Abstract

Enterprise architecting is becoming critical for most modern organizations whose competitive strategies are tightly linked to the underlying information technology (IT) infrastructure. This is so because an enterprise architecture takes a holistic view of the business processes and functions and the information technologies supporting them, rather than the more detailed perspectives provided with application-by-application views. Our understanding of effective enterprise architecting activities is still evolving and this practice is replete of challenges. These challenges are further compounded by the fact that organizations are often geographically dispersed. Furthermore, business processes, technology infrastructure components, information and the people involved may be distributed in different geographic configurations, making it very difficult to comprehend. In this article, we make a first attempt at providing a theoretical framework to guide our thinking for practice and research in this area. We build on the foundations of coordination theory and geographically distributed collaboration research.

1. Introduction

As noted in prior research [22], Circular A130 of the U.S. Office of Management and Budget describes enterprise architecture as “the explicit description and documentation of the current and desired relationships among business and management processes and information technology.” Armour and colleagues defined an enterprise architecture as the “blueprint for creating enterprise-wide information systems” [4]. They argued that when an organization’s information management system is conceptualized and designed system by system, the result is an eclectic set of patched-up legacy systems commingled with new systems. This is an inadequate strategy in today’s dynamic environments, which demand flexible and scalable systems that are responsive to rapidly changing business and technological environments. Without an enterprise architecture system managers have no guidance on how to grow their systems to meet business needs and no shared vision on how these systems should evolve.

While having a sound and flexible enterprise architecture is critical to success in information intensive and fast paced competitive environments, the process of developing and maintaining this the enterprise architecture—i.e., enterprise architecting—is very complex. As Kaisler and colleagues have articulated, there are three important areas in which enterprise architecting can encounter critical problems: (1) modeling—when the enterprise architecture is conceptualized and designed; (2) management—ensuring that individual systems are implemented in compliance with the enterprise architecture and that they interoperate properly; and (3) maintenance—ensuring that the enterprise architecture evolves consistently with organizational needs [22]. Furthermore, architecting activities in any of these three areas involve managing multiple and complex interdependencies among organizational goals, work processes, business functions, information needs, information technology (IT) infrastructure and people [4].

We argue that it is precisely because of this complex web of interdependencies that coordination is critical to successful enterprise architecting. Coordination theory defines coordination as “the management of dependencies among task activities” [28] and therefore provides a useful theoretical lens to begin to articulate a research framework to study enterprise architecting. At the same time, we also argue that this theoretical lens is not sufficient because a large portion of the enterprise architecture dependencies in organizations often need to be managed across geographic boundaries, thus presenting unique challenges. Two such boundaries that have a substantial effect on system implementations are distance and time [6]. There is a substantial amount of research in the related fields of virtual teams [32], geographically distributed system development [1] and geographically dispersed communication and coordination [23, 30], which can inform research in geographically distributed enterprise architecting.

In this article we develop an initial theoretical framework of geographically distributed enterprise architecting by integrating three different bodies of research: enterprise architecture, coordination theory and geographically distributed collaboration research. These multiple perspectives provide an excellent opportunity to develop such a framework to guide further studies of enterprise architecting. In the next 3 sections we discuss each of these 3 perspectives in more detail. In section 5 we integrate these perspectives into a unified theoretical framework. In section 6 we provide some concluding remarks.

2. Enterprise Architecture Research

Enterprise architecture provides a blueprint that helps develop a shared vision for the development of enterprise-
wide information systems [4]. This shared vision provides some common ground about how the various organizational stakeholders communicate about information system issues and a frame of reference to make IT investment and implementation decisions. The main goal of the architecting process is to align the IT vision with business goals, objectives, functions and processes. Armour et. al [4] suggest that there are 5 important views of an enterprise architecture: business, work, function, information and infrastructure views. The business, work and function views describe the business processes, where these processes need to take place, and the corresponding functional requirements. The work view also describes who needs to do what and how these people need to communicate and coordinate. The information view describes the various information entities necessary to support these processes and how these entities relate to each other. The infrastructure view describes the underlying IT necessary to support these processes. For the purposes of our theoretical development in this paper, these can be consolidated into four more general components previously suggested for information system development efforts [26]: business processes (i.e., business and function), information, technology (i.e., infrastructure) and people (i.e., work, locations).

But enterprise architecting is more than just developing the architecture. According to Armour et al [5], it involves: (1) establishing a target vision for the architecture and other initial activities (e.g., scoping the project, building the team, etc.); (2) describing the baseline architecture—i.e., the current architecture we want to change—in terms of its business processes, information, technology and people; (3) describing the target architecture—also in terms of business processes, information, technology and people; (4) planning the transition from the baseline architecture to the target architecture; (5) planning the implementation of the architecture; and (6) the actual implementation of individual systems according to the architecture.

Establishing the target vision for the enterprise architecture is important because it helps develop a shared understanding of the architecture at a high level of abstraction among the stakeholders. Developing such a vision is a complex process because it involves various stakeholders with different views of the architecture—e.g., customers, users, company managers, system architects, system developers, etc. A coordinated high level vision of the architecture needs to emerge from these various views. Good communication is of paramount importance in this activity and this communication is likely to be frequent and interactive.

Describing the baseline architecture involves developing an inventory of the current business processes, the information that needs to be maintained to carry out these processes, the current IT infrastructure and applications supporting the processes, and the people and locations involved in the processes. This activity will require a substantial amount of modeling and written documentation, but the communication and gathering of the necessary information is more likely to be structured and planned (compared to the vision development).

The modeling and written documentation necessary to describe the target architecture is similar to the views provided by the baseline architecture, except that the target architecture describes where the enterprise will be after the implementation of the architecture, rather than where it is now. In that sense, the descriptions of baseline and target architectures should contain similar types of models and written documents. However, the process of developing the target architecture is likely to be more complex because it will need to be agreed upon and validated by the stakeholders, which will require a substantial amount of frequent interaction. This validation will need to be done against the target vision discussed earlier.

Planning the transition from a baseline architecture to the target architecture is also a complex undertaking because it involves multiple projects and it is affected by uncertainties and changing conditions in the business and technical environments [3]. Furthermore, transition planning must carefully consider the interdependencies across each system implementation project and among the new systems being implemented and between the new and current systems that will not be updated. For example, one needs to evaluate how the vintage technology and legacy systems that are being retained will interoperate with new systems being deployed. Another important aspect of transition planning is deciding on how and when each old system will be phased out and when each new system will be faced in so that the current business processes are not affected during the transition. The plan needs to describe how each current system will be changed and how new ones will be rolled out. This will depend on which system is being updated, migrated to a new infrastructure platform or totally replaced [3]. In addition, if the enterprise architecture effort is part of a larger business process re-engineering effort, then the architecture transition effort needs to be synchronized to meet the needs of the re-engineering goals and plans. Naturally, all stakeholders need to be involved in this process and communication is likely to be frequent and interactive. In addition, there needs to be a substantial amount of technical and domain knowledge sharing during this process.

The target architecture implementation plan differs from the transition plan in that it contains more specific details about resources and activities necessary for the implementation. Once the transition plan is in place and agreed upon by stakeholders, the implementation plan needs to specify how individual systems will be implemented in compliance with the target architecture. This plan is the master plan for the
general implementation of systems per the target architecture and, as such, it focuses on a general allocation of budgets, personnel, and technical resources for the various projects identified in the transition plan. The communication required during this phase is more structured and will require regular meetings of stakeholders to review these allocations and substantial written documentation. The last activity in the enterprise architecting process derives from the implementation plan and it involves the individual implementation of each specific system. In essence, this is not a single activity, but a number of activities bundled as individual system development projects. These projects proceed in the same fashion as any other system implementation project, but the overall system requirements are subordinate to the target architecture, and the implementation plans are subordinate to the enterprise architecture transition and implementation master plans. Communication and coordination requirements for each project is no different than any other system implementation project, except that all enterprise architecture stakeholders need to participate in this communication even if the specific system being implemented does not affect them directly.

In sum, enterprise architecting requires developing a high level vision of an architecture that is aligned with business goals, describing the baseline architecture, developing the target architecture, planning the transition to and implementation of the target architecture, and the planning of individual information system development in accordance with the architecture. The description of baseline and target architectures require that those involved in the architecting effort develop accurate views of business processes, information, technology and people associated with the architecture. Naturally, this process is fraught with complex dependencies, which need to be managed, thus the importance of incorporating coordination theory into our research framework. We discuss this next.

3. Coordination Theory

4.1 Foundations

Coordination theory defines coordination as “the management of dependencies among task activities” [27]. This definition is very useful because it implies that if task activities can be carried out independently, then coordination is less necessary or not necessary at all for collective performance. Malone and colleagues [28] extended this concept in a more interdisciplinary way by arguing that what matters is how the various activities in a task depend on each other, but that these dependencies are not only among people, but also among processes and resources. For example, the work of a software developer depends on the work of a software architect, thus the need to coordinate their activities. Similarly, two pieces of software code that need to be integrated have a dependency with each other at the interface. Finally, the use of hardware resources may be dependent on the availability of financial resources to acquire them and expert personnel to use them correctly.

To better understand the nature of these dependencies it helps to think in terms of topology of dependencies formulated by Thomson [33], which include, in increasing order of complexity: (1) independence—i.e., task activities can be carried out individually without coordination; (2) pooled dependency—i.e., task activities can still be carried out individually, but depend on and compete for the same pool of resources; (3) sequential dependency—i.e., one task activity depends on another, but not the other way around, thus only the dependent activity needs to be coordinated; and (4) reciprocal dependency or interdependency—i.e., both activities are dependent on each other, thus requiring tight coordination.

One other distinction is important to understanding coordination. The word “coordination” can have a dual interpretation, depending on whether one is referring to the process of coordinating—i.e., managing dependencies—or coordination outcomes—i.e., extent to which an activity was effectively coordinated. For example, when team members communicate they are coordinating—a process. But this doesn’t ensure that the team will be well coordinated in the end. Once their dependent activities have been carried out in a synchronized fashion per project schedules the activity can be said to have been successfully coordinated—an outcome. We now discuss coordination processes and outcomes in more detail.

4.2 Coordination Processes

The process of coordinating is an old problem discussed in the classic organizational literature. March and Simon discussed coordination processes in detail way back in the 50’s [29]. They argued that coordination can be accomplished “mechanistically” or “organically”.

- **Mechanistic coordination**, also referred to as task programming and coordination by plan [29, 33, 34] is useful for task activities that are more routine and certain, which can be more easily programmed. The implementation and use of mechanisms like project schedules, interface specifications, plans, procedure manual, and workflow automation are a few examples of mechanistic coordination processes.

- **Organic coordination**, also referred to as coordination by communication, feedback or mutual adjustment [29, 33, 34] is more effective with less certain and non-routine task activities that cannot be coordinated mechanistically because conditions change often. This communication can take place in a number of ways, including informal and spontaneous or more formally and planned [25, 31], interpersonal or in groups [34], and verbal or non-verbal.
• **Team Knowledge.** More recent research in team cognition suggests a third type of coordination process based on team knowledge [7, 8, 24]. This type of coordination is some times referred to as “implicit coordination” described as “unspoken assumptions of what others in the group are likely to do” [36]. This body of research suggests that when team members share knowledge about each others’ task activities they can better plan their own activities in coordination with the activities of others. Similarly, when team members have knowledge about each other they can anticipate each other’s actions more accurately and locate and access expertise more effectively. Research with software teams has provided evidence that knowledge sharing is critical to coordination success [12, 13, 17]. This body of research is quite extensive, but the main implication for our research is that team knowledge helps members coordinate in two ways directly because members can program their actions more effectively; and indirectly because this knowledge helps them coordinate organically—i.e., members have more common ground to communicate effectively—and mechanistically—i.e., members have more common ground to communicate effectively—and mechanistically—i.e., team members understand how to use plans, schedules, etc.—more effectively.

4.2 Coordination Outcomes

Coordination outcomes can be more easily distinguished from coordination processes because coordination processes are what team members do whereas coordination outcomes are the results of those actions. Coordination outcomes are more evident when lacking, manifesting themselves as coordination problems. There are many types of coordination outcomes (or problems), but research with software teams has identified three main types of coordination outcomes, which are important in IT projects: technical, temporal and process coordination [18].

• **Technical coordination** problems arise when different parts of a system don’t integrate or interoperate well—i.e., technical dependencies have not been managed effectively. For example, if two systems work well individually and are compliant the enterprise architecture but there are severe problems at their interface.

• **Temporal coordination** problems arise when the deliverables of different interdependent parts of the project are not completed on schedule. For example, if the two projects in the example above integrate and interoperate well but one of them was completed way behind schedule it may throw the whole architecting process in disarray and future activities may need to be re-planned.

• **Process coordination** problems arise when there are established processes with embedded dependencies, which are not followed. In the case of enterprise architecting process coordination problems can originate either because some established processes for the architecting process are not followed—e.g., priority conflicts, activities starting out of sequence, escalation issues, etc.—or because the architecture itself is out of synch with the business processes it needs to handle.

In sum, coordination is important for enterprise architecting because of the pervasiveness and complexity of interdependencies involved in the process. The process of coordination of enterprise architecting activities can be mechanistic, organic and/or cognitive, resulting in technical, temporal and process coordination outcomes. However, the conventional wisdom of which coordination processes are more adequate for a particular task activity changes when members and stakeholders are separated by distance and time, impairing their ability to communicate (i.e., coordinate organically) effectively. We now discuss this in more detail.

4. Geographically Distributed Collaboration Research

4.1 Bridging Global Boundaries

Information system development projects are increasingly carried out across global boundaries because of proximity to clients and skilled personnel [9] and offshore outsourcing resources [10]. Given this very strong trend it is not a stretch by any means to extrapolate it to enterprise architecting efforts. At the very least many of the individual project implementations will be carried out across more than one location around the world and in many cases the stakeholders, implementation team, information, business processes and IT infrastructure will be spread out across several locations. These global boundaries represent barriers that those collaborating in the architecting effort need to bridge to get the job done. While there are often various global boundaries represented in large information system projects (e.g., language, cultural, organizational), time and distance are two particular boundaries that are most important in system implementations [6]. Both of these boundaries have a substantial effect on how people communicate. We focus on these two boundaries in this paper.

4.2 Geographic Distance

Much has been written about geographic distance in collaboration. The main effect of geographic distance is that it eliminates or severely impairs the benefits of co-location. It has been known for a long time [2] and it has been confirmed more recently [23] that even small distance separations have a strong negative effect on communication frequency. Distance separation also affects the timeliness [19, 35] and richness [14] of the communication because the media through which communication takes place are less interactive with reduced shared context references [11]. Geographic dispersion also reduces shared understanding
[21] and common ground in communication [30] among collaborators. These arguments about the difficulties of coordinating work across geographic distance are consistent with the research literature on geographically distributed software teams showing that geographic separation causes increased coordination overhead and more substantial delays, partly because it takes longer to initiate contacts and resolve difficult issues [9, 20]. Because most of the effects of geographic distance have to do with communication, distance will have a strong influence on how members coordinate organically in enterprise architecting activities.

4.2 Time Separation

Once the effect of geographic distance is taken into account, time separation affects primarily the synchronicity of communication [16]. Time separation may occur for several reasons (e.g., differences in work schedules, holidays, vacation time, telecommuting, etc.) the primary form of time separation in geographically dispersed teams is due to time zones [15]. In contrast to geographic distance—in which once members are separated the magnitude of the distance makes little difference—the magnitude of time separation makes a big difference in how team members can interact. Larger time zone differences reduce the window of opportunity for synchronous interaction. Furthermore, when collaborators are scattered across multiple time zones, simply figuring out when to meet can become daunting tasks [15]. At the same time, and in contrast to geographic distance, time zone differences can be advantageous because work may be carried out in one site while members in other sites sleep overnight. As with distance, time zone differences affect how collaborators coordinate organically, but in contrast to distance, the ability to program and time interactions will make a big difference in whether time zone differences facilitate or hinder coordination.

5. Integrated Framework

The discussion we have presented above leads us to formulate the research framework depicted in Figure 1. In this section we describe, elaborate and support our framework with a series of propositions. Because, to the best of our knowledge, there are no other research frameworks for geographically distributed enterprise architecting, we propose this framework as preliminary for discussion. Additional work will need to be done to further develop theory to guide research in this field, but we view this as a very important first step. We develop our framework in a backwards fashion, starting with coordination outcomes, followed by enterprise architecting process, then by coordination processes, and ending with task context factors that influence the effectiveness of coordination processes.

5.1 Coordination Outcomes

From the perspective of our framework, the goal of the enterprise architecting process is to develop an enterprise architecture with minimal or no technical, temporal and process coordination problems. Prior research has found that different types of task activities are more prone to some type of coordination problems than others. For example, a prior study found that in large scale software projects, technical personnel are more sensitive to technical coordination problems, whereas managers are more sensitive to temporal and process coordination problems [18]. Consequently, we posit that:

*Proposition 1a: the importance of the difference types of coordination outcomes will vary for each of the six enterprise architecting activities presented in the framework.*

The type of coordination outcomes that will matter the most for each enterprise architecting activity will depend on the type of dependencies that are more critical for that activity. We argue that technical coordination will be more important for implementation activities, whereas process coordination will be more important for conceptual activities. We also argue that temporal coordination is important for all activities, but its importance will be affected by the priority and time pressures of the architecting effort, more than the activities per se. Finally, we argue that temporal coordination is important between activities because each activity needs to start and end according to a schedule to avoid delaying the next activity. For example, temporal coordination is necessary between the activities involved in developing the transition plan and those of developing the implementation plan (due to their sequential dependency) so that the start of the implementation plan’s activities is not delayed. Thus we posit:

*Proposition 1b: technical coordination is most important for implementation activities—i.e., describing the baseline architecture, developing the target architecture, and developing the implementation plan.*

*Proposition 1c: process coordination is most important for conceptual activities involving more stakeholders—i.e., developing the enterprise architecture vision and developing the transition plan.*

*Proposition 1d: temporal coordination is most important between activities with sequential dependencies, so that one activity does not delay subsequent activities.*

*Proposition 1e: beyond proposition 1d, the importance of temporal coordination is dictated by the priority and time pressures of the architecting process.*
Proposition 1f: because each individual system implementation is different, the importance of technical, temporal and process coordination will vary for each project, depending on which type of dependencies are more salient in the respective project.

5.2 Enterprise Architecting Processes

A coordination strategy involves employing a particular mix of coordination processes. A specific coordination outcome can be accomplished with different coordination strategies. For example two software teams may be effectively coordinated, but one of the teams may use little communication and more mechanistic coordination with software tools, while another team may rely on team knowledge and frequent communication. But we argue that an optimal mix of coordination processes will be one that is best suited for the particular nature of the task activities. Generally speaking, more interactive, uncertain and less routine activities will require more organic coordination. Similarly, more structured, certain and more routine activities will require more mechanistic coordination. Finally, activities that require more integration of knowledge from more stakeholders will benefit from more team knowledge. Thus we posit:

Proposition 2a: the optimal mix of coordination processes for different enterprise architecting activities will vary depending on the routine-ness, certainty and equivocality of the activity.

Proposition 2b: the optimal coordination mix for more equivocal tasks—i.e., developing the enterprise architecture vision, developing the target architecture, and developing the transition plan—will be more heavily weighted on organic coordination—i.e., communication.

Proposition 2c: the optimal coordination mix for more defined, certain tasks—i.e., describing the baseline architecture and developing the implementation plan—will be more heavily weighted on mechanistic coordination.

Proposition 2d: the optimal coordination mix for activities that require more integration of domain and technical
knowledge—i.e., developing the enterprise architecture vision, developing the target architecture, and developing the transition plan—will be more heavily weighted on team knowledge.

Proposition 2e: the optimal coordination mix for individual system implementations will depend on the volatility and certainty of requirements of the individual systems.

5.3 Coordination Processes and Geographic Boundaries

The propositions we just articulated are generally applicable to co-located enterprise architecting activities. However, because distance and time boundaries impair communication, we posit that:

Proposition 3a: the optimal mix of coordination processes for any enterprise architecting activity will be affected by the presence of geographic boundaries—i.e., distance and time.

Geographic distance reduces or eliminates many of the benefit of co-presence—e.g., team members don’t know each other that well, interaction is less frequent and less spontaneous, it is more difficult to ascertain who is where and what is going on with the task, etc. Therefore, the reduced effectiveness of organic coordination due to distance separation needs to be offset with increased mechanistic coordination and team knowledge. Thus, we posit:

Proposition 3b: as more geographic locations are involved in the enterprise architecting process, the optimal coordination mix will benefit from less organic coordination and more mechanistic coordination and team knowledge. Team knowledge will help members coordinate implicitly and make their limited communication more effective.

Time separation changes the synchronicity of the team’s communication. As more time zones are represented in the team and as the time zone spanned by the team increases, the window of opportunity for synchronous interaction is reduced and members need to be more careful about structuring and timing their communication to be better synchronized with workflow activities in each location. Team members need to make conscious choices about when and how to communicate—i.e., wait to communicate synchronously with other sites during the overlapping time or communicate asynchronously outside of the overlapping period. The most predictable activities can be programmed so that task requests are sent by site A to site B to be completed while A sleeps. Conversely, if a task request is sent by A while B is sleeping it will introduce delay because nothing will be done until B comes to work. For less predictable activities, time zone differences often introduce delay because the team can’t communicate as frequently and spontaneously. Thus, we posit:

Proposition 3c: as more time zones are represented in a team and as the time zone difference spanned by the team increases, the effectiveness of organic coordination will not be severely affected for activities that require periodic communication that can be timed and programmed to meet workflow needs.

Proposition 3d: as more time zones are represented in a team and as the time zone difference spanned by the team increases, the effectiveness of organic coordination can actually be enhanced if dependencies are sequential and communication for task requests can be timed such that one site advances the work while the other site sleeps.

Proposition 3e: as more time zones are represented in a team and as the time zone difference spanned by the team increases, the effectiveness of organic coordination will be progressively more severely affected for more equivocal and uncertain activities that contain reciprocal interdependencies, which require frequent and spontaneous interaction. The optimal mix of coordination processes for these activities will need to be much more heavily weighted on mechanistic coordination and team knowledge to offset this deficiency.

5.4 Other Context Factors

The propositions formulated above are generally applicable, other things being equal. However, not every enterprise architecting project is the same and the coordination needs of different projects will vary depending on the context of the architecting task. We have discussed two such factors to some extent throughout this paper, but it is important to discuss more specifically how these factors may affect the coordination mix. The first task context factor has to do with the nature of the dependencies in the enterprise architecting task. As dependencies become more complex and tightly coupled, the importance of coordination increases. Thus, we posit:

Proposition 4a: as the enterprise architecting activity dependencies move from independent, to pooled, to sequential, to interdependent, the effect of the coordination processes on the effectiveness of the enterprise architecting process will be amplified—i.e., an interaction effect.

The second factor has to do with the uncertainty, dynamism, complexity, priorities and time pressures of the implementation, all of which will affect the importance of coordinating and the effectiveness of the respective coordination processes. Based on our arguments throughout the paper, we posit:

Proposition 4b: the higher the uncertainty of the enterprise architecting task, dynamism and volatility of the requirements of the task, complexity of the target architecture, and priorities and time pressures, the stronger the effectiveness of organic coordination processes will be, thus making communication more important and
mechanistic coordination less important, which is typical of agile methodologies of system development.

6. Concluding Remarks

We have proposed and formulated a research framework to help us guide studies of geographically distributed enterprise architecting work. Our framework is by no means complete, but we hope that it will spark much needed discussion in this area. This framework has limitations because it has not been validated empirically and because it only incorporates distance and time boundaries. Further research is necessary to expand our framework to include other global boundaries. Despite this limitation, our proposed framework is soundly grounded on well established bodies of research—i.e., coordination theory and geographically distributed collaboration—and practice—i.e., enterprise architecting. This framework is a first step to providing theoretical foundations for studies in this area.

7. References


