Positioning the Normalized Systems Theory in a Design Theory Framework

Philip Huysmans, Gilles Oorts and Peter De Bruyn
Normalized Systems Institute
Department of Management Information Systems, University of Antwerp, Prinsstraat 13, 2000 Antwerp, Belgium
{*philip.huysmans, gilles.oorts, peter.debruyn*}*@ua.ac.be*

Keywords: Normalized Systems, Design Science, Design Theory, Design Theory Anatomy.

Abstract: Several frameworks have been proposed to define design science and design theory over the last decades. For this reason, positioning a research stream within both paradigms has become a difficult exercise. In this paper, the Normalized System (NS) theory is positioned within design science and design theory, in particular the design theory framework formulated by Gregor & Jones (2007). Normalized Systems theory has been proposed as a way to cope with the ever increasingly agile environment to which organizations and their software applications need to adapt. NS achieves this evolvability by articulating theorems that modular structures need to comply with in order to be evolvable. The results of positioning NS within the presented framework for design theories show that NS almost fully incorporates all components of the design theory anatomy. An application of NS theory in other fields is also discussed, which confirms the applicability of the anatomy of Gregor & Jones (2007) within other disciplines. By positioning Normalized System theory within design science and design theory, we also believe to contribute to the definition of both fields in this paper.

1 INTRODUCTION

During the last decades, the design science research approach has increasingly become of interest in information systems (IS) research. One of the first acknowledgments of this evolution was articulated by March & Smith (1995), who stated that design research (aimed at developing IT artifacts that solve relevant IS problems) could be argued to be more important within IS research than traditional (natural) science (aimed at understanding IT phenomena). As the relatively new IS research area took shape, most researchers agreed with the importance of design science approach as an alternative to the behavioral research approach. On the content of IS design science however, opinions still differ greatly (Baskerville, 2008). For this reason, giving a clear and agreed upon definition of design science is close to impossible. However, some fundamental properties of the discipline seem to exist nevertheless.

The notion of design science was first formulated in 1969 by Simon, who stated that researchers can achieve their research goals "to the degree that they can discover a science of design, a body of intellectually tough, analytic, partly formalizable, partly empirical, teachable doctrine about the design process" (Simon, 1996, p. 113). March & Smith (1995, p. 253) articulate this idea more specific for IS research, stating that "design scientists produce and apply knowledge of tasks or situations in order to create effective artifacts." The same authors state that ultimate purpose of IS design science research is the formulation of IT artifacts that provide utility in solving IS problems, as opposed to formulating and testing hypotheses in a conceptual framework as in traditional natural science.

Although most researchers agree on these principles, one of the discussion points in defining IS design science is whether or not theory should be included in the design science paradigm. Based on the literature review of Venable (2006), one can clearly detect the polemic between proponents and adversaries regarding the inclusion of theory in design science. Whereas Hevner, March, Park & Ram (2004) are ambiguous on the role of theory, March & Smith (1995) clearly state that theory should be solely reserved for natural science and should not be part of design science (Venable, 2006). Many other authors however disagree and argue theory should in fact be an essential part of design science (Nunamaker, Chen & Purdin, 1990; Walls, Widmeyer & El Saway, 1992; Venable & Travis, 1999; Markus, Majchrzak & Gasser, 2002;

Huysmans P., Oorts G. and De Bruyn P.
Positioning the Normalized Systems Theory in a Design Theory Framework.
DOI: 10.5220/0004460900330042
In *Proceedings of the Second International Symposium on Business Modeling and Software Design (BMSD 2012)*, pages 33-42
ISBN: 978-989-8565-26-6

Gregor, 2006; Gregor & Jones, 2007).

Recently Normalized Systems (NS) theory has been proposed as an approach to design evolvable modular structures in software in a deterministic and proven way. While the purpose of this paper is not to define design science or substantiate whether or not design theory is a part of design, we will attempt to position the state of art of the NS research stream within the design science and design theory paradigms. Indeed, as Normalized Systems theory prescribes specific constraints for building IT artifacts in a purposeful way, the research stream seems to be at least closely related to design science and theory. Positioning the stream into current design science methodologies may thus provide us the opportunity to check the approach for its theoretical completeness, while possible arising gaps might identify future research directions. This method is endorsed by Walls, Widermeyer & El Sawy (2004), who state that the notion of an Information System Design Theory can be used as guiding framework to structure design research.

This classification is a challenging exercise as both design science and design theory are not yet decisively defined. In this paper, we will argue that Normalized Systems theory is both a design theory and design science, as it is a proven and normative way of building IT modules (design theory) and it formulates IT artifacts (design science). The point of view on design science and design theory that will be used in this paper, is the distinction expressed by Walls et al. (2004). These authors defined design science different than other authors such as March & Smith (1995) and Hevner et al. (2004) by separating design science from design practice. According to this distinction, design practice is concerned with the creation of instances of artifacts, whereas design science "should create the theoretical foundations of design practice" (i.e. design theories) (Walls et al., 2004, p.48). As this paper will show, Normalized Systems is such a design theory that creates the theoretical foundation for designing evolvable systems.

Positioning Normalized Systems as a design theory will also express the different nature of NS from most of existing design science. According to March & Smith (1995, p. 253) design science should be concerned with creating effective artifacts by producing and applying apply knowledge of tasks or situations. These authors also specify the aforementioned should be the primary goal of design science, "rather than producing general theoretical knowledge" (March & Smith, 1995, p. 253). Normalized Systems theory however uses a theoretical foundation to prove that using Normalized Design Theorems leads to Normalized Systems theory that are stable with respect to the anticipated changes. This alternative approach of Normalized Systems to design science will become clear in the next sections of this paper.

The remainder of this article is structured as follows: in the following section, we discuss the origin and structure of Normalized Systems theory. In the third section, we will position Normalized system theory within the design theory anatomy suggested by Gregor & Jones (2007). In the discussion we will take a look at the results of this evaluation. The anatomy used in this paper will also be placed within its context and the applications of the evaluation will be discussed.

2 NORMALIZED SYSTEMS THEORY

In the current information-intensive and agile business environment, organizations as well as their supporting software applications need to cope with changes in their structure and functionality. The way in which an organization can assimilate these changes, determines its evolvability. On top of these changes, systems are also faced with an ever increasing size and complexity of their structure and functionality. To tackle these two challenges, *modularity* has frequently been suggested to divide complex system in easier to handle subsystems and cope with the evolvability requirement by allowing the modules to change independently (Baldwin & Clark, 2000). In fact, modularity is a core concept from systems theory and has been applied in many different application domains, including software engineering (Parnas, 1972) were multiple modular primitives have been defined for representing *data* (e.g., structures), *actions* (e.g., procedures) or both (e.g., classes) as the core of information systems. However, (hidden) coupling between those modules tends to limit the anticipated evolvability (De Bruyn & Mannaert, 2012). Thoroughly based on concepts and reasoning from systems theory and thermodynamics, Normalized Systems theory (NS) was recently proposed to strictly guide engineers in devising such *evolvable modularity*. While it was originally applied for software architectures, its relevance for other application domains (e.g., organizational design) has been demonstrated previously (Van Nuffel, 2011; Huysmans, 2011).

First, with its primary aim of enabling evolvability in software architectures, NS defines the occurrence of so-called *combinatorial effects*: changes within the modular structure of which the impact is dependent on the size of the system they are applied to (Man-

naert, Verelst & Ven, 2011, 2012). Such combinatorial effects are clearly undesirable in the context of evolvability. Indeed, as a system grows throughout time, a combinatorial effect would imply that the effort required to implement a same kind of change becomes ever more complex as time goes by. Moreover, the concept of *stability from systems theory* is highly related to this reasoning and offers clues in how to avoid combinatorial effects. In systems theory, stability is regarded as an essential property of systems and implies that a bounded input function should always result in a bounded output function, even if an unlimited time period $T \rightarrow \infty$ is considered. Applied to information systems, this means that a bounded set of changes (selected from the so-called *anticipated changes* such as "add additional data attribute" or "add additional action entity") should result in a bounded impact to the system, even for $T \rightarrow \infty$ (i.e., an unlimited systems evolution is considered). Consequently, stability reasoning confirms that the impact of changes should not be dependent on the size of the system, but only related to the kind of change performed (and hence, combinatorial effects should be avoided). As such, normalized systems are defined as systems exhibiting stability and lacking any combinatorial effects towards a defined set of anticipated changes (Mannaert et al., 2011, 2012).

In order to obtain such normalized systems, NS theory proposes four *theorems* which should be systematically adhered to (Mannaert et al., 2012):

- *Separation of Concerns*, stating that each change driver (concern) should be separated from other concerns;
- *Action Version Transparency*, stating that action entities should be updateable without impacting their calling action entities;
- *Data Version Transparency*, stating that data entities should be updateable without impacting their calling action entities ;
- *Separation of States*, stating actions in workflow should be separated by state (and called in a statefull way).

For each of these theorems it has been formally proven that any violation against them at any time will result in a combinatorial effect (Mannaert et al., 2012). Consequently, if the aim is to obtain a true normalized system, these principles for building software architectures should all be consistently applied. In reality, the systematic isolation of all concerns prescribed by the theory, result in a very fine-grained modular structure. While exhibiting a proven degree of evolvability, the structure in itself may be regarded as complex in the sense that the system becomes an aggregation of many fine-grained instantiations of the modular primitives offered by the employed programming language, many more than in commonly developed software applications.

Therefore, a set of *elements* were proposed to make the realization of normalized systems more feasible. These elements are each a structured aggregation (i.e., encapsulation) of the available software primitives and together providing the core functionality of information systems, be it in a highly generic and reusable way. As such, the internal structure of these elements could be considered to be a set of design patterns regarding higher-level modular building blocks, adhering to the above-described theorems. These five elements are (Mannaert et al., 2011):

- *action element*, a structured composition of software constructs to encapsulate an action construct into an action element;
- *data element*, a structured composition of software constructs to encapsulate a data construct into a data element;
- *workflow element*, a structured composition of software constructs to create an encapsulated workflow element (representing a sequence of action elements);
- *trigger element*, a structured composition of software constructs to create an encapsulated trigger element (controlling for and representing the activation of a workflow or action element)
- *connector element*, a structured composition of software constructs to create an encapsulated connector element (allowing the stateful connection of data elements with external systems).

Each of these elements are described in more detail in Mannaert et al. (2011) and illustrated to be able to cope with a set of anticipated changes in a stable way. Normalized systems are then typically implemented by creating a set of *n* instantiations of the five elements.

Additionally, recent research efforts have demonstrated that reasoning based on the concept of *entropy from thermodynamics* (1) supports and (2) further extends NS theory (Mannaert, De Bruyn & Verelst, 2012a). More specifically, the definition of entropy as employed in statistical thermodynamics was used for this purpose, i.e., the number of microstates consistent with the same macrostate (Boltzmann, 1995). Applied to information systems, *microstates* are operationalized as binary values expressing the correct or erroneous processing of a programming language construct, while the *macrostate* is associated with loggings or database entries representing the correct or erroneous processing of the software system.

Hence, when it is uncertain which microstate configuration brought about the observed macrostate, entropy (uncertainty) is present in the system. The larger the number of microstates consistent with the same macrostate, the higher is the amount of entropy in the system. In Mannaert et al. (2012a), it was further illustrated that the aim of minimizing or controlling entropy in modular system, highly coincides with the four theorems explained above. Moreover, a set of of two new theorems were suggested based on this entropy reasoning (Mannaert et al., 2012a):

- *Data Instance Traceability*, requiring each version and values of an instance of a data structure to be tracked;
- *Action Instance Traceability*, requiring each version and thread of an instance of a processing structure to be tracked.

Finally, several real-life implementations of NS applications have been successfully deployed up to this moment. Some of them have been already briefly discussed in Mannaert et al. (2011).

3 CLASSIFICATION OF NORMALIZED SYSTEMS THEORY

In this section, Normalized Systems theory will be classified within a design theory framework. The Information System Design Theory (ISDT) framework that will be used in this paper is the design theory anatomy proposed by Gregor & Jones (2007). This anatomy is an extension of the ISDT framework proposed by Walls et al. (1992), whose sources date back to the work of Dubin (1978) on theory of the natural science type and the work of Simon (1996) on sciences of the artificial. In their work, Walls et al. (1992) attempted to formulate a prescriptive theory that articulates how a design process can effectively be carried out. Looking back on the formulation of their ISDT, (Walls et al., 2004) conclude that, although used scarcely, it can be helpful as a guiding framework that helps in structuring the "how to" of the design progress based on a theoretical foundation. As the work of Walls et al. (1992) was just an initial attempt to define an Information Systems Design Theory, several reflections and reactions have been published on their work (Gregor & Jones, 2007; Walls et al., 2004). For example Gregor & Jones (2007) clarify that the goal of a design theory can be either a methodology or a product, which was not yet clearly defined by Walls et al. (1992) (Gregor & Jones, 2007). The remainder of this section will show that Normalized System is an example of a design theory defining the design of a product (e.g. a system). Furthermore Gregor & Jones (2007) argue that two structural components of a theory formulated by Dubin (1978) are lacking from the anatomy formulated by Walls et al. (1992), namely "units" and "system states". These constructs are defined as the building blocks of theory and the range of system states that the theory covers respectively (Gregor & Jones, 2007), constructs that will be shown to be part of the Normalized System paradigm in this paper.

Considering these missing constructs, Gregor & Jones (2007) argue that an information system design theory consists of eight components. The first six components are called the Core Components, as they are essential to determine how an artifact can/should be constructed. The other two components can have a positive influence on the credibility of the work, but can be defined later (Gregor & Jones, 2007). The results of the classification of Normalized Systems Design Theory by these components are shown in Table 1. In the next paragraphs, we will discuss the classification of the NS theory within these eight components.

The first component Gregor & Jones (2007) define is the *purpose and scope* of a design theory. This component states "what the system is for, or the set of meta-requirements or goals that specifies the type of system to which the theory applies" (Gregor & Jones, 2007, p. 325). According to the same authors, the environment in which the artifact should operate is an important factor to consider. To understand the purpose and goal of Normalized Systems, it is indeed important to keep in mind the agile environment in which modern systems should operate. As mentioned earlier, the purpose of Normalized Systems theory is the elimination of combinatorial effects (towards a set of anticipated changes), as a means to achieve evolvability of information systems. Combinatorial effects however not only appear in information systems, but can be observed in a very broad spectrum of domains. Therefore Normalized Systems Design Theory is not limited to information systems and, as will be discussed in Section 4.3 of this paper, can apply to many different natural and artificial phenomena. This is in agreement to Gregor & Jones (2007) who state that the applicability of their design theory anatomy is possibly wider than the IS discipline, as it is in itself based on sources from other disciplines.

Constructs are "the entities of interest in the theory", and "are at the most basic level in any theory" (Gregor & Jones, 2007, p. 325). Other authors, such as March & Smith (1995, p. 256), define constructs

Table 1: Classification of Normalized Systems within the anatomy of Gregor & Jones (2007) and the components of Walls et al. (1992).

Gregor & Jones (2007)	**Walls et al. (1992)**	**Normalized Systems theory**
Core Components		
1. Purpose and scope	Meta-requirements	Evolvable software architectures by elimination of combinatorial effects
2. Constructs		Combinatorial effects, modularity, action/data
3. Principles of form and function	Meta-description	Five Normalized Elements Four Normalized Design Theorems
4. Artifact mutability		Anticipated changes
5. Testable propositions	Product hypothesis Process hypothesis	When theorems are applied, no combinatorial effects occur with regard to the anticipated changes
6. Justificatory knowledge	Product kernel theories Process kernel theories	Systems theory (cf. stability) Thermodynamics (cf. entropy)
Additional Components		
7. Principles of implementations	Design method	Supporting applications (e.g., "Prime radiant")
8. Expository instantiation		Real-life NS implementations (by means of instantiations of the elements)

as "the vocabulary of a domain [. . .] used to describe problems within the domain and to specify their solutions". Although the combination of these definitions gives a clear understanding of a construct, the authors believe there is still a certain subjectivity in determining what "vocabulary" or shared knowledge should be considered a construct and what should not. Within the Normalized System theory, the authors recognize three constructs. The first and second construct are generally accepted and known within IT, namely the concept of modularity and the basic building blocks of an information systems: data and actions. The third construct of Normalized Systems theory is combinatorial effects which the NS theory is determined to eliminate, as formulated by Mannaert & Verelst (2009). These three concepts constitute the main constructs of NSDT, although due to the subjectivity of the definition of this component it does seems possible to argue there are other constructs that were not mentioned.

To avoid combinatorial effects, the NS theory states that applications should be structured using the five elements defined by Mannaert & Verelst (2009), which are based on the body of thought of the Normalized Design Theorems. As the *principles of form and function* are defined as "the principles that define the structure, organization and functioning of the design product" by (Gregor & Jones, 2007, p. 325), it is clear the Normalized Elements and Normalized Design Theorems make up this component of the design anatomy. The Normalized design Theorems and Elements specify both structural and functional properties of artifacts by providing guidelines and patterns for constructing instances of the artifacts. Whereas the theorems articulates the general principles that need to be applied, the elements are in fact a design pattern as they specify a possible way of comply with the Normalized Theorems. For the imposed internal structure of the five types of Normalized Elements make applications free of combinatorial effects, the aggregation of these instances of elements subsequently makes up a Normalized System. The Normalized Design Theorems on the other hand can also be considered as a principle of form and function, as they are the guiding principles for constructing the Normalized Elements.

The next component has to do with a special nature of an IS artifact that Gregor & Jones (2007, p. 326) recognize, stating that IS artifacts are very mutable and constantly evolving. They also believe that:

> "the way in which they [i.e. IT artifacts] emerge and evolve over time, and how they become interdependent with socio-economic contexts and practices, are key unresolved issues for our field and ones that will become even more problematic in these dynamic and

innovative times".

Evolvabilty and agility are in fact exactly the goal and purpose of Normalized Systems theory, since Normalized Systems are highly evolvable and stable systems based on structured elements that minimize combinatorial effects. For this reason the component of *artifact mutability*, which is formally defined as "the changes in state of the artifact anticipated in the theory" (Gregor & Jones, 2007, p. 322) can be cleared recognized as the anticipated changes of NS.

The next component Gregor & Jones (2007) define, is *testable propositions*. These are claims or predictions about the quality of a system or tool when the design theory is applied. As such, the testable proposition of NS theory can be formulated as the elimination of combinatorial effects when the principles of form and function are pursued consistently. Although this component seems clearly defined within NS theory, the definition of the component also requires the propositions to be testable. According to Walls et al. (1992), the assertion that applying a set of design principles will result in an artifact that achieves its goal can be verified by building (an instance of) the artifact and testing it. Applying this verification method on Normalized Systems, the proposition of NS theory (the elimination of combinatorial effects) can be verified/tested by building a Normalized system according to the principles of form and function and proving that the system is exempt of combinatorial effects.

Walls et al. (1992) formulated the idea that kernel theories should be part of an Information System Design Theory (ISDT). According to these authors, kernel theories govern both the design requirements and the design process. As Walls et al. (1992) believe, these two aspects should be separated and therefore defined both product kernel theories and process kernel theories. Walls et al. (2004) elucidate the importance of kernel theories for design science, by stating that the design science process uses these theories and combines them with existing artifacts to formulate new design theories. Gregor & Jones (2007, p. 327) however argue that the two types of kernel theories (i.e. process and product kernel theoris) are "a linking mechanism for a number, or all, of the other aspects of the design theory" and should be considered as one component, the *justificatory knowledge* that explains why a design works. This symbiosis is substantiated by the argument that the design process and design product are mostly founded by a single kernel theory (e.g. the justificatory knowledge). They define this component as "the underlying knowledge or theory from the natural or social or design sciences that gives a basis and explanation for the design" (Gregor & Jones, 2007, p. 322). According to this definition, the underlying justification for Normalized Systems theory is twofold. First the central idea of Normalized Systems theory is systems stability, as formulated in the systems stability theory which states that a bounded input should always result in a bounded output (BIBO-principle). In Normalized Systems theory, this is interpreted as the transformation of functional requirements into software primitives (Mannaert et al., 2011). Secondly, Normalized Systems theory shows compatibility with the concept of entropy, as has been discussed in Section 2. Initial research efforts largely validate the use of Normalized System principles when studying the NS theory from the point of view of entropy theory (Mannaert, De Bruyn & Verelst, 2012b).

The next component of a design theory is its *principles of implementation*. Gregor & Jones (2007) consider this and the next components as additional components that are not a no essential part of a design theory but should be formulated if the credibility of the theory is to be enhanced. Concerning this component, we can refer to the model taxonomy of Winter, Gericke & Bucher (2009). According to this taxonomy, Normalized Systems theory should be classified rather as a prescriptive model with result recommendation ("a model") than a model with activity recommendation ("a method"), referring to the clear principles of form and function of NS mentioned earlier. Although the emphasis within NS theory is on the "result view" rather than the "activity view", Winter et al. (2009) argue that both are views on the same "problem solving artifact". This similarity of methods and models is also apparent in the classification used in this paper, in the form of the similarities between the principles of form and function and the principles of implementation. The form or architecture of an artifact can in itself be used as a underlying principle and target on which a method and guidelines for construction of the artifact are based. As opposed to clearly defined principles of form and function, such a formal methodology in the form of a procedure that explicitly articulates the steps that need to be followed to construct normalized elements, does not exist. The formulation of the Normalized elements simply happens while keeping the theorems at the back of one's mind, and is helped by some supporting applications (e.g. "Prime radient"). These applications are more than a tool, as they provide guidelines for constructing Normalized elements. For this reason, they could be considered the principles of implementation of Normalized System theory.

The final component, *expository instantiation*, has two functions: it shows the applicability of a design

theory and it can be used as a way to explain a design (Gregor & Jones, 2007). Instantiated artifacts are the embodiment of this component. Normalized Systems theory has been used in the development of several several software applications. According to the definition of Gregor & Jones (2007), the expository instantiation of NS are these applications and the Normalized Elements they consist of. The seven applications that have been implemented according to the NS principles range from a system for distribution of multimedia to a system responsible for the management of power units on high-speed railroads. Mannaert et al. (2012) describe these implementations in greater detail.

4 DISCUSSION

4.1 Reviewing NS According to the Anatomy

By positioning NS theory within the design theory anatomy of Gregor & Jones (2007), it becomes clear that NS theory incorporates close to all components of the anatomy. One could argue that there still is one significant gap between the anatomy and NS theory: an explicit method to guide the construction of Normalized elements (part of the principles of implementation). However, overall the NS theory shows great similarities with the design anatomy studied in this paper. In conclusion to this strong similarity, we could argue the term Normalized System Design Theory is appropriate to describe the presented theory.

4.2 Design Science vs Design Theory

Although there was opted to compare Normalized System theory to the design theory anatomy of Gregor & Jones (2007) in this paper, the authors acknowledge several other frameworks exist that attempt to define design science and design theory. To put the used anatomy into context, an overview of the different definitions and point of views on design science and design theory will be discussed in this section.

Over the last decades, several definitions and frameworks have been published that tried to formalize and structure IS design science. These publications however show some notable differences in beliefs and definition of IS design science and theory. The work of March & Smith (1995) mainly focuses on the processes and research outputs of IS design science, and emphasizes that theory should not be part of design science. In their opinion, theories are reserved for natural sciences, and design science should "strive to create models, methods and implementations" (March & Smith, 1995, p. 254) On the other hand, authors such as Nunamaker et al. (1990), Walls et al. (1992), Gregor (2006) tried to define this very notion of an IS design theory. It should be mentioned that these authors distinguish a design theory form a theory in natural sciences. According to their definition, a design theory is as a prescriptive theory that "says how a design process can be carried out in a way which is both effective and feasible", in contrast to the "explanatory and predictive theories found in natural or physical sciences" (Walls et al., 1992, p. 37).

Another point of disagreement seems to be the outcome of design science. Some academics (March & Smith, 1995; Hevner et al., 2004; Baskerville, 2008) believe the core of design science is "directed toward understanding and improving the search among potential components in order to construct an artifact that is intended to solve a problem" (Baskerville, 2008, p.441). According to this point of view, design science should therefore be limited to defining artifacts. The work of Walls et al. (1992), Walls et al. (2004) and Gregor & Jones (2007) however suggests design science should also be concerned with building design knowledge in the form of design theories, something that is simply not mentioned by the aforementioned authors. Walls et al. (2004) clearly indicate this difference by distinguishing between design science and design practice. Whereas design practice actually creates instances of artifacts, design science "should create the theoretical foundations of design practice" (i.e. design theories) (Walls et al., 2004, p.48).

Previous differences clearly show that there is an agreement that design science can not be equated with design theory (Baskerville, 2008). To the authors knowledge and feeling, there are however no publications that explicitly and decisively specify the relationship between design theory and design science, nor has there been a discussion between the proponents of both points of view. Therefore, other than claiming NS is both design science and design theory, precisely positioning Normalized Systems theory within design science is simply impossible at this point. The nature of Normalized Systems theory also makes it very difficult to relate NS to the generally accepted conception of IS design science formulated by Hevner et al. (2004). As discussed earlier, Normalized Systems theory is based on proven theorems that deterministically define that combinatorial effects will be eliminated when the four theorems are systematically applied to the construction of elements. According to Hevner et al. (2004) however, design science is typically concerned with designing artifacts

that are evaluated and improved upon and which outcome is a final design artifact that performs better than other artifacts. Although it is clear that Normalized systems theory is in contrast with this design process, NS in its essence still is design science as it is "concerned with devising artifacts to attain goals" (March & Smith, 1995, p. 253). In the authors opinion, this also shows that the current conception of design science is too stringent to allow design theories such as Normalized Systems theory to be positioned within design science. Even the notion of "theory-based design science" (Baskerville, 2008) does not cover the basic principles of Normalized Systems theory, as this type of design science should be concerned with theory testing rather than formulating a theory (which Normalized Systems does).

4.3 Applications of NS

The positioning of NS in the presented framework is an important basis for further research in the context of the NS theory. Within the design science methodology, the framework allows the positioning of at least two research directions. First, existing gaps between the current research results and the framework components can be considered, as discussed in Section 4.1. Since NS was not developed by following a specific design science methodology, it is possible that certain components are missing or insufficiently described. Consequently, this research direction would focus on the original domain of NS, i.e., software. Second, different domains can be researched using the NS theory. As a motivation for this research approach, consider the use of the systems theoretic concept of *stability* and the thermodynamic concept of *entropy* in the justificatory knowledge component in Table 1. The interpretation of such fundamental concepts indicates that the NS theory could possibly be applied to other research fields as well: in itself, these concepts do not originate from the software domain. As discussed, the application of successful solutions from related research fields is an important goal of the design science methodology. As an example of such a research project, we mention how the NS theory has already been applied to the research field of Business Process Management (Van Nuffel, 2011). In a research project in this research direction, one first has to validate whether the purpose and scope of the NS theory can be applied to that research field as well. Consequently, it is crucial to consider the research field as a *modular structure*, where *combinatorial effects* between the modules are a relevant issue. Put differently, one has to apply the constructs defined by the NS theory in the specific research field. For the Business Process Management field, Van Nuffel (2011, p. 89) explicitly mentions: "a business process essentially denotes a *modular structure*, of which the building blocks should be loosely coupled in order to avoid the described *combinatorial business process effects*". Next, the framework prescribes that principles of form and function should be described. In the software domain, NS describes four theorems, which provide the guidelines for the design of five software elements. In the Business Process Management domain, the four principles have been applied to the design of business processes. This has resulted in the formulation of 25 guidelines which describe necessary modular structures to avoid combinatorial business process effects. An example of such a guideline is the "*Notifying Stakeholders*" guideline (Van Nuffel, 2011, p. 143): "Because notifying, or communicating a message to, stakeholders constitutes an often recurring functionality in business processes, a designated business process will perform the required notification". Indeed, omitting to separate this process would result in a combinatorial effect when changes to the notification process need to be applied. Similar to NS on the software level, this insight is not necessarily new: other authors, such as Erl (2008), Papazoglou & Van Den Heuvel (2006) and Cohen (2007) provide similar guidelines. However, these authors do not provide a theoretical framework to motivate the formulation of such guidelines. In order to mature the field towards a design *science*, the authors of this paper believe that such a theoretical framework is vital. Consequently, the proposed framework is important as a methodological guide for future research. Moreover, the framework allows not only to position current research results, but also to identify missing elements. It can be noted that these guidelines can be positioned between the four NS principles and the NS elements: they are an application of the NS principles, but are not sufficient to completely specify process elements. Therefore, additional research is required to further the insight on these guidelines, and arrive at such elements. Consistent with the approach provided in the framework in Table 1, anticipated changes will need to be formulated (as prescribed in the artifact mutability component), and the absence of combinatorial effects with regard to these anticipated changes needs to be demonstrated (as prescribed in the testable propositions component).

5 CONCLUSIONS

In this paper, we made an attempt to position Normalized System theory within design science and de-

sign theory frameworks. Although we argued that NS theory in its essence is design science, we also showed that it does not completely fit in existing design science frameworks. Positioning Normalized Systems within the design theory anatomy of Gregor & Jones (2007) however showed Normalized Systems strongly resembles a design theory. The similarities are to the extent that NS can be formulated as the Normalized Systems Design Theory.

Positioning NS in the anatomy of Gregor & Jones (2007) has shown to present several contributions in this paper.

First, it endorses the applicability of the framework by showing that a theory such Normalized Systems can indeed be soundly positioned within the framework. Furthermore the positioning ratifies the premise of Gregor & Jones (2007) who stated that the possibility of applying the anatomy in other disciplines could be a question for further research. Indeed, this paper showed that it is initially possibly to position the application of NS within Business Process Management within the anatomy. This proves that the design theory anatomy of Gregor & Jones (2007) can indeed be applied in other disciplines than IS design science.

A second contribution is that positioning NS within the anatomy gave the opportunity to check NS for its theoretical completeness, which is consistent with the view of Walls et al. (2004) who state that an ISDT framework can be used as guiding framework to structure design research. The gaps that were identified according to the application of NS within Business Process Management also indicated future research directions.

Finally the authors believe that this papers contributes to the discussion on design science and design theory. It has been shown that Normalized Systems theory is an apparent example of a design theory. As Normalized System theory should however also be considered design science, the authors believe to have contributed to both the definition of design science and the elucidation of the relationship between design science and design theory.

ACKNOWLEDGEMENTS

P.D.B. is supported by a Research Grant of the Agency for Innovation by Science and Technology in Flanders (IWT).

REFERENCES

Baldwin, C. Y. & Clark, K. B. (2000). *Design Rules: The Power of Modularity*. Cambridge, MA, USA: MIT Press.

Baskerville, R. (2008). What design science is not. *European Journal of Information Systems*, *17*, 441–443.

Boltzmann, L. (1995). *Lectures on Gas Theory*. Dover Publications.

Cohen, S. (2007). Ontology and taxonomy of services in a service-oriented architecture. *The Architecture Journal*, *11*, 10.

De Bruyn, P. & Mannaert, H. (2012). Towards applying normalized systems concepts to modularity and the systems engineering process. In *Proceedings of the Seventh International Conference on Systems (ICONS 2012)*.

Dubin, R. (1978). *Theory building, revised edition*. London: Free Press.

Erl, T. (2008). *SOA: Principles of Service Design*. Prentice Hall Service-Oriented Computer Series. Upper Saddle River, NJ: Prentice Hall.

Gregor, S. (2006). The nature of theory in information systems. *MIS Quarterly*, *30*(3), 611 – 642.

Gregor, S. & Jones, D. (2007). The anatomy of a design theory. *Journal of the Association for Information systems*, *8*(5), 312–335.

Hevner, A., March, S., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, *28*(1), 75–105.

Huysmans, P. (2011). *On the Feasibility of Normalized Enterprises: Applying Normalized Systems Theory to the High-Level Design of Enterprises*. PhD thesis, University of Antwerp.

Mannaert, H., De Bruyn, P., & Verelst, J. (2012a). Exploring entropy in software systems: Towards a precise definition and design rules. In *Proceedings of the Seventh International Conference on Systems (ICONS)*.

Mannaert, H., De Bruyn, P., & Verelst, J. (2012b). Exploring entropy in software systems: Towards a precise definition and design rules. In *Proceedings of ICONS 2012: The Seventh International Conference on Systems*, (pp. 93 – 99). IARIA.

Mannaert, H. & Verelst, J. (2009). *Normalized Systems: Re-creating Information Technology Based on Laws for Software Evolvability*. Koppa.

Mannaert, H., Verelst, J., & Ven, K. (2011). The transformation of requirements into software primitives: Studying evolvability based on systems theoretic stability. *Science of Computer Programming*, *76*(12), 1210–1222. Special Issue on Software Evolution, Adaptability and Variability.

Mannaert, H., Verelst, J., & Ven, K. (2012). Towards evolvable software architectures based on systems theoretic stability. *Software: Practice and Experience*, *42*(1), 89–116.

March, S. T. & Smith, G. F. (1995). Design and natural science research on information technology. *Decision Support Systems*, *15*(4), 251 – 266.

Markus, M. L., Majchrzak, A., & Gasser, L. (2002). A design theory for systems that support emergent knowledge processes. *MIS Quarterly*, *26*(3), 179 – 212.

Nunamaker, Jr., J. F., Chen, M., & Purdin, T. D. M. (1990). Systems development in information systems research. *J. Manage. Inf. Syst.*, *7*(3), 89–106.

Papazoglou, M. P. & Van Den Heuvel, W.-J. (2006). Service-oriented design and development methodology. *International Journal of Web Engineering and Technology*, *2*(4), 412–442.

Parnas, D. L. (1972). On the criteria to be used in decomposing systems into modules. *Communications of the ACM*, *15*(12), 1053–1058.

Simon, H. (1996). *The Sciences of the Artificial* (Third edition ed.). Cambridge, MA: MIT Press.

Van Nuffel, D. (2011). *Towards Designing Modular and Evolvable Business Processes*. PhD thesis, University of Antwerp.

Venable, J. (2006). The role of theory and theorizing in design science research. In *DESRIST Proceedings*.

Venable, J. & Travis, J. (1999). Using a group support system for the distributed application of soft systems methodology. In *Proceedings of the 10th Australasian Conference on Information Systems, Wellington, New Zealand*, (pp. pp 1105–1117).

Walls, J., Widmeyer, G., & El Saway, O. (1992). Building an information system design theory for vigilant eis. information systems research. *Information Systems Research*, *3*(1), 36–59.

Walls, J. G., Widermeyer, G. R., & El Sawy, O. A. (2004). Assessing information system design theory in perspective: How useful was our 1992 initial rendition? *Journal of Information Technology Theory and Application (JITTA)*, *6*(2), 43–58.

Winter, R., Gericke, A., & Bucher, T. (2009). Method versus model - two sides of the same coin? volume 34 of *Lecture Notes in Business Information Processing*, (pp. 1–15). 5th International Workshop on Cooperation and Interoperability, Architecture and Ontology, Amsterdam, NETHERLANDS.