THE UNIVERSITY OF
WARWICK

Original citation:
Henfridsson, Ola, Mathiassen, Lars and Svahn, Fredrik. (2014) Managing technological change in the digital age : the role of architectural frames. Journal of Information Technology, 29 (1). pp. 27-43.

Permanent WRAP url:
http://wrap.warwick.ac.uk/76098

Copyright and reuse:
The Warwick Research Archive Portal (WRAP) makes this work by researchers of the University of Warwick available open access under the following conditions. Copyright © and all moral rights to the version of the paper presented here belong to the individual author(s) and/or other copyright owners. To the extent reasonable and practicable the material made available in WRAP has been checked for eligibility before being made available.

Copies of full items can be used for personal research or study, educational, or not-for profit purposes without prior permission or charge. Provided that the authors, title and full bibliographic details are credited, a hyperlink and/or URL is given for the original metadata page and the content is not changed in any way.

Publisher's statement:
"This is a post-peer-review, pre-copyedit version of an article published in Journal of Information Technology. The definitive publisher-authenticated version Henfridsson, Ola, Mathiassen, Lars and Svahn, Fredrik. (2014) Managing technological change in the digital age : the role of architectural frames. Journal of Information Technology, 29 (1). pp. 27-43. doi:10.1057/jit.2013.30 is available online at: http://www.palgrave-journals.com/jit/journal/v29/n1/full/jit201330a.html "

A note on versions:
The version presented here may differ from the published version or, version of record, if you wish to cite this item you are advised to consult the publisher's version. Please see the 'permanent WRAP url' above for details on accessing the published version and note that access may require a subscription. For more information, please contact the WRAP Team at: publications@warwick.ac.uk

warwick**publications**wrap

highlight your research

http://wrap.warwick.ac.uk

[Research Article]
MANAGING TECHNOLOGICAL CHANGE IN THE DIGITAL AGE: THE ROLE OF ARCHITECTURAL FRAMES

post-print version of :
Henfridsson, O., Mathiassen, L., and Svahn, F. 2014. "Managing Technological Change in the Digital Age: The Role of Architectural Frames," *Journal of Information Technology* (29:1), pp 27-43.

Ola Henfridsson*
University of Warwick, Coventry, CV4 7AL United Kingdom
E-mail: ola.henfridsson@wbs.ac.uk
Phone: +46703779729

Lars Mathiassen
Georgia State University, PO Box 4015, Atlanta, GA 30302-4015
E-mail: lmathiassen@ceprin.org

Fredrik Svahn
Chalmers University of Technology, 412 96 Göteborg, Sweden
&
Viktoria Institute
E-mail: Fredrik.Svahn@chalmers.se

* Corresponding author

42 pages including figures, tables, references, appendix, and this title page
Running title: Managing Technological Change in the Digital Age

Abstract. Inspired by Herbert Simon's notion of nearly decomposable systems, researchers have examined modularity as a powerful approach to manage technological change in product innovation. We articulate this approach as the hierarchy-of-parts architecture and explain how it emphasizes decomposition of a design into loosely coupled parts and subsequent aggregation of these into an industrial product. To realize the scale benefits of modularity, firms successively freeze design specifications before production and therefore only allow limited windows of functionality design and redesign. This makes it difficult to take advantage of the increased speed by which digitized products can be developed and modified.

To address this problem, we draw on Christopher Alexander's notion of design patterns to introduce a complementary approach to manage technological change that is resilient to digital technology. We articulate this approach as the network-of-patterns architecture and explain how it emphasizes generalization of ideas into patterns and subsequent specialization of patterns for different design purposes. In response to the increased digitization of industrial products, we demonstrate the value of complementing hierarchy-of-parts thinking with network-of-patterns thinking through a case study of infotainment architecture at an automaker.

As a result, we contribute to the literature on managing products in the digital age: we highlight the properties of digital technology that increase the speed by which digitized products can be redesigned; we offer the notion of architectural frames and propose hierarchy-of-parts and network-of-patterns as frames to support innovation of digitized products; and, we outline an agenda for future research that reconsiders the work of Simon and Alexander as well as their followers to address key challenges in innovating digitized products.

AUTHOR BIOGRAHIES

Ola Henfridsson is a Professor of Information Systems and Management at Warwick Business School, University of Warwick, UK. His research interests include digital innovation, digital infrastructure, platforms, and technology management. The outcome of this research has been published in *MIS Quarterly, Information Systems Research*, *Organization Science*, and other top journals. He is a Senior Editor of *Journal of Information Technology* and a former Senior Editor of the *MIS Quarterly.* Ola also serves on the editorial boards of *Information Technology and People* and *Journal of the Association for Information Systems*. He is a Visiting Professor at Department of Applied Information Technology, Chalmers University of Technology, Sweden.

Lars Mathiassen is Georgia Research Alliance Eminent Scholar, Professor at the Computer Information Systems Department, and co-Founder of The Center for Process Innovation at Georgia State University. His research focuses on development of software and information services, on IT-enabled innovation of business processes, and on management and facilitation of organizational change processes. He has published extensively in major information systems and software engineering journals and has co-authored several books on the subject including Professional Systems Development, Computers in Context: The Philosophy and Practice of Systems Design, Object Oriented Analysis & Design, and Improving Software Organizations: From Principles to Practice. He can be reached at lmathiassen@ceprin.org.

Fredrik Svahn is Postdoctoral Research Associate at the Department of Applied Information Technology, Chalmers University of Technology, Sweden. He is also active at the applied research institute Viktoria Swedish ICT. Fredrik earned his Ph.D. in Informatics at Umeå University in 2012, and a M.Sc. in Applied Physics and Electrical Engineering at Linköping Institute of Technology in 1996. He has more than 10 years of professional experience from Defense, Telecom and Automotive industries. Fredrik's research is focused on new patterns of innovation emerging in response to digitalization of manufactured products. In particular, he studies how organizations can architect digital products to improve its generative capability.

1. INTRODUCTION

There is hardly any watchmaker receiving more attention in the scholarly literature than Hora. Serving as key actor in Herbert Simon's (1962) renowned parable, Hora, unlike his colleague Tempus, had designed subassemblies of the many parts of his watches. In turn, he aggregated these subassemblies into larger subassemblies, forming a hierarchical system. Organizing "his work such that there was a one-to-one mapping between functions and sub-assemblies" (Garud et al. 2003, p.2), Hora radically decreased the design complexity, meaning that he, when interrupted by new customers, only lost a fraction of the assembly work compared to Tempus. This story and the idea of a product decomposed into a hierarchy of stable subassemblies have served as inspiration for a substantive literature in technology and innovation management (Baldwin and Clark 2000, Clark 1985, Garud and Kumaraswamy 1995, Garud et al. 2003, Henderson and Clark 1990). This literature argues that modularity offers a powerful architecture for achieving economies of scale by drawing on the wealth of external production capabilities available on a global market (Langlois 2007, Langlois and Robertson 1992, Sturgeon 2002).

The parable of Hora is primarily about creating a smart design for production. As Hora created subassemblies consisting of ten elements each, he anticipated customers would want different watches based on similar components. Following Hora's modular design practice, firms may split a product "into a set of components with low variety and high reusability, and another set with high variety and low reusability" (Baldwin and Woodard 2009, p. 25). In order to reap the scale benefits of modularity, the consequential architecture is then frozen for suitable periods of time (Baldwin and Clark 2000, Iansiti 1995). In this way, modularity has served as a powerful way to address technological change in aircrafts, automobiles, consumer electronics, household appliances, personal computers, software, test instruments, and power tools (Clark 1985, Garud and Kumaraswamy 1995, Sanchez and Mahoney 1996).

However, the increasing digitization of products[i] affords extended windows of redesign that challenge the idea of stable sub-assemblies in managing technological change. First, products become programmable as they embed digital technology and design of new functionality can therefore be integrated at any time, even after production (Kallinikos et al. 2013,Yoo et al. 2010, Zittrain 2006). Second, economics of scale in production is less relevant as anything that is encoded into a digital format can be reproduced instantly without virtually any marginal cost (Benkler 2006, Shapiro and Varian 1999). As products become increasingly digitized, firms therefore need to consider new software-based approaches to introduce, or improve, functionality in their products that parallel the traditional practice of replacing manufactured parts (Tiwana et al. 2010).

As we will argue in this paper, Hora's decomposition into hierarchies of physical components, which we refer to as the hierarchy-of-parts frame, represents only one form of architectural thinking. A complementary option devotes attention to designers' problem-solving practices to exploit the speed of change made possible by digital technology. The dominant hierarchy-of-parts frame is inspired by Simon's principle of near decomposability and invites designers to structure products into loosely coupled parts (Baldwin and Clark 2000, Murmann and Frenken 2006, Wilson 1969). The complementary frame, which we refer to as network-of-patterns, is inspired by the adoption of Christopher Alexander's pattern theory in software engineering (Alexander 1979, Alexander 1999, Alexander et al. 1977) and invites firms to structure products into loosely coupled patterns. A pattern "describes a problem that occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without doing it the same way twice" (Alexander et al. 1977, p. x). Because patterns are generic solutions to recurring problems that stay at an arm's length distance to specific product implementations, their use in architectural thinking support reuse of ideas and functionality re-design. Hence, patterns resonate well with properties of digital technology, they make design relevant over the entire product lifecycle, and, they facilitate scale innovation rather than scale economics.

Our paper makes a number of contributions. First, we highlight the properties that increase the speed by which digitized products can be redesigned. Second, we contribute two architectural frames for thinking about, representing and eventually producing an industrial product, and we argue that product-developing firms can leverage the complementarity between the frames to help effectively manage technological change in the digital age. We demonstrate the value of the two frames through an empirical analysis of an automaker's attempts to introduce a new architecture for infotainment system design in response to technological change. Lastly, we outline a future research agenda that reconsiders the work of Simon and Alexander as well as their followers to address key challenges in innovating digitized products.

2. MODULARITY AND TECHNOLOGICAL CHANGE

Management cognition is challenged when product-developing firms face technological change (Kaplan and Tripsas 2008, Tripsas and Gavetti 2000). Based on experience, managers develop mental representations of markets, products, and technologies that are often imperfect (Kaplan and Tripsas 2008, March and Olson 1976, Weick 1979). Over time, these representations become taken-for-granted and expressed in a dominant design (Anderson and Tushman 1990), often embedded in technical architectures, standards, and day-to-day routines, that influence a product-developing firm's ability to recognize the threats and opportunities created by new technology.

As the digitization of products presents new opportunities for innovation in long-established industries such as automotive, telecom, media, consumer electronics, and publishing (Hanseth and Lyytinen 2010, Henfridsson and Bygstad 2013, Lee and Berente 2012, Lucas and Goh 2009, Selander et al. 2013, Tilson et al. 2010), it challenges us to rethink architectures (Yoo et al. 2010) and the extent to which established approaches represent barriers to unleash digital opportunities (cf. Tripsas and Gavetti 2000). Product architecture cannot be separated from industrial dynamics and economic organization (Langlois 2007, Yoo et al. 2010), which suggests that examining its cognitive basis and assumptions can be valuable for understanding the challenges that product-developing firms face in the digital era.

Modularity has almost a canonical status as a template for managing technological change in the product architecture literature. We therefore conceive of modularity as an important frame for thinking about and representing a product's architecture in product-developing firms[ii]. Modularity serves as a technological frame-of-reference (Davidson 2002, Kaplan and Tripsas 2008, Orlikowski and Gash 1994) that fosters a "built-up repertoire of tacit knowledge" (Gioia 1986, p. 56) for managing technological change when the key source of value creation is economics of scale. We refer to this dominating architectural frame as the hierarchy-of-parts frame[iii] and in the following we detail its benefits in responding to technological change as well as its limits in leveraging the opportunities afforded by digital technology.

2.1 The Hierarchy-of-Parts Frame

We define the hierarchy-of-parts frame as a schema that views design processes as acts of decomposition and aggregation to achieve architectures that preserve and enhance a hierarchy of loosely coupled parts. Table 1 summarizes the characteristics that distinguish this frame.

<Table 1 about here>

Simon (1962) observed that a complex system is "one made up of a large number of parts that interact in a non-simple way" (p. 468) and suggested that decomposing systems into parts reduces complexity and increases design flexibility (cf. Garud et al. 2003, Henderson and Clark 1990, Murmann and Frenken 2006, Schilling 2000; Simon 2002). *Decomposition* increases design flexibility and scalability by dividing a product into a hierarchy of parts which can serve as a blueprint for materializing its components. The parts can later be assembled into a ready-made product through *aggregation*. The goal of decomposition is to increase independence among *parts*. Such independence relies on interfaces intended to resolve potential conflicts among a complex product's interacting parts (Sanchez 1995, Ulrich 1995).

Following Baldwin and Clark (2000), *interfaces* are predefined procedures for exchanging necessary information about the functional interactions between components. Interfaces encapsulate, through information hiding, a component's implementation details from other components (Parnas 1972, Schilling 2000). As a result, interfaces can be designed to reduce the implications of changes in individual components for the overall design and at the same time enable concurrent design of individual components (Baldwin and Clark 2000, Murmann and Frenken 2006).

The hierarchy-of-parts frame assumes a part-whole view of a product, where each part is associated with the whole through many-to-one relationships in design *hierarchies* (Clark 1985). A component on one level is part of precisely one component on the level above and consists of several components on the level below. As an example, a navigation system might be decomposed into user interface, GPS receiver, data storage, and a computing platform. In turn, user interface might be further decomposed into display, speakers, and controls. Each component normally performs one primary function and a design therefore reflects a hierarchy of physical parts as well as a hierarchy of functions.

2.2 Hierarchy-of-parts as Response to Technological Change

As a template for managing technological change in product innovation (Garud et al. 2003), the benefits of the hierarchy-of-parts frame can be located to two distinct episodes[iv] in the product life cycle: design and production. In the design episode where “a complete description of the structural elements of a particular artifact” (Baldwin and Clark 2000, p.42) is created, modularity provides design flexibility. We refer to design flexibility as the degree to which a firm is unconstrained by previous design decisions in making new ones. Compared to an integral architecture that “includes a complex (no one-to-one) mapping from functional elements to physical components and/or coupled interfaces between components” (Ulrich 1995, p. 422), a modular architecture facilitates redesign of the product by implementing one-to-one mapping between functions and physical components (Brusoni et al. 2001, Ulrich 1995). Although just nearly decomposable in practice (Simon 2002), the product-developing firm can then substitute parts of the design to improve performance and increase fit with customer needs (Garud and Kumaraswamy 1995). Splitting the modular system into a set of components that remain a stable platform, and another set that allows variation, this adaptability can be reinforced (Baldwin and

Woodard 2009). Firms in capital-intensive industries such as commercial vehicles and white goods often develop clusters of components, so-called product platforms, from which derivate products are generated to achieve customer differentiation (Muffatto and Roveda 2000, Robertson and Ulrich 1998).

In the production episode, modularity enables scale economics, i.e., it enables firms to decrease the unit costs as the volume of materials increases (Chandler 1990). Product decomposition presents significant opportunity to accomplish scale advantages on the component level by drawing on external manufacturing capabilities (Langlois 2007). Specialized suppliers are in a much better position than the product-developing firm to offer low unit prices as their fixed costs can be distributed across a higher volume (Sturgeon 2002). Even though market segmentation matters for integration decisions (Argyres and Bigelow 2010), decomposition of products provides the basis for vertical disintegration (Christensen 2006) and allows firms to benefit "from the external capabilities of the entire economy" (Langlois 2003, p. 375). As a result, the 1990s witnessed the emergence of modular clusters (Baldwin and Clark 2000) and modular production networks (Sturgeon 2002) that Garud and Kumaraswamy (1995) succinctly refer to as an economics of substitution. Economics of substitution increases the speed by which the product can be changed, although design decisions will be constrained by the availability of new components among suppliers.

As noted above, successful combination of design flexibility and scale economics requires that the design and production episodes be kept apart. Since mass production needs to be nearly algorithmic with well-synchronized stages for assembling the product and enabling high-speed throughput (Chandler 1977), this puts high demands on the specificity of the design. It leaves little room for ambiguity about the structure of the product and underlines why the design needs to be successively frozen each time it is released for production (Baldwin and Clark 2000, Iansiti 1995). While this separation of the design and production episodes is essential for reaping the benefits from external economics of scale, it squarely brackets the time period during which the design is open to redesign in response to emergent customer needs. Applying platform thinking (Baldwin and Woodard 2009, Cusumano and Gawer 2002, Gawer

2009), as in the case of Apple’s iOS (Ghazawneh and Henfridsson 2013), the window can be kept open over an extended amount of time for particular parts of the product.

3. New Opportunities in the Digital Age

Some research recognize the unique opportunities afforded by digital technology by distinguishing between a hierarchy of control and a hierarchy of inclusion as two structures of complex products (Murmann and Frenken 2006, Wilson 1969). Since unforeseen interactions between components may not be accommodated through “simple” modularity, scholars of system integration (Brusoni et al. 2001, Hobday 1998, Prencipe 2000) consider complex forms of modularity with hierarchies of control to enable coordination of operations across multiple distributed components. The system integration perspective recognizes digital technology’s integration capabilities and provides a useful lens for understanding its capacity to process complex real-time information for control purposes. Working in this tradition, Lee and Berente (2012) recognize how the integration of digital controls in products such as automotive emission control systems involve a generative aspect, where their re-programmability tends to spawn new applications that were unforeseen. In what follows, we pick up on this observation by considering more closely how digital technology introduces new and powerful forms of design flexibility and scalability.

Design Flexibility. Digital technology supports design flexibility across the product lifecycle. While the hierarchy-of-parts frame positions functionality attribution as an activity that precedes production of the physical product (Baldwin and Clark 2000, Ulrich 1995), digital technology radically relaxes the requirement of successive freezing of the product design. Tangible components get functionality instantiated at the time of production, but digital components may modify subsidiary functionality, add supplemental functionality, or introduce entirely new functionality over the product lifecycle. This form of unbounded design flexibility can be traced to the programmability of digitized artifacts and enables more timely responses to a changing environment.

As Yoo et al. (2010) point out, computers use a processing unit for executing digitally encoded instructions, and a storage unit for holding these instructions and the data manipulated by them. Traced to Von Neumann and Turing, the stored-program concept has worked as the basis for the modern computer since the late 40s (Langlois 2002). The power of the stored-program concept is that it separates functional logic from the physical hardware that executes it. This means that the same physical artifact can perform new functions if equipped with new instructions or programs. As tangible artifacts embed digital technology, they therefore become increasingly programmable, enabling artifacts to perform new functions after their production (Kallinikos et al. 2013, Yoo et al. 2010, Zittrain 2006).

The degree of functionality change through reprogramming may vary, largely determined by the governance framework used by its proprietor. At one end of the continuum, in a control system with specific functions to monitor or regulate the behavior of another system (cf. Lee and Berente, 2012), reprogramming may amount to changing subsidiary functions. For instance, programmable power-train control units (PTCU) in a car allow modification of design parameters such as fuel mixture, ignition timing, and idle speed of a combustion engine. New parameter configurations give the engine new behaviors by attributing new subsidiary functions to the engine while its primary function remains untouched. At the other end of the continuum, an open-ended system such as a smart phone may allow entirely new software to be installed, making it a multi-functional artifact supporting calls, picture-taking, gaming, reading, movie-watching, and so on. Its users essentially determine what should be regarded as the artifact's primary functions (Yoo 2010).

Design Scalability: Digital technology also supports the scaling of new designs. While the hierarchy-of-parts frame seeks to achieve external scale economics (Sturgeon 2002), digital technology is characterized by swift reproducibility, making it inexpensive to scale a new design. We refer to design scalability as the degree to which a firm can efficiently translate new design ideas into volume products.

As underlined by economists of information, an information good such as software or contents is associated with high fixed costs but negligible marginal costs (Shapiro and Varian 1999). This is because anything that is digitized, i.e., encoded into a digital format, can be reproduced without virtually any marginal cost (Benkler 2006). In contrast, tangible artifacts are typically dependent on significant investment in production resources to avoid diminishing returns (Arthur 1996). Mass production plants secure that the cost per unit drops when increasing the volume of processed materials (Chandler 1990).

Comparing software and tangible products from the perspective of the product lifecycle, the distinction between design and production is largely meaningless in the former case, while it is fundamental in the latter. In fact, the reproducibility feature of software implies the design becomes the product, that is, once a detailed design is in place, there is virtually no time lag before the product can be marketed, sold, and distributed to users. Instead of large investments in production technology, all that is required for realizing the software design are software tools such as compilers and operating systems (Yoo et al. 2010). This makes it comparably inexpensive to scale ideas and it enables democratization in innovation (von Hippel 2005, Baldwin and von Hippel 2011).

In an idealized case, reproducibility is endless. In practice, however, the degree of reproducibility varies. At one end of the continuum, we find simple updates of software and contents, which can be distributed to all clients in a network of usage. For instance, the modern car increasingly relies on software updates, or exchanges, to deal with software errors and avoid expensive return calls. Today, this is typically done at authorized workshops, but the future promises such updates online (de Boer et al. 2005). At the other end of the continuum, entirely new functionality can be added to digitized products by distributing new applications. For instance, many leading automakers, most notably GM and Ford, have announced making so-called “appstores” available, from which car owners can download new automaker and third-party developed functionality to their vehicle.

In summary then, the hierarchy-of-parts frame supports design flexibility and scale economics as powerful responses to technological change. However, exploitation of these benefits requires the design of a product to be frozen before production and therefore only allows limited windows of functionality design and redesign. The consequential time lag in innovation makes it challenging for firms to take advantage of two important properties of digital technology that increase the speed by which digitized products can be redesigned: digital technology affords (a) unbounded design flexibility across the product lifecycle, and (b) design scalability at virtually no marginal cost. As a result, the increased digitization challenges the dominant position of modularity and the hierarchy-of-parts frame in design and management of industrial products.

4. PATTERNS AND TECHNOLOGICAL CHANGE

Next, we draw on the adoption of Alexander's pattern theory in software engineering research to suggest a complementary architectural frame for managing technological change. While a significant body of technology and innovation management research points to the similarities in Simonian and Alexandrian thinking (Baldwin 2008, Langlois 2006, Murmann and Frenken 2006, Schilling 2000, Ulrich and Eppinger 2003, von Hippel 1990), the increasing digitization of products (Kallinikos et al. 2013; Lee and Berente 2012, Lindgren et al. 2008, Yoo 2010) calls for consideration of their differences. To substantiate this claim, it is necessary to turn the attention to Alexander's thinking on design patterns (Alexander 1979, 1999, Alexander et al. 1977), something that is rarely done within technology and innovation management research. Such focused attention can take heart from the fact that Alexander's design pattern theory already has had profound impact on the software engineering community[v] without devaluing Simon's legacy (Alexander 1999, Gabriel 1996, Gamma et al. 1995).

4.1 Alexander's Patterns Theory

At the heart of Alexander's theory is the idea that "the actual substance of which the environment is made consists of *patterns* rather than *things*" (Grabow 1983, p. 11). A pattern describes the properties of a generic solution to a recurring problem (Alexander 1979, Gabriel 1996, Mehaffy 2007). It allows reuse of ideas without concern for implementation details. In this regard, a well-formulated pattern serves as a resource for a product-developing firm confronted with a problem. It affords a general solution yet offers the possibility of specializing the solution to the unique conditions of the current problem setting.

For instance, to position (i.e., determine the geographical location of) an object is a recurring problem in many settings. Different techniques exist (such as celestial navigation, dead reckoning, and satellite-positioning), each serving as a pattern that provides a high-level description of how to locate an object. Consider satellite positioning as an example. It can be viewed as a pattern offering a solution for how to determine the position of an object on earth by measuring its distance to at least four different satellites in orbit. This pattern can be reused in many different settings, where it is engaged together with other patterns to help design well-adapted location-based products and services.

Pattern theory's focus on recombination (cf. Arthur 2009, Schumpeter 1934) is shared with modularity and the hierarchy-of-parts frame. However, rather than recombining parts based on substitution (Garud and Kumaraswamy 1995), pattern theory assumes recombination of problem-solving procedures based on abstractions (Gamma et al. 1995). As Alexander (1979, p.84) observe, "it is very puzzling to realize that the 'elements', which seem like elementary building blocks, *keep varying*, and are different every time that they occur." Patterns abstract from this variation and focuses on the general properties of a solution shared across many settings. Hence, the utility of patterns depends on the product-developing firm's ability to establish generic solution properties based on prior experience from different settings within a particular design domain. Generic solution properties can be traced to "the implicit, language-like system of rules that determines their structure" (Grabow 1983, p. 45) and a pattern language[vi] explicates such rules within a particular design domain that a designer, or a group of designers, can follow (Alexander 1979). In designing new solutions to particular settings, product-developing firms may combine and adapt existing patterns into new sub-patterns that reflect the particulars of the considered problem. Pattern theory facilitates thinking about abstract and generic solutions to recurring problems and how such solutions can be specialized in indefinite ways to solve specific problems.

4.2 The Network-of-Patterns Frame

The network-of-patterns frame views design processes as acts of generalization and specialization to achieve an architecture that preserves and enhances a network of loosely coupled solution patterns. The strengths of the network structure are that it increases reusability of patterns and affords multiple relations between patterns. Table 2 summarizes the characteristics that distinguish this frame.

<Table 2 about here>

The frame directs attention to generalization and subsequent specialization as a means for managing technological change (Mathiassen et al. 2000). We refer to *generalization* as a cognitive process that increases design flexibility and scalability by deriving generic concepts, or patterns, that abstract from irrelevant information related to its implementation. To accomplish functional fit with specific settings (Alexander 1964), patterns are contextualized through *specialization*, i.e., the process of creating a pattern by combining and adapting existing patterns to create a new solution for a recurring problem. When using patterns to support a new solution (i.e., specializing), the solution (i.e., specialized pattern) inherits important properties from generic patterns in addition to its own unique properties. Hence, *inheritance* is the process by which a specialized pattern receives some or all of the properties from another pattern (Shalloway and Trott 2005). For instance, a smartphone-enabled application for locating tourist attractions in a city inherits properties from the positioning pattern among other patterns, but adds specific properties valuable for navigating the city as a place. This approach allows the designer of the tourist attraction application to focus entirely on the particular features important for city tourists without having to worry about positioning.

Rather than focusing on hierarchies of parts, the network-of-patterns frame focuses on *networks* of patterns. As epitomized in Alexander's (1966) article "A City is not a Tree", seemingly unrecognized in the technology and innovation literature, patterns on lower and higher levels in the Alexandrian frame are related in many-to-many relationships in network architectures (Alexander et al. 1977, Mehaffy 2007). Rather than deriving a hierarchic structure of parts defining how a hybrid engine can substitute a standard combustion engine, the Alexandrian frame draws attention to how a general solution for hybridization of powertrains can be applied and reused to solve specific problems in different contexts. Sticking to this example, the network-of-patterns frame captures the shared functional roots of a sport utility vehicle (such as Lexus RS) and a luxury sedan (such as Lexus GS), applying the same concept for hybridization. In other words, the structural representation in the Alexandrian frame is a network in which patterns can inherit properties from multiple patterns on the level above, and patterns on higher levels support description of several patterns on the level below.

Patterns and pattern languages work as generic resources that product-developing firms can enact to manage technological change. However, to benefit from these resources, they must translate patterns into actual products. Accordingly, designers can create (many) specific *instances* of any particular pattern as part of the digital and physical components that constitute a given product. Although instantiation, i.e., the process that generates specific digital and physical components from a pattern, of digital components (say software for positioning) requires the pattern to be expressed in executable code, such patterns may be instantiated again and again without virtually any marginal cost (Benkler 2006). In contrast, instantiation of physical components (say sensors for measuring car speed) requires investments in production resources and involves economics of scale consideration (Chandler 1990).

4.3 Network-of-Patterns as Response to Technological Change

Although modifications and supplements (such as adopting the notion of platform) to the hierarchy-of-parts frame can increase its resilience to digitized products, we argue that complementing it with the network-of-patterns frame makes it considerably easier for managers to take advantage of the new forms of design flexibility and scalability induced by digital technology. Innovations in general-purpose programming languages have over the past decades allowed software designers to increasingly focus on details of the problem at hand rather than on the idiosyncrasies of the supporting computer technology (Bergin and Gibson 1996). Although this development has allowed software designers to become increasingly productive, patterns represent complementary opportunities by offering domain specific, high-level language constructs that represent solutions within a particular problem setting.

By having access to a positioning pattern, a designer of car navigation technology achieves two advantages. First, the pattern enables reuse of software code in the design of new specific applications, thereby curbing the cost of repetitive design. Second, since the positioning code is used across many different mobile services, investments can be made in ensuring that the code performs according to specification, thereby increasing the reliability of the application. Hence, taking a network-of-patterns frame perspective, product-developing firms can achieve flexibility by developing languages of patterns specified by many designers (inside and outside the firm) within a particular area, thereby propelling the space of possible digital solutions readily available at any point in time. The resulting pattern languages introduce new opportunities for each individual designer. Subsequent instantiations of specialized patterns serve as a critical way to quickly respond to emerging and varying customer needs in volatile environments. In this way, pattern languages leverage the unbounded flexibility of digital technology by offering a schema for maintaining functional fit between a product and its use environment and by encouraging product-developing firms to develop architectures that support "living structures" (Alexander 1999, Zittrain 2006).

Moreover, since the marginal cost of reproducing software is close to zero, there are fewer incentives for product-developing firms to treat digital components as standardized parts. Hence, taking a network-of-patterns frame perspective, product-developing firms can leverage design scalability by sharing pattern languages with many designers, even outside their organizational boundaries. Such a move affords the opportunity to specializing and instantiating new applications or services for particular groups of users (cf. von Hippel 2005). This, in turn, accommodates variations in needs amongst the product-developing firm's customers. For instance, Apple's flourishing developer program for the iPhone and iPad has stimulated many third-party application developers, leading in their customer segment, to port, or reproduce, their solution patterns for Apple's devices (cf. Boudreau, 2012, Eaton et al. 2011).

While Apple maintains control over which instantiated patterns (applications) end up on its digitized product, the company wishes their customers to have access to a wide variety of functionality. Accordingly, they offer open access to their platform's development environment (pattern language) (Ghazawneh and Henfridsson 2013). This approach leverages digital technology's scalability of ideas by increasing the value proposition of Apple's products for consumers as well as leading developers within the industry. As such, it provides a powerful approach to the management of technological change.

5. METHODS

5.1 Case Context and Selection

To assess the proposition that the network-of-patterns frame adds value as a complement to the hierarchy-of-parts frame, we draw on a longitudinal study of an automaker's management of technological change in the area of infotainment[vii] systems (Svahn 2012). The single case study design can be motivated by the need of rich and detailed data about the architectural thinking of a firm that develops and manufactures digitized products, something that can difficult to obtain if focusing on multiple cases.

The automaker is CarCorp, a small, international automaker with a devoted customer base in Europe and the US. In 2007, when this study started, CarCorp sold around 125,000 cars across five product lines and employed approximately 4,300 people. For the purpose of this paper, we zoomed in on a service-oriented architecture for infotainment system design called Media-Oriented Systems Transport (MOST). A key element of MOST is the function block framework, essentially serving as a language for conceptualizing and describing functionality independently from hardware. This language affords automakers the opportunity to develop harmonized infotainment solutions that share basic resources and functions.

Infotainment is a suitable empirical context for this research as it represents a product area in the automotive industry that is (a) going through radical digital transformation (Henfridsson and Yoo 2013),

(b) challenged by non-automotive competitors operating at a higher clockspeed of change (Fine 1998), and (c) facing architectural challenges to integrate data from various, heterogeneous source systems (Yoo et al. 2010). The specific case of the MOST architecture is particularly interesting because it carries rich evidence of the transition to complement hierarchy-of-patterns as the traditional architectural frame within the automotive industry with pattern-oriented architectural thinking.

5.2 Data Collection

The presented data was collected between 2007 and 2009. As summarized in Table 3, it comprised multiple data sources, including interviews, participant observation, and archival data. We organized and recorded this data in a single research database using the Atlas.ti software for qualitative analysis. First, we conducted 31 semi-structured interviews (μ=70 minutes, σ= 27 minutes) with a total of 23 respondents. All interviews were tape-recorded and transcribed, producing 36 hours of recorded material amounting to approximately 342,000 words. Our respondents came primarily from CarCorp (15/23), but also from consultancy organizations engaged in architecture-related projects at CarCorp (8/20). Using a snowball approach (Knoke and Yang 2008), we interviewed CarCorp employees of various rank, including engineers, departmental managers, and directors, and from a variety of functions, including advanced engineering, software and control, and infotainment.

Participant observation was another valuable data source. Over the most intensive period (September 2007 to March 2008) we spent more than 30 days at CarCorp, acting as embedded researchers in CarCorp's different infotainment projects. Over the entire data collection period, we also took part in various meetings, including project meetings, workshops, steering committee meetings, and supplier meetings. All in all, we participated in 47 meetings, summing up to 142 hours (μ = 181 minutes, σ=101 minutes). It was generally problematic to record these meetings, so we took extensive field notes for inclusion in our research database. In addition, we conducted recurring debriefings after interviews and meetings.

Finally, we had largely unlimited access to CarCorp’s internal project documentation. In particular, we made use of different specifications of the automaker’s first MOST-based infotainment system. Apart from being boundary objects for informal discussion with CarCorp practitioners in situ, these specifications provided a solid foundation for reconstructing the details of how architectural framing evolved. We selected 29 specifications related to bus architecture, system architecture, MOST function catalog, function specifications, and function partitioning to be included in the case database.

<Table 3 about here>

5.3 Data Analysis

Our data analysis was driven by the theoretical proposition that the network-of-patterns frame adds value as a complement to the prevailing hierarchy-of-parts frame in explaining technological change. We therefore approached the case findings with the ambition to sensitize the concepts deriving from our conceptual basis. As to avoid imposing our lens without sensitivity to the story that the data tell, however, we initiated our analysis by using open coding to discover concepts and their properties and dimensions in the data material (Charmaz 2006, Strauss and Corbin 1998). The first and third authors conducted the coding process together, which yielded an initial set of approximately two hundred descriptive concepts. To reduce overlap, we reviewed and compared all concepts by eliminating duplicates and merging closely related ones. In the comparison process, we formulated preliminary definitions to capture concept properties and dimensions. This eventually resulted in a list of 128 mutually exclusive concepts.

We then applied key concepts of our conceptual basis for making sense of the coded material. This process was conducted in an iterative fashion where we frequently revisited Alexander’s work and the literature on product architecture and modularity. The MOST-specifications provided strong basis for identifying the use of patterns at CarCorp, but this step in the data analysis also involved constructing the dense texture of relationships between these patterns and the automaker’s challenges to respond to technological change.

6. ARCHITECTURAL FRAMES AT CARCORP

At the turn of the century, CarCorp engineers could look back on a period of exceptional growth of infotainment functionality. Moving beyond a simple radio, rapid digitalization had opened up for in-car phones, navigation, telematics, TV, CD, and rear-seat entertainment. As a result, the complexity had increased dramatically. In particular, the couplings between components skyrocketed as speakers, displays, controls, and various sensors were shared over a range of infotainment functions to (a) support a coherent end-user experience (e.g., coordinated speaker output from navigation and the telephone) and (b) leverage economy of scale (e.g., single speaker system for several functions). Among CarCorp's infotainment engineers, there was an increasing recognition that the encapsulation of software in hardware components was at the heart of their problems. While the use of software afforded suppliers the opportunity to quickly change functionality at the component level, this advantage did not play out at the system level as the specificity of interfaces effectively prevented flexible redeployment of functionality.

In 2000, CarCorp decided to adopt MOST as a response to the experienced problems. CarCorp engineers needed an architecture in which software-enabled functionality was less coupled to particular components. Such decoupling between hardware and software was expected to facilitate change at the system-level and, at the same time, increase the degrees of freedom when decomposing the system. An architect reflected upon the early impressions of the MOST architecture:

> *I think we all realized – at least the people involved in [architecting] infotainment – that this was the future. We needed to focus on the system, solving problems at the system level. We could not remain in the hands of suppliers, making stand-alone components. Instead, we had to make these suppliers part of a larger whole. [...] I think, at the heart of MOST, there is a kind of system level thinking that is not component-oriented. Instead, it centers on the structure of logical elements or functionality. (Senior system architect)*

There was consensus among engineers that this new architectural thinking was a necessary complement to the existing hierarchy-of-parts thinking to cope with the increasing complexity and fast-changing functional requirements of infotainment systems. Although the electronics division had undergone significant growth in the aftermath of the past years' radical digitalization, CarCorp engineers were concerned about the way the firm's manufacturing heritage negatively affected the clockspeed by which they could introduce new functionality. The temporal differences in product design between the automotive industry on one hand, and the consumer electronics, information technology, and telecom industries on the other hand, came to dominate architectural thinking at CarCorp's advanced engineering unit:

> *The providers are pushing new functionality, updates, and so on, into the market space in a rapid pace. The car industry is a bit schizophrenic. We want all these new features in our cars, but our development cycle is so slow. Once we design and implement a new function, it is not attractive on the market anymore. (Infotainment product manager)*

The new competitive landscape made CarCorp engineers and managers reflect upon current innovation practices, especially in terms of what made automakers slow in introducing new functionality. They traced this difficulty to the timing of requirements determination:

> *The problem is that the automotive industry designs and constructs their cars over three or four years. We nail down requirements early on, and then it is immensely difficult to make change requests along the way. On the consumer device side, time-to-market is totally different. If they miss a deadline with six months, then that model is commercially dead. (Senior consultant)*

> *We are struggling with a reality where we cannot update functionality because of our processes. Therefore, there is no other way than telling the customer that this is what we have, nothing more. We can schedule an update for the next generation of systems, but it is a very long process integrating a new component into a car. (Infotainment product manager)*

Early freezing of functionality was a natural ingredient of CarCorp's hierarchy-of-parts thinking, allowing for division of labor and effective sourcing. However, the flip side was significant inflexibility. Once the functional purpose of a component was defined through its interface, it was virtually impossible to modify it. CarCorp engineers generally traced this inflexibility to the product architecture:

> *If we could get away from a hardware solution, we might address the problem of long lead times for introducing new functionality in the car…. The hardware should remain the same over time, while the software modules should enable the adaptation needed. ... We need architectures that can enable new functionality in a flexible way. (Infotainment R&D manager)*

6.1 A Wave of New Architectural Thinking[viii]

MOST paved the way for a wave of new architectural thinking. In particular, it offered a new approach to conceptualizing functional patterns independently from hardware components. Traditionally, CarCorp engineers specified functions in relation to a component where an interface specification defined the relationship between components and a functional requirement. In contrast, the MOST architecture centered on description of available functional patterns without any a priori assumptions on how functions eventually would be deployed in a particular car infotainment system. Documented in the so-called "MOST Function Catalog", this description essentially served as a pattern language. First, the catalog described how patterns were to be instantiated in software, including all the details on how to instantiate a particular function. Second, function-partitioning specifications described how different patterns related to each other and how general functions could be combined to form more specific functionality. In 2004, after ten revisions over a two-year period, CarCorp's function catalog consisted of 280 general patterns (see Table 4 for an excerpt), frequently reused in more specific patterns.

The most general patterns in the catalog served as templates for orchestration of shared system resources (e.g., see Table 4 pattern IDs 0x001-0x111). They made the ground rules for how to make use of speakers, displays, button, and many other critical assets, referred to as "sinks" and "sources". Other patterns – resolving positioning (0xC54), phone book search (0xD42), voice recognition feedback (0xE87), or vehicle speed (0xC46) – had more specific functionality. They could, in turn, be inherited by other patterns to make up complete solutions for end-user functions within navigation, phone, or telematics. Since the MOST architecture did not enforce any particular deployment strategy, this network of patterns could be reconfigured without affecting the hardware setup. Accordingly, as new functional requirements emerged for a subsystem, MOST offered a common pattern language for developing functionality to specific physical infotainment components.

<Table 4 about here>

The MOST architecture offered generalized patterns that largely remained relevant across generations of infotainment products. In other words, MOST supported reuse of patterns as engineering teams worked on different subsystems with similar problems. Reflecting a network-of-patterns frame, the MOST architecture afforded a seemingly unambiguous way to create, structure, and maintain the functional patterns that defined the infotainment system and its various components over time. In this way, system architects turned into platform designers and platform users.

6.2 Tensions between Architectural Frames

Despite initial optimism, CarCorp's adoption of MOST turned out to be rather painful, as the new network-of-patterns frame was introduced into an organization that for decades had practiced architectural thinking shaped by the hierarchy-of-parts frame. Just like the product structures were hierarchical with horizontal independency between components, the organization design was hierarchical with clear division of labor. To control such design hierarchies, CarCorp followed a strictly linear innovation process, reflecting a waterfall model of product development. Accordingly, requirements were gradually broken down alongside the design hierarchy. Business objectives, general system topics, and overall functional properties were managed by CarCorp, while the design of components and detailed functionality was assigned to highly autonomous suppliers, operating on lower levels of the hierarchy. As witnessed by a consultant deeply involved as system engineer in developing CarCorp's MOST architecture, this traditional hierarchy-of-parts thinking did not change easily:

> *They thought the traditional model would work, where each [supplier] had responsibility for his own function, embedded in his own component. […] Down the road, they saw the flip side. It didn't work since the whole system – end-to-end – was so incredibly distributed. (Senior consultant)*

CarCorp had invested considerable efforts in generalizing patterns, trying to build a powerful infotainment platform where functional patterns were consistently reused by more specific patterns. At the time of deployment, when patterns were allocated to physical components, tensions between the two architectural frames became clear: functions and components did not match anymore. General patterns, inherited by many specific patterns, were instantiated where it made best sense from an economic or complexity perspective. As a result, a given function, such as navigation, was distributed across several different components. This had significant implications for suppliers, contracted to design and produce components, not software. Relying on the existing legal practices at CarCorp, these suppliers were formally made liable to functionality that was distributed across a range of other components, outside their immediate control.

Neither suppliers nor manufacturers were comfortable with this situation. Without dedicated software suppliers, taking full responsibility for component-spanning functions, innovation would most likely slow down. CarCorp saw no other option than bridging the gap between suppliers and themselves by specifying not only interfaces between components but also the general patterns to be widely reused across the system. This transition of responsibility increased the automakers' stakes in functional design dramatically:

> *You are taking a [new] responsibility as a manufacturer, when specifying this stuff. It becomes… I mean, they [suppliers] cannot even do anything! When I think about it, it's not them rejecting responsibility; it's us taking it from them. Yes, that's what it is. We are telling them that "the only thing you're about to do is to support this [our solution]… Earlier, when things were more component-oriented, they had an opinion of their own on things, they had tested it – possibly with other manufacturers – and knew what was good and what was bad. With this approach [MOST] we more or less lost such feedback. (Project manager)*

Clearly, these problems were grounded in an emerging and fundamental mismatch between the existing organizational structures and the MOST's approach to conceptualize software-enabled functionality. Taking the network-of-patterns frame seriously, CarCorp's engineers and managers had to increasingly background the physical hardware. At the same time, they remained organized to match the hardware structure of the system.

6.3 Leveraging Network-of-Patterns Thinking

Knowing that this mismatch could not be easily resolved, the automaker initiated two different measures to smooth the implementation of a MOST-based infotainment solution. First, they found that leveraging network-of-patterns thinking required reorganizing the workforce at a local level. Realizing that hardware components were no longer at the core of architectural thinking, managers started to structure the infotainment group to enable system level control and general pattern development:

> *Originally, it was a component-oriented group. They were expected to work with functional specifications as well. Later on, this didn't work out, so they invited some people working with functions only. They needed more and more such people and, eventually they were a group of their own. Probably 10-12 [persons], maybe even more. Most of them were consultants since it was running so fast, and we wanted it implemented. We underestimated the efforts significantly. (Project manager)*

Rather than obliterating the hierarchical structure, the manufacturer rebalanced the workforce, with old roles essentially remaining the same, but the center of design moved upwards in the waterfall model, from the component level to the system level.

Second, as designers reinforced the network-of-patterns frame, they had to break with the strictly linear model of innovation associated with component-based modularity. The new architectural practices pushed new forms of collaboration and new relationships – some temporary and some more permanent – that were not supported by the traditional hierarchy. Moreover, as CarCorp's design focus shifted, from specific solutions (guiding decomposition) to general solutions (guiding specialization), it was necessary to adopt an iterative approach to innovation. The traditional development processes stated few iterations, each resulting in the production of a pre-series car, but the new way of designing infotainment systems called for endless series of iterations. Although management formally approved the new emphasis on patterns thinking, solutions to these challenges emerged bottom-up from designers' daily need to make progress. When proposed patterns were ambiguous to suppliers, CarCorp designers initiated workshops with relevant stakeholders. When suppliers failed to reuse patterns due to various misconceptions, designers built extensive system-level test environments to identify and solve problems in collaboration with suppliers. When progress was too slow, designers increased iterations dramatically to boost learning from constant interplay between generalizations and specializations. Sometimes, the clockspeed of iterations exceeded one software release per week, a stark contrast to the traditional development process with only a handful of releases for an entire 3-4 year car project.

Struggling to combine the hierarchy-of-parts and the network-of-patterns frames, engineers and managers gradually found a reasonably stable way forward. On one hand, the hierarchy-of-parts frame remained. Specific functional patterns were defined and used to guide decomposition of infotainment systems, eventually sourced to various suppliers following existing principles. On the other hand, designers engaged in generalization and specialization of infotainment systems with suppliers. These iterations were performed in a fluid structure of more or less temporary, cross-organizational design teams. Relations and arenas for collaboration were established and destroyed according to project needs. Together these informal teams and processes made up a network-based structure of innovation, augmented to the formal hierarchy.

To help balance the two ways of organizing, CarCorp tried to locate specific patterns characterized by particularly high pace of change to just a few components. As demonstrated by CarCorp's user interface guidelines, this strategy was expected to give a malleable infotainment system that could be effectively changed, without involving tensions between different suppliers:

> *The infotainment system is a user interactive and user intensive (application) with continuous changes in the user interface, but with core functionality that in some degree is defined as stable. Therefore it is a good idea to split the core functionality from the user interface. (MOST system architecture specification)*

However, splitting the more durable core functionality from specific user interface patterns reinforced the need for a shared approach to generalization. A scenario where navigation, telematics, and media player had different user interface logics, would eventually confuse end-users. Therefore, as described in CarCorp's architectural specification, all designers had to adopt the same strategy when applying generalization to their respective functionality:

> *In many cases there exist design issues that do not map onto a single component, neither physical nor logical. These issues are more general in nature and must be addressed and expressed in form of strategies that must be followed by all designers involved in the design of the infotainment system family. (MOST system architecture specification)*

In retrospect, the MOST architecture helped develop powerful and flexible infotainment systems, but they were technically complex, expensive, and in 2008 they had not yet delivered expected novelty.

However, despite a massive range of teething problems, the MOST architecture survived and became a first-hand choice for many automakers. Over the coming years, standard practices emerged for how to deploy functionality to physical components. These standard practices allowed automakers and suppliers to identify and advance their respective roles in innovation. It also restored the basis for exercising scale production. While this can be viewed as a return to modularity, it can just as well be portrayed as a breakthrough for network-of-patterns thinking in the automotive industry. Today navigation, telematics, and media playback are described, modeled, and manifested as networks of interacting functional patterns rather than hierarchies of nearly decomposable parts. Since the first car was introduced based on the MOST architecture in 2001, the technology has been increasingly accepted as a reliable and profitable solution for in-car infotainment. In 2012, MOST was integrated in over 115 vehicle models, provided by 16 automakers worldwide. This makes MOST a significant basis for the use of pattern languages in the ongoing digitalization of one the most profound industries in the world.

7. DISCUSSION

New technologies arise from the combination of existing technologies (Arthur 2009, Schumpeter 1934). While such combination is powered by forward-looking visions and a desire to accomplish new goals, it is also characterized by its legacy – the genesis of a particular technology largely defines how it can be reused for new purposes. The legacy simply makes some directions of progression "much more compelling of attention than others" and often "advance seems to follow advance in a way that appears almost inevitable" (Nelson and Winter 1982). The legacy largely shapes our thinking about technology and its management (Kaplan and Tripsas 2008, Tripsas and Gavetti 2000). Hence, as product-developing organizations engage in architectural thinking, rigid templates may emerge for how to manage the complexity of innovation in the midst of technological change. In this regard, architectures constitute a link between historical achievements and future potentialities. An architecture is a strategic tool that, properly orchestrated, can be used to gradually reinforce sound ideas in a series of "structure-preserving and structure-enhancing transformations" (Alexander 1999, p.79). In other words, architecture is an instrument for path creation, but, at the same time, a shackle of path dependency. Whether product-developing firms will be able to transform innovation practices and leverage the opportunities of digital technology depend, to a significant extent, on their capability to fertilize new architectural perspectives that resonate with the opportunities afforded by digital technology.

CarCorp adopted network-of-patterns thinking in the appropriation of MOST. The new architecture was founded on the idea of developing a language of functional patterns for how to solve different problems in the context of infotainment. These patterns opened up for functions to be designed independently from the physical realization of the system. Drawing on this capability, designers engaged in generalization of the system, resulting in a whole range of shared general patterns that could be inherited by specific applications. This investment paid off through specialization, when functions such as navigation, telematics, and media playback could be designed by reusing the same general solutions for volume control or positioning.

At the same time, the organization as a whole viewed MOST from a hierarchy-of-parts perspective. Fiber optics offered an exceptionally simple interface. In fact, the same, standardized interface could be applied to all the different parts constituting the system. With such a clean and simple template for how to build the physical structure of an infotainment system, CarCorp saw a great opportunity to reinforce modularity. Therefore, the physical infotainment system continued to be decomposed into a wide range of components, each expected to enable a well-defined piece of functionality. This would not just preserve the existing hierarchy of engineers, managers and suppliers, but would also allow them to aggregate the system aggressively to differentiate the product portfolio and launch a range of new, attractive offers.

Trying to exercise both perspectives, CarCorp uncovered a strong tension between the two architectural frames at the point when patterns were instantiated and deployed to physical parts. While hierarchy-of-parts thinking prescribed functional decomposition to provide nearly independent parts, network-of-patterns thinking had generated numerous shared patterns that, when instantiated, increased coupling dramatically. The clash between architectural frames became obvious, when suppliers realized their components were not functionally independent, but deeply intertwined with other components, outside their control. In some sense, they lost their creative leeway as CarCorp strived for general functionality. As a result, this tension pushed them to adopt a defensive strategy, largely leaving for CarCorp to define how to improve navigation, telematics, and other infotainment functions over time. Validating its new infotainment solution, CarCorp could establish that it resulted in much needed and appreciated harmonization, but little new specific end-user functionality. Network-of-patterns thinking had offered the automaker new opportunities, but to the price of a crashed innovation model. Unless resolving this issue through appropriate mixtures of different approaches to work design and collaboration, CarCorp would not be able to release the potential in a continuous interplay between hierarchy-of-parts and network-of-pattern thinking.

Drawing on these insights, this research contributes to the literature on management of technological innovations. As a first contribution, we have highlighted how the properties of digital technology increase the clockspeed by which digitized products can be redesigned. Within the technology and innovation management literature, these properties have typically been masked by the separation between design and production implicated in the product lifecycle of tangible artifacts (cf. Baldwin and Clark 2000). While programmability allows for design flexibility through functionality change of the digitized product (Yoo et al. 2010, Zittrain 2006) and reproducibility enables design scalability through instant reproduction of software and content (Benkler 2006, Shapiro and Varian 1999), seminal works maintain that functionality is both attributed and fixed to physical components, and manifested in the design specification serving as the basis for production (Baldwin and Clark 2000, Brusoni et al. 2001, Ulrich 1995). There is little doubt that the emergence of digitized products motivates a modification of this view. Our research extends the technology and innovation management literature (Baldwin and Clark 2000, Garud et al. 2003, Schilling 2000, Ulrich 1995) by focusing on the particular challenges and opportunities related to digitized products.

As a second contribution, our research introduces the notion of architectural frames, and proposes two complementary frames for managing technological change in product innovation. Our view on architectural frames draws on traditional notions of modularity from the technology of innovation management literature inspired by Simon’s notion of nearly decomposable systems and on complementary approaches to complexity inspired by the adoption of Alexander’s patterns theory in the software engineering literature. The hierarchy-of-parts frame emphasizes decomposition of products into parts and the network-of-patterns frame focuses on generalization of solution patterns across contexts and implementations (Table 3). Moreover, we posit that both frames are needed since they represent complementary ways to think about and manage digitized product architectures: the hierarchy-of-parts frame supports the economics of scale (Chandler 1990, Langlois 2007, Sturgeon 2002) needed for the physical elements of the product, while the network-of-patterns frame helps building architectures that leverage the design flexibility and scalability of its digital elements.

The technology and innovation management literature offers modularity as an effective template to manage technological change when production follows design and consumption follows production. However, when these stages of the product lifecycle intermingle due to digitization, this frame is insufficient in its value creation legacy in economics of scale. For instance, when Baldwin and Clark (2000, p. 49) state that an artifact "*only* comes into existence when the design is converted into the real artifact," they assume that a product is a bounded physical thing and the design is a blueprint for something to be produced in almost an algorithmic fashion to maximize throughput (Chandler 1977). Since production is so important, the window for functionality redesign therefore becomes narrow, usually limited to when a new product line is established rather than on a continuous basis. Thinking about product components as patterns instead of parts provides a cognitive frame for unleashing properties of digital technology, which is independent of production in its traditional sense. Patterns redirect attention from aggregation and interfaces (which connect mass-produced parts) to specialization and inheritance (which supports the creation of new functionality). Inheritance makes use of existing solution patterns and offers the possibility to specialize the solution (oftentimes together with other solutions) to the unique conditions of the product-developing firm's current problem setting.

The proposed architectural frames perspective also contributes to the system integration and complex products literature by providing a perspective that goes beyond viewing digital technology as controls (Brusoni et al. 2001, Hobday 1998, Lee and Berente 2012, Murmann and Frenken 2006, Prencipe 2000). The network-of-patterns frame may indeed be used to understand recurring problems of coordinating across multiple distributed components such as those found in emission control systems (Lee and Berente 2012). Such analyses would use generalization-specialization thinking to develop generic control capabilities that would be valuable across functional areas in a product-developing organization, especially as "digital controllers are being increasingly *decoupled* from the subassemblies that they control" (Lee and Berente 2012, p. 3). Such generic control patterns may then be reused through specialization to local control contexts in a traditional hierarchical structure. However, as far as it is true that "digital controls have a generative aspect to them" and that "their inclusion in a complex system tends to stimulate additional, often unforeseen, digital applications" (Lee and Berente 2012, p. 3), the unbounded aspect of digital technology may lead to open-ended systems. In these cases, the digital technology is not necessarily destined to control an inclusionary hierarchy (Murmann and Frenken 2006, Wilson 1969), but offers the possibility of multiplicity where product boundaries are spanned and meanings are created (cf. Verganti 2009, Yoo et al. 2010). For such contexts, our conception of the relationships between design patterns as many-to-many provides a powerful avenue for analyzing product design and redesign efforts in product-developing firms. In other words, in situations where novel, digitally-enabled services are created on the basis of a product-developing firms' combination of inbound and outbound generic capabilities, the network-of-patterns frame complements the dual hierarchy framing of digital technology found in the system integration and complex products literature (Lee and Berente 2012, Murmann and Frenken 2006) by accommodating the multiple inheritances found in such settings.

The findings of our study suggest possible directions for how managers of product-developing firms may organize product design and redesign in the digital era. Our research suggests that managers should view product architectures as representations that help responding to technological change over time. Such a view on architecture allows for maintaining multiple views on architecture to better cater for the increasing demands on continual innovation. Since the hierarchy-of-parts frame is reasonably well-established in product-developing firms, efforts should primarily be geared towards developing patterns that capture the essence of solutions to recurring problems. Such patterns typically inhibit capabilities of the product-developing firm and its network of collaborators. Confronted with new types of customer demands, firms can then combine existing patterns through specialization thereby reusing well-tested ideas. This type of innovation is greatly facilitated when patterns are implemented as software affording design scalability through swift reproduction and unbounded design flexibility through reprogramming to create the variation needed for particular situations. In short, the network-of-patterns frame provides a focus on product ideas and ways to establish and maintain a viable product offer over time.

Future studies could address several limitations in our work. First, our study suggests how digitization of tangible products motivates a dual view on architecture. Yet, our study does not sufficiently address how architectural frames impact our understanding of the relationship between organization design and product design. As an area of considerable attention in the literature (Sanchez and Mahoney 1996), this relationship, referred to as the “fundamental isomorphism” (Baldwin and Clark 2000) or the “mirroring hypothesis” (Baldwin 2008), has been examined with particular attention to alignments and misalignments (Sosa et al. 2004). As showed by Lee and Berente (2012), this relationship is challenged by digital technology. However, it remains unclear how recognizing complementary architectural frames would alter our understanding of this key relationship. The organization design cannot simply mirror the modular design of the product, since there are multiple frames to consider. It would therefore be useful to investigate how the adoption of Alexandrian thinking would influence organization design as it operates in parallel with the hierarchy-of-parts frame.

Second, our case story does not fully reflect the organizational tensions and struggle involved in pursuing a path that breaks with existing practices and institutional arrangements. In view of recent literature (e.g., Benner 2007, 2010, Lucas and Goh 2009), one would think that firms in long-established industries respond to digital technology by slowly incorporating it within the dominant design. Through such a process, they may come to see programmability and reproducibility as new, but similar features of their products, rather than exploring the unique challenges and opportunities digital technology represents for design and production. In other words, a more comprehensive derivative from our model of architectural frames would usefully embrace the structurational and sociomaterial aspects of architecting digitized products in settings with a long tradition of manufacturing tangible artifacts.

Third, recent studies suggest that digitization can provide the basis for collaboration among firms in previously unrelated industries (Yoo et al. 2010). Our analysis implies that such collaboration is greatly facilitated when firms give the network-of-patterns frame a complementary role in deploying a digitized product. Such a balanced architectural take on managing technological change can be realized only through a cognitive shift, wherein the manufacturing heritage of incumbent firms is reoriented by embracing generalization-specialization thinking. Noting that similar mental shifts occur in the design of complex products ranging from consumer devices such as cameras to logistics technology such as ship cranes (Jonsson et al. 2008, Tripsas 2009, Yoo 2010), further research is needed to more closely trace the dynamics of this cognitive change.

Finally, our detailing of the nature of the two architectural frames suffers from lack of detailed examination of evolutionary aspects. Further research is needed to more precisely elaborate the evolutionary aspects of the two frames and their interaction. Technology management researchers have successfully examined the evolutionary aspects of modularity (Baldwin and Clark 2000, Murmann and Frenken 2006, Schilling 2000), but more research is needed to sufficiently address such characteristics in the context of digitized products and the interplay between architectural frames.

8. CONCLUSION

Concurring with leading technology and innovation management research (Baldwin 2008; Langlois 2006; Murmann and Frenken 2006; Schilling 2000; Ulrich and Eppinger 2003; von Hippel 1990), we have introduced the notion of architectural frames to address limitations to manage technological change in the digital age. Our research sets an agenda for future research that reconsiders the work of Simon and Alexander as well as their followers to address key challenges in innovating digitized products. The extant technology and innovation management literature (Baldwin 2008, Langlois 2006, Murmann and Frenken 2006, von Hippel 1990) typically argues that modularity theory is grounded in the works of Simon and Alexander, viewing the two approaches to technological change as essentially the same. For instance, Murmann and Frenken (2006, p. 931) state: “we are certainly not the first ones to argue that complex technological systems are hierarchically organized. The idea has its roots in design theory going back to Simon (1962) and Alexander (1964)". Similarly, Baldwin (2008) argues that "modularity theory is rooted in the design theories of Simon (1962, 1969) and Alexander (1964)." Although this line of argument makes sense when considering how both Simon and Alexander discuss decomposing systems with specified dependencies between units, our research challenges us to reconsider these interpretations. Drawing on insights from software engineering research, we introduce a different take on Alexander's legacy by articulating how his later work on patterns (Alexander 1979, 1999, Alexander et al. 1977) paves the way for a different approach to manage technological change. It is also important to note, that the dominant notion of modularity relies on a specific and arguably quite narrow interpretation of Simon's work that does not fully leverage his notion of design as a bounded rational problem-solving process.

It is the particular characteristics of digitized products that challenge us to reconsider the work of Simon and Alexander, focusing not only on their similarities but also their differences. Digitized products are characterized by distinct trajectories of physical components and digital components, meaning that the components of such products cannot be understood as a unified entity that follows the same clockspeed regardless of their properties. Given that digital components are increasingly decoupled from the components that they support (Langlois 2002, Lee and Berente 2012, Yoo et al. 2010), architectural frames offer a lens with which to meaningfully study the temporal differences of these nearly decoupled systems and understand how product-developing firms should manage and take advantage of increasing clockspeed. The suggested complementarity of architectural frames enables simultaneous study of the specificity required to transition to production and the flexibility required to leverage the generative capability of digital technology throughout a product's lifecycle. In this regard, we have drawn on insights from information systems and software engineering research to challenges the conventional wisdom in technology and innovation management research and to set a new agenda for studying digitized products as they embed technologies from fast-moving industries.

REFERENCES

Alexander, C. 1964. *Notes on the Synthesis of Form*. Harvard University Press, Cambridge, MA.

Alexander, C. 1966. A City is not a Tree. *Design*.

Alexander, C. 1979. *The Timeless Way of Building*. Oxford University Press, New York.

Alexander, C. 1999. The Origins of Pattern Theory: The Future of the Theory, and the Generation of a Living World. *IEEE Software* **16**(5) 71-82.

Alexander, C., S. Ishikawa, M. Silverstein, M. Jacobson, I. Fiksdahl King, S. Angel. 1977. A *Pattern Language: Towns, Buildings, Construction*. Oxford University Press, New York.

Anderson, P., M. L. Tushman 1990. Technological Discontinuities and Dominant Designs: A Cyclical Model of Technological Change. *Administrative Science Quarterly* **34**(4), 604-633.

Argyres, N., L. Bigelow. 2010. Innovation, Modularity, and Vertical Deintegration: Evidence from the Early U.S. Auto Industry. *Organization Science* **21**(4) 842-853.

Arthur, W.B. 1996. Increasing Returns and the New World of Business. *Harvard Business Review* (July/August) 100-109.

Arthur, W.B. 2009. *The Nature of Technology: What it is and how it evolves*. Free Press, New York.

Baldwin, C.Y. 2008. "Where Do Transactions Come From? Modularity, Transactions, and the Boundaries of Firms," *Industrial and Corporate Change* **17**(1), pp 155-195.

Baldwin, C.Y., K.B. Clark. 2000. *Design Rules - The Power of Modularity*. MIT Press, Cambridge, MA.

Baldwin, C.Y., E. von Hippel. 2011. Modeling a Paradigm Shift: From Produce Innovation to User and Open Collaborative Innovation. *Organization Science* **22**(6) 1399-1417.

Baldwin, C. Y., and Woodard, C. J. 2009. "The Architecture of Platforms: A Unified View," in *Platforms, Markets and Innovation,* A. Gawer (ed.), Edward Elgar: Cheltenham, UK.

Benkler, Y. 2006. *The Wealth of Networks: How Social Production Transforms Markets and Freedom*. Yale University Press, New Haven and London.

Benner, M. 2007. The Incumbent Discount: Stock Market Categories and Response to Radical Technological Change. *Academy of Management Review* **32**(3) 703-720.

Benner, M. 2010. Securities Analysts and Incumbent Response to Radical Technological Change: Evidence from Digital Photography and Internet Telephony. *Organization Science* **21**(1) 42-62.

Bergin, T.J., R.G. Gibson, eds. 1996. *History of Programming Languages - II*. ACM Press and Addison-Wesley.

Boudreau, K.J. 2012. Let a Thousand Flowers Bloom? An Early Look at Large Numbers of Software "Apps" Developers and Patterns of Innovation. *Organization Science* **23**(5) 1409-1427.

Brusoni, S., A. Prencipe, K. Pavitt. 2001. Knowledge Specialization, Organizational Coupling, and the Boundaries of the Firm: Why Do Firms Know More than They Make? *Administrative Science Quarterly* **46** 597-621.

Chandler, A.D. 1977. *The Visible Hand: The Managerial Revolution in American Business.* Cambridge, MA and London, England: The Belknap Press of Harvard University Press.

Chandler, A.D. 1990. *Scale and Scope: The Dynamics of Industrial Capitalism*. The Belknap Press of Harvard University Press Cambridge, MA and London, England.

Charmaz, K. 2006. *Constructing Grounded Theory: A Practical Guide Through Qualitative Analysis*. SAGE Publications, Thousands Oaks, CA.

Christensen, J.F. 2006. "Whither Core Competency for the Large Corporation in an Open Innovation World?," in: *Open Innovation: Researching a New Paradigm*, H. Chesbrough, W. Vanhaverbeke and J. West (eds.). Oxford: Oxford University Press, pp. 35-61.

Clark, K.B. 1985. "The Interaction of Design Hierarchies and Market Concepts in Technological Evolution," *Research Policy* **14**, pp 235-251.

Cusumano, M., and Gawer, A. 2002. "The Elements of Platform Leadership," *Sloan Management Review* (43:3), pp 51-58.

Davidson, E. 2002. Technology Frames and Framing: A Socio-Cognitive Investigation of Requirements Determination. MIS Quarterly **26**(4) 329-358.

de Boer, G., P. Engel, W. Praefcke. 2005. Generic Remote Software Update for Vehicle ECUs Using a Telematics Device as a Gateway *9th International Forum on Advanced Microsystems for Automotive Applications*, Berlin, Germany, 371-380.

Eaton, B., S. Elaluf-Calderwood, C. Sørensen, Y. Yoo. 2011. Structures of Control and Generativity in Digital Ecosystem Service Innovation: The Cases of the Apple and Google Mobile App Stores. *Working paper series of the Department of Management, London School of Economics and Political Science* (183).

Fine, C.H. 1998. *Clockspeed: Winning Industry Control in the Age of Temporary Advantage*. Reading, MA: Perseus Books.

Gabriel, R.P. 1996. *Patterns of Software: Tales from the Software Community*. Oxford University Press, New York.

Gamma, E., R. Helm, R. Johnson, J. Vlissides. 1995. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, Reading, MA.

Grabow, S. 1983. *Christopher Alexander: The Search for a New Paradigm in Architecture*. Oriel Press, Stocksfield, MA.

Garud, R., and Kumaraswamy, A. 1995. "Technological and Organizational Designs for Realizing Economics of Substitution," *Strategic Management Journal* **16**, pp 93-109.

Garud, R., A. Kumaraswamy, R.N. Langlois, eds. 2003. *Managing in the Modular Age: Architectures, Networks, and Organizations*. Blackwell Publishing, Malden, MA.

Gawer, A. (ed.) 2009. *Platforms, Markets, and Innovation*. Edward Elgar.

Ghazawneh, A., and Henfridsson, O. 2013. "Balancing Platform Control and External Contribution in Third-Party Development: The Boundary Resources Model," *Information Systems Journal* (23:2), pp 173-192.

Gioia, D.A. 1986. Symbols, scripts, and sensemaking Creating meaning in the organizational experience. H.P. Sims, D.A. Gioia, eds. *The Thinking Organization: Dynamics of Organizational Social Cognition*. Jossey-Bass, San Fransisco, CA, 49-74.

Hanseth, O., and Lyytinen, K. 2010. "Design Theory for Dynamic Complexity in Information Infrastructures: The Case of Building Internet," *Journal of Information Technology* (25), pp 1-19.

Henderson, R. M., and Clark, K. B. 1990. "Architectural Innovation: The Reconfiguration of Existing Product Technologies and the Failure of Established Firms," *Administrative Science Quarterly* (35), pp 9-30.

Henfridsson, O., and Bygstad, B. 2013. "The Generative Mechanisms of Digital Infrastructure Evolution," *MIS Quarterly,* **37**(3), pp. 907-931.

Henfridsson, O., and Yoo, Y. 2013. "The Liminality of Trajectory Shifts in Institutional Entrepreneurship," *Organization Science*, forthcoming

Hobday, M. 1998. Product Complexity, Innovation, and Industrial Organization. *Research Policy* **26** 689-710.

Iansiti, M. 1995. "Technology Integration: Managing Technological Evolution in a Complex Environment," *Research Policy* **24**(4), pp 521-542.

Jonsson, K. 2010. *Digitalized Industrial Equipment: An Investigation of Remote Diagnostic Services* (Doctoral Dissertation). Umeå University, Umeå.

Jonsson, K., U.H. Westergren, J. Holmström 2008. Technologies for Value Creation: An Exploration of Remote Diagnostics Systems in the Manufacturing Industry. *Information Systems Journal* **18**(3) 227-245.

Kallinikos, J., Aaltonen, A., and Marton, A. 2013. "The Ambivalent Ontology of Digital Artifacts," *MIS Quarterly* (37:2), pp 357-370.

Kaplan, S., M. Tripsas. 2008. Thinking about Technology: Applying a Cognitive Lens to Technical Change. *Research Policy* **37**(5) 790-805.

Knoke, D., S. Yang. 2008. *Social Network Analysis*. SAGE Publications, Thousand Oaks.

Langlois, R.N. 2002. "Computers and Semiconductors," in: *Technological Innovation and Economic Performance*, B. Steil, D.G. Victor and R.R. Nelson (eds.). Princeton, New Jersey: Princeton University Press, pp. 265-284.

Langlois, R.N. 2003. "The Vanishing Hand: The Changing Dynamics of Industrial Capitalism," *Industrial and Corporate Change* **12**(2), pp 351-385.

Langlois, R.N. 2006. The Secret Life of Mundane Transaction Costs. *Organization Studies* **27**(9) 1389-1410.

Langlois, R.N. 2007. *The Dynamics of Industrial Capitalism: Schumpeter, Chandler, and the New Economy*. London and New York: Routledge.

Langlois, R.N., and Robertson, P.L. 1992. "Networks and Innovation in a Modular System: Lessons from the Microcomputer and Stereo Component Industries," *Research Policy* **21**, pp 297-313.

Lee, J., and Berente, N. 2012. "Digital Innovation and the Division of Innovative Labor: Digital Controls in the Automotive Industry," *Organization Science* (23:5), pp 1428-1447.

Lindgren, R., M. Andersson, O. Henfridsson. 2008. Multi-Contextuality in Boundary-Spanning Practices. *Information Systems Journal* **18**(5) 641-661.

Lopreato, J., L. Alston. 1970. Ideal Types and the Idealization Strategy. *American Sociological Review* **35**(1) 88-96.

Lucas, H.C., J.M. Goh. 2009. Disruptive Technology: How Kodak missed the Digital Photography Revolution. *Journal of Strategic Information Systems* **18** 46-55.

March, J.G., J.P. Olsen. 1976. *Ambiguity and Choice in Organizations*. Universitetsforlaget, Bergen.

Mathiassen, L., A. Munk-Madsen, P.A. Nielsen, J. Stage. 2000. *Object Oriented Analysis and Design*. Marko Publishing, Aalborg, Denmark.

Mehaffy, M.W. 2007. Notes on the Genesis of Wholes: Christopher Alexander and his Continuing Influence. *Urban Design International* **12** 41-49.

Muffatto, M., and Roveda, M. 2000. "Developing Product Platforms: Analysis of the Development Process," *Technovation* (20), pp 617-630.

Murmann, J.P., K. Frenken. 2006. Toward a systematic framework for research on dominant designs, technological innovations, and industrial change. *Research Policy* **35** 925-952.

Nelson, R.R., S.G. Winter. 1982. *An Evolutionary Theory of Economic Change*. Harvard University Press, Cambridge, MA.

Ohlsson, S., and Lehtinen, E. 1997. "Abstraction and the Acquisition of Complex Ideas," *Journal of Educational Research* (**27**:1), pp 37-48.

Orlikowski, W.J., D.C. Gash. 1994. Technological Frames: Making Sense of Information Technology in Organizations. ACM Transactions on Information Systems **12**(2) 174-207.

Parnas, D.L. 1972. On the Criteria to be Used in Decomposing Systems into Modules. *Communications of the ACM* **15**(12) 1053-1058.

Prencipe, A. 2000. Breadth and Depth of Technological Capabilities in CoPS: The Case of the Aircraft Engine Control System. *Research Policy* **29** 895-911.

Robertson, D., and Ulrich, K. 1998. "Planning for Product Platforms," *Sloan Management Review* **39**(4), 19-31.

Sanchez, R. 1995. Strategic Flexibility in Product Competition. *Strategic Management Journal* **16** 135-159.

Sanchez, R., and Mahoney, J.T. 1996. "Modularity, Flexibility, and Knowledge Management in Product and Organization Design," *Strategic Management Journal* **17**(Winter), pp 63-76.

Schilling, M.A. 2000. Towards a General Modular Systems Theory and its Application to Interfirm Product Modularity. *Academy of Management Review* **25**(2) 312-334.

Schumpeter, J.A. 1934. *The Theory of Economic Development*. Harvard University Press, Cambridge, MA.

Selander, L., Henfridsson, O., and Svahn, F. 2013. "Capability Search and Redeem across Digital Ecosystems," *Journal of Information Technology* (28), pp 183-197.

Shalloway, A., J.R. Trott. 2005. *Design Patterns Explained: A New Perspective on Object-Oriented Design*. Addison-Wesley, Boston.

Shapiro, C., H.R. Varian. 1999. *Information Rules - A Strategic Guide to the Network Economy*. Harvard Business School Press, Boston, MA.

Simon, H.A. 1962. The Architecture of Complexity. *Proceedings of the American Philosophical Society* **106** 467-482.

Simon, H.A. 1996. *The Sciences of the Artificial*, 3rd ed. The MIT Press, Cambridge, MA and London, England.

Simon, H.A. 2002. Near Decomposability and the Speed of Evolution. *Industrial and Corporate Change* 11(3) 587-599.

Sosa, M.E., S.D. Eppinger, C.M. Rowles. 2004. The Misalignment of Product Architecture and Organizational Structure in Complex Product Development. *Management Science* **50**(12) 1674-1689.

Strauss, A., J. Corbin. 1998. *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*, 2 ed. SAGE, Thousands Oaks, CA.

Sturgeon, T.J. 2002. "Modular Production Networks: A New American Model of Industrial Organization," *Industrial and Corporate Change* **11**(3), pp 451-496.

Svahn, F. 2012. "Digital Product Innovation: Building Generative Capability through Architectural Frames," Doctoral Thesis, Umeå University, Umeå, Sweden.

Takeishi, A., T. Fujimoto. 2003. Modularization in the Car Industry: Interlinked Hierarchies of Product, Production, and Supplier Systems. A. Prencipe, A. Davies, M. Hobday, eds. *The Business of Systems Integration*, 254-278.

Tilson, D., Lyytinen, K., and Sørensen, C. 2010. "Digital Infrastructures: The Missing IS Research Agenda," *Information Systems Research* (21:4), pp 748-759.

Tiwana, A., Konsynski, B., and Bush, A.A. 2010. "Platform Evolution: Coevolution of Platform Architecture, Governance, and Environmental Dynamics," *Information Systems Research* (21:4), pp 685-687.

Trispas, M. 2009. "Technology, Identitiy, and Inertia through the Lens Of "The Digital Photography Company"," *Organization Science* **20**(2), pp 441-460.

Trispas, M., G. Gavetti. 2000. Capabilities, Cognition, and Inertia: Evidence from Digital Imaging. *Strategic Management Journal* **21** 1147-1161.

Ulrich, K. 1995. "The Role of Product Architecture in the Manufacturing Firm," *Research Policy* **24**(3), pp 419-440.

Ulrich, K.T., and Eppinger, S.D. 2003. *Product Design and Development,* (3 ed.). Boston: McGraw-Hill.

Verganti, R. 2009. *Design-Driven Innovation: Changing the Rules of Competition by Radically Innovating What Things Mean*. Harvard Business Press, Boston, MA.

Von Hippel, E. 1990. Task Partitioning: An Innovation Process Variable. *Research Policy* **19** 407-418.

Von Hippel, E. 2005. *Democratizing Innovation*. The MIT Press, Cambridge, MA/London, England.

Weber, M. 1949. *The Methodology of the Social Sciences*. Free Press, Glencoe, IL.

Weick, K.E. 1979. *The Social Psychology of Organizing*, 2 ed. McGraw-Hill, New York.

Wilson, D. 1969. Forms of Hierarchy: A Selected Bibliography. *General Systems* **14** 3-15.

Yoo, Y. 2010. Computing in Everyday Life: A Call for Research on Experiential Computing. MIS Quarterly **34**(2) 213-231.

Yoo, Y., O. Henfridsson, K. Lyytinen. 2010. The New Organizing Logic of Digital Innovation: An Agenda for Information Systems Research. *Information Systems Research* **21** (4), pp. 724-735.

Zittrain, J.L. 2006. "The Generative Internet," *Harvard Law Review* **119**, pp 1974-2040.

Table 1. The Hierarchy-of-Parts Frame

Dimension	**Characteristic**
Approach to Complexity	Decomposition-aggregation
Notion of Component	Part
Components Relation	Interface
Structural Representation	Hierarchy

Table 2. The Network-of-Patterns Frame

Dimension	**Characteristic**
Approach to Complexity	Generalization-specialization
Notion of Component	Pattern
Components Relation	Inheritance
Structural Representation	Network

Table 3. Data Collection

Data Source	**Total**	**Details**
Interviews	31	*Interview statistics:* -∑: 36:17:55 h -μ : 1:10:15 h -σ : 27:22 m - word count: 341,997 - 23 respondents
Participant observation	31 full days & 47 meetings	*Meeting statistics:* -∑: 142 h -μ : 3:01:17 h -σ : 1:41:03 h
Archival data	29 specifications	*Specification topics:* - bus architecture (1) - system architecture (1) - MOST function catalog (2) - function specification (2) - function partitioning (21)

Table 4. Examples of Patterns in the MOST Architecture

ID	Pattern	Domain	Description from CarCorp's MOST Function Catalog
0x001	NOTIFICATION	System	This property administrates the Notification Matrix of a function block
0x116	SYNCDATAINFO	System	This property can be used to query the function block on how many connections it may serve as sink or source
0x101	ALLOCATE	System	With this method, the source is caused to occupy synchronous channels.
0x102	DEALLOCATE	System	The method DeAllocate causes the source to free occupied synchronous channels
0x100	SOURCEINFO	System	This property gives particulars about the type of synchronous source data
0xC60	SOURCEACTIVITY	System	This property signals whether a source is active or not (On/Off).
0x112	DISCONNECT	System	By use of this method, synchronous channels for audio reception will be disconnected
0x110	SINKINFO	System	The property SinkInfo can be used to query the sink about the type of data it can handle
0x114	SINKNAME	System	By using property SinkName, a name for the synchronous data can be requested.
0x111	CONNECT	System	By use of this method, synchronous channels for audio reception will be connected.
0xC54	GPS_RAWDATA	Positioning	This property is used to transmit GPS Raw Data. All parameters will be transmitted at the same time as: Longitude, latitude, Fix, HDOP, VDOP, Speed, Heading, Height, Year, Month, Day, Hour, Minute, Second.
0xC56	GPS_SATELLITEIDS	Positioning	Satellite IDs (according to NMEA protocol) used for the position calculation
0xC55	GYRO	Positioning	Function returns the yaw rate as calculated from the vehicle's gyro.
0xD40	PHONELISTS	Phone book	This method administrates the PhoneList.
0xD41	ARRAYINSERT	Phone book	This method is used to insert an arbitrary number of PhoneBook entries into the PhoneBook.
0xD42	ARRAYSEARCH	Phone book	This method is used to search the PhoneBook(s) with a search string.
0xD43	ARRAYDELETE	Phone book	This method is used to delete entries in the PhoneBook(s).
0xD44	ARRAYREAD	Phone book	This method is used to read "Quantity" number of entries with a certain "WindowType"and at a certain "PosX" in the PhoneBook.
0xD45	SUBPHONEBOOKS	Phone book	This property is used to read important information about different SubPhoneBooks.
0xD46	PHONEBOOKSTATUS	Phone book	Status of the phone book.
0xE80	VRSTATUS	Voice recognition	This property is used to set and signal the status of the voice recognition engine.
0xE84	PTTPRESSED	Voice recognition	This property is used to send button presses from the Speech Manager.
0xE85	VOICEFEEDBACKSWITCH	Voice recognition	This property is used to toggle voice feedback On/Off.
0xE86	VOICEDISABLED	Voice recognition	This property is used to signal if the voice recognizer has been disabled (On) by OnStar or enabled (Off)
0xE87	FEEDBACKPROMPT	Voice recognition	This property is used to request voice feedback messages from the Voice Recognizer.
0xC42	ALARMSTATUS	Car status	Shows the status of the alarm system.
0xC45	SYSTEMPOWERMODE	Car status	Function is used for ignition key power modes.
0xC46	VEHICLESPEED	Car status	Function returns the speed of the vehicle in km/h.
0xC47	TILTSENSORSTATUS	Car status	This function shows the tilt sensor status
0xC48	VIN_2_9	Car status	This property is used for the Vehicle Identification Number (VIN). Eight ASCII Characters.
0xC4A	INTRUSIONSENSINGSTATUS	Car status	This function shows the intrusion sensing status.
0xC4B	EXTERIORLIGHTSON	Car status	This function shows the exterior lights status
0xCA2	AIRBAGDEPLOYED	Car status	This function shows the status of the airbags.
0xCB2	VIN_10_17	Car status	This property is used for the Vehicle Identification Number (VIN). Eight ASCII Characters.

---

[i] We use the term "digitized products" as a short-name for digitized tangible products. We refer to digitized products as assemblages of digital and physical components that are commonly recognized as an end to a customer need. Examples of digitized products are everyday consumer products such as cars and cameras, but also an entire of range industrial equipment including ship cranes and underground mining vehicles (Jonsson 2010).

[ii] It should be emphasized that Simon (1996, p.215) noted that "how complex or simple a structure is depends critically upon the way in which we describe it". In this regard, Simon underlined that complexity is not an invariant aspect of technology but is primarily a matter of identifying and enacting appropriate representations.

[iii] We recognize that the hierarchy-of-parts frame presented here represents a general frame from which different variants emerge in the contingency of everyday practices. Our objective is to present an ideal type "formed by the one-sided accentuation of one or more points of view" (Weber 1949, p. 90). In this regard, the frame represents conceptual constructs that may not appear in reality in its purest form, but represent a manifestation of theorizing through idealization (Lopreato and Alston 1970, Ohlsson and Lethinen 1997).

[iv] In addition to modularization in product architecture (design) and modularization in production, Takeishi and Fujimoto (2003) also highlight modularization in inter-firm systems as an additional "facet" of modularization.

[v] It is somewhat ironic that the network-of-patterns frame has had more impact in software engineering than in architecture (Alexander 1999, Gabriel 1999, Mehaffy 2007). The software engineering community's interest in Alexander's work has boomed since Gamma et al. (1995) outlined a language consisting of 23 patterns for recurring problems in object-oriented software design.

[vi] Alexander et al. (1977) exemplifies a comprehensive pattern language consisting of 253 patterns for how to address city planning, building, and construction problems. The description of each pattern follows the same syntax, including the problem, core of the solution, archetypical example, context of the pattern, empirical background of the pattern, and evidence of its validity.

[vii] The term infotainment refers to media providing a combination of information and entertainment. In the automotive industry it includes navigation, telematics, rear-seat entertainment, and similar systems.

[viii] MOST also addressed the physical layer of infotainment architecture. It offered a fiber-optical bus network, providing bandwidth far beyond hitherto established solutions. This network interconnected the different components through a generic, non-functional interface in a ring topology. In such a ring topology, components are not nested to hide complexity. Instead, all components are found at the same level, regardless of potential functionality dependences. Seen as a layer in a higher-level hierarchy – e.g. a car – such a system is flat, having a wide span (Simon 1962) at that level. This allowed engineers to mount components just about anywhere in a car, as long as it was possible to connect a tiny fiber-optical wire.