The Effect of Time Separation on Coordination Costs in Global Software Teams: A Dyad Model

J. Alberto Espinosa  
American University, Washington DC  
Kogod School of Business  
alberto@american.edu

Erran Carmel  
American University, Washington DC  
Kogod School of Business  
carmel@american.edu

Abstract

Research to date has not addressed the difficulties of coordinating across time zones in global software development. We present a preliminary collaboration model to help us understand the consequences of time separation on coordination costs. The model is for a team composed of dyads and each dyad consists of a task requestor and a task producer who have a sequential workflow dependency. The model is constructed with formulas for: production, coordination, and vulnerability costs for a number of: (1) collaboration modes; (2) time overlap conditions; (3) asynchronous and synchronous communications mechanisms, each of different quality and cost; and (4) production and delay cost rates. We describe the model and evaluate it with regression analysis using randomly generated observations. Our evaluation shows that the model adequately represents time-separated work and that time-separation effects are: (1) different and more complex than distance-separation effects; (2) asymmetric, depending on whether work time overlap between the two actors occurs at the beginning or end of an actor’s day; and (3) dependent on the amount of this overlap.

1. Introduction

Working across geographic distances is not easy. Coordination in these contexts is challenging because of lean communication media, fewer opportunities for spontaneous interaction, less contextual reference, and lack of other benefits of co-location (e.g., presence awareness, workspace awareness), among other things. In spite of these difficulties, team work is increasingly carried out globally for a number of reasons: transport cost for digital products is low; production costs in developing nations are very low; and companies need to access specialized software talent and technical resources [4]. However, the cost-benefit tradeoffs for Global Software Teams (GSTs) are complex and not well understood. This has led researchers to study coordination in GSTs [4, 7, 12] and geographically dispersed teams in general [14, 20].

However, distance and time separation are often correlated [6, 22, 25], making it difficult to tease out the true effects of time zone separation. One important aspect that differentiates distance and time effects has to do with the fact that geographic distance is symmetric (i.e., distance A→B = distance B→A) while time separation is not (i.e., work time overlap occurs at the beginning of the day in one site and at the end of the day in the other site). Consequently, the timing when interactions occur and when tasks are processed may not make much difference in pure distance-separated conditions, but they do make a big difference in time-separated conditions. For example, the concept of “follow-the-sun” or “round-the-clock” software development [4] takes advantage of time zone differences to speed up project work, thus offsetting some of the problems of distance separation. While there are not many successful cases of “follow-the-sun” software work, the idea that team members in other sites (e.g., India) can advance the work while members in one’s site (e.g., Eastern U.S.) are sleeping has a lot of appeal in terms of potential productivity gains, at least in principle. However, because of the asymmetric nature of time-separated work, these teams need to pay close attention to timing issues. In fact, GSTs separated by time zones often adjust their work practices to deal with temporal factors – e.g., they establish overlap work hour windows; use liaison personnel whose work hours are the same as the other site; and program work to deliver output in batches to the other site at the end of the day.

The present study is motivated by our interest in learning more about working arrangements involving different time-zones, which is very typical of today’s global software organizations. Because the research literature on global software development is almost silent on the issue of time separation, we feel that it is important to start from the ground up. In a prior study [5], we formulated a model to represent coordination costs for a single dyad with a workflow dependency,
in which we distinguished distance from time separation. In the present study we extend the model to handle larger teams consisting of multiple dyads collaborating in various time- and distance-separated conditions, using various communication tools.

In this paper we first discuss the theoretical foundations of our study. We then provide a more detailed description of our extended dyad coordination model. Next we use regression analysis of randomly generated observations to evaluate the model and develop insights on the effects of time-separation. We then present a discussion section and concluding remarks. Given the paucity of theoretical models and empirical research in this area, this study represents an important contribution because it helps us understand how dependencies can be better coordinated when team members are separated by time.

2. Theoretical Foundations

Consistent with coordination theory, we define coordination as “the management of dependencies among task activities to achieve a goal” [17, 18]. Malone and colleagues defined coordination theory as “a body of principles about how activities can be coordinated” – i.e., how team members can effectively manage task dependencies. Two important principles derive from this definition. First, if task activities can be carried out independently, then there is no need to coordinate. Conversely, more complex tasks like software development have substantial dependencies among its tasks, thus the need to coordinate them. For example, when many software individuals and teams are working in parallel to build a single software product, different software parts need to interoperate properly and tasks (e.g., coding) need to be completed on schedule to avoid delaying other tasks (e.g., testing). Second, if a task is analyzed with a fine-grained level of detail, such that the dependencies among task activities are well understood, one can begin to identify different coordination mechanisms to manage these dependencies effectively.

Dependencies in a task can be pooled (i.e., two tasks depend on a shared same resource like hardware or tools), sequential (i.e., task A cannot proceed until task B is completed) or reciprocal (i.e., tasks A and B are interdependent) [23]. For example, one team member R, the “task requestor”, may be working on a task (e.g., software coding) and may reach a point in which the work needs to be handed over to another team member P, the “task producer”, who needs to perform a task (e.g., testing), such that R’s work on this task cannot continue until P is finished. This sequential dependency among two members needs to be effectively managed to achieve coordination.

The organizational research literature suggests that team members coordinate the routine aspects of their work via task programming mechanisms (e.g., tools, schedules), but they need to communicate to coordinate more uncertain aspects of the task [19, 23]. The effectiveness of task programming mechanisms is less likely to be affected with separation. For example, a configuration management system helps developers work in parallel on the same code without affecting each other, regardless of time zone or location, except when there are non-routine circumstances that require communication. Coordination itself is costly [18], so we need to better understand how the effectiveness of various coordination mechanisms differ between co-located, distributed and time-separated contexts.

Our focus in this study is on coordination by communication because of a number of reasons. First, software development is a complex task with substantial non-routine work, thus requiring a fair amount of communication to coordinate. Second, communication is an obvious way for team members to generate coordination processes [18]. Third, the frequency and effectiveness of communication can be adversely affected when team members are not in close proximity [1, 14]. Fourth, coordination in time-separated conditions is affected not only by the volume of communication, but also by the timeliness of communications [8, 24]. Finally, empirical evidence suggests that communication mediates the effect of other coordination mechanism on performance [9].

When team members are separated by time zones, their coordination requires that both sites pay close attention to the timing of work activities, which is not necessary when working hours fully overlap. In global team work, teams separated from east to west have less work overlap time than teams separated from north to south. Even small time zone differences can bring surprising difficulties. For example, a study on GST coordination observed that a time-zone difference of one hour between two sites substantially affected the team’s ability to communicate interactively because it reduced their overlapping time by 4 hours – one hour at the beginning of the day, one hour at the end of the day, and 1 hour during each site’s lunch break [10]. If a situation, which requires interaction with members at the other site occurs during the other site’s off-work hours, being unable to pick up the phone and call other members can slow down the team’s progress. As O’Leary and Cummings discussed, different team member distributions across time zones, geographic locations, and number of sites...
can have a substantial effect on how members interact in a team [21].

The choices for a team member in such a situation are to either send a request asynchronously (e.g., e-mail) or wait until work hours overlap next and then make the request synchronously (e.g., phone call). Often, requests are not clear, requiring additional communication to clarify things, further delaying the whole process. When team members are working face-to-face, the clarification may be nearly instantaneous. Even when members are distant, but in same time zones, clarifications can be made very quickly through phone calls, instant messaging, or videoconference. However, when team members are separated by time, the need to clarify things will introduce further delay, unless this happens during work hours overlap.

Such unclear communication exposes the team to “vulnerability costs” because of the need for further communication and possibly rework. In sum, because of the asymmetric nature of time separation, the effect of time separation on GST coordination can be modeled by analyzing timing issues—e.g., time when an interaction occurs, task duration time, amount of overlap in work hours—and evaluating how they affect production costs (i.e., the cost of carrying out tasks), coordination costs (i.e., the cost of communication and delays when coordinating), and vulnerability costs (i.e., the cost of additional communication and possibly rework because of miscommunication). Furthermore, we argue that empirical studies involving time-separated conditions must consider such timing issues to avoid confounds between the effects of geographic distance and those of time separation. In the next section we describe the model in more detail.

3. The Model

Our model is an extension of a prior model [5] and is based on coordination theory [17, 18], in which we treat coordination as the management of dependencies between activities in a task. As we discussed, we focus on the use of communication as a coordination mechanism in a variety of distance and time separated conditions. Since coordination theory does not explicitly address issues of distance and time separation, we incorporate this separation in our analysis by evaluating how the total costs of carrying out a task are influenced by the use of different communication tools in various collaboration modes (i.e., co-located and separated by distance and or time) and by time delays caused by time separation.

Prior studies of coordination in software development have used software development time as the main dependent variable [7, 11]. While shorter task duration may be the important goal in competitive environments with strong time-to-market pressures, there are tradeoffs between task duration and cost. For example, longer task duration may be preferred in offshore outsourcing arrangements if it leads to substantial cost savings. Therefore, our model establishes a relationship between time separation and the total cost of carrying out a software task, rather than task duration.

Our model, illustrated in Appendix A, focuses on a single collaboration act between two actors, a task Requestor (R), who makes a request to a task Producer (P) because of a workflow dependency. For simplicity, we only illustrate the different-time portion of the model in the Appendix. Our model is based on a sequential workflow dependency1 in which R’s work depends on P completing the task and sending a notification to R that the task has been completed [18]. R and P may interact in four possible collaboration modes, depending on whether the dyad is separated by distance and/or by time: face-to-face (F), separated by distance only (D), separated by time only (T) or separated by distance and time (DT) [2]. In the two same-time modes, F and D, the dyad interacts synchronously. In the two different-time modes (T and DT) the dyad interacts asynchronously during non-overlap work hours or synchronously during overlap work hours. A central aspect of time separation in the model is that the work overlap (O) between R and P can take place either at the beginning of R’s work day (Ob) or at the end of R’s work day (Oe)—i.e., it is asymmetric. Work overlap between actors is measured in our model using an overlap index [21] ranging from 0 (no overlap) to 1 (full workday overlap).

The communication costs for two actors include the cost of maintaining a communication link (CI) and the cost of sending a single message (Cm) [16]. The cost of maintaining a face-to-face link and the cost of face-to-face communication are assumed to be negligible for co-located teams, compared to other communication costs. The cost of maintaining an synchronous and asynchronous communication links are (CIs) and (Cla) and the cost of sending a synchronous and an asynchronous message are (Cms) and (Cma) respectively.

A single collaboration act in this context consists of the following: (1) R communicates at time (Rt) a task request to P; (2) P completes the requested task in

---

1 Pooled or reciprocal dependencies can be accommodated in our model as two sequential dependencies.
a time equal to \((T_t)\); and (3) \(P\) acknowledges the completion of the task to \(R\) at time \((A_t)\). The total cost of carrying out a software task in our model is composed of three components: (1) \(P\)’s production costs \((P_c)\) incurred to produce the requested software piece; (2) \(R\)’s coordination costs \((C_c)\) incurred because of communication costs and delays (i.e., waiting until \(P\) completes the task); and (3) vulnerability costs \((V_c)\) incurred because of \(R\)’s additional coordination costs because of the need for rework. This breakdown of total costs into production, coordination and vulnerability costs is similar to the breakdown suggested by Malone in his theoretical modeling of coordination in organizations and markets [16]. This model has been widely used in theoretical and simulation research involving coordination [3, 13, 15].

Our model follows Malone’s model, but we make some adjustments because our model needs to take into account delays resulting from distance separation or time zones differences. First, we specifically model time and distance separation between actors. Second, Malone’s model analyzes different coordination structures for a set of actors, while our model employs only two actors who need to carry out a task with a tightly coupled workflow dependency, who coordinate via communication. Third, Malone’s model assumes that actors employ their production capacities optimally, but we don’t need to make this assumption because there are only two actors in our model. Finally, like in Malone’s work, we model production, coordination and vulnerability costs, but our definition of these costs have been adjusted to better fit the case of two actors collaborating on a software task.

Malone defines production costs as those incurred due to delay in processing a task, but since Malone’s model does not incorporate time delays due to time separation, he only includes the time it takes to carry out the task, which is consistent with our definition of production costs. Malone defines coordination costs based on the cost of maintaining communication links and the cost of sending messages among nodes in the coordination structure, but messages arrive instantly in his model. Our definition of coordination costs is similar, but we also include the time delay introduced due to time separation (e.g., \(R\) may send a task request during \(P\)’s off-work hours). Finally, Malone defines vulnerability costs as those resulting from task failures, which require the task to be reassigned. Because our model involves only one dyad, there is no re-assignment. Instead, failures lead to further communication to clarify a message clarification, the resulting delay, and possibly rework, with a probability \((P_r)\), of a portion of the task \((R_w)\). A message can be unclear, with some probability \((P_u)\), and this probability is different for each collaboration mode – i.e., \(P_u(F) \neq P_u(D) \neq P_u(T) \neq P_u(DT)\), where \(F, D, T\) and \(DT\) represent face-to-face, distributed only, time-separated only, and both distributed and time separated modes. The respective formulas are:

\[
P_c = \lambda C_p T_t
\]

\[
C_c = C_l + 2\lambda C_m + \lambda T_d C_d
\]

\[
V_c = P_u(PrRwPc+Cc),
\]

Where \(\lambda\) is the number of task requests per day made by \(R\); \(C_p\) is \(P\)’s daily cost of production; and \(T_d\) is the delay time, equal to the elapsed time from \(R_t\) until the task is completed and acknowledged back to \(R\) at \(A_t\). A few things are worth noting: (1) \(C_l\) and \(C_m\) will vary depending on whether \(R\) and \(P\) communicate face-to-face, synchronously or asynchronously; (2) \(T_t\) is measured from \(P\)’s perspective because it affects \(P\)’s production costs; (3) \(T_d\) is measured from \(R\)’s perspective because it affects coordination costs due to delay in \(R\)’s work; and (4) \(T_d\) is computed as elapsed time during \(R\)’s work hours only, not counting time during \(R\)’s off-work hours (i.e., \(P\) produces while \(R\) sleeps).

We made the following simplifying assumptions in this model: (1) The probability of unclear message (\(P_u\)) for a given collaboration mode (\(F, D, T\) or \(DT\)) is affected by the cost (i.e., quality) of the communication tool used, such that \(P_u = 1\) if the cost of the tool is 0 (i.e., inexpensive communication tools increase the probability of miscommunication) and goes asymptotically to 0 as the cost of the tool goes to infinity.; (2) Only one clarification message is necessary to resolve unclear messages; (3) The production object (i.e., software) and messages can be sent/received instantly across; (4) The task is high priority and time constrained; (5) When working hours overlap actors prefer to communicate synchronously (e.g., telephone, video conference); they communicate asynchronously (e.g., e-mail, shared databases) otherwise; and (6) All tasks requested by \(R\) are immediately accepted by \(P\) and carried out competently with no parallel multi-tasking.

Communication costs \((C_c)\) are the sum of the cost of maintaining a communication link \((C_l)\) and sending two communication messages \((C_m)\), one by \(R\) to request the task and one by \(P\) to acknowledge its completion. The asymptotic relationship between \(P_u\) and \(C_c\) is established as \(P_u = K/(C_c + K)\), where \(K\) can be any constant. In our simulation, we assume that \(P_u(F) < P_u(D) < P_u(T) < P_u(DT)\), to reflect the fact more team boundaries make clear team communication more difficult [6, 25]. The constant \(K\) for each
collaboration mode (Pu(F), Pu(D), Pu(T) and Pu(DT)) was selected so that Pu is arbitrarily set to 0.1, 0.3, 0.5, and 0.7 for the average cost of all communication tools used in F, D, T and DT collaboration modes respectively. For example, if a dyad operates in a time-separated collaboration mode (T) using a communication tool of higher quality and cost than average, then the resulting probability of sending or interpreting messages unclearly through that tool Pu(T) is lower than the average 0.5 for that mode. Again, these parameters can be changed in different simulation exercises.

### 4. Model Evaluation Method

We used Ordinary Least Squares (OLS) regression analysis of several thousand randomly generated observations to evaluate the model and to develop a better understanding of how different variables affect coordination and vulnerability costs. We generated the observations similar to a Monte Carlo simulation, by drawing random values for all exogenous variables from expected probability distributions. We then computed all other variables using our model. The main exogenous variables included were: Overlap Type (binary – 1 if at the end of R’s work day, 0 if at the beginning), Overlap (Ob or Oe – 0 to 1 in increments of 0.1), Request Time (Rt) drawn from a uniform distribution [0,1]; All other exogenous variables were drawn from a normal distribution with the values indicated in Table 1. The cost of maintaining a communication link (Cl) and sending messages (Cm) were randomized to reflect the use of multiple communication tools of various qualities and costs. R’s Delay Cost per Day (Cd) were set intentionally high to reflect high opportunity costs of delay due to time-to-market pressures, with a high variance to cover a wide range of delay costs; P’s Production Cost (Cp) were set high for the same reason. Lower production costs could be simulated to evaluate less costly outsourcing arrangements for P.

Because the timing of activities has a substantial effect when parties are separated by time (e.g., time zones), we were particularly interested in understanding the timing dynamics of the model. Therefore, we included a number of interaction variables between main effects and time-related variables (i.e., Overlap Type, Overlap, Request Time and Task Completion Time). The regression model was estimated in a hierarchical fashion by entering all main effect variables first and then adding all interaction effect variables as a block, so that we could evaluate the respective changes in main effects. Because of the large number of possible interaction variables to include and to avoid problems of multicollinearity, we only retained significant variables in the model. We first entered all interaction variables and then removed those that were not significant at the p<0.001 level, one by one in a backward fashion to avoid omitted variable bias. Work overlap can occur at the beginning (Ob) or end (Oe) of R’s day. Similarly, task requests and acknowledgements can occur during overlap or non-overlap hours. Therefore, we generated a few thousand random observations for each of 11 overlap index values, ranging from 0 (i.e., no overlap) to 1 (i.e., full overlap), in increments of 0.1. Half of the observations were generated for Ob and the other half for Oe. All dependent variables were calculated with the model formulas provided above. The simulation model first evaluates if the request time falls within or outside the overlapping work hours and then computes the communication costs (Ca and Cs) and time of delay (Td) accordingly. As previously discussed, one important aspect of the simulation model is that Td is only computed during R’s work hours. The delay time during R’s off-work hours are not included in Td.

### 5. Evaluation Results

Regression results presented in Table 2 suggest that the model behaves according with expectations. As the table shows, the inclusion of interaction variables significantly increased the explained variance ($R^2$) in all models, suggesting that these interaction terms are important. Consequently, we

<table>
<thead>
<tr>
<th>Variables and Parameters</th>
<th>Mean</th>
<th>Std. Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pu(F)</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>Pu(D)</td>
<td>0.3</td>
<td></td>
</tr>
<tr>
<td>Pu(T)</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>Pu(DT)</td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>Fr</td>
<td>0.3</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Variables and Simulation Parameters
only discuss below the results of the interaction models. Since our main interest is in time separation, we first discuss briefly the main effects of non time-related variables and then discuss main and interaction effects of time-related variables in more detail. Also, since total costs are a function of coordination and vulnerability costs, we only discuss effects on total costs when the effects on coordination costs and vulnerability costs have different signs. Finally, we report on results significant at the p<0.001 level, except when noted.

As expected, P’s daily production costs (β=0.03) and the proportion of rework (β=106.12) increase vulnerability costs. Similarly, R’s delay costs increase coordination costs (β=0.36) and vulnerability costs (β=0.20). Interestingly, an increase in asynchronous and synchronous communication costs raises coordination costs (β=0.76 and β=0.30 respectively), but reduce vulnerability costs (β=−1.53 and β=−0.10 respectively), underscoring the importance of understanding the tradeoffs between the cost increase of using more expensive communication tools and the reduction in vulnerability costs resulting from fewer unclear messages resulting from using higher quality communication tools. On the other hand, distance separation increases both coordination costs (β=361.70) and vulnerability costs (β=567.55). In contrast, time separation alone (controlling for distance separation) reduces coordination costs (β=−619.89), but it increases vulnerability costs (β=228.27). Delay costs are reduced because P can produce all or part of the task during R’s off-work hours, but are increased because time separation increases the probability of unclear messages.

With respect to time-related variables, the actual work overlap time substantially reduced both coordination costs (β=−1,475.23) and vulnerability costs (β=−686.25) (i.e., every 0.1 increment in overlap decrease costs by $147.52 and $68.62 respectively). Coordination and vulnerability costs increased when only one of the two interactions occurred during overlap time (β=769.88 and β=481.96 respectively) or when none of the interactions occurred during the overlap time.
While work overlap time reduces coordination and vulnerability costs, these beneficial effects are quickly offset with cost increases when the requests are made later in R’s work day (β = 5,193.07 and β = 3,110.64 respectively) and as the time it takes to complete the task goes up (β = 4,338.99 and β = 1,584.99 respectively). Also, the beneficial effects of overlap are somewhat offset as asynchronous (n.s. and β = 2.44 respectively) and synchronous (β = 0.40 and β = 0.38 respectively) communication costs increase. Similarly, the beneficial effects of overlap are somewhat offset with geographic distance (β = 609.98 and n.s. respectively).

On the other hand, the beneficial effects of overlap are enhanced when R’s daily cost of delay are higher (β = −0.31 and β = −0.24 respectively). Whether the overlap occurs at the beginning or the end of R’s work day (i.e., Overlap Type) only made a small difference in our results and, in the case of coordination costs, the effect was marginally significant (p=0.042). But overlap type does make a difference when it interacts with other variables. Having overlapping work hours at the end of R’s day reduces coordination costs and vulnerability costs more strongly with longer overlaps (β = −85.53 and β = −59.42 respectively), later task request times (β = −359.17 and β = −205.39 respectively), and tasks of longer duration (β = −165.47, p=0.032 and β = −113.68, p=0.026 respectively). So, the time when tasks request are made have different effects, depending on whether the work overlap occurs at the beginning or end of R’s work day, supporting our asymmetry arguments.

The time when R makes task requests to P makes a difference too. Task request time significantly reduces coordination costs (β = −2,720.63) and vulnerability costs (β = −1,339.16). The benefits of later request times on coordination and vulnerability costs are even stronger with higher R’s delay costs (β = −0.53 and β = −0.31 respectively), distance separation (n.s. and β = −609.86 respectively) and longer tasks (β = −5,431.02 and β = −3,326.03 respectively). On the other hand the benefits of later request times on vulnerability costs weaken with higher asynchronous (β=3.78) and synchronous (β=−0.53) communication costs because higher communication tool quality reduces the negative effects of time separation on vulnerability costs.

Finally, as expected, task completion time increases coordination costs (β=2624.61) and vulnerability costs (β=1,181.21) because it increases R’s delay, but more so when delay costs are higher (β = 0.59 and β = 0.30 respectively). Also, the effect of task completion time on vulnerability costs is stronger with distance separation (n.s. and β = 529.31 respectively). On the other hand, the effect of task completion time on vulnerability costs is weaker with higher asynchronous (β = −2.12) and synchronous (β = −0.47) communication costs (i.e., higher quality).

6. Discussion and Future Research

6.1 The Model

Our results are intuitive and according to expectations, suggesting that our model is theoretically sound and that it can be used for empirical research of GSTs coordination across time zones. A key finding is that interaction variables add significant predictive power to the regression models, suggesting that we need to pay attention to the multiple interaction factors present in time-separated collaboration and not just to the main effects. This is so because time separation is asymmetric and because all cost variables are very sensitive to the timing of events.

In our model, coordination costs represent the scenario when all goes well. When things don’t go well, vulnerability costs from additional production and/or communication are incurred. When particular effects on these two costs are in opposite directions, our model helps us understand the tradeoffs involved. For example, consistent with “follow the sun”, time separation reduces coordination costs because P can do some of the work while R sleeps. However, vulnerability costs increase because of lower communication quality that may lead to miscommunication, which may explain the paucity of success stories in this endeavor. The opposite is true for communication costs, which increase coordination costs but reduce vulnerability costs because of higher communication quality.

6.2 Time-Separation

Some of the statements in this subsection are intuitive on reflection. We are pleased with this because it shows that our model is robust and can be expanded. Our contribution is in arriving at a set of propositions about coordination across time-zones:

1. Time separation reduces coordination costs. This may seem counter-intuitive but coordination costs in our model are largely based on delays in R’s work caused by a tight dependency with P when all goes well, and that these costs can be reduced if P
produces during R’s off-work hours, which can only happen with time separation (see next point).

2. **Time separation increases vulnerability costs.** These represent further production and/or coordination costs needed because of miscommunication which, unlike pure coordination costs, are affected by the quality of the communication tools used.

3. **Increasing work overlap time reduces both coordination and vulnerability costs.** The fact that time separation reduces coordination costs while reduced work overlap time increases coordination costs seems counterintuitive. But this actually underscores an important tradeoff between the benefits of time separation – P works while R sleeps, which reduces R’s delay – and the increased delays incurred in message delivery – a member sends a message during the other member’s off-work hours – which are reduced with work overlap time. On the other hand, the effects on vulnerability costs are clear: more time separation and less overlap time increase coordination costs.

4. **Vulnerability and coordination costs are reduced due to the following: co-location, more overlap in work hours, and when communication between R and P occurs during overlap hours.** However, these benefits can be quickly offset by other time-related variables. For example, the benefits of work time overlap are offset: (a) if only one interaction (R → P or P → R), or no interaction, occurs during the overlap, because this often causes more delay in message delivery and increases the probability of miscommunication; (b) when task requests are made later in the day; (c) with tasks that take longer to complete; and (d) with geographic dispersion. On the other hand, the benefits of work overlap time are stronger when R’s daily cost of delay is higher.

5. **The type of overlap (i.e., at the beginning or end of R’s work day) makes a difference, but primarily when interacting with other variables.** While making task requests later in the day diminishes the benefits of overlap time, making late requests are somewhat more beneficial when the work overlap time occurs at the end of R’s day. Also, having work overlap time at the end of R’s day reduces coordination costs and vulnerability costs more strongly with longer overlaps and tasks of longer duration.

6. **Making task requests late in the day reduce coordination and vulnerability costs, except when overlap occurs at the beginning of R’s day and the request is made during the overlap.** After the overlap period, the later the better. These benefits are enhanced when R’s delay costs are high, with distance separation, and with longer tasks. On the other hand, these benefits are weaker when communication costs are high, most likely because higher communication tool quality reduces the negative effects of time separation on vulnerability costs, making the request time less important.

7. **Task duration time increases coordination costs and vulnerability costs because it increases R’s delay.** But this effect is stronger when R’s delay costs are high and with geographic dispersion. On the other hand, this effect is weaker when communication costs are high because of high quality communication tools that reduce the probability of miscommunication, which help offset these increased costs.

### 6.3 Distance vs. Time Separation

The number of dyad interaction patterns increases substantially with time separation. In same-time contexts, there are only two possible collaboration modes, co-located or distributed. However, in different-time contexts, there are 16 possible collaboration modes, depending on whether the collaboration is co-located or distributed (2x); R makes a task request during overlapping work hours or not (2x); P completes the task during overlapping work hours or not (2x); and whether the overlap occurs at the beginning or end of R’s work day (2x). This underscores the substantial difference with pure distance-separated contexts where time-related variables don’t have a strong influence on coordination and vulnerability costs. We found that time-related variables – i.e., task request time, task duration time, time separation, overlap time, and the number of interactions that take place during overlap hours – have significant main and interaction effects on coordination and vulnerability costs. On the other hand, except for task duration time, these variables have no role in pure distance-separated conditions (i.e. no time zone differences). In summary, our results support our arguments that: (1) time-separation effects are different and more complex than distance-separation effects; (2) time-separation effects are asymmetric depending on the amount of work overlap time and whether this overlap occurs at the beginning or end of R’s day; and therefore (3) researchers studying the effects of geographic dispersion need to control for the effects of time separation.

### 6.4 Future Research

We plan to apply and adapt the dyadic coordination model to other collaboration contexts: bi-
modal production costs to simulate offshore outsourcing alternatives; substantially raise P's production costs to simulate expensive equipment like testing labs; simulate mission critical applications by increasing the proportion of re-work necessary; simulate investments in communication quality by increasing cost and lowering the probability that communications in a given collaboration mode (F, D, T or DT) will generate unclear messages. In sum, we plan to expand the model to analyze more complex coordination structures like those analyzed by Malone and other structures representative of larger teams found in today's software development organizations.

6. References


Appendix A: Model Diagram

Overlap Conditions:
- Full Overlap (F or D, Ob=Oe=1)
- Beginning of R's day (Ob=[0,1])
- End of R's day (Oe=[0,1])
- No overlap (Oe=Ob=0)

Collaboration Modes:
- Face-to-face (F)
- Separated by Distance (D)
- Separated by Time (T)
- Separated by Distance & Time (DT)

Co-located

Distance Separated

Face-to-Face

Synch Comm

Asynch Comm

Distance Separated

Co-located

Asynch Comm

Pu(F) Cl=Cm=0

Pu(D) Cls, Cms

Pu(T) Cla, Cma

Pu(TD) Cla, Cma

Pu = Prob of unclear messages
Cl = Daily cost of comm link
Cm = Cost of sending 1 message

Task Completion Time (Tt)
Daily production cost for P (Cp)

Overlap condition & Collaboration Mode

R Requests Task to P

Request time (Rt)
Daily task request rate (λ)

Overlap condition & Collaboration Mode

R Requests Task to P

R receives acknowledgement

Total delay time (Td)
Daily cost of delay for R (Cd)

R sends Acknowledgment to R

Acknowledge Time (At)
From R's perspective
At= Rt+Tt, depending on overlap conditions

P sends Acknowledgment to R

From R's perspective

Hours overlap at At

Hours don't overlap at At

P begins task

Original Request

From clarification

P completes task

Message is unclear (Pu)

Clarification only
(no re-work)

Prop of work to be re-done (Rw)

Prob unclear message leads to re-work (Pr)

End

Start

λ