An Application of the Design Science Research Theoretical Framework and Methodology in the Construction of an Approach for Designing Applications in 3D Virtual Worlds

William Sawyerr
Anglia Ruskin University, Cambridge, United Kingdom
Email: william.sawyerr@anglia.ac.uk

Abstract—**This paper discusses the use of the design science research theoretical framework and methodology for constructing an approach for designing applications in 3D virtual worlds. The research is about the need for a suitable method and tools for designing virtual world applications, which is evidenced by many of the problems that virtual worlds continue to suffer due to poor design. The proposed approach uses the generative design grammar framework and the generative design agent model as tools for designing virtual world applications. Preliminary analyses suggest that, the use of a method and tools created specifically for designing virtual world applications provides a suitable and robust approach for designing these types of applications.**

Index Terms—**Methodologies, Design Science Research, Methods, Design, Tools, Toolkits, Frameworks, Models**

I. Introduction

In software engineering (SE), the application development process requires the design of specifications of applications that are intended to accomplish certain goals, using a set of primitive components and subject to constraints [1]. Using this rule, the application development process in virtual worlds (VWs) also requires the design of specifications that comprise goals, components, and constraints. SE already provides methods and tools for designing applications. However, VW applications are characteristically different from the classic types of applications in SE. For example, VW applications are three-dimensional and run in three-dimensional environments, while classic applications are two-dimensional and run in two-dimensional environments. Such differences have implications for design. Those methods and tools in SE that were developed for designing classic types of applications, are unable to effectively capture and describe the characteristics that are important for designing VW applications. Therefore, the VW application design activity requires methods and tools that are specifically for designing VW applications.

Manuscript received February 28, 2016; accepted March 10, 2016; revised April 10, 2016.

In spite of these ideals, the application development process in VWs is currently at an early stage of development, largely utilising existing methods and tools for application design. These methods and tools were mostly developed for managing application design activities for the classic types of applications in SE. Therefore, their use is unsuitable for designing VW applications. The use of unsuitable methods and tools for designing VW applications results in poorly designed specifications. Poor design is one of the causes of usability problems in VW applications [2]. Furthermore, poor design can also adversely affect the adoption and use of VWs [3], [4]. By continuing to use unsuitable methods and tools, VW applications will continue to suffer from problems related to poor design, as well as other problems exacerbated by poor design.

The use of a method created specifically for designing VW applications could serve as a suitable guide for VW application design. In addition, the use of tools created specifically for designing the specifications of VW applications could enable capturing and describing characteristics that are specific to VW applications, in an effective and efficient manner.

This paper is about the use of the design science research (DSR) theoretical framework and methodology for constructing a new approach for designing VW applications. DSR is a set of analytical techniques and perspectives traditionally used for performing research in information systems (IS). It involves the development or building of artefacts to address unsolved problems, and the justification or evaluation of such artefacts with respect to the utility provided in solving those problems. According to Livari [5], artefacts may be constructs, models, methods, or instantiations. Consequently, not only does DSR provide a natural framework for developing the proposed approach, but it has in fact also been used for decades by computer scientists and software engineers to develop artefacts such as new architectures for computers, new programming languages, new compilers, new algorithms, new data and file structures, new data models, and new database management systems.

The proposed approach establishes the use of a method and tools inspired by architecture and built environments, for designing dynamic VW applications. Dynamic VW applications have artificial intelligence (AI) properties embedded in their specification to allow anyone to eventually be able to develop VW applications. VWs are democratic platforms that were created so that any user could develop any type of application for any purpose. Therefore, by designing VW applications that are dynamic, the hope is to be able to contribute towards a future in which VW application development is a democratised process.

The method that is established by the proposed approach determines the scope and limit of the VW design activity. Furthermore, the method enables the systematic use of the tools for designing VW applications. The tools are for designing the specifications of VW applications. They are used for capturing and describing the characteristics of VW applications, and to add dynamic properties to VW applications. Overall, the proposed approach helps to provide the necessary focus and direction for the VW design activity.

An overview of the DSR theoretical framework and methodology is next, followed by a brief review of current methods and tools for designing VW applications. A discussion of the proposed approach is then provided, and the paper concludes with a summary of the initial findings.

II. DSR Theoretical Framework and Methodology

DSR involves two main activities in a cycle: (1) build and (2) evaluate. March and Smith [6] identified four design artefacts or deliverables created by DSR: (1) constructs, (2) models, (3) methods, and (4) instantiations. Constructs are the elementary concepts of a problem or solution space. Models are relationships between relevant constructs. Methods specify how to perform a design task. The product of a design task is a design (i.e., specification of an artefact). Instantiations are the realisations of designs as physical or abstract products.

Drawing on the earlier work of March and Smith, Hevner et al [7] developed an overall framework and guidelines, which is the most recent and accepted for conducting and reporting DSR. They extended the above DSR cycle and renamed the two main activities develop/build and justify/evaluate.

According to Venable [8], DSR should be informed by both business needs (what can also be considered organisational or domain needs) and applicable existing theoretical knowledge. The products of DSR include applications of the new instantiations to organisational or domain environments and additions to the theoretical knowledge. The quality of these two products (i.e., instantiations and theoretical knowledge) corresponds respectively to relevance and rigour.

Nunamaker et al [9] proposed a framework for contextualising the role of systems development, which was mainly focused on instantiation of information systems. The framework included four areas of research activities: (1) theory building, (2) system development, (3) experimentation, and (4) field studies. Much of the early research in IS was focused on systems development approaches and methods. Nunamaker's framework focused only on computer-based systems as the artefact. Therefore, Venable and Travis [10] extended the artefact to include systems development methods. Furthermore, they also proposed a revision to Nunamaker's framework that includes the following four areas of research activities: (1) theory building, (2) solution technology invention, (3) artificial evaluation, and (4) naturalistic evaluation. Figure 1 shows an illustration of the revised framework.

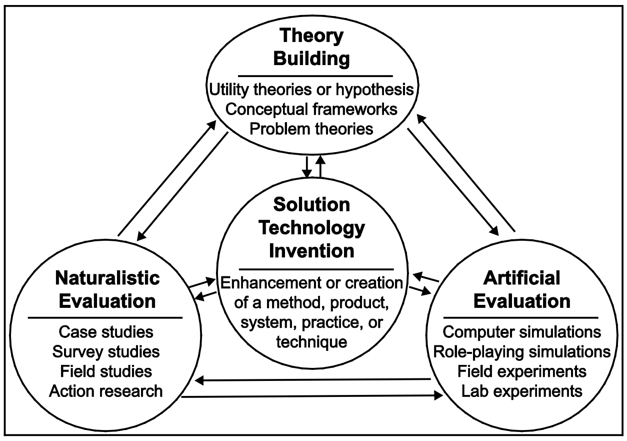

Figure 1. Venable and Travis' framework and context for DSR (A revision of Nunamaker's framework for contextualising the role of systems development).

Solution technology invention is the core of DSR. A solution technology is any approach to making an improvement in an organisation or a domain, including IS, IT, systems development methods, algorithms, managerial practices, etc. Solution technology invention involves high-level and detailed design, building, and possible functional testing of a hypothesised solution technology.

Any or all of the activities in the revised framework shown in Figure 1 may be part of a particular piece or programme of research. The arrows indicate that over time the researcher can alternate between the different activities as research design dictates. The different activities involve multiple research methods and paradigms that Venable suggests should not be performed in isolation. The following provides further details on each of the four activities in the revised framework in Figure 1:

Theory building should occur both as a precursor and as a result of DSR. As a precursor to solution technology invention, the researcher should formulate a utility theory or hypothesis of an approach to reduce the problem. Utility hypothesis formulation compares to the use of abductive reasoning. The following are types of utility theories as proposed by Venable [11]:

- Solution technology X (when applied properly) will help solve problems of type Y.

- Solution technology X (when applied properly) will provide improvement of type Y.
- Solution technology X (when applied properly to problems of type Y) is more effective, efficacious, or efficient than solution technology Z.

Any utility theory proposed should be precise about the problem it addresses, the way it addresses the problem, and the benefit that would occur from applying the solution technology.

Evaluation results in understandings of a solution technology's efficiency, efficacy, or effectiveness for solving or alleviating the problem.

Solution technology invention is the activity in which the core idea of the hypothesised solution technology is thought out and explained in detail. In this activity, notations of diagrams are developed, descriptions of steps, stages, or phases of new methods or practices are written, or software is developed and tested for correct functioning according to requirements. The development of a solution technology may be just a small refinement of an existing solution technology or it may be the invention of a wholly new, complex solution technology. The process of the invention or creation of a new solution technology are many and varied. The process may involve many small iterations with theory building and evaluation activities, or it may be an entire, top-down development approach, with the resulting solution technology not being evaluated until the whole technology is put together.

Solution technology evaluation is the activity in which the solution technology, as well as the utility theory on which it is based, are tested and evaluated. Solution technologies and utility theories may be evaluated in three main areas as follows:

- In terms of their effectiveness and efficacy in solving or alleviating the problem
- In comparison to other solution technologies
- For other undesirable impacts

Evaluation research is usually empirical and may use methods from the natural or social sciences, depending on the nature of the problem and solution. This leads to the following two broad classes of evaluation activities: (1) artificial evaluation and (2) naturalistic evaluation.

Artificial evaluation is evaluating a solution technology in a contrived, non-real manner. It includes evaluation research methods such as the following:

- Laboratory experiments
- Field experiments
- Simulations

The particular steps to be taken in artificial evaluation depends on the particular research methodology chosen.

Naturalistic evaluation enables a researcher to explore how well or poorly a solution technology works in its real environment (i.e., the organisation or domain). Studies of solution technologies in use, and of technology transfer and adoption of the new technology, can reveal the new problems introduced by the technology itself or problems related to its introduction. Studies can also focus on organisation or domain impact, even after the technology has been in use for many years.

Naturalistic evaluation may be difficult and costly because it must discern the effects of many confounding variables in the real world. For example, it may be impossible to compare with other solution technologies because a project can only be carried out once with the same people, in the same state of mind, and so on. Naturalistic evaluation can be conducted using research methods that include the following:

- Case studies
- Field studies
- Surveys
- Ethnography
- Action research

Naturalistic evaluation is empirical and what is observed or studied are sometimes people's opinions or perceptions rather than a phenomenon itself. For example, successfully solving a problem may be based on whether people perceive it to be solved, rather than some objectively verifiable phenomenon.

III. Methods and Tools

Methods and tools for designing VW applications usually fall into one of four general categories: (1) classic, (2) adapted, (3) ad-hoc, and (4) pertinent.

Classic methods and tools are those originally developed for managing the development process for the classic types of applications in SE, and are used without any adaption for designing VW applications. An example in this category includes the use of questionnaires, online surveys, interviews, and other user-centred design tools and techniques for designing educational VW applications [12].

Adapted methods and tools are those developed by modifying classic methods and tools for use in designing VW applications. An example in this category is the use of VW heuristics (an adaption of Nielsen's Heuristics) for the designing of gaming VW applications [13].

Ad-hoc methods and tools are those developed on a makeshift basis and in response to the lack of pertinent design methods and tools, for use in designing VW applications. An example in this category is the four-dimensional framework for designing and evaluating educational VW applications [14].

Pertinent methods and tools are those developed as suitable or more relevant solutions for designing VW applications. This paper proposes an approach for designing VW applications that uses a pertinent method and tools.

IV. Towards Dynamic VW Applications

Generative design is an established method in art, architecture, built environments, and product design [15], in which a set of rules or an algorithm generates the output of architectural models using a computer program. The proposed approach includes a method for designing VW applications that combines the use of a generative design grammar (GDG) framework and a generative design agent (GDA) model [16] as tools for designing VW applications. Figure 2 illustrates the method, reflecting the integration of the GDG and the GDA.

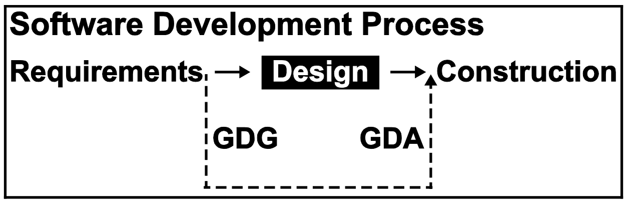

Figure 2. Method for designing VW applications.

The concept of place is a common design metaphor in VWs. This metaphor provides a rich basis for designing VW applications, and allows the use of generative design in the application development process. As such, a VW application is a virtual place built to support various VW activities. Typical VW activities include gaming, socialising, or attending events. A VW application has a layout, includes objects for supporting various VW activities, and interaction rules embedded in the various objects. Interaction rules are typically in the form of scripts.

A GDG is a set of rules that describe a design style. A GDG G is comprised of design rules R, an initial design D_i, and a final state of the design D_f: $G = \{R, D_i, D_f\}$. The basic components of the GDG are the design rules R. The general structure of the GDG for designing VW applications consists of four sets of design rules, which are layout rules R_a, object design rules R_b, navigation rules R_c, and interaction rules R_d: $R = \{R_a, R_b, R_c, R_d\}$. The four sets of design rules correspond to the four phases of designing VW applications, which are as follows:

Layout design: developing the layout of the application in the VW, where each area has a purpose that accommodates certain intended VW activities.

Object design: configuring the application with objects that provide visual boundaries of the VW application, as well as visual cues for supporting the intended VW activities.

Navigation design: specifying navigation methods that use wayfinding aids such as hyperlinks and teleportation devices for assisting the movements of users (using avatars) between different areas of the VW application.

Interaction design: designing algorithms, writing code, and ascribing scripts to objects, to enable users to interact with the VW application.

The GDG framework provides guidelines and strategies for developing GDGs. It enables specifying the general structure of a GDG and its basic components, which are the design rules. By using the GDG framework, GDGs can be developed for designing VW applications, reflecting a certain design style. In order to design an application, the GDG is applied first to generate a design specification of the VW application. Next, the GDA interprets the design specification for the design to be instantiated.

The GDA is a computational agent model with computational processes for reasoning, and therefore can be used for designing VW applications. The GDA's reasoning mechanism uses sensors and effectors as an interface between the GDA and the VW, and for constructing the VW application based on the design specification. Figure 3 is an illustration of the GDA model.

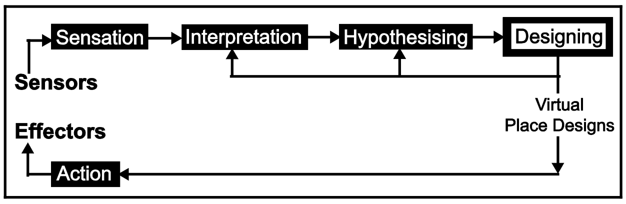

Figure 3. The GDA model and its computational processes.

The GDA wraps around the GDG and provides mechanisms for sensing and changing attributes of VW applications. The GDA's reasoning mechanism has five computational process as follows:

Sensation - using sensors to retrieve raw data from the VW to prepare for the process of interpretation.

Interpretation - interpreting the current design needs and the current state of the VW.

Hypothesising - setting up design goals that aim to eliminate mismatches between the current design needs in the VW and the current state of the VW.

Designing - providing the design of a virtual place (i.e., the VW application) in order to satisfy current design goals.

Action - planning actions for implementing the design specification in the VW, as well as activating the planned actions in the VW.

V. Conclusion and Outlook

Preliminary analyses of the method and tools of the proposed approach, suggest that they are capable of capturing and describing several facets of VW applications (i.e., layout, objects, navigation, and interaction). This suggests that the approach is suitable and robust for designing VW applications. Detailed analyses are currently underway, including comparative evaluations between the proposed approach and existing approaches for developing VW applications. Furthermore, there are also plans to validate and verify the new approach by using simulations such as designing some VW applications whose prototypes are dynamically generated.

References

[1] P. Ralph and Y. Wand, "A proposal for a formal definition of the design concept," in *Design Requirements Engineering: A Ten-Year Perspective*, K. Lyytinen, P. Loucopoulos, J. Mylopoulos, and B. Robinson, Eds. Berlin: Springer-Verlag, 2009, pp. 103-136.
[2] W. Sawyerr and M. Hobbs, "Designing for Usability in 3D Virtual Worlds," in *Proc. 2nd International Workshop on Usability and Accessibility Focused Requirements Engineering*, Karlskrona, 2014, pp. 32-35.
[3] A. Luse, B. Mennecke, and J. Triplett, "The changing nature of user attitudes toward virtual world technology: a longitudinal study," *Science Direct*, vol. 29, pp. 1122-1132, November 2012.

[4] T. E. Yoon and J. F. George, “Why aren’t organizations adopting virtual worlds?,” *Science Direct*, vol. 29, pp. 772-790, May 2013.

[5] J. Livari, “A Paradigmatic Analysis of Information Systems as a Design Science,” *Scandinavian Journal of Information Systems*, vol. 19, no. 2, 2007, pp. 39-64.

[6] S. March and G. Smith, “Design and Natural Science Research on Information Technology.” *Decision Support Systems*, vol. 15, 1995, pp. 251 - 266.

[7] A. Hevner, S. March, J. Park, and S. Ram, “Design Science in Information Systems Research,” *MIS Quarterly*, vol. 28, no. 1, 2004, pp. 75-105.

[8] J. Venable, “A Framework for Design Science Research Activities,” in *Emerging Trends and Challenges in Information Technology Management*, M. Khosrow-Pour, Ed. 2006, pp. 184-187.

[9] J. F. Nunamaker, M. Chen, and T. D. M. Purdin, “Systems Development in Information Systems Research,” *Journal of Management Information Systems*, vol. 7, no. 3, 1991, pp. 89-106.

[10] J. R. Venable and J. Travis, “Using a Group Support System for the Distributed Application of Soft Systems Methodology”, in B. Hope and P. Yoong, Eds. in *Proc. of 10th Australasian Conference on Information Systems*, Wellington, New Zealand, 1999, pp. 1105-1117.

[11] J. R. Venable, “The Role of Theory and Theorising in Design Science Research”, in A. Hevner and S. Chatterjee, Eds. Proc. 1st International Conference on Design Science, 2006, pp. 1-18.

[12] S. Minocha and A. J. Reeves, “Design of learning spaces in 3D virtual worlds: an empirical investigation of Second Life,” *Learning, Media and Technology*, vol. 35, no. 2, 2010, pp. 111-137.

[13] R. Munoz, T. Barcelos, and V. Chalegre, “Defining and Validating Virtual Worlds Usability Heuristics,” in *Proc. 30th International Conference of the Chilean Computer Science Society*, 2011, pp. 171-178.

[14] S. D. Freitas, G. Rebolledo-Mendez, F. Liarokapis, G. Magoulas, and A. Poulovassilis, “Learning as immersive experiences: Using the four-dimensional framework for designing and evaluating immersive learning experiences in a virtual world,” *Brit. J. Educ. Technol.*, vol. 41, no. 1, 2010, pp. 69-85.

[15] T. R. Gruber and D. M. Russell, “Generative Design Rationale: Beyond the Record and Replay Paradigm,” in *Design Rationale: Concepts, Techniques, and Use*, T. P. Moran and J. M. Carroll Eds. Mahwah, NJ: Lawrence Erlbaum, 1996, pp. 323-349.

[16] N. Gu and M. L. Maher, “Dynamic designs of 3D virtual worlds using generative design agents,” in *Computer Aided Architectural Design Futures 2005*, B. Martens and A. Brown, Eds. Berlin: Springer-Verlag, 2005, pp. 239-248.

William Sawyerr is an Associate Lecturer and Researcher at Anglia Ruskin University. His teaching and research experience are in software engineering, and his current research interests are in the area of software design, especially in the context of 3D virtual worlds.