FOUR PERSPECTIVES ON ARCHITECTURAL STRATEGY

Research-in-Progress

C. Jason Woodard
Singapore Management University
School of Information Systems
80 Stamford Road, Singapore 178902
jwoodard@smu.edu.sg

Joel West
Keck Graduate Institute
535 Watson Drive
Claremont, CA 91711
joel_west@kgi.edu

Abstract

A recurring theme in the literature on technology and organizations is the concept of mirroring, which posits a duality between technological and organizational design decisions. In this paper we highlight a second, orthogonal duality between components and interfaces: designers of both products and organizations must decide what information to hide within component boundaries and what to expose to other designers. Although the component–interface duality appears in many settings, it presents especially vexing strategic challenges in the design and production of complex digital artifacts. We present a typology of four interlinked perspectives on these kinds of strategic design problems, and discuss the tensions that can arise between them. We conjecture that the ability to resolve these tensions may be a significant and underappreciated source of competitive advantage, and suggest future empirical research that could use this typology to develop new ways of thinking about architectural strategy in IT-intensive industries.

Keywords: Product architecture, organizational design, strategic alignment, digital goods, IT strategy
Introduction

Research on technology and organizations has long recognized the importance of product architecture, and many scholars have explored its relationship to the organizational structure of the product development process. A recurring theme of this literature is the concept of mirroring, which posits a duality between technological and organizational design decisions. Here we highlight a second, orthogonal duality between components and interfaces: designers of both products and organizations must decide what information to hide within component boundaries and what to expose to other designers. Although the component–interface duality appears in many settings, it presents especially vexing strategic challenges in the design and production of complex digital artifacts.

In this paper, we present a typology of four interlinked perspectives on these kinds of strategic design problems (Figure 1), each of which is associated with a distinct body of theory and practice defined by a focal design domain (technological or organizational) and a focal type of design element (components or interfaces). This matrix typology suggests two novel research questions for scholars of organizations, strategy, and information systems:

1) Given that design activities in firms tend to be dispersed among people and groups whose primary perspectives differ (e.g., the people who design product components tend to be different from those who design organizational interfaces such as contracts with suppliers), what tensions can arise?

2) How can firms improve the alignment between these various design activities?

Answers to these questions could suggest new ways of thinking about architecture as a source of strategic advantage, and lead to actionable insights about how to execute sophisticated architectural strategies.

To explore these issues, we review prior research on two paired dimensions. The first dimension links the technological and organizational domains of design through research into topics such as interfirm modularity and the mirroring hypothesis (Colfer and Baldwin 2010; Sanchez and Mahoney 1996; Schilling 2000). The second dimension relates components and interfaces, the basic types of design elements in a technological or organizational architecture (Baldwin and Clark 2000; Ulrich 1995).

We observe that both dimensions exhibit dual relationships in that when one member of the pair is in the foreground, the other is invariably in the background. Thus almost any discussion of a technological architecture implies a corresponding organizational architecture, because most technologies are designed and produced by people working in organizations. Likewise, a discussion of components necessarily involves the interfaces between them, because an interface is essentially the informational boundary of a component. Rather than binary opposites or extremes on a continuum, we contend that these pairs are analogous to particles and waves in quantum physics, where matter displays properties of both, but in a given experiment one aspect may be more prominent than the other.

These two orthogonal dualities identify four distinct architectural perspectives. Each perspective draws attention to a particular set of design problems. For example, the design of technological components focuses on the problem of decomposing a complex product or system into interrelated parts. We believe that there is an underappreciated need to reconcile these perspectives, and that doing so could have important implications for research and practice.

While we argue that alignment between the four perspectives should be a central goal of architectural strategy, it appears that few firms seem to achieve it consistently. We suggest several factors that can give rise to tensions between perspectives, and thus between the various groups of designers and managers who gravitate toward one or two perspectives to the exclusion of the others. We conjecture that the ability to develop and execute a well-aligned architectural strategy may be a significant and sustainable source of competitive advantage, especially in industries that produce IT-enabled products and services.
The Role of Architecture in Product and Organization Design

Research on technological innovation and new product development has highlighted the importance of product architecture in managing the complexity of product development processes, creating successful product families, and stimulating the development of complementary assets. A distinct but related body of literature on organization design has emphasized the need to align the structure of an organization with its external environment, promote loose coupling among organizational units to make them more adaptable, and manage the tensions that arise when firms pursue conflicting strategies simultaneously (e.g., incremental and radical innovation). A third stream of work explores the intersection of the first two, positing a similarity between technological and organizational architectures which has become known as the mirroring hypothesis. This section briefly reviews each of the three streams with an emphasis on their implications for the design and production of complex digital artifacts.

Product Architecture

Ulrich (1995: 419) defines product architecture as “the scheme by which the function of a product is allocated to physical components.” This scheme includes “(1) the arrangement of functional elements; (2) the mapping from functional elements to physical components; (3) the specification of the interfaces among interacting physical components.” Components may be hierarchically nested, from systems and subsystems down to individual parts (Murmann and Frenken 2006; Simon 1962). Components with strong internal interdependencies but relatively weak linkages to the rest of the system are called modules (Baldwin and Clark 2006). The extensive literature on product modularity explores the ways in which modular designs enable decentralized innovation, rapid product evolution, mass customization, and economies of scale and scope (Garud and Kumaraswamy 1995; Langlois and Robertson 1992; Mikkola and Gassman 2003; Ro et al. 2007; Sanchez and Mahoney 1996; Schilling 2000).

Architectural Strategy

Morris and Ferguson (1993) introduced the concept of architectural strategy in an influential article which advanced the proposition that “architecture wins technology wars.” In particular, they argued that in information technology and other complex system industries, “competitive success flows to the company
that manages to establish proprietary architectural control over a broad, fast-moving, competitive space” (1993: 87). The idea that seemingly mundane engineering decisions can have important strategic consequences resonated among industry practitioners, although the article’s specific claims remain surprisingly underexplored in the academic literature. Much of this work centers around the costs and benefits of modularity more generally, with some scholars emphasizing the power of modular architectures to transform industries into dynamic “modular clusters” (Baldwin and Clark 1997), while others warn that modularity can commoditize profitable components and facilitate imitation by competitors (Baldwin and Clark 2000; Christensen and Raynor 2003; Ethiraj et al. 2008).

Implications for Software-based Systems

Product architecture choices take on special strategic significance when the products involved are malleable, in the sense that they can be extended or reconfigured at low cost. Software-based systems can be extremely malleable because their architectures are only minimally constrained by the physical laws that dominate the design of mechanical systems like automobiles and jet engines. Modern software architects enjoy unprecedented freedom to add or change components without adversely affecting the performance of the system as a whole, or even requiring the end user’s direct knowledge or intervention. As the performance cost of malleability continues to decrease due to hardware advances enabled by Moore’s Law, it becomes easier to make architectural changes for reasons that are primarily competitive rather than driven by customer requirements or technological necessity. For example, firms can choose to provide compatibility with rival technologies by emulating the infrastructure they depend on; thus Microsoft Windows can run on an Apple Macintosh and Linux can run on an IBM mainframe. Firms can also try to disable or degrade the performance of a competitor’s product through subtle changes to key components, such as those made by Microsoft to “embrace and extend” Sun’s Java technology. Taking advantage of (or guarding against) these possibilities requires coordination between activities that have traditionally been divided between engineering and management roles.

Organization Design

Not coincidentally, organization scholars have also employed architectural concepts to describe the structure and evolution of human social systems. The idea of treating organizational structure as a design problem was widely explored in the 1960s and 1970s (Galbraith 1973; Mintzberg 1979; Thompson 1967) and continues in the modern literature on organization design (Weick 2004). Garud and Kumaraswamy (1995) draw an explicit parallel between technological and organizational systems:

Technological systems consist of components that together provide utility to users. Similarly, firms that manufacture the components of a technological system together comprise an organizational system for that technology. Relationships between these firms are analogous to interactions between components of the technological system. The mosaic of rules, procedures, and norms that comprise the institutional environment of this organizational system parallels the architecture of a technological system. (Garud and Kumaraswamy 1995: 98)

Organizational Architecture

A key concern of organization design is the division of tasks among organizational units such as firms, departments, and teams. In research on this topic, the term organizational architecture has been used to describe the formal structure of an organization, such as a multi-divisional hierarchy, as well as the linking mechanisms that coordinate interactions between individuals and groups, such as interdepartmental liaisons or matrix reporting relationships (Nadler and Tushman 1997). A recurring theme in this literature is the strategic challenge of creating “ambidextrous” organizations that can simultaneously explore new opportunities and exploit existing capabilities (Raisch et al. 2009; Tushman and O’Reilly 1996).
Levels of Analysis

The study of organizational architecture is complicated by the fact that different levels of analysis present different design issues, which in turn have attracted the attention of different scholarly communities. For example, problems related to interfaces between individuals and teams are well described by the literature on boundary objects (Carlile 2002; Star and Griesemer 1989). Other work examines the design of coordinating structures at the level of a firm and its value network (Brusoni et al. 2001; Christensen et al. 2002; Maula et al. 2006), and even an entire industry (Jacobides et al. 2006). We find it helpful to elide these distinctions by using the terms organizational components and organizational interfaces to refer to elements of organizational architecture at any level of analysis, analogous to the way technological components and interfaces are understood by product and system designers. Just as a product architecture can be specified at various levels of abstraction, an organizational architecture could describe a set of organizational units within a single firm, a supply chain or business ecosystem that spans firms, or even the “architecture of participation” (Baldwin and Clark 2006; West and O’Mahony 2008) that governs the development of an open-source software project.

The Mirroring Hypothesis

Henderson and Clark (1990: 27) suggested an intriguing relationship between product architecture and organization design: “We have assumed that organizations are boundedly rational and, hence, that their knowledge and information-processing structure come to mirror the internal structure of the product they are designing.” This assumption has come to be viewed as both a normative prescription (Sanchez and Mahoney 1996) and a testable proposition known as the “mirroring hypothesis” (Baldwin 2008; Colfer and Baldwin 2010). There are two categories of studies:

In the within-firm group of studies, developers are employees of a single firm, thus by definition share one organizational tie. Accordingly this literature focuses on the patterns of communication and collocation within a product development organization. ... In the across-firm literature, developers are distributed across two or more firms, thus by definition, some lack the organizational tie of firm co-membership. This literature seeks to determine whether the distribution of design work across firms corresponds to the underlying pattern of technical dependencies. Specifically, do firm boundaries coincide with module boundaries in the technical architecture? And are formal transactions located where technical dependencies are sparse (Baldwin 2008)? (Colfer and Baldwin 2010: 11–13)

Empirical evidence on the mirroring hypothesis has been largely supportive (see, e.g., MacCormack et al. 2011), with important exceptions in which organizations “break the mirror.” For example, Brusoni and Prencipe (2006) found that the introduction of robotic manufacturing technology induced Pirelli Tires to shift from a modular organization to an integrated one in order to successfully manufacture a product that was transformed by a new production process from integrated to modular.

Exceptions to the Mirroring Hypothesis

There are two categories of exceptions. Colfer and Baldwin (2010: 17) observe that “contributors with rich organizational ties may develop a modular product consisting of largely independent components.” They have a ready explanation: “[I]n theory nothing prevents a tight-knit team or an individual with adequate knowledge from creating a design made up of largely independent components.” On the other hand, “contributors with highly constrained organizational ties may make interdependent contributions to the design of a single technical system or sub-system.” Building on West and O’Mahony’s (2008) study of participation architecture in open-source communities, they propose that this exception can be explained by the presence of actionable transparency in the design process: contributors can observe each other’s activities, make sense of them, and use this knowledge to act on the design itself.

The Product–Organization Duality

Research on the mirroring hypothesis suggests a dual relationship between products and the organizations that make them: their architectures are linked by the nature of the product development process, such that information about one yields information about (or guidance for designing) the other.
In the hypothetical case of perfect mirroring, there would be a one-to-one correspondence between technological and organizational components, with flows of information among designers exactly following the pattern of interdependencies among components. Even under partial mirroring (the only kind observed in the empirical literature), product architecture can generate important constraints on the structure of product development organizations and vice versa. This is especially true in the context of multi-product systems that span firm boundaries, where typically no single firm controls the design of the entire system. In these settings, product designers must think carefully about the interactions between the parts of the system they control and the parts they do not. These interactions are governed by the interfaces between components, as discussed in the next section.

The Dual Relationship Between Components and Interfaces

Interfaces describe how components interact, “including how they will fit together, connect, and communicate” (Baldwin and Clark 1997: 86). They consist of visible information that can be used by designers to reason about the behavior of components designed by others. In software engineering, it is common to think of an interface as a contract between a component and its clients (Meyer 1992). In organizational systems, interfaces between firms are often literally specified by legal contracts, although Puranam and Jacobides (2006) note that organizational interfaces need not be formally defined, and that interactions between organizational units within a firm are often highly unstructured (cf. Langlois 2002).

Just as an organizational architecture lies behind every product architecture, any partitioning of a technological or organizational artifact into components implies a set of interfaces between them. There is thus a dual relationship between components and interfaces, which is orthogonal to the duality between products and organizations in the sense that it works the same way in both domains. The two domains differ in the nature of their architectural elements (technological vs. organizational), while the two types of elements play diametrically opposing roles in a design hierarchy: component design is fundamentally about information hiding, while interface design is about information sharing.

Although these concerns are two sides of the same coin, they are often decoupled from each other in theory as well as in practice. The remainder of this section elaborates on this observation with the aim of highlighting both opportunities for synthesis and strategic tensions that may arise between component and interface designers. While the opportunities and tensions in the that arise in product design are analogous to those faced by organization designers, they present different managerial challenges (and different questions for strategic management scholars), making it worthwhile to consider them separately. The result is a two-dimensional framework (Figure 1) that identifies four complementary perspectives on architectural strategy.

Components and Decomposition

Ever since the seminal work of Herbert Simon (1962), modular decomposition has been seen as a primary goal of component design in both technological and organizational systems. A nearly decomposable or loosely coupled system (Weick 1976) is characterized by parts that can be designed, operate, and/or evolve independently of each other. Although there are many ways to achieve these properties, Parnas (1972) proposed “information hiding” as a criterion for decomposing systems into modules. The principle that a good modularization should minimize the amount of information that needs to be shared among designers remains central to modern software engineering approaches such as object-oriented programming (Booch 1986), although it is often violated in practice (Parnas 2002). Modular architectures based on information hiding are partitioned into visible information that needs to be widely shared, and hidden information that is encapsulated within components (Baldwin and Clark 1997; Langlois 2002).

Benefits of Information Hiding

Information hiding is a powerful technique because it allows designers of individual components to focus on the design decisions that relate to their own components and ignore those that are delegated to other component designers. This form of “separation of concerns” (Dijkstra 1982) facilitates the division of labor among product development teams — both within and between firms — and simplifies the task of
integrating components into a working system. At the industry level, the separation of concerns among component designers plays out as horizontal and vertical disintegration into networks of competing and cooperating firms (Baldwin and Clark 2000; Langlois and Robertson 1992; Jacobides et al. 2006).

**Platform Architectures**

Recently, attention has focused on a particular form of modular decomposition known as a platform architecture (Evans et al. 2006; Gawer and Cusumano 2002; Meyer and Lehnerd 1997). Platform architectures divide systems into a set of relatively stable core components surrounded by a diverse and rapidly evolving population of peripheral components (often but not always packaged as separate products), enabling economies of scale in the core and economies of scope at the periphery (Baldwin and Woodard 2009). The management of platforms and their surrounding business ecosystems has become a substantial research topic in its own right (Boudreau 2010; Eisenmann 2008; Iansiti and Levien 2004; Iyer et al. 2006; Sanchez 2004).

**Interfaces and Coordination**

Competition in platform ecosystems often hinges on the design and control of interfaces to a far greater extent than the functionality of the underlying components. For example, Apple exercises tight control over the set of application programming interfaces (APIs) available to developers on its iOS platform, rejecting those it deems threatening or simply inconsistent with its design goals.¹

Understanding these issues requires engaging another major body of literature, broadly known as network economics (David and Greenstein 1990; Farrell and Klemperer 2007; Katz and Shapiro 1994; Shy 2001). Most of this research takes the modular structure of a system as given, and focuses instead on how to design its interfaces.

**Compatibility and Openness**

A central concern of the network economics literature has been whether the interfaces between components permit compatibility between products made by different firms, which in turn gives rise to consumption externalities among users who can interact with a greater number of others or choose among a wider variety of complementary products (Katz and Shapiro 1985). Compatibility can arise through ex-ante agreement on an interface standard or the ex-post addition of an adapter, converter, or gateway component that bridges the differences among incompatible interfaces (David and Bunn, 1988; Farrell and Saloner 1985, 1992).

Systems in which key interfaces are shared by multiple vendors are typically labeled “open” while systems characterized by proprietary interfaces are said to be “closed” (Garud and Kumaraswamy 1993; Saloner 1990). This bifurcation is often problematic in practice, and thus many intermediate or hybrid possibilities can also occur (West 2003, 2007). Indeed, standardization itself has come to be viewed as a dynamic and often messy process influenced by (and in turn potentially reshaping) alliances and other interactions among firms (Garud et al. 2002; Rosenkopf et al. 2001).

**Strategic Interface Design**

Here we emphasize that interface design presents strategic choices that are related to but distinct from choices about modularity and component boundaries. The ability to influence the design of key interfaces and control the disclosure of these designs can either encourage or deter both complementors and competitors, with profound effects on the value of a system and its architecture (Farrell and Saloner 1992, Morris and Ferguson 1993). These effects are especially stark in software-based systems, where the fine details of an interface — down to the exact sequence of binary digits needed to invoke a particular function

— can make the difference between compatible and incompatible components. System architects can manipulate interfaces in many ways, such as adding an interface to provide compatibility across existing components (West and Dedrick 2000) and “inverting” hidden components to expose a new interface that encapsulates their common functionality (Baldwin and Clark 2000).

Information Leakage

In some cases, an interface fails to fully encapsulate a component, letting design information “leak out.” Schilling (2000) coined the term synergistic specificity for the degree to which a component is optimized to work with components on the opposite side of an interface. Such optimization may yield better performance or extra functionality because it entails designers taking advantage of information that is supposed to be hidden, but it tends to undermine “mix and match” compatibility for the same reason. The differential ability to optimize may be a source of competitive advantage for firms that supply mutually complementary products, like platforms and applications. During the 1990s, for example, Microsoft was alleged to engage in incomplete disclosure of its operating system programming interface specifications to benefit its own applications (Sheremata 2004).

Industry Architectures

Jacobides, Knudsen and Augier extend the concept of interfaces into the organizational domain, and describe their role in the emergence of industry architectures:

We define interfaces quite broadly, as the technological, institutional, or social artifacts that allow for two or more independent entities to divide labor. Interfaces are both the catalysts and the evidence of co-specialization between players [in an industry]. They can emerge through conscious action or through happenstance; they both reflect and amplify the division of labor among industry participants. ... Such a system of interfaces moderates a set of productive units (firms) whose functions are co-specialized so their interaction is based on a well-defined distribution of roles (division of labor). To the extent that the individual players receive positive feedback, the emergent interfaces and co-specialized players will tend to coalesce, inviting newcomers to define their business in a way that aligns with the emergent architecture. (Jacobides et al. 2006: 1203)

They note that the interfaces that emerge through this process can, in turn, reinforce the competitive advantage of the firms that define them:

Often, as “winners” emerge in some parts of the value chain (because of their idiosyncratic, superior capabilities), potential upstream suppliers or downstream retailers come to co-specialize. Thus, an industry architecture will emerge on the basis of the interfaces defined by firms that initially happen to hold superior capabilities, in terms of technical efficiency (Jacobides and Winter 2005). The stability of such a system increases with positive feedback from current operations and negative feedback from trying to change the architecture (cf. Padgett et al. 2003). This results in one, or, at most, a small number of rival “platforms,” co-specialized “business ecosystems,” with their own sponsors, orchestrators, and keystone members (Gawer and Cusumano 2002; Iansiti and Levien 2004). (Jacobides et al. 2006: 1203–04)

Four Architectural Perspectives and Their Design Problems

Taken as a whole, the literature that bears on architectural strategy spans an enormous range of phenomena and intersects a variety of other literatures across several disciplines, including engineering, economics, and organizational theory. Figure 1 maps this vast intellectual landscape into a two-dimensional framework defined by the design domain (technological or organizational) and design elements (components or interfaces) that are the main focus of attention in a given strategic design problem. The horizontal dimension corresponds to the familiar duality between product and organizational architecture that was discussed above. The vertical dimension represents the widely known but less widely appreciated duality between components and interfaces, as discussed earlier in the paper. The resulting matrix contains four quadrants, each of which identifies a distinct architectural perspective.
We label the perspectives using adjective–noun pairs (e.g., “technological components”), although for ease of exposition we also refer to them using the Roman numerals shown in the figure.

Some design problems can be mapped closely onto a single perspective. For example, the technological component perspective (I) concerns the modular decomposition of a product or system into interrelated parts; this is the main focus of much of the product architecture literature. The technological interface perspective (II) concerns the linkages among these parts, especially the extent to which they afford compatibility between components. Much of the literature on standardization focuses on these issues. Similarly, the organizational component perspective (III) emphasizes the division of labor among teams, firms, and industry sectors; each of these has been studied extensively. The organizational interface perspective (IV) highlights the coordinating mechanisms that link these entities, which often take the form of contracts in the important case of linkages between firms — again, a topic of abundant research.

Other problems in strategic design require multiple perspectives to address effectively. For example, designing a complete system architecture entails defining both the components of the system (I) and the interfaces between them (II). Producing such a system requires the components themselves to be implemented by a specific set of organizational units (III). These units, in turn, may be linked in complex ways that require attention to organizational interfaces (IV) such as boundary objects (Carlile 2002; Star and Griesemer 1989), interdivisional charters (Galunic and Eisenhardt 2001), and networks of strategic relationships across firms (Gulati et al. 2000). In these kinds of settings, the four perspectives are interdependent rather than modularly decomposable. The key design problems cannot be solved by sorting them into four boxes and handing them off to people whose job responsibilities correspond to each box. On the contrary, these problems may need to be examined from several angles, each of which reveals a different set of opportunities and challenges.

**Toward a Useful Theory of Architectural Strategy**

We conjecture that effectively integrating across the four perspectives is hard, and few firms manage to do it consistently well. Tensions can arise between engineers and managers, who tend to focus respectively on technological and organizational design problems. Tensions can also arise between designers in both domains who focus on hidden information (e.g., optimizing the performance of a particular module or process) versus the visible information that enables coordination and compatibility. We propose that these tensions tend to arise in industries that produce complex malleable artifacts like digital goods and services, and that resolving them can yield sustainable competitive advantage.

**Current Status of Research**

We are investigating these and related propositions in ongoing research on architectural strategy in platform ecosystems (see, e.g., Woodard and West 2011).

One of the goals of this effort is to explore the antecedents and consequences of architectural control: the ability to influence the design of other firms’ products and services indirectly by virtue of their dependence on components directly owned or controlled by a firm. Progress on this goal would in turn shed light on questions that have attracted interest among academics and practitioners alike, such as whether and when “open” approaches to standardization, system design, and software development can return sufficient profits to fund long-term innovation.

**Acknowledgements**

An earlier version of this paper was accepted at the DRUID Summer Conference 2009. We are grateful to Carliss Baldwin, Dominie Garcia, Richard Tee, George Westerman, and Youngjin Yoo for thoughtful comments on subsequent drafts, and to participants in the session on Design Choices in Modular Organizations at the Academy of Management 2010 Annual Meeting for additional suggestions and encouragement. Several anonymous reviewers also gave us valuable feedback, and the School of Information Systems at Singapore Management University provided generous financial support.
References


