)
User Effort	'[uses menu option] up-lot <-click menu/share x <-click (or keyboard shortcut)/share y/ multipanel'	'(1) load data (1) input parameters (2) play with the results (3) have to have some background to input "valid" parameters'	'Users set up parameters, then run a mathematical simulation, and generate both the static configuration of the system, and the initial position of the vortex – antivortex pair./Then users run a simulation of the dynamics of the system/ challenge: to finetune the parameters'	(A) 'They drag files around' (B) 'Finding information, opening the website'
System Rules	'Ask user for plot info – label/scale. Type of presentation. Create empty plot'	'(1) It should present good results comparing to high-speed video (2) for offline data analysis and online monitoring'	'The systems start with plottings of the latest simulation. It remembers parameters and last visited location to save snapshots/it should be used by inputting data parameters, and once it has run first, it should be interactive, so the user can rotate or zoom-in and out the system'	(A)'File manager GUI for HPC to OneDrive' (B)'Step by step learning tutorial'
System Representation	'System Representation Advantage [it can] be standard size. Labels with right font etc.'	'The images from each modality should (be) fused as a single image, and mean concentration should be presented. The results should be exported.'	'The scientific subject is represented with graphics, and it is experienced with a visualization interface of both the statics and dynamics of the system'	(A)'Datafiles and Model files' (B)'Organized in steps'
Outcomes Quantification	'Outcome is a plot of user's choice'	'Phase distribution and phase decomposition'	'The outcome is measured with respect to known results in the literature./It is materialized with plotting and animations'	(A)'Time consumed in data management' (B)'Qualityofthe videos'
User Satisfaction	'Plot close to expectations. Fewer button presses. Plot looks better'	'If the results are reasonable comparing to reference'	'Nice plottings, easy to export in a LateX-friendly way/Animations of the dynamics of the system/ an intuitive interface that "learns" how the user interacts would make it more pleasurable'	(A)'Cause is faster' (B)'Fast troubleshooting'

point at data visualization output as the outcome for their work with software. In participants' accounts, as in Popper's assertion on the role of scientists in the evaluation of their work (2005), they were users and close associates who decided on whether outcomes should be deemed of value or not. At that stage, it is the scientist who upholds the criteria for outcome valorization and success – which could be either corroborated or refuted by peers in later stages. In a game of science, it would be fair to say that users do not approach software as a game, but as a tool or a stage within a game that stretches beyond that particular stage and tools, toward a wider ecosystem of software (as in the case of P4, who tried whose challenge was to get data *into* scientific software), machinery, and metrics for success. Although there is an explicit sense of progression (P1 wanted to 'move onto the next stage,' and P3, to 'achieve the goal'), the evaluation of users' performances and adoption of traditional gamification metrics (scores, badges, levels, etc.) could be challenging, and not a true

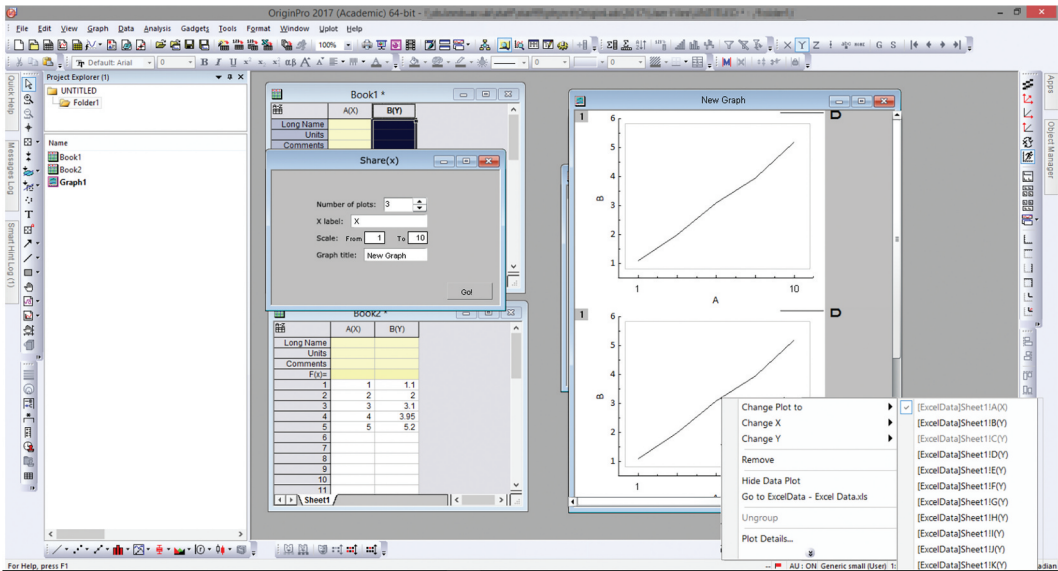


Figure 9. Interactive prototype based on P1’s concept and OriginPro’s original interface (OriginLab Corporation, 2017).

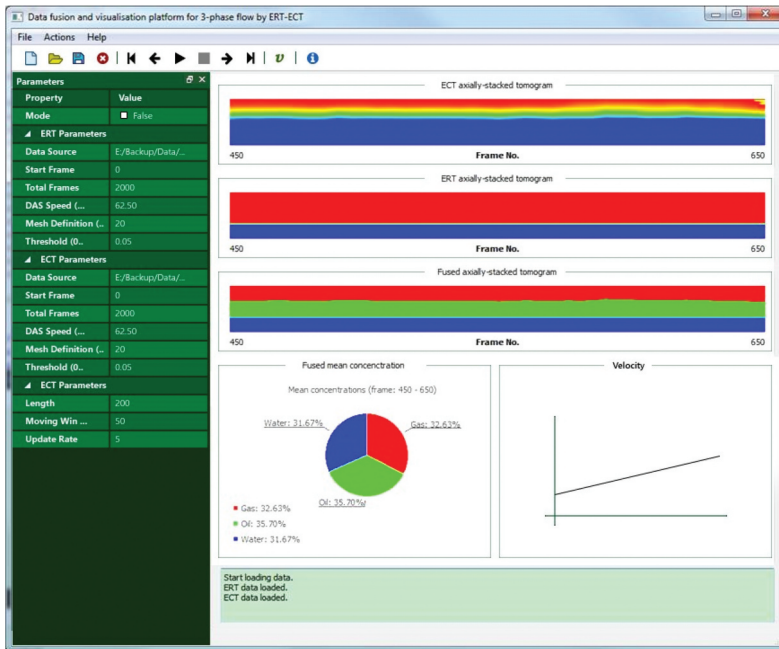


Figure 10. Interactive prototype based on P2 solution.

reflection of their progress or state of their research. Moreover, although a sense of progression is evident in P3’s description of literature – it can be a starting point (review), end goal (publication), and metric for researchers’ success (references) – reading, writing, and managing papers are activities undertaken through other types of software, and as far as scientific software is concerned,

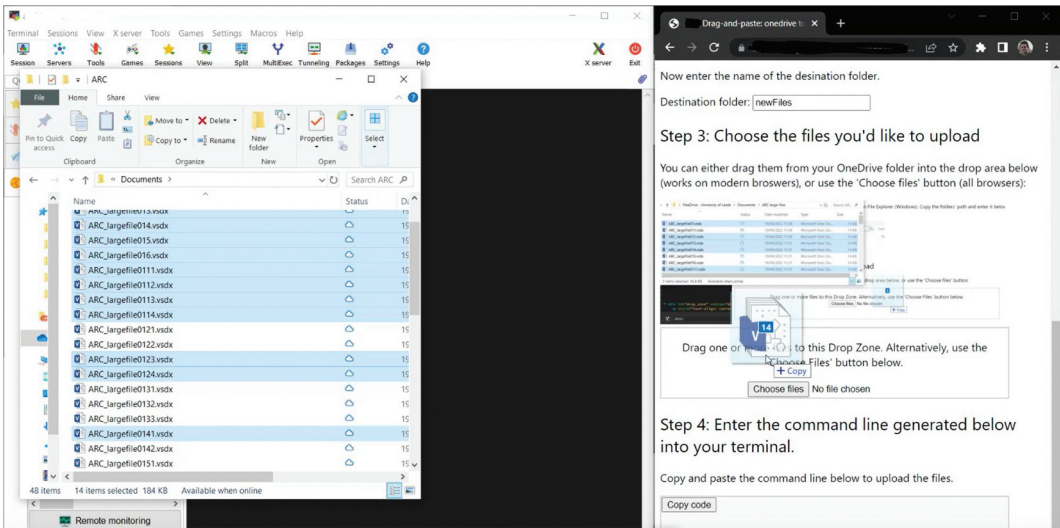


Figure 11. Still from instructional video of solution generated for P4, used in conjunction with terminal software MobaXterm (Mobatek, 2008).

most users' end goals seem to be outputting data and displays for insight and publication. P4 provided a different perspective, focusing on initial stages of computational science (modeling) and, before that, on learning how to 'play around with (...) supercomputers.'

The role of participants themselves in the evaluation of outcomes reflects their relationship with rules. As made evident by P3 – who codes algorithms that determine simulations' behavior – researchers actively design their methodologies, experiments, and models, making rules as much as following them. Arguably, the software developed by P2 is used in a more constrained way (as it is a 'black box' which users cannot freely manipulate). Still, its users are required to judge outcomes themselves based on their knowledge of the domain.

Regarding negotiable consequences, P1 and P2 cases presuppose laboratorial conditions for the monitoring of phenomena by instruments and a mathematization of the gathered data. A separation between real-world and scientific software, however, is most evident in P3's experience: first, for its level of abstraction; second, for not relying on real-world observation. A third reason, however, is related not to the nature of the work, but to the researcher's attitude: his ability to experience fun when failing to find an answer indicates a sense of meaningfulness and personal gain that goes beyond 'winning.' Juul argues that '[t]he freedom found in regular games can only be preserved if we are given room to experiment and the freedom to fail, at least temporarily' (Juul, 2013, p. 122). A similar sense of freedom seems to approximate scientific research to games, contributing to a sense of separation between that activity and 'real life.' A similar regard for intrinsic fun was present in P4's account of his pleasure in 'playing around' in experimenting with code.

Finally, regarding attachment to outcome, P1 and P3 emphasized their satisfaction with pictures that were clear and adequate for publication. P2 stressed the satisfaction of having adequate results, and all participants expressed a concern about increasing software's ease and speed of use – particularly P4, for whom the valued outcome would be time itself. P1 attempted to reduce the user's workload by getting the software to automate the plotting of multiple graphs. His proposed solution of manually activating contextual menus throughout a sequence of graphs, combined with the instant visual feedback provided by the graph itself, could foster a sense of flow through a rather repetitive task. Through this design decision, he manifested, perhaps unconsciously, the concepts of 'agility' and 'mastery'

present in the ninja probe card, attempting to minimize a sense of conflict ('software is working against me'). Interestingly, whereas video games usually establish a sense of conflict through obstacles and opponents, participants have described complex and clumsy interfaces as the main source of struggle within their work. P4, for instance, struggled with repetitive actions needed to transfer data files. P3, on the other hand, devised a tool for easily plotting, annotating, and exporting graphs – actions that are time-consuming through his tools of choice. P2's software allows users to control the playback of simplified graphs – a minimalist, more effective approach to visualization than complex methods such as cross-sectional views – which it suggests that – despite the excitement it could generate – the imposition of visually rich forms of representation inspired by video games could have, in some cases, a negative impact.

By encouraging participants to think their research in the terms proposed by reflection cards, the method seemed successful in highlighting structural elements of the model through which to consider the interplay between user and software, emphasizing how the interaction between them should deliver satisfactory outcomes and more pleasurable experiences.

All participants successfully ideated solutions for their issues, albeit through different approaches: P1 designed a single feature he would like to have in his software of choice – a feasible solution, given that software's tools for plug-in development. In that case, probe cards made issues clear and tangible, providing inspiration to potential solutions. P1 used the concepts expressed in the reflection cards to clearly dictate how users should complete a task, how software should behave and respond, and how the designed solution would provide outcomes and experiences that would be more satisfactory. Interestingly, although the concepts expressed in reflection cards originally applied to fully formed solutions, they were successful in providing a framework for elaborating a solution concerning a specific task.

P2's role as developer would seem to contradict a sense of user-centered design. However, since scientific software is 'developed by scientists for scientists' (Sletholt et al., 2012, p. 24), and borders between users and developers seem blurred in that area, we hoped to take advantage of that proximity, instigating the developer to adopt a user's perspective. In our experience, however, those roles were not interchangeable, and end-users were indeed more effective at identifying issues. Still, the method served as an interesting way of describing a software's features, challenges, and design decisions.

P3 ideated a complex solution that would demand much more effort and expertise to be properly developed – or even prototyped. In that case, probe cards pinpointed the most important goals and efforts in his work as a researcher (read, write, solve problems, imagine, and visualize), around which he envisioned several functionalities – In particular, the annotation tool, and image export button – without, however, describing most in detail.

Finally, P4 sketched solutions for easier use of HPC systems, supported by better documentation. Unlike other participants, P4 made *learning* those systems part of his goal, and chose probe cards to succinctly point at desired solutions rather than explain the problem in detail.

Recommendations favoring small, incremental changes in scientific software, Wilson et al. (2014), suggest that addressing a single functionality – as P1, P2, and P4 did – could be a productive, practical use of the method. P4's issue, for instance, was addressed not by a completely new solution, but a complementary step to an existing workflow, which indeed could be implemented. Still, its use in speculative design – an approach similar to P3's – should also be considered.

6. Conclusion

This paper proposed a game-inspired conceptual model for scientific inquiry, based on a systemic view of games' defining traits, and its application as a method for conceptualizing scientific software interfaces.

Through the proposed method, participants were encouraged to think of their research and work in terms of a game, hopefully attaining clarity on their goals and ways to achieve them. In many ways, participants' outputs reflected similarities between science and games identified by literature of reference: as a game, scientific work is permeated by challenges, goals, metrics, stages, rules, and fun. Such game, however, is not played through scientific software alone, being extensible to previous and subsequent stages of research, often cyclic or erratic, involving specialized literature, peers, advisers, equipment, institutions, and so forth. Additionally, scientific practice is complex, open to experimentation, and difficult to evaluate. In that case, the method could be proposed as an alternative to traditional gamification techniques in scientific software, such as borrowing game design elements as proposed by Hutson et al. (2016) and Queiroz et al. (2017). Moreover, the method does not aim at the sense of fun and engagement through gamification proposed by Wolff (2015), being closer to activity-based, systematic view proposed by Nacke and Deterding (2017), taking into consideration inherent aspects of scientific practice.

In more general terms, the contribution of this study resides in the initial mapping of the concepts relating science to games, as discussed by its corpus of references, into a conceptual model informed by an objective and unambiguous definition that sets it apart from previous attempts by Cunningham (1988) and McCain and Segal (1988), as well as the adaptation of that model to scientific software use.

Although limited to four pilot studies, findings from our research suggest that the proposed method could be helpful for users and developers to communicate issues and ideate solutions through a game-like perspective that is attuned to users' needs. The reflection cards seemed to communicate the structure of the game-like model, whereas probe cards helped participants to establish priorities, describe challenges, and conceptualize solutions.

The method seemed particularly suitable for addressing specific needs in an iterative fashion – which is considered the most adequate approach in scientific software development. The method could be refined through further research – the number of probe cards could be potentially reduced, and the systemic relationship between user, system, and outcomes emphasized. Moreover, it would be advisable to test the method through additional case studies and, also, within the wider context of software development processes, potentially extending and adapting the toolkit for use without the help of a mediator. In addition to the reduced number of participants, limitations of this study include the lack of investigation on subsequent stages of the design process (such as prototyping and evaluation), as well as the use of workshops as the only method for data collection (which could be complemented by interviews and questionnaires, for instance). Additionally, future studies could investigate the model and toolkit through a co-design study involving multiple participants to expand the discussion further, as the present study was limited to individuals. Finally, model and method could be adapted to other target activities rather than scientific software through the transposition of game defining traits to other domains, such as education and healthcare.

Acknowledgments

Authors would like to thank the School of Design, the School of Maths, the School of Physics and Astronomy, the School of Chemical and Processing Engineering, the School of Politics and International Studies, the Advanced Research Computing team, and the International Student Office at the University of Leeds; the Department of Art Design, the Central Coordination for International Cooperation, and the Tecgraf Institute at PUC-Rio. Additional photographs and notes from design sessions were taken by Tian Tian and Ana Signorini. The authors would like to thank Ahmed Nassereldin, Chelsea Sullivan, Hoyoung Hyun, Tian Tian, and Zheng Wang for their comments about the toolkit, Alessandro Medici for his comments on co-design, Antonio Augusto Passos Videira for his suggestions on Philosophy of Science, and Ken Friedman for his feedback on an earlier version of the manuscript. Finally, the authors would like to thank the editor and anonymous reviewers for their useful and constructive comments and feedback. This study was approved by the Arts, Humanities and Cultures Faculty Research Ethics Committee at the University of Leeds, reference LTDESN-081.

Disclosure statement

No potential conflict of interest was reported by the author(s).

ORCID

Francisco Queiroz  <http://orcid.org/0000-0002-2685-2653>

Maria Lonsdale  <http://orcid.org/0000-0003-0315-6169>

Rejane Spitz  <http://orcid.org/0000-0002-9837-3387>

References

- Ahmed, Z., Zeeshan, S., & Dandekar, T. (2014). *Developing sustainable software solutions for bioinformatics by the butterfly paradigm*. *F1000Research*, 3, 71. <https://doi.org/10.12688/f1000research.3681.1>
- Anupam, V., & Bajaj, C. (1993). SHASTRA-an architecture for development of collaborative applications. In [1993] *Proceedings Second Workshop on Enabling Technologies - Infrastructure for Collaborative Enterprises* 155–166.
- Atkinson, M., Gesing, S., Montagnat, J., & Taylor, I. (2017). *Scientific workflows: Past, present and future* (Vol. 75). Elsevier.
- Bachelard, G. (1984). *The New Scientific Spirit*. Beacon Press.
- Badal, V. D., Wright, D., Katsis, Y., Kim, H.-C., Swafford, A. D., Knight, R., & Hsu, C.-N. (2019). Challenges in the construction of knowledge bases for human microbiome-disease associations. *Microbiome*, 7(1), 1–15. <https://doi.org/10.1186/s40168-019-0742-2>
- Bergmann, T., Balzer, M., Hopp, T., Van de Kamp, T., Kopmann, A., Jerome, N. T., Zapf, M. (2017). Inspiration from VR gaming technology: Deep immersion and realistic interaction for scientific visualization 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, VISIGRAPP 2017 Porto, Portugal Vol. 3 Linsen, L., Telea, A., Braz, J. (SciTePress). doi:10.5220/0006262903300334.
- Bila-Deroussy, P., Bouchard, C., & Kaba, S. D. Addressing complexity in design: A systemic model of creativity and guidelines for tools and methods. (2017). *International Journal of Design Creativity and Innovation*, 5(1–2), 60–77. <https://doi.org/10.1080/21650349.2015.1116412>
- Blohm, I., & Leimeister, J. M. (2013). Gamification. *Business & Information Systems Engineering*, 5(4), 275–278. <https://doi.org/10.1007/s12599-013-0273-5>
- Bogost, I. (2014). Why gamification is bullshit. In S. P. Walz & S. Deterding (Eds.), *The gameful world: approaches, issues, applications* 65–79. MIT Press.
- Coopmans, C. (2014). Visual Analytics as Artful Revelation. In C. Coopmans & J. Vertesi (Ed.), *Representation in Scientific Practice Revisited* (pp. 37–59). Cambridge, MA: The MIT Press.
- Cunningham, A. (1988). Getting the game right: Some plain words on the identity and invention of science. *Studies in History and Philosophy of Science Part A*, 19(3), 365–389. [https://doi.org/10.1016/0039-3681\(88\)90005-2](https://doi.org/10.1016/0039-3681(88)90005-2)
- da Rocha B Pinto, G., Strauch, J., de Souza, J., Oliveira, J., Cardoso, L., Botelho, L., & da Justa Medeiros, S. (2002). A framework to support scientific knowledge management: A case study in agro-meteorology. In *The 7th international conference on computer supported cooperative work in design* 320–324.
- Damen, N. B., & Toh, C. (2021). Investigating information: A qualitative analysis of expert designers' information representation and structuring behaviors. *Journal of Mechanical Design*, 143(8): 081403. <https://doi.org/10.1115/1.4046647>
- Daniluk, A. (2012). Visual modeling for scientific software architecture design. A practical approach. *Computer Physics Communications*, 183 (2), 213–230. <https://doi.org/10.1016/j.cpc.2011.07.021>
- Daston, L., & Galison, P. (2007). *Objectivity*. Zone Books.
- Dos Santos, J. R. L., Werner, H., Ribeiro, G., & Belmonte, S. L. (2016). Combination of non invasive medical imaging technologies and virtual reality systems to generate immersive fetal 3D visualizations. In *International conference on digital human modeling and applications in health, safety, ergonomics and risk management* 92–99.
- 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* 659–662.
- Feibush, E., Gagvani, N., & Williams, D. (2000). Visualization for situational awareness. *IEEE Computer Graphics and Applications*, 20(5), 38–45. <https://doi.org/10.1109/38.865878>
- Feyerabend, P. (1993). *Against method*. Verso.
- Galison, P. (1987). *How Experiments End*. University of Chicago Press.
- Gu, Y., Tinn, R., Cheng, H., Lucas, M., Usuyama, N., Liu, X., , and Poon, H. (2021). Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1), 1–23. doi:10.1145/3458754.
- Hacking, I. (1983). *Representing and intervening: Introductory topics in the philosophy of natural science*. Cambridge university press.

- Hamari, J., Koivisto, J., & Sarsa, H. (2014). Does gamification work?—A literature review of empirical studies on gamification. In *2014 47th Hawaii international conference on system sciences* 3025–3034.
- Hatton, L., & Roberts, A. (1994). How accurate is scientific software? *IEEE Transactions. on Software Engineering*, 20 (10), 785–797. <https://doi.org/10.1109/32.328993>
- Heaton, D., & Carver, J. C. (2015). Claims about the use of software engineering practices in science: A systematic literature review. *Information and Software Technology*, 67 (4), 207–219. <https://www.sciencedirect.com/science/article/pii/S0950584915001342>
- Houstis, E. J., & Rice, J. (2000). *On the Future of Problem Solving Environments* (00-009). Purdue University Department of Computer Science Technical Reports. Accessed 20 06 2022. <https://docs.lib.purdue.edu/cstech/1487>
- Hsieh, H.-F., & Shannon, S. E. (2005). Three approaches to qualitative content analysis. *Qualitative Health Research*, 15(9), 1277–1288. <https://doi.org/10.1177/1049732305276687>
- Huizinga, J. (1949). *Homo ludens: Astudy of the play element in culture*. In *Homo ludens: Astudy of the play element in culture*. London: Routledge & Kegan Paul.
- Hutson, H., Johnson, D., Chua, X.-Y., Hogan, J., Breerton, M., Rittenbruch, M., . . . O'Donoghue, S. (2016). Can videogame players inform better scientific visualization'. In *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play Companion Extended Abstracts* 181–187.
- Johanson, A., & Hasselbring, W. (2018). Software engineering for computational science: Past, present, future. *Computing in Science & Engineering*, 20(2), 90–109. <https://doi.org/10.1109/MCSE.2018.021651343>
- Juul, J. (2005). *Half-real: video games between real rules and fictional worlds*. The MIT Press.
- Juul, J. (2013). *The art of failure: An essay on the pain of playing video games*. MIT press.
- Keefe, D. F. (2010). Integrating visualization and interaction research to improve scientific workflows. *IEEE Computer Graphics and Applications*, 30(2), 8–13. <https://doi.org/10.1109/MCG.2010.30>
- Keller, R. M., & Rimon, M. (1992). A knowledge-based software development environment for scientific model-building. In *Proceedings of the seventh knowledge-based software engineering conference* 192–193.
- Kim, T. W. (2015). Gamification ethics: Exploitation and manipulation. In *Proceedings of the ACM SIGCHI gamifying research workshop*. <https://doi.org/10.13140/RG.2.1.2672.4321>
- Knorr-Cetina, K. (1981). *The manufacture of knowledge: An essay on the constructivist and contextual nature of science*. Elsevier Science Limited.
- Knorr-Cetina, K. (2009). *Epistemic cultures: How the sciences make knowledge*. Harvard University Press.
- Kuhn, T. S. (1977). *The essential tension: Selected studies in scientific tradition and change*. University of Chicago Press.
- Kuhn, T. S. (2012). *The structure of scientific revolutions*. University of Chicago press.
- Larkin, Narasimhan K., Raffuse, Sean, Pryden, Daniel, Healy, Alan, Unger, Kevin, Strand, Tara, Solomon, Robert, . . . U.S. Joint Fire Science Program (2009). Conversion of the BlueSky Framework into collaborative web service architecture and creation of a smoke modeling application Accessed 20 06 2022. Retrieved from <https://digitalcommons.unl.edu/jfspresearch/143/>
- Latour, B., & Woolgar, S. (1986). *Laboratory life: the construction of scientific facts*: Vol. 80. Princeton, N.J.: Princeton University Press.
- Latour, B. (1987). *Science in action: How to follow scientists and engineers through society*. Harvard university press.
- Leavy, P. (2017). *Research design: Quantitative, qualitative, mixed methods, arts-based, and community-based participatory research approaches*. Guilford Publications.
- List, M., Ebert, P., Albrecht, F., & Markel, S. (2017). Ten simple rules for developing usable software in computational biology. *PLoS Computational Biology*, 13(1), 1–5. <https://doi.org/10.1371/journal.pcbi.1005265>
- Lundstrom, M., & Klimeck, G. (2006). The NCN: Science, simulation, and cyber services. In *2006 IEEE Conference on Emerging Technologies - Nanoelectronics* 496–500.
- MacLeod, R. S., Johnson, C. R., & Matheson, M. A. (1992). Visualization of cardiac bioelectricity—a case study. In *Proceedings visualization '92* 411–418.
- Mårtensson, P., Fors, U., Wallin, S.-B., Zander, U., & Nilsson, G. H. (2016). Evaluating research: A multidisciplinary approach to assessing research practice and quality. *Research Policy*, 45(3), 593–603. <https://doi.org/10.1016/j.respol.2015.11.009>
- McCain, G., & Segal, E. M. (1988). *The game of science*. Thomson Brooks/Cole.
- Milewicz, R., Pinto, G., & Rodeghero, P. (2019). Characterizing the roles of contributors in open-source scientific software projects. In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)* 421–432. <https://doi.org/10.1109/MSR.2019.00069>
- MobateX. (2008) *MobaXterm* [software]. MobaXterm. <https://mobaxterm.mobatek.net/>
- Nacke, L. E., & Deterding, C. S. (2017). The maturing of gamification research. *Computers in Human Behaviour*, 71, 450–454. <https://doi.org/10.1016/j.chb.2016.11.062>
- Nersessian, N. (2008). *Creating scientific concepts*. MIT Press.
- Oinn, T., Greenwood, M., Addis, M., Alpdemir, M. N., Ferris, J., Glover, K., Wroe, C., Goderis, A., Hull, D., Marvin, D., Li, P., Lord, P., Pocock, M. R., Senger, M., Stevens, R., Wipat, A., & Wroe, C. (2006). Taverna: Lessons in creating a workflow environment for the life sciences. *Concurrency and Computation: Practice and Experience*, 18 (10), 1067–1100. <https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.993>

- OriginLab Corporation. (2017). *Origin(Pro) 2017* [software]. OriginLab Corporation. <https://www.originlab.com/>
- Patton, M. Q. (2002). *Qualitative research & evaluation methods*. SAGE.
- Poincaré, H. (1921). *The Foundations of Science: Science and Hypothesis, The Value of Science, Science and Method*. Science Press.
- Popper, K. (2005). *The logic of scientific discovery*. Routledge.
- Popper, K. (2013). *All life is problem solving*. Routledge.
- Queiroz, F., Spitz, R., Elias, P., Pinheiro, R., Azevedo, T., Rodrigues, V., Reis, L., Raposo, A. (2017). Video games as inspiration for scientific software. In Proceedings of SBGames 2016 (pp. 387–396). Sao Paulo, Brazil. <http://www.sbgames.org/sbgames2016/downloads/anais/157338.pdf>
- Rios, F. (2017). Preserving and Sharing Software for Transparent and Reproducible Research: A Review Open Science Framework doi:10.17605/OSF.IO/D4KEF
- Segal, J., & Morris, C. (2008). Developing scientific software. *IEEE Software*, 25(4), 18–20. <https://doi.org/10.1109/MS.2008.85>
- Sletholt, M. T., Hannay, J. E., Pfahl, D., & Langtangen, H. P. (2012). What do we know about scientific software development's agile practices? *Computing in Science Engineering*, 14(2), 24–37. <https://doi.org/10.1109/MCSE.2011.113>
- Sochat, V. V., Prybol, C. J., Kurtzer, G. M., & Antoniewski, C. (2017). Enhancing reproducibility in scientific computing: metrics and registry for singularity containers. *PloS one*, 12(11), e0188511. <https://doi.org/10.1371/journal.pone.0188511>
- Tran Luciani, D., Löwgren, J., & Lundberg, J. (2020). Designing fine-grained interactions for automation in air traffic control. *Cognition, Technology & Work*, 22(4), 685–701. doi:10.1007/s10111-019-00598-9.
- Wang, G., Tran, T. N., & Andrade, H. A. (2010). A graphical programming and design environment for FPGA-based hardware. In *2010 International Conference On Field-Programmable Technology* 337–340. <https://doi.org/10.1109/FPT.2010.5681433>
- Wang, J., Bennett, K. J., & Guinness, E. A. (2012). Virtual astronaut for scientific visualization—A prototype for santa maria crater on Mars. *Future Internet*, 4(4), 1049–1068. <https://doi.org/10.3390/fi4041049>
- Wang Q, Wang M, Wei K and Qiu C. (2017). Visualization of Gas–Oil–Water Flow in Horizontal Pipeline Using Dual-Modality Electrical Tomographic Systems. *IEEE Sensors J.*, 17(24), 8146–8156. [10.1109/JSEN.2017.2714686](https://doi.org/10.1109/JSEN.2017.2714686)
- Wilson G *et al* . (2014). Best Practices for Scientific Computing. *PLoS Biol*, 12(1), e1001745 [10.1371/journal.pbio.1001745](https://doi.org/10.1371/journal.pbio.1001745)
- Wolff, C. (2015). The case for teaching “tool science” taking software engineering and software engineering education beyond the confinements of traditional software development contexts. In *2015 IEEE global engineering education conference (educon)* 932–938.
- Woollard, D., Medvidovic, N., Gil, Y., & Mattmann, C. A. (2008). Scientific software as workflows: from discovery to distribution. *IEEE Software*, 25(4), 37–43. <https://doi.org/10.1109/MS.2008.92>