Computational Science: A Field of Inquiry for Design Science Research

Veda C. Storey
J. Mack Robinson College of Business
Dept. of Computer Information Systems
Georgia State University
Atlanta, Georgia 30338
VStorey@gsu.edu

Richard L. Baskerville
J. Mack Robinson College of Business
Dept. of Computer Information Systems
Georgia State University
Atlanta, Georgia 30338
Baskerville@acm.org

Abstract

The digitalization of science has resulted in the development of essential, specialized, devices and software. Computational science, as a branch of science, is specifically identified as an important, potential area for which it would be helpful to apply design science research. This paper examines computational science, identifies its past and ongoing challenges, and suggests that progressing computational science with design science research can serve as an important area of inquiry for continuing design science research.

Keywords: Computational science, design science research, sciences of the artificial, digital science, digital artifact, Science 2.0

1. Introduction

In 1969 (now over fifty years ago), Nobel Laureate Herbert Simon set a broad distinction between two kinds of academic disciplines: the *sciences of the natural* that study and describe our natural environment, and the *sciences of the artificial* that prescribe and create artifacts that change our environment [1]. These *sciences of the artificial* spanned professional schools, such as law, business, and information systems, that primarily design and create artifact that are useful to society.

But even fifty years ago, many scientists in nearly every field of natural science were creating artifacts to assist them in their work. In Simon's time, these devices were typically measuring devices for sensing and observing natural phenomena. Today, however, much work in science occurs *in silico*: within digital simulation systems where sensors, calculations, displays, etc., reflect observations of a reality that is not natural, but rather simulations that are digitally created [2]. The natural sciences have become increasingly dependent on complex, special purpose, digital assemblies of computational devices, software, and data.

Over the course of these five decades, the natural sciences and its disciplines have spawned computational science and its various branches. Computational biology emerged from biology, computational geoscience emerged from geoscience, etc. Computational science is the use of computers, software, and algorithms to solve complex problems and needs [3]. In computational science, there are many rapid advances via simulations (e.g., astronomy), mining of massive data sets (e.g., bioinformatics and medicine), and other technology-based discovery techniques. These natural science advances require the development of essential, specialized, devices and software.

As computational science become ubiquitous within each branch of science, a natural scientist increasingly must design and program their experiment in the digital world. In a survey of 2000 scientists, 38% spent at least a fifth of their time developing software, 45% reported this workload was increasing, 47% lacked an understanding of software testing; yet only 34% felt software development training was necessary [4]. In other words, natural scientists are poorly prepared, yet deeply engaged in designing, creating, and depending on digital artifacts.

Furthermore, engineering branches have emerged in concert with the computational sciences. Biological engineering emerged with computational biology, ecological engineering emerged with computational geoscience, etc. In this sense, contemporary developments have gradually advanced the natural sciences to become what Simon considered to be sciences of the artificial.

It seems obvious from the names given to these advances (*computational* geoscience, ecological *engineering*, etc.) that the fields of computer science and engineering are important. But what of information systems? As a branch of systems science concerned with "information and the complementary networks of hardware and software that people and organizations use to collect, filter, process, create and

URI: https://hdl.handle.net/10125/80043
978-0-9981331-5-7

also distribute data" [5], there should be a contribution to the modern version of the natural sciences.

The field of information systems certainly offers the sciences knowledge about organizational infrastructure systems. These systems enable data and information functions in scientific organizations as well as any other kind of organization. But further, the information systems field has proven to be a pioneer in developing design science research methods. The concept of a *scientific* design of computational artifacts is a direct contribution to the fundamental principles of any computational science.

Design science research traces its genesis to a chapter in Simon's *Sciences of the Artificial* entitled The "Science of Design" [6]. In this chapter, Simon complains, "In terms of the prevailing norms, academic respectability calls for subject matter that is intellectually tough, analytic, formalizable, and teachable. In the past much, if not most, of what we knew about design and about the artificial sciences was intellectually soft, intuitive, informal, and cook-booky" (p. 112).

While designing inevitably engages the creativity of the designer [7], Simon advocated professional design practices that are not only soft and intuitive art, but also seek to incorporate as much teachable, formal, analytical science as possible. The resulting designs are not only reusable, but testable and verifiable. In the field of information systems today, the paradigm of design science research has provided methods for theorizing, designing, and developing digital artifacts.

In this paper, we briefly survey examples in the research literature of how important digital artifacts in the natural sciences have reflected the more formative kinds of design practices, soft and intuitive -- like those in information systems that predate Simon's vision. Our objective is to show the feasibility of computational science as a potential area of inquiry for design science research and especially show how design science research can contribute to computational science activities that involve the role of science in the design, creation, evaluation, and reuse of digital artifacts. This contribution reveals broad new arenas, and very important arenas, *viz.* the natural sciences, for information systems design science researchers. In this way, design science research that emerged from information systems can contribute the theory-based design practices and well-validated digital artifacts that science demands.

2. Artifact Design in Computational Science

Like the rest of society, the natural sciences have gone digital. Terms like computational science, digital science [8], science 2.0 [9], etc., are indicators that the sciences are leading, or at least keeping up with, the trends. But going digital is also creating new risks and new challenges across society [10]. Computational science is no different. For example, scientists are using data analytics and machine learning with massive amounts of data [11-13] gathered through interdisciplinary research that combines open scientific outputs, citizen science, and data-intensive science [9, 14]. There are risks in the dependence on nontransparent AI and challenges to social well-being through occupational and cultural shifts [10]

While evidence that justifies such social and cultural concerns is still formative, the evidence of poorly designed and developed digital artifacts in science is more substantial.

2.1 Design Risks in Computational Science

Research in computational science is important because of its potential high impact on people in multiple societies. Nevertheless, there is a history of failures in computational design and development [3, 4, 15].

Scientific programming often involves researchers developing their own software artifacts with little or no training in programming. Yet, this software is critical for correctly carrying out research that involves modeling biological structures, or simulating data on evolution, and so forth [4]. These software artifacts are also needed by other researchers who attempt to build upon prior work; hence the need for making software open source [16].

The development of software artifacts can result in reuse problems when the programs do not scale, or work properly in different applications, or provide errors or inaccurate results that would not have been obtained had the software been designed properly. For example, a UC Davis biologist designed code for comparing genomes of closely related organisms. Unknown to him, other biologists used the code to compare genomes of distantly related organisms beyond the program's working range. The result was a publication of totally wrong results [4].

The digital artifacts developed in computational science are often complex. Failure to deal properly with this complexity can lead to the creation of artifacts that are poorly designed, constructed, or tested, thus limiting the falsifiability, repeatability, and reproducibility of the results [17]. For example, a specific small error occurred when a program mistakenly changed a minus sign to a plus sign. The development process was designed without proper testing and evaluation. The result was the retraction of five published research papers from the Scripps Research Institute [4].

Although incorrect conclusions are obviously problematic, some situations have further consequences. For example, the "Climategate" scandal resulted from the exposure of incorrect data [3]. In another situation, researchers Robiou-du-Pont et al. [18] tested a popular web-based bioinformatics tool, SNAP (single nucleotide polymorphisms (SNPs)), and found evidence of 17.6% and 36.6% false negatives. In the Duke cancer research scandal, the data analysis was conducted using graphical and spreadsheet tools, which led to questionable or incorrect results [19, 20]. That investigation led to a full retraction of 10 published research articles and either corrections or partial retraction of an additional seven [21]. In high-risk applications, mistakes can have a very large impact. Scientists need to be knowledgeable in software design and development and in the application domain. It is important that they realize when they are creating a solution to a narrow problem [22].

2.2 Design Challenges in Computational Science

Many observers may be quick to conclude that the risks evident in computational science are often due to poor programming. This conclusion is not unreasonable from a programming perspective. A software engineering perspective would lead us to a parallel conclusion that bad software engineering is the problem. Undoubtedly, it would help to get more people in the sciences to follow received software engineering practices. But there is the rub. As noted in the introduction, most scientists view formal software training as unnecessary. Part of the problem is people: a part that researchers in sociotechnical systems, like those in information systems, understand well.

The defects in computational science *systems* exist on multiple levels if they additionally have poor project management, poor software engineering, and poor systems design. These are the types of issues that design science research addresses by foregrounding *design of artifacts*: constructs, models, methods, as well as software. Not only do such methods improve the science in natural science, but they also elevate reusability and reproducibility to the level of design theories and principles. Research in computational science is also an important, societal area of research.

Table 1 shows six examples of current challenges in computational science that provide important opportunities for new, groundbreaking design science research. We selected these examples because: (1) they provide indications of the breadth of research opportunities; and (2) the underlying challenges have been published in the scientific literature. While the list is likely incomplete, it shows deep opportunities for expanding our knowledge of design science research by tackling problem arenas that may be more complex than business and management.

Table 1: Examples of Challenges in Computational Science

Computational Science Challenge	Published Examples of Challenges
Semantics/culture gap	Semantics of technical language different in computational science. Fields develop unique information systems cultures. [3]
Lacking broad systems design view	Computational scientists view the problem as a matter of programming errors; there is a failure to recognize that digitalization of experimental science has created a digital ecosystem that spans all branches of science (and beyond). Example: [23]
Dynamic goals of computational artifacts	Science is exploratory; therefore, its software development is exploratory. Requirements discovered iteratively. Formal software processes overly constrain research. Verification and validation are emergent. Example: [3]
Technology-push diffusion	Cutting-edge technologies adopted rapidly without deep understanding (e.g., machine learning, data analytics). Example: [24]
Technical debt	Future working obligations that are the consequence of technical choices made for short-term benefit. E.g., writing bad code under time/resource pressure. Example: [25]
Partial understanding of system life cycles	Scientists often self-taught programmers; broader training in systems development lacking. Example: [15]

Some scientists may be simply unprepared to program. While many of the sciences include formal training in common educational formats, there may be only informal training in software development. In the sciences, software developers are more reliant on learning from individuals and activities, such as mentors and peer learning, which exist primarily outside of a normal education environment. Coding skills are typically learned informally [15]. Under the assumption that the essential problem is poor programming habits, one obvious solution to the challenge of creating digital artifacts is to train researchers in software engineering. While this solution is an important start, the solution scope will need to be broader for the future; otherwise, such challenges with computer systems (Table 1) will continue to inhibit science. The "gap" or "chasm" between software engineering and scientific programming remains a serious risk to the production of reliable scientific results [17].

For scientific software artifacts, Johanson and Hasselbring [3] argue that sound engineering practices are needed, but that software engineering should be separated from computational science because of the nature of scientific challenges, limitations of computers, and cultural environment of scientific software development. They identify three specific reasons:

[1] *Requirements are not known up front*.
[2] *Overly formal software processes restrict research*.
[3] *Verification and validation are difficult and strictly scientific*.

These are fundamentally deep problems in the systems of computational science. From an information systems perspective, poor programming skills and habits may not be the main problem. They may, instead, be symptoms of the problem, which is why software engineering can only be a partial solution.

Digitalization means that systems are being constructed using multiple devices and platforms. Integrating the devices and platforms so they can be used together is more than just software engineering. It is actually an architecture. Moreover, the problem is sociological, organizational, and cultural. Computational science needs to value high quality digital artifacts. Scientific research organizations need scientists who are qualified software engineers; they need software project management; they need systems designers and architects; they need systems designed to fit the scientific enterprise. In short, they need information systems expertise.

3. Perspective on Research

Design science research focuses on theory and methods for developing useful artifacts to address real-world problems [26, 27]. Design science, thus, strives to produce well-validated digital artifacts, consistent with the needs of science. It also recognized the need to maintain the novelty and creativity required to address complex, real-world problems and to be able to represent the generative process of artifact creation in addition to the artifact itself [7]. Information systems broadly, and design science, specifically, deliver the kind of theory-based designs, with well-validated digital artifacts, required by science. What is more, the challenges that computation science places before design science research offer a novel range of research venues and questions for design scientists [28].

Table 2 summarizes possible dual contributions of design science research with computational science.

Table 2: Complementary Research: Design Science Research and Computational science

Computational science Perspective (DSR -> D-Science)	Design Science Research Perspective (D-Science -> DSR)
Well-developed design theory about embedded phenomena, nascent design theory (knowledge as operational principles/architecture), situated implementation of artifact to Computational science	Computational science provides area of research for fundamental improvements to design science research or information systems per se
Design theories and principles, constructs, methods, models, technological rules, instantiations of software products or implemented processes	New design science methods, concepts, constructs, applications, subjects, scope, or scale

In conjunction with Table 1, Table 3 shows how computational science presents new research opportunities for design science research. These opportunities will improve our knowledge of design science research and broaden the impact of information systems research. Table 3 lists six common design science research activities, together with examples of how such activities can find new research questions (RQ) posed by computational challenges detailed in Table 1. Each of these activities and the features of the challenges in computational science is discussed in the following subsections.

Table 3. Design science research applied to computational science

Design Science Research Activities	Research opportunities in computational science
1. Problem formulation	Challenge: Semantics/culture gap RQ: How can a design science approach reduce gaps between scientific fields? • More dynamic kind of wicked problem • Problem formulation unstable • New social theories need to be applied • Dynamic methods of problem formulation
2. Meta problem identification	Challenge: Lacking broad systems design view RQ: What generalized goals inhabit computational science artifacts? • Assess availability and value of scientific information platforms and tools • Organize scientific procedures to operate with digital artifacts
3. Theorize the design	Challenge: Dynamic goals of computational artifacts RQ: How can design principles and theories define classes of computational science artifacts? • Understand how scientific theories interacts with a design or kernel theory • Explore how relationship between design theory, kernel theory, and context's scientific theory can enhance science with better digital artifacts
4. Design the artifact	Challenge: Technology-push diffusion RQ: How will design principles and theories affect innovation diffusion in computational science? • Accommodate uniqueness of the design challenges of digital artifacts • Create new design guidelines to account for uniqueness scientific data or artifact
5. Make the artifact	Challenge: Technical debt RQ: How will iteratively matching artifact characteristics and environments affect technical debt? • Build correct and efficient artifact that forms foundation for research problem • Recognize importance of artifact in scientific endeavor • Identify new ways of making an artifact or new guidelines • Create or identify development techniques for dealing with the uniqueness of digital artifacts in the scientific community
6. Evaluate the artifact	Challenge: Partial understanding of life cycles RQ: How will user (scientist) participation in full, iterative artifact life cycles improve evaluation? • Create new assessment techniques to accommodate changing access to large databases, ontologies, or other stocks of knowledge. • Adapt and extend existing evaluation guidelines to natural sciences. May require new methods for scientific code.

3.1 Problem Formulation

A major challenge for system designers in computational science is the vast difference in terminology and technical semantics present from the beginning of their technology design projects. Variations in technical semantics means that different fields refer to the same underlying technology with different technical terms. Essentially these language conflicts reflect a technological culture gap developed by dominance of different technological architectures in different scientific fields.

The challenge is one of translating ideas that form local truths in different scientific fields. There are not really any agreed upon global semantics (a global truth) between fields. Designers faced with defining problems in one science field, must currently relearn the language for problems and technologies when defining problems in a different field [29].

This activity is usually an opportunity for a design scientist to find a problem that is common across similar settings. Ideally, the challenge is not only a part of a unique, one-time problem, but is also one that is generalizable as a class to other scientific applications. In the sciences, identifying a class of design problems is complicated by differing research cultures, terminology, and even the exact semantics of common terminology. Discovering the problem may be iterative itself in order to deal with these differences.

Most design science approaches commence by clarifying the fundamental problem. In computational science, the general class of problems are wicked: questions in natural science involve unknowns that are often difficult to formulate, confusing, value-laden, and stated in terms of different solutions (alternative hypotheses) [30, 31]. Design science researchers may, thus, discover new kinds of "wicked" real-world problems that require an artifact [32].

Scientific applications have a more dynamic kind of problem in which the artifact and its environment co-define each other along a continuum of technical and scientific change. The design problem formulation will rarely be sufficiently stable to permit designs to be deduced from their context. In computational science, problems can among themselves be made dynamic by the rapid progress of science in general.

Design science research can encompass new social theories and dynamic methods of problem formulation. Science settings are of a kind where the design context (research problem) is highly interactive with the design problem. Therefore, design science research can emphasize studies on the reflexive quality of digital artifacts. That is, how these artifacts affect the formulation of their context. For example, rather than regarding problem formulation as a separate stage, the iterative methods of design science allow a continuous problem formulation process that operates in parallel throughout the lifespan of a design science research project. This characteristic has parallels with agile systems development. By using design science research, building science systems can become more closely aligned with mature work in building information systems.

For example, Johanson and Hasselbring [3] elaborate how the semantics of technical language are different in computational science, both from the language of computing fields and from the language of other scientific fields. They show how different fields develop unique information systems cultures.

These issues with problem formulation raise new kinds of research problems for design science research. These issues regard formulating problems when context is one of wicked problems described in differing terminology, with differing technological cultures. An example of a research question for design science research could be: "How can a design science approach reduce gaps between scientific fields?" Another example could be: "How can broader views of system life cycle models help better define artifact design problems in the computational sciences?"

3.2 Meta Problem Formulation

Many of the software problems endemic to computation sciences arise from the one-time-use assumptions of the developers. They assume their problem is so unique that related types of systems solutions would not exist. Users in science may be deeply familiar with digital systems. Nevertheless, the notion that their problem solutions may represent a class of scientific artifacts applicable in many fields of science may be overlooked. They may even be able to adapt existing approaches to solve their problem.

Design science researchers should consider the availability and value of scientific information platforms and tools to adapt in their own work. They can investigate better ways to organize scientific procedures to effectively operate with digital artifacts. Science settings also present design science researchers with a different kind of user. More work is needed to determine if general categories or classes of scientific problems exist and the degree to which scientific problems can be generalized to a class of problems, as well as whether researchers can discover, or develop, generalized tools or platforms.

Even though scientific projects and experiments have become increasingly complex and digitalized, many are still not formally organized. Consequently, many settings in science appear to operate as clan organizations in which information systems folks, if present at all, operate as a separate clan. Although the digital artifacts may actually define much of the research framing, these artifacts might be delegated to an independent clan or constructed by clan-based amateurs. Either way, there are too few in science project organizations that are knowledgeable about the broader availability, technical quality, and value of scientific information platforms and tools. The grounding of design science research in information systems connects known solutions to such problems. For example, information systems research discovered processes by which systems can help informate clans and enable more constructive clan-based control [33].

By operating in the sciences, we broaden the scope of design science research in information systems. Design science research should address better ways to organize scientific procedures in order to more effectively operate with digital artifacts. Researchers will confront the need to further classify their problems. Science needs theories, principles, and methods to organize their projects, classify science and design science research problems, better characterize the conditions under which different design science research methods operate, and develop problem frameworks that are broader in scope than existing ones defined in the information systems field. For example, a computational science issue is that digital science views the problem as a dependency on complex computer code or as a failure to recognize that experimental science has grown into a complete digital ecosystem [23].

Design science research can provide further research into this issue because it could be addressed in the phase that deals with meta problems. This work includes access availability and the value of scientific information platforms and tools. This work could address the need to manage computational science artifacts within the ecosystem of computational science platforms and tools. An example of a design science research question could be: "What generalized goals inhabit computational science artifacts?" Another example is: "How do different technological terminologies affect instances of similar problems in different branches of computational science?"

3.3 Theorize the Design

For digital artifacts in science, the formulation of the design theory or design principles is deeply entwined in the formulation of the digital artifact and the class of research problems or questions being considered. The requisite class must be theoretically feasible. Such an entwinement is inevitable because the range of research problems/questions must be "researchable," just as with specific research problems or questions. This entwinement means the requisite class of digital artifacts must be theoretically feasible.

Theorizing the design involves establishing a relationship between the class of problems (i.e., the general problem) and a class of solutions (i.e., the general solution). This design activity takes place at an abstract level where the researcher develops a theoretically general solution for a theoretically general problem. In this way, design science research helps respond to challenges of computational science by generating a solution to a range of problems.

Science settings present design science researchers with increased importance of design theories and design principles because the problem formulations are unstable and dynamic. Empirical work is needed to demonstrate that theoretically sound design science, within the context of the sciences, is both possible and desirable. Researchers need to understand how scientific theories interact with a design or kernel theory. It may be possible that the relationship amongst the design theory, the design science research kernel theory, and the context's scientific theory can provide the key to bettering overall science with better digital artifacts. We need to learn whether this relationship means that these kinds of theories collide within design science research, whether they form a junction, or whether they interact in the form of a nexus [34].

The scientific applications can lead to the use of new kernel theories specific to science. These may be required because of the need to process different kinds of data. For example, text mining theories and applications may be needed. The scientific context's theories are likely to interact with design theories in a manner similar to the interaction of the science problem and the design problem. We have little understanding of how a theory driving a design context interacts with a design theory or a kernel theory in design science research. There are multiple theories interacting to drive a class of design problems and solutions.

Needed is an exploration of how relationships between design theory, kernel theory, and the context's scientific theory can enhance science with better digital artifacts. Since computational science involves poorly defined and wicked problems, this is a good area for research into design theorizing for computational science projects. Examples of design science research questions that arise from this need include: "How can design principles and theories define classes of computational science artifacts?" Also: "How does design theorizing in the computational sciences differ from design theorizing in business and organizational fields?"

3.4 Design the Artifact

Artifacts are designed to address ill-structured, novel problems. Generalizable designs for scientific digital artifacts need to be created. Digital science requires essential, specialized devices and software to support scientific research. The artifacts may need to be designed with the capability to accommodate the large volumes of various types of data that are being generated due to digitalization.

In this activity, design science researchers translate the general, theorized design from the previous activity into the specific, unique instance at hand. Design science researchers need to understand, describe, and generalize the uniqueness of the design challenges of computational science artifacts. They need to create new design principles and guidelines to account for the unique characteristics of the scientific data or equipment for specific application domains. Because of the expansion of the application domains to science, it may be possible to identify new ways of designing novel artifacts: for example, the creation of a digital artifact that simulates past behavior of the spread of a contagious disease when accurate data is not available.

Design in the computational sciences can be complicated by the mode of technology diffusion called technology-push [35]. Scientists in every field will be kept aware of the latest available information technologies and often work to push these technologies into their labs. While this technology may indeed be ideal, research-based designs may find that the specific problem does not inherently demand an advanced technology. For example, Sculley et al. [24] explain how computational scientists are motivated to rapidly adopt cutting-edge technologies such as AI, machine learning, and big data analytics. Such adoptions are often made without a deep understanding of how these technologies work, and how they might contribute to the scientific research being undertaken.

Computational science offers design science researchers a novel challenge to accommodate the uniqueness of design problems for digital artifacts. This may require the creation of new design guidelines to account for such uniqueness in the scientific data or artifact. Examples of design science questions that arise from this challenge include: “How will design principles and theories affect innovation diffusion in computational science?” Another is: “How can design science research examine the role of platforms in highly unique problem and design settings?”

3.5 Make the Artifact

Essential artifacts must be built that match the scientific research problem. Researchers require an accurate understanding of the importance of such artifacts. Digital artifacts need to be developed for new applications and design knowledge extracted from these efforts. The creation of digital scientific artifacts should follow the same guidelines as those established in design science research. This is true even for scientific artifacts that might appear to be simplistic and employ easy-to-use technology, such as the spreadsheets in the public debt project economies [17, 36, 37].

In this activity, design scientists implement the design from the previous step. In the computational sciences, digital artifacts have particularly stringent needs to follow and must adhere to established development and implementation guidelines. For design science researchers, such artifact creation can be more critical than in many business and organizational settings. Given the potential for societal impact, a lack of exactness can be disastrous. This is a new challenge for design science researchers whose artifacts in the past have stopped as administrative prototypes. For computational science, these artifacts must often solve their scientific problem and be able to replicate their results exactly.

From the development of new digital artifacts for new application domains in the natural sciences, it should be possible to extract design knowledge. Building a correct and efficient artifact that forms the foundation for the research problem may require a new way of making an artifact or new guidelines for doing so. New efforts may be required to address issues of large databases and large numbers of variables in analysis and simulations [38]. Thus, it is necessary to create or identify development techniques for dealing with the uniqueness of digital artifacts in the scientific community. The uniqueness could be large sets of, perhaps, biological or disease data; interfaces to known corpus of data for testing and assessment; or use of, for example, biomedical ontologies [39]. Design science research offers science its experiences in designing big data analytics processes such as social media analytics and health analytics.

We need to build a correct and efficient artifact that forms the foundation for a research problem, thus recognizing the importance of an artifact in a scientific endeavor. This may require identifying new ways of making an artifact or new guidelines. It may involve creating or identifying development techniques for dealing with the uniqueness of digital artifacts in the scientific community. Examples of design science research questions driven by this requirement include: “How will iteratively matching artifact characteristics and environments affect technical debt?” Another example is: “How can we build prototypes that do not create or deepen technical debt?”

3.6 Evaluate the Artifact

Artifacts need to be created that avoid costly mistakes and/or decrease credibility within the scientific community. This requires the development and diligent application of stringent evaluation techniques and procedures. Any appropriate evaluation method needs to take into consider the types of digital artifacts being developed, with the most recognized guidelines for artifact evaluation found in FEDS [40] and its adaptations. The guidelines include observation and participation techniques, such as case studies, as well as empirical assessments, involving experiments, simulations, and prototypes. Other efforts to understand an appropriate evaluation method for an artifact have also been carried out; for example, Prat et al. [41]’s taxonomy of artifact evaluation methods.

In computational science, standards for artifact performance are stringent. Like making the artifact, evaluation must meet higher standards than many business and management applications. Indeed, computational science aspires to a degree of proof-of-performance that sustains very exact repeatability. This degree of performance confronts design science research with challenges to improve evaluation processes to meet much higher expectations for exactness in performance and replicability.

Existing evaluation guidelines may need to be adapted or extended for the natural sciences, which may require new methods for scientific code. Needed, for example, may be new methods to identify when scientific code has not been tested thoroughly or at the extremes, and case examples for both routine and unusual situations. New assessments techniques may need to be developed to deal with other challenges in computational science such as: changing access to large databases that might be used or shared by scientists; large sample sizes; large number of variables; simulations (e.g., those that re-create historic weather and environmental patterns); and other characteristics as they appear in contemporary scientific endeavors. Examples of design science research questions that proceed from these research opportunities include: “How will user (scientist) participation in full, iterative artifact life cycles improve evaluation?” Also, “How do we evaluate the goals of dynamic computational science artifacts?”

4. Discussion and Conclusion

Computational science has emerged from the recognition of the need for digital artifacts (research software) to support research in science, which, as described by Hasselbring et al. “can be an object of study itself [because the software is used to advance our] understanding of natural systems, answering questions that neither theory nor experiment alone is equipped to answer” [16] (p.84). However, there are many instances where the development of digital artifacts for computational science have been considered as a one-time development activity and the digital artifacts have suffered from the inability to be reused. In this sense, both the digital artifact and the design knowledge inherent in the artifact become what we call *digital artifact exhaust.*

More importantly, though, the field of computational science is a meaningful area of inquiry for design science research, thus expanding the fields to which design science research can make a contribution. Much research in design science has focused on small relatively small problems, such as those one might find in business or non-profit organizations. Some of these efforts have resulted in well-defined artifacts, but are presented only as prototypes or proof of concept attempts. In contrast, many outputs from computational science can have significant impacts on society. The problems addressed in computational science require more than proof of concept efforts, which would make design science research ever more meaningful.

Researchers in information systems can contribute to addressing grand challenges as related to science, and in doing so, significantly contribute to society. Future work is needed to further formalize and address the proposed research questions outlined in this paper and to apply and expand design science research to many of the varied kinds of inquiry found in computational science.

5. References

[1] H. A. Simon, "The Sciences of the Artificial," *MIT Press. Cambridge,* 1969.

[2] P. Ruiz, A. Sack, M. Wampole, S. Bobst, and M. Vracko, "Integration of In Silico Methods and Computational Systems Biology to Explore Endocrine-Disrupting Chemical Binding with Nuclear Hormone Receptors," *Chemosphere,* vol. 178, pp. 99-109, 2017.

[3] A. Johanson and W. Hasselbring, "Software Engineering for Computational Science: Past, Present, Future," *Computing in Science & Engineering,* vol. 20, no. 2, pp. 90-109, 2018.

[4] Z. Merali, "Computational Science: ... Error ... Why Scienific Programming Does Not Compute," *Nature,* vol. 467, no. 7317, pp. 775-777, 2010.

[5] "Information System," *Wikipedia: The Free Encyclopedia*, 26 July 2021. Accessed on: 14 August 2021Available: https://en.wikipedia.org/wiki/Information_system

[6] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design Science In Information Systems Research," *MIS Quarterly,* vol. 28, no. 1, pp. 75-105, Mar 2004 2004.

[7] R. L. Baskerville, M. Kaul, and V. C. Storey, "Aesthetics in Design Science Research," *European Journal of Information Systems,* vol. 27, no. 2, pp. 140-153, 2018.

[8] D. Dougherty and D. D. Dunne, "Digital Science and Knowledge Boundaries in Complex Innovation," *Organization Science,* vol. 23, no. 5, pp. 1467-1484, 2012.

[9] K. Szkuta and D. Osimo, "Rebooting Science? Implications of Science 2.0 Main Trends for Scientific Method and Research Institutions," *Foresight,* vol. 18, no. 3, pp. 204-223, 2016/06/13 2016.

[10] R. Hodson, "Digital Revolution," *Nature Outlook,* vol. 563, no. 7733, 29 Nov 2018.

[11] H. Chen, R. H. Chiang, and V. C. Storey, "Business Intelligence and Analytics: From Big Data to Big Impact," *MIS Quarterly,* pp. 1165-1188, 2012.

[12] C. Dobre and F. Xhafa, "Intelligent Services for Big Data Science," *Future Generation Computer Systems,* vol. 37, pp. 267-281, 2014.

[13] G. George, M. R. Haas, and A. Pentland, "Big Data and Management," *Academy of Management Journal,* vol. 57, no. 2, pp. 321-326, 2014.

[14] R. Lukyanenko, A. Wiggins, and H. K. Rosser, "Citizen Science: An Information Quality Research Frontier," *Information Systems Frontiers,* pp. 1-23, 2019.

[15] L. Hwang, A. Fish, L. Soito, M. Smith, and L. H. Kellogg, "Software and the Scientist: Coding and Citation Practices in Geodynamics," (in English), *Earth and Space Science,* Article vol. 4, no. 11, pp. 670-680, Nov 2017.

[16] W. Hasselbring, L. Carr, S. Hettrick, H. Packer, and T. Tiropanis, "Open Source Research Software," *Computer,* vol. 53, no. 8, pp. 84-88, 2020.

[17] T. Storer, "Bridging the Chasm: A Survey of Software Engineering Practice in Scientific Programming," (in English), *ACM Computing Surveys,* Article vol. 50, no. 4, p. 32, Nov 2017, Art. no. 47.

[18] S. Robiou-du-Pont, A. H. Li, S. Christie, Z. N. Sohani, and D. Meyre, "Should We Have Blind Faith in Bioinformatics Software? Illustrations from the SNAP Web-Based Tool," (in English), *Plos One,* Article vol. 10, no. 3, p. 8, Mar 2015, Art. no. e0118925.

[19] K. R. Coombes, J. Wang, and K. A. Baggerly, "Microarrays: Retracing Steps," *Nature Medicine,* Correspondence vol. 13, no. 11, p. 1276, 11/01/online 2007.

[20] M. N. Fienen and M. Bakker, "HESS Opinions: Repeatable Research: What Hydrologists can Learn from the Duke Cancer Research Scandal," *Hydrology and Earth System Sciences,* vol. 20, no. 9, pp. 3739-3743, 2016.

[21] R. M. Califf and S. Kornbluth, "Establishing a Framework for Improving the Quality of Clinical and Translational Research," *Journal of Clinical Oncology,* vol. 30, no. 14, pp. 1725-1726, 2012.

[22] D. Kelly, "Scientific Software Development Viewed as Knowledge Acquisition: Towards Understanding the Development of Risk-Averse Scientific Software," *Journal of Systems and Software,* vol. 109, pp. 50-61, 2015/11/01/ 2015.

[23] R. M. Betz, R. C. Walker, and Engineering, "Streamlining Development of a Multimillion-line Computational Chemistry Code," *Computing in Science & Engineering,* vol. 16, no. 3, pp. 10-17, 2014.

[24] D. Sculley *et al.*, "Machine Learning: The High Interest Credit Card of Technical Debt," 2014.

[25] K. Hinsen, "Technical Debt in Computational Science," *Computing in Science & Engineering,* vol. 17, no. 6, pp. 103-107, 2015.

[26] J. Iivari, "A Critical Look at Theories in Design Science Research," *Journal of the Association for Information Systems,* vol. 21, no. 3, p. 10, 2020.

[27] J. Vom Brocke, R. Winter, A. Hevner, and A. Maedche, "Special Issue Editorial–Accumulation and Evolution of Design Knowledge in Design Science Research: A Journey through Time and Space," *Journal of the Association for Information Systems,* vol. 21, no. 3, p. 9, 2020.

[28] V. C. Storey and R. L. Baskerville, "The Digital Science Field of Design Science Research," in *International Conference on Design Science Research in Information Systems and Technology*, 2021, pp. 343-355: Springer.

[29] B. Czarniawska and B. Joerges, "The Travels of Ideas," in *Translating Organizational Change,* vol. 56, B. Czarniawska and G. Sevón, Eds. Berlin: Walter de Gruyter, 1996, pp. 13-47.

[30] C. W. Churchman, "Guest Editorial: Wicked Problems," *Management Science,* vol. 14, no. 4, pp. B141-B142, 1967.

[31] H. W. J. Rittel and M. M. Webber, "Dilemmas in a General Theory of Planning," *Policy Sciences,* vol. 4, no. 2, pp. 155-169, 1973.

[32] R. Baskerville, A. Baiyere, S. Gregor, A. Hevner, and M. Rossi, "Design Science Research Contributions: Finding a Balance Between Artifact and Theory," *Journal of the Association for Information Systems,* vol. 19, no. 5, p. 3, 2018.

[33] R. Kohli and W. J. Kettinger, "Informating the Clan: Controlling Physicians' costs and outcomes," *MIS Quarterly,* vol. 28, no. 3, pp. 363-394, 2004.

[34] J. Pries-Heje and R. Baskerville, "The design theory nexus," *MIS Quarterly,* pp. 731-755, 2008.

[35] R. Baskerville and J. Pries-Heje, "A Multiple-Theory Analysis of a Diffusion of Information Technology Case," *Information Systems Journal,* vol. 11, no. 3, pp. 181-212, 2001.

[36] T. Herndon, M. Ash, and R. Pollin, "Does High Public Debt Consistently Stifle Economic Growth? A Critique of Reinhart and Rogoff," *Cambridge Journal of Economics,* vol. 38, no. 2, pp. 257-279, 2014.

[37] C. M. Reinhart and K. S. Rogoff, "Growth in a Time of Debt," *American Economic Review,* vol. 100, no. 2, pp. 573-78, 2010.

[38] P. B. Goes, "Editor's Comments: Big Data and IS Research," *MIS Quarterly,* vol. 38, no. 3, pp. iii-viii, 2014.

[39] K. W. Fung and O. Bodenreider, "Knowledge Representation and Ontologies," in *Clinical Research Informatics*: Springer, 2019, pp. 313-339.

[40] J. Venable, J. Pries-Heje, and R. Baskerville, "FEDS: A Framework for Evaluation in Design Science Research," *European Journal Information Systems,* vol. 25, no. 1, pp. 77-89, 2016.

[41] N. Prat, I. Comyn-Wattiau, and J. Akoka, "A Taxonomy of Evaluation Methods for Information Systems Artifacts," *Journal of Management Information Systems,* vol. 32, no. 3, pp. 229-267, 2015.