1-1-2006

Coordination theory and its application in HCI

Kevin Crowston

Joseph Rubleske

James Howison
Carnegie Mellon University, jhowison@cs.cmu.edu

Coordination Theory

Kevin Crowston, Joseph Rubleske and James Howison

Syracuse University
School of Information Studies
4–206 Centre for Science and Technology
Syracuse, NY 13244–4100 USA

crowston@syr.edu, jrublesk@twcny.rr.com, jhowison@syr.edu,

+1 (315) 443–1676
Fax: +1 (866) 265–7407

Draft of 29 September 2004

To appear in Zhang, P. and Galletta, D. (Eds.) Human-Computer Interaction in Management Information Systems, Vol I. M. E. Sharpe.

Coordination Theory: A Ten-Year Retrospective

Abstract

Since the initial publication in 1994, Coordination Theory (Malone and Crowston, 1994) has been referenced in nearly 300 journal articles, book chapters, conference papers and theses. This chapter will analyze the contribution of this body of research to determine how Coordination Theory has been used for user task analysis and modelling for HCI. Issues that will be addressed include: 1) how the theory has been applied; 2) factors that led to the success of the theory; and 3) identification of areas needing further research.

Coordination Theory: A Ten-Year Retrospective

An increasingly ubiquitous application of computer systems is to help a group of people work together better. To do so requires an appreciation of what the group is doing and how its members might work together in a more efficient or effective manner. Such user/group task analysis and modelling is at the core of MIS HCI research. A key issue in the analysis of group work is an understanding of the dependencies between the tasks the different group members are carrying out and the way the group coordinates their work. However, many studies describe dependencies and processes only in general terms, without characterizing in detail differences between dependencies, the problems dependencies create or how the proposed coordination processes address those problems (Grant, 1996; Medema, 1996). This vagueness makes it difficult or impossible to determine what alternative processes might be useful in a given circumstance. Similarly, it is hard to translate from dependencies to specifications of individual activities or to uses of information and communication technologies (ICT) to support a process (e.g., as part of system development during a business process redesign effort (Davenport & Short, 1990; Hammer, 1990; Harrington, 1991; Harrison & Pratt, 1993)).

In 1994, Malone and Crowston described a new approach to these problems, an approach they called Coordination Theory (CT) (Malone & Crowston, 1994). Their 1994 paper presented examples of similar coordination problems encountered in a variety of disciplines and analyzed them as arising from dependencies. For example, approaches to sharing resources (i.e., ways to manage the dependency created when multiple tasks require the same resources) have been analyzed in economics, organization theory and computer science, among others. Other dependencies identified by Malone and Crowston are producer/consumer dependencies,

simultaneity constraints and task/subtask relations. Since the 1994 publication, nearly 300 papers and dissertations have referred to or made use of the CT approach. The purpose of this chapter is to introduce CT, review the impact it has had for MIS HCI and its limitations, to discuss factors contributing to its level of impact and to identify areas needing further research.

Before introducing CT in detail though, a comment on the name. Some commentators have objected to CT, stating that it is not a theory. This critique presumes a definition of theory such as:

> A set of statements or principles devised to explain a group of facts or phenomena, especially one that has been repeatedly tested or is widely accepted and can be used to make predictions about natural phenomena" (American Heritage Dictionary, 2000).

Such statements are also known as scientific laws. Malone and Crowston chose the name Coordination Theory in part because many of the fields synthesized do have such laws and the hope was that CT would develop to include them, even if it did not in its initial statement. However, the critique overlooks the fact that theories include scientific concepts as well as laws (Kaplan, 1964/1998, p. 297). Theory in this sense helps make sense of data and makes observed events meaningful. Concept and theory generation go hand in hand, and an initial set of concepts guides the search for data and for laws. As we will discuss, CT definitely includes a set of concepts that can be used to label phenomena and link them to others, and thus fulfills this important function of theory. In its current state though, we would describe CT as a pattern model (Kaplan, 1964/1998, p. 327), meaning that it seeks to explain phenomena by showing how they fit a known pattern.

Contributions of Coordination Theory

The primary purpose of Malone and Crowston (1994) was to synthesize work done on coordination from a variety of fields. The work started with an interest in how groupware (that is, software designed to support groups of people working together) might help people to coordinate their activities better. The paper made three key contributions.

First contribution: Definition of coordination

The first contribution of the paper was a concise definition of coordination as "managing dependencies between activities". Coordination has been a long-standing interest of organizational scholars and more recently of computer scientists, so many definitions for this term had been proposed. Malone and Crowston (1994) and Weigand, van der Poll & de Moor (2003) list several, including:

- Structuring and facilitating transactions between interdependent components (Chandler, 1962)
- The protocols, tasks and decision-making mechanisms designed to achieve concerted actions between interdependent units (Thompson, 1967)
- The integrative devices for interconnecting differentiated sub-units (Lawrence & Lorsch, 1967)
- Composing purposeful actions into larger purposeful wholes (Holt, 1988)
- The integration and harmonious adjustment of individual work efforts towards the accomplishment of a larger goal (Singh & Rein, 1992)

- Establishing attunement between tasks with the purpose of accomplishing that the execution of separate tasks is timely, in the right order and of the right quantity (Reezigt, 1995)

In contrast to the first 3 definitions, drawn from the organization studies literature, Malone and Crowston (1994) conceptualize dependencies as arising between tasks rather than individuals or units. This approach has the advantage of making it easier to model the effects of reassignments of activities to different actors, which is common in process redesign efforts. Compared to the final three definitions, taken from the CSCW literature, the Malone and Crowston (1994) definition focuses attention on cause for a need to coordinate, rather than on the desired outcome of coordination. This focus has advantages again for modelling, as will be discussed below.

Second contribution: Modelling framework

The second contribution of the 1994 paper was to provide a theoretical framework for analyzing coordination in complex processes, thus contributing to user task analysis and modeling. Consistent with the definition proposed above, Malone and Crowston (1994) analyzed group action in terms of *actors* performing *interdependent tasks*. These tasks might require or create *resources* of various types. For example, in the case of software requirements development, actors include the customers and various employees of the software company. Tasks include translating aspects of a customer's problem into system requirements and checking requirements for consistency against other requirements. Finally, resources include the information about the customer's problem, existing system functionality and analysts' time and

effort. In this view, actors in organizations face *coordination problems* arising from dependencies that constrain how tasks can be performed.

It should be noted that in developing this framework, Malone and Crowston describe coordination mechanisms as relying on other necessary group functions, such as decision making, communications and development of shared understandings and collective sensemaking (Britton, Wright, & Ball, 2000; Crowston & Kammerer, 1998). To develop a complete model of some process would involve modelling all of these aspects: coordination, decision making and communications. In practice, our analyses have tended to focus on the coordination aspects, bracketing the other phenomenon, though clearly groups might face problems in all of these.

Third contribution: Typology of dependencies and coordination mechanisms

The main claim of coordination theory is that dependencies and the mechanisms for managing them are general, that is, a given dependency and a mechanism to manage it will be found in a variety of organizational settings. Thus the final contribution of coordination theory is a typology of dependencies and related coordination mechanisms, as shown in Table 1. (For continuity, in this paper we will use the original typology presented by Malone and Crowston (1994), though it has been refined in more recent work; see Malone et al. (1999) and Crowston (2003).)

Insert Table 1 here

For example, a common coordination problem is that certain tasks require specialized skills, thus constraining which actors can work on them. This dependency between a task and an actor arises in some form in nearly every organization. CT suggests identifying and studying

such common dependencies and their related coordination mechanisms across a wide variety of organizational settings.

Another common dependency is the *producer/consumer* (or *flow*) dependency, in which one task creates a resource that is needed by another. Malone and Crowston (1994) analyzed this dependency into three subdependencies, *usability*, *transfer* and *precedence*. The *usability* subdependency means that the resource created by the first task must be appropriate for the needs of the second task. The *transfer* subdependency means that the resource must be moved from where it was created to where the consuming task will be performed. And finally, the *precedence* subdependency means that the actor performing the second task must learn when the resource is available and the task can be started.

To overcome these coordination problems, actors must perform additional work, which Malone and Crowston (1994) called *coordination mechanisms*. For example, if particular expertise is necessary to perform a particular task (a task-actor dependency), then an actor with that expertise must be identified and the task assigned to him or her. There are often several coordination mechanisms that can be used to manage a dependency. For example, mechanisms to manage the dependency between an activity and an actor include (among others): (1) having a manager pick a subordinate to perform the task, (2) assigning the task to the first available actor and (3) a labour market in which actors bid on jobs. To manage a usability subdependency, the resource might be tailored to the needs of the consumer (meaning that the consumer has to provide that information to the producer) or a producer might follow a standard so the consumer knows what to expect. Mechanisms may be useful in a wide variety of organizational settings. Conversely, organizations with similar goals achieved using more-or-less the same set of

activities will have to manage the same dependencies, but may choose different coordination mechanisms, thus resulting in different processes.

CT suggests that, given an organization performing some task, one way to generate alternative processes is to first identify the particular dependencies and coordination problems faced by that organization and then consider what alternative coordination mechanisms could be used to manage them. In particular, we can look for processes supported by extensive use of ICT, thus contributing to system design for collaborative systems.

Summary of contributions

To summarize, Malone and Crowston (1994) made several contributions in defining coordination theory. Substantively, their 1994 paper provided 1) a succinct and actionable definition of coordination, 2) a framework for task analysis and modelling for collective processes and 3) the beginning of a typology of dependencies and coordination mechanisms. As well, the paper advanced the field by drawing attention to coordination as a topic for research and providing examples of coordination topics in multiple disciplines. Coordination mechanisms are now an integral topic in fields such as distributed artificial intelligence and multi-agent systems.

Example: Coordination in restaurant service

In order to make the previous presentation of CT more concrete, we will present two examples of its use for MIS HCI, in restaurant service and in software development. The first example is a short analysis of the customer service aspects of a restaurant. Restaurants have long been studied as important forums for coordination. The essential characteristics of

restaurants—many customers, many orders, frequent deliveries, continuous monitoring of customers and of personnel in accomplishing work, and perishable products—makes them particularly illuminating for studies of logistical flows, information flows, and resultant needs for coordination. As Whyte (1948, pp. 18–19) noted, "Failure of coordination is perhaps the chief enemy of job satisfaction for the worker. And the varying and unpredictable demands of customers makes this coordination always difficult to achieve." He noted further that in a small restaurant, everyone was in direct contact "and the problems of communication and coordination are relatively simple," while in a larger restaurant, "coordination must be accomplished through people who are not generally in face-to-face contact with each other" (p. 47). Finally, we assume that all readers are familiar with the basic process of restaurant service, allowing us to focus on the coordination aspects of the analysis.

The first step in a CT-based analysis is to develop a description of the activities involved in the process. A simple description of these steps is shown in Figure 1, which shows actors on the left and activities performed by each across the page in time-order. Activities performed jointly are connected by dotted lines. While there may be some disagreements about details (note in particular that the kitchen is modelled as a single collective actor and cooking as a single high-level activity), the sequence of activities is recognizable as representative of a traditional sit-down cook-to-order restaurant.

Insert Figure 1 here

Dependencies in restaurant process

The next step in the analysis is to identify dependencies in the process. In this case, a particularly important type of dependency is the producer/consumer dependencies between

activities. These dependencies can be easily identified by noting where one activity produces a resource required by another. The resource flows and resulting dependencies for the restaurant process are shown in Figure 2. For example, the activity of cooking creates food that can then be served and eaten; customers' departure produces a table ready for busing; and busing and resetting a table produces a table ready for another customer. One could also analyze shared resource dependencies between customers needing the same table or attention in the kitchen, but for brevity, we will omit this discussion.

Insert Figure 2 here

Viewing the process in terms of producer/consumer dependencies further suggests that the wait staff act as intermediaries between the kitchen and the customer, taking an order which is then transmitted to the kitchen, or taking food prepared by the kitchen and delivering it to the customer. In other words, there is a high-level dependency—between the cooking activities performed by the kitchen and the eating activities performed by the customer—that is managed by activities performed by the wait staff. More specifically, the *usability* subdependency between cooking and eating is managed by taking an order (allowing the customer to select the most appropriate food and in some case, the kitchen to tailor the food to the customer). The *transfer* subdependency is managed by having the wait staff move the food from the kitchen to the table. The *precedence* subdependency is managed by the customer's waiting for the food to be delivered. Similarly, activities performed by the host or hostess manages a dependency between a customer's needing a table and the activities involved in preparing tables. This model is shown in Figure 3.

Insert Figure 3 here

The analysis can also be performed for dependencies that arise between the initial set of tasks and the tasks that comprise the chosen coordination mechanisms. In particular, for a traditional restaurant there is a producer/consumer dependency between serving and cooking that itself requires management, for the *precedence* subdependency in particular. That is, there must be some way for the wait staff to know when the food is ready to be served.

Alternative processes

Having identified the dependencies, we can now use the taxonomies of dependencies and related coordination mechanisms to imagine alternative processes. In particular, we can look for processes that rely on ICT support, using the models to suggest the coordination functions that the systems need to perform.

For example, different actors might perform the order and transfer coordination mechanisms. In many busy restaurants, drinks and food are delivered by "runners" waiting in the kitchen rather than by the wait person who took the order. Similarly, rather than walking an order to the kitchen, wait staff in many restaurants enter orders on a computer system that transmits them to the kitchen. Some restaurants even provide wait staff with wireless terminals with which to transmit orders from table to kitchen. One could imagine providing such a terminal to the customer; orders would then be directly transmitted to the kitchen and delivered when they are ready, thus eliminating the role of the waitress or waiter, i.e., disintermediation of the relationship between kitchen and customer (Benjamin & Wigand, 1995). (Indeed, we say such a system in one Internet café: customers visited the café's website to place orders for coffee and cakes, which were then delivered by the wait staff.)

Restaurants have employed a large variety of information systems to manage the precedence subdependency between activities. In Malone and Crowston's (1994) analysis, a precedence dependency can be managed in one of two ways: either the person performing the first activity can notify the person performing the second that a resource is ready, or the second can monitor the performance of the first. A number of mechanisms have been devised to notify wait staff that food is ready in the kitchen, ranging from a cook's shouting "Order up!", to bells, numbered lights on the wall, and most recently pagers. In the absence of a notification mechanism, actors must instead spend time monitoring the status of the previous activity. For example, a bused table, ready for a customer, waits until the host or hostess notices it. A paging system can be used to let the buser notify the host or hostess that a table has been bused and is ready. Similarly, the wait staff can monitor the kitchen to notice when an order is ready or the kitchen staff can page the wait staff to notify them that it is. Once a suitable notification system is available, it can be employed at all stages of the process. For example, wait staff can be paged when new customers arrive at a table; a buser can be paged when the table has been vacated and is waiting to be bused. These changes can be employed in combination: for example, in a restaurant that employs runners, the wait staff can be paged to meet the runner at tableside to make the final presentation of the food, thus maintaining the appearance of a single server.

In summary, this example illustrates the main uses of CT. A process can be analyzed in terms of the dependencies among resources and tasks and the current coordination mechanisms identified. This analysis can then be used to suggest alternative processes, created by substituting one coordination mechanism for another.

Example: Coordination in software requirements analysis

As a second more substantive example, we present a coordination theory analysis of the software requirements analysis process, drawn from Crowston & Kammerer (1998). Requirements analysis is the stage in the software development process of determining what functionality and interfaces a new system should have. Figure 4 places the requirements analysis process in the context of the overall software development process. This stage is especially important because it establishes the framework for the rest of the development effort. Early mistakes become increasingly more difficult and expensive to fix later in the process, so inadequate requirements definition will have adverse effects throughout the development process and beyond (Pressman, 1982).

Insert Figure 4 about here

Dependencies in software requirements analysis

A CT analysis involves identifying the dependencies that arise in this process and the coordination mechanisms being used to manage them. The primary tasks in requirements analysis are translating domain knowledge, such as customers' statements of their needs or descriptions of interacting systems, into requirements that can be used to develop software. As well, these requirements need to be checked to ensure they are complete and consistent with each other and with prior decisions about the system. Resources include the knowledge and effort of the analysts, other people whom the analysts consult and other sources of information. Between these tasks and resources, numerous dependencies are possible.

Task-resource dependencies. Most tasks require resources, if only the effort of the actor performing the task, and an important class of coordination processes manage the assignment of resources to tasks. Crowston (2003) suggests four steps to such resource assignments: identifying the type of resources needed, identifying available resources, selecting the resources and making the assignment. Most work on resource assignment (e.g., in economics, organization theory or computer science) has focused on the middle steps, that is, techniques for identifying or selecting available resources. By contrast, in the requirements analysis process, the first step—identifying what resources are needed to perform a particular task—seems equally important. Analysts can easily refer problems to each other, but they must somehow first determine who should be consulted for a particular problem based on the nature of the problem and the expertise and assignment of different developers. This problem is exacerbated by the potential for dependencies between requirements (that is, between the products of these tasks), which increases the need for consultation.

Producer-consumer dependencies. A second major type of dependency is a producer-consumer dependency, where the output of one task is the input to another. Malone and Crowston (1994) note that such dependencies often impose additional constraints, in particular usability—ensuring that the output is of a form usable by the next task—and transfer—ensuring that the output is available to the consumer when needed. This type of dependency exists at a high level between the requirements process and further downstream software development processes. While there are some transfer problems, e.g., making sure that the requirements are finished on schedule, usability seems to pose a key problem. Indeed, at the highest level shown in Figure 4, the requirements development process itself can be viewed as a coordination mechanism that ensures that the output of the software development process is usable by the

eventual customers for the system. Malone and Crowston (1994) suggest alternative approaches to satisfying a usability dependency, including standardization (i.e., producing the output in an expected form), asking the users and participatory design.

For the software developers who are the user of the requirements, the main approach to usability appears to be standardization. That is, most companies have standards for the format and content of requirements documents to ensure that they are easily usable by software developers. For the eventual customer of the software though, a standard output is assumed to be unsatisfactory, since the companies studied developed customized systems. Making the system usable required the application of a lot of specialized knowledge about application domain (Walz, Elam, & Curtis, 1993) and members of requirements development groups were often experts in their fields. However, gathering the information about the particular circumstances of the customer required considerable additional work.

Task-subtask dependencies. Third, some tasks are decomposable into subtasks. For example, the task of writing requirements must typically be decomposed into units of work that a single individual can perform. A key issue in any decomposition, however, is ensuring that the overall goal is satisfied by the performance of the subtasks. Without such a check, it is easy for analysts to focus on their component of the system and lose sight of the overall goals of the system.

Feature and requirement interdependencies. Finally, there are dependencies between entities that need to be identified and managed (a dependency that Malone et al. (1999) labelled "fit"). In particular, interactions between features are often a problem in large systems with many features. Analysts must determine how each feature interacts with every other feature in the

system and control for undesirable interactions. When there are hundreds of features, each supported by different teams of people, these interactions may be extremely difficult to detect. Modifying the software for one feature may affect other features in very subtle ways which are unanticipated by the analyst specifying the modification, with the result that those in charge of the affected feature might not become aware of the changes until their feature "breaks". In order to solve the feature interaction problem, one must know how to get important information to the appropriate people without overloading everyone by broadcasting all information to all people. As well, the requirements themselves may have dependencies. Developing a consistent specification when many different people must write parts of the specification is difficult.

Coordination mechanisms for requirements analysis

Many of the solutions proposed for requirements analysis correspond to the suggestions of CT, focusing on better ways to manage dependences or even to reduce and eliminate them. An example of the second is the decomposition of a system into decoupled subparts with well-defined interfaces or using design methodologies that only allow certain kinds of interactions, with the goal of forcing dependences along particular channels.

Strategies for managing dependences focus on more efficient ways to transfer information among people, such as formal languages or prototyping, to surface dependences. These include techniques for eliciting requirements from people, modeling techniques, requirements definition methods and computer support tools (Brackett, 1990). Techniques for eliciting requirements include a variety of general techniques for information gathering, e.g., as surveyed by Davis (1983) and Powers, Adams, & Mills (1984). Joint Application Development Technique (JAD), which was developed by IBM, provides a technique for analysts to get

information from and negotiate with clients in intensive workshop sessions. SADT (Marca & McGowan, 1988) provides a method for iterative review of requirements by developer and client. Walk-throughs (Freedman & Weinberg, 1982) and technical reviews (Collofello, 1988) can be used to assess the quality and progress of the requirements definition process.

Modeling techniques are used to represent aspects of the information gathered. The methods are typically graphical, although physical and simulation models are occasionally used. Most modeling methods allow the system to be described in terms of several of the following in order to gain a sufficiently comprehensive view of the system: interfaces to external entities, functions to be performed, data transformations, structure of input/output data, relationships among information and system behavior (Brackett, 1990).

There are many computer-based tools to support requirements definition and model development. Most of these are method-specific tools. Non-method-specific tools also exist, with PSL/PSA (Teichroew and Hershey, 1977) being one of the most well known of these. The non-method-specific tools generally require either customization to the method used by the developer or translation from the developer's notation to the tool's notation. Finally, many tools have been developed to support prototyping. These include languages and packages designed for creating prototypes, fourth generation languages, windowing system tool-kits and many others (Brackett, 1990).

In summary, a CT analysis of requirements analysis suggests viewing the requirements analysis process itself as a coordination mechanism for managing the usability of the software development process for the end user. Within the process, developers must identify and manage dependences inherent in software requirements development, specifically dependencies between

tasks and actors with the knowledge to contribute to the tasks, producer/consumer dependencies between requirements developers and programmers (especially with regards to usability) and dependencies among requirements and features themselves.

Coordination theory: Impact

In this section, we will discuss the impact that CT has had in general and on MIS HCI in particular. A common way to measure the impact of a theory is by counts of citations to the seminal articles. By this measure, CT has had a moderate impact, as the CT articles (the 1994 Computing Surveys article (Malone & Crowston, 1994), an earlier conference paper (Malone & Crowston, 1990), and working paper versions (Malone, 1988; Malone & Crowston, 1991)) have been cited in at least 287 journal articles, conference papers and dissertations. This level of citations is well above the average, though not as high as a citation classic, such as TAM (F. D. Davis, 1989), which has closer to 800 citations.

To better understand the impact of CT, we examined the use of CT in these publications in more detail. In the remainder of this section, we will describe the method used to find and code citing articles, then discuss the factors that seem to have contributed to the success that CT has enjoyed. We conclude with a brief discussion of the progress that has been made on Malone and Crowston's (1994) research agenda and suggestions of areas for further research.

Method

Citations were found by searching for citations to the CT articles mentioned above in the ISI Social Science and Science Citation Indexes, which we accessed via Dialog. To find citations in conference papers and other sources not indexed by ISI, we also searched for citations in

Citeseer (http://citeseer.psu.edu/). These two sources provided a total of 287 references between 1989 (a citation to the earliest working paper presentations of CT) through 2004. Note though that there is a delay between publication and indexing by citation databases, meaning that the data for recent years is likely incomplete. As well, these two databases do not cover numerous sources, such as book chapters (e.g., those in Malone, Crowston and Herman (2003), most of which cite Malone and Crowston (1994)). As a result, the count of citations should be considered a lower bound.

We next obtained abstracts and full text for as many of the articles as we could. We were able to obtain the text for 232 of the articles. We then coded these articles according to the general topic of the work, how the research reported used CT, and if the research extended CT in some way. The coding system was developed through discussion among the authors to refine the definitions of the categories. Once we had come to agreement about how to code a subset of the articles, each author independently coded a portion of the remaining articles.

Uses of coordination theory

For how the research used CT, we coded articles into four categories, based on which of the three contributions of the Malone and Crowston (1994) contributed to the research.

- The majority of articles (115/232 or about half) were coded as "mention", meaning that a CT article was cited (e.g., in a discussion of coordination), but the research did not use the definition, modelling framework or typology of dependencies and mechanisms proposed by Malone and Crowston (1994). (Indeed, a few articles included a CT reference without any discussion in the text of the paper.)

- The next largest group of articles (63/232 or just over a quarter) used the definition of coordination proposed by Malone and Crowston (1994) ("coordination is management of dependencies"), but not other aspects of the work.

- A small number of papers (10/232 or 4%) used the definition and the modelling framework (actors working on tasks that create or use resources) but not the typologies.

- The final set of articles (46/232 or 20%) made some use of the typology of dependencies and coordination mechanism. We focused most of our attention on these articles, since they represented the most substantive use of CT. Examples of the work in this group of articles are discussed below.

Figure 5 shows the count of articles by classification over time. The figure suggests a steady interest in CT over time (note that the citation counts for 2003 and especially 2004 are incomplete).

Insert Figure 5 here

Factors in the impact of Coordination Theory

In this section, we briefly discuss the factors involved in the level of adoption of CT. To analyze this success, we draw on work by Martens (2003). Based on a study of the life history of theories, she suggested eight factors important to the success of a theory, grouped into empirical, theoretical and social factors.

Empirical factors

Martens notes three empirical factors for the success of a theory: that the theory is applicable to a wide variety of phenomenon; that the phenomena are salient; and that the constructs are defined in a way that facilitates replication and testing. CT does generally well on these empirical criteria. First, our review of the citations discussed above shows that CT has been applied to a diverse range of setting. Major areas include:

- software engineering (e.g., Crowston, 1997; Crowston & Kammerer, 1998; Faraj & Sproull, 2000; Krieger, Vigder, Dean, & Siddiqui, 2003; McChesney, 1997),
- systems design (e.g., Bernstein & Klein, 2002a; Bernstein & Klein, 2002b; Bui, 2000; Bui & Lee, 1999; Klein & Bernstein, 2004; Klusch, Bergamaschi, & Petta, 2003; Ossowski & Omicini, 2002; Ricci, Omicini, & Denti, 2003; van Breemen & de Vries, 2001),
- business processes (e.g., Albino, Pontrandolfo, & Scozzi, 2002; Lee & Lee, 1999; Lizotte & Chaib-draa, 1997; Sikora & Shaw, 1998),
- supply chains (e.g., Bello, Chelariu, & Zhang, 2003; Britton et al., 2000; den Hengst & Sol, 2002; Kobayashi, Tamaki, & Komoda, 2003; Mehring, 2000) and
- organizational simulations (e.g., Clancey, Sachs, Sierhuis, & van Hoof, 1998).

For example, Den Hengst and Sol (2002) focus on "interorganizational coordination structures" (IOCSs) and the factors (such as ICTs) that influence them, and in particular, to assess how e-commerce will affect IOCSs. The container-transport industry is used as a case study. Crowston (1997) uses CT to analyze the software problem fixing process of a large mini-computer manufacturer in order to identify source of difficulties and possible alternative

processes. CT has also been used to analyze the cataloguing processes of the *Flora of North America* digital library (Schnase et al., 1997), the benefits of active sick leave (Scheel, Hagen, & Oxman, 2002), methods for organizational process change (Kim, 2000), knowledge management (Holsapple & Joshi, 2000, 2002) and the use of genres for coordination (Yoshioka, Herman, Yates, & Orlikowski, 2001).

Second, as we noted in the introduction, in these diverse areas, many research have noted the importance of coordination problems and the potential for computer systems to help groups of people work together better, arguing for the salience of coordination as a research topic.

For the final factor though, testability, we note a negative result, since CT is not currently presented in way that facilitates testing. Indeed, an early critique of CT, as mentioned above, was that it is not a theory at all, because it did not provide a list of hypothesized relationships. Martens's research suggests that it would be useful for the progress of CT to be able to state a set of testable propositions or hypotheses. These would likely be in the form of a process model, identifying factors that are necessary but not sufficient for better coordination of a process.

Theoretical factors

Martens identifies two theoretical factors for the success of a theory: multiple uses for theory; and fit with other theories or approaches. Again, CT does well on these criteria. First, CT can be applied to several purposes, such as modelling a process to understand how it works or to suggest improvements, or for designing new coordination mechanisms, particularly those that rely on information and communications technologies. A number of articles build on the modelling approach proposed by Malone and Crowston (1994). For example, Tolksdorf used the

definition and modelling concepts from CT to develop an XML schema for describing business process models (Tolksdorf, 2000) and a dependency markup language (Tolksdorf, 2003).

Second, CT has been combined with other process modelling techniques. For example, Crowston & Osborn (2003) developed a six-step technique for documenting organizational processes by embedding CT in Checkland's (1981; 1990) Soft Systems Methodology. They propose two general heuristics for identifying dependencies: dependency-focused analyses (identify dependencies then search for mechanisms) and activity-focused analyses (identify mechanisms then search for dependencies). Examples of each approach are provided in the form of a small marketing services company. Barbuceanu, Gray and Mankovski (1999) offer a four-level model of social interaction and behavior into agent-based programming tools. In this model, an agent's request consists of a set of "obliged and forbidden behaviors." Social laws, expressed as obligations, are positioned at the highest level of this model. Below this level is a layer that considers each agent's unique priorities and authorities. The lowest levels of this model consist of scheduling and executing adopted behaviors, respectively.

Kim (2000) uses CT as a basis for organizational process change by applying the Massachusetts Institute of Technology Process Handbook (Malone et al., 1999) with a simulation technique. The Process Handbook modelling approach extends the framework of CT described above by adding the notions of inheritance and decomposition. A process can be decomposed into smaller steps, which in turn can be decomposed further. Having decomposed a process, the dependencies among the steps can then be analyzed. A novel feature of the Process Handbook is the use of inheritance, adopted from object-oriented programming. A process in the Handbook can be viewed as a specialization of a more general process (e.g., the processes for selling in a store or selling on an e-commerce web site are both special cases of selling; the process the

company Lands End uses to sell using a web site is still more specialized). This specialization can be done at any level of decomposition. The combination of the two approaches provides great flexibility in composing novel processes. Kim gives an example of this approach in a case of an organizational change in a hospital.

Several authors have developed new coordination mechanisms, particularly those in distributed artificial intelligence. For example, Decker in his PhD thesis designs and evaluates a "family" of coordination mechanisms for cooperative computational task environments. Chen and Decker describe 17 coordination mechanisms to handle the dependency relationships among multiple agents' tasks. They describe these as components that can enable agents to adapt to changing environments. Each mechanism was catalogued into one of eight groups (p. 6):

- Avoidance;
- Reservation schemes;
- Simple predecessor-side commitments;
- Simple successor-side commitments;
- Polling approaches;
- Shifting task dependencies (by learning or transferring mobile code);
- Various third-party mechanisms; and
- More complex multi-stage negotiation strategies.

Schmidt and Simone (1996) focus on the use of artifacts for coordination purposes in cooperative settings. They suggest that coordination mechanisms "are characterized by a specific and crucial relationship between protocol and artifact," and consider computational coordination mechanisms, "in which the allocation of functionality between actor and artifact is changed in

such a way that the coordination mechanism, as a software device, incorporates the artifact in a computational form as well as aspects of the protocol which, again in a computational form, operates on the artifact" (p. 185).

As well, CT has clear links to other theories of social behaviour. The original presentation of CT indicated its reliance on theories of group decision making and communications. Englert et al. (1996) extend this layer framework to include and describe relationships between coordination, cooperation (modified from "group decision making"), communication and selected communication media. Gittel (2000) notes the importance of human relations for successful coordination. Crowston and Kammerer (1998) combine CT and collective mind theory (Weick & Roberts, 1993) to identify sources of problems in managing teams of requirements analysts at two companies (extending the example presented above). The two theories complement each other in that CT assumes that individuals have shared understandings of the problems they face, while collective mind theory explains how those shared understandings develop. Weigand, van der Poll and de Moor (2003) similarly note that the problem of coordination is aggravated by several factors, such as the information asymmetry between actors. They suggest that "when communicative action is aimed at mutual understanding, it is an integration mechanism indeed, and thus also a coordination mechanism" (p. 123).

Social factors

Finally, Martens identifies three social factors for the acceptance of a theory: the theory is relatively easy to understand and to use; it first appears in a widely-read journal; and it is communicated in a variety of settings (e.g., via students and lab members). CT scores generally

well on the first and third criteria but not as well on the second. The definition of coordination proposed by Malone and Crowston is quite simple and easy to understand, perhaps accounting for the high number of citations to the definition. The modelling framework is similar to many already in use, again suggesting that it is easy to use.

On the other hand, while *ACM Computing Surveys* is well known in the computer science field, it is hardly known outside of it. The choice of the original outlet explains the preponderance of citations in computer- and information-related fields and the relative paucity of citations in organizational-focused disciplines, even though CT was intended to unify these treatments of coordination. The article has recently been reprinted in two books, which may help in the dissemination of the ideas. Finally, CT has been picked up and further disseminated by several of Malone's students and colleagues (11 of the 46 substantive papers or about 25% were authored by Malone and his associates).

Conclusion

In the conclusion to their paper, Malone and Crowston set out a research agenda for CT. To conclude this paper, we will briefly discuss the progress that has been made in accomplishing this agenda and note places where more work is still needed.

Malone and Crowston's (1994) first agenda item was how to represent coordination processes. They note several possibilities, such as flow charts, Petri nets and state transition diagrams. In more recent work, Malone et al. (1999) propose a representation technique (the "Process Handbook") that represents processes at various levels of abstraction, as described above. A further advantage of this approach is its ability to suggest new processes by navigating through the dual hierarchies, abstraction and composition. As well, the work reviewed above on

modelling techniques shows that coordination can be incorporated into many modelling techniques.

The second agenda item was to determine what kinds of dependencies there are. Crowston (1994; 2003) conceptualized dependencies as arising from shared use of resources by multiple tasks, thus providing a conceptual basis for a typology of dependencies, though the resulting typology was nearly the same as shown in Table 1. Resources in this framework include anything produced or needed by a task. For example, Faraj and Sproull (2000) introduce the notion of expertise coordination ("the management of knowledge and skill dependencies") that "entails knowing where expertise is located, knowing where expertise is needed, and bringing needed expertise to bear".

The third agenda item was how to classify different coordination processes. The initial approach to this problem was to list coordination processes by the dependency they addresses. Malone et al. (1999) take this approach further by proposing a hierarchy of coordination processes from general to specific. Other authors have proposed different organizations. For example, Etcheverry, Lopisteguy and Dagorret (2001a; 2001b) propose a catalog of coordination patterns. A pattern is defined as "a solution to a problem in a given context", so the catalog is organized by coordination contexts. More work could be done to bring together and organize the mechanisms that have been studied.

The fourth agenda item was about the generality of coordination mechanisms. Most of the work applying the typology of coordination mechanisms has assumed rather than tested the generality of the mechanisms. Nevertheless, the list of mechanisms does seem to have been useful in a variety of settings.

A final agenda item was how to analyze specific coordination practices, e.g., resource allocation. Malone and Crowston (1994) asked, "Can we characterize an entire 'design space' for solutions to this problem and analyze the major factors that would favor one solution over another in specific situations?" Most applications of CT are not very explicit about evaluation or factors that make particular coordination mechanisms more or less desirable. There has been some work addressing specific metrics for coordination. For example, Frozza & Alvares (2002) offer a list of criteria for comparing mechanisms: predictivity, adaptability, action control, communication mode, conflicts, information exchange, agents, applications, advantages and disadvantages. Albino, Pontrandolfo and Scozzi (2002) develop the notion of coordination load, "a quantitative index that measures the effort required to properly coordinate a given process", based on an analysis of the workflow in the process. The goal of this index is to allow a comparison of alternative coordination modes. Nevertheless, it is clear that we are far from characterizing the design space for any of the identified dependencies or coordination mechanisms.

In conclusion, the past ten years have seen considerable progress on the Coordination Theory agenda laid out by Malone and Crowston. The basic outlines of the theory are clear. Methods for capturing and documenting processes and coordination mechanisms and systematic typologies of dependencies and mechanisms have been developed. There are numerous examples of the application of CT in a variety of settings. Taken together, this body of work provides a solid basis for the application of CT. Challenges for future research include developing testable hypotheses (e.g., about the generality of coordination mechanisms) and more structured approaches to evaluate and choose between alternate coordination processes.

References

Albino, V., Pontrandolfo, P., & Scozzi, B. (2002). Analysis of information flows to enhance the coordination of production processes. *International Journal of Production Economics, 75*, 7–9.

American Heritage Dictionary. (2000). (4th ed.). Boston: Houghton Mifflin.

Barbuceanu, M., Gray, T., & Mankovski, S. (1999). Role of obligations in multiagent coordination. *Applied Artificial Intelligence*.

Bello, D. C., Chelariu, C., & Zhang, L. (2003). The antecedents and performance consequences of relationalism in export distribution channels. *Journal Of Business Research*.

Benjamin, R., & Wigand, R. (1995). Electronic markets and virtual value chains on the information superhighway. *Sloan Management Review*(Winter), 62–72.

Bernstein, A., & Klein, M. (2002a). Discovering services: Towards high-precision service retrieval. In *Web Services, E-Business, and the Semantic Web, Lecture Notes in Computer Science* (Vol. 2512, pp. 260-275).

Bernstein, A., & Klein, M. (2002b). Towards high-precision service retrieval. In *Semantic Web—ISWC 2002, Lecture Notes in Computer Science* (Vol. 2342, pp. 84-101).

Brackett, J. W. (1990). Software Requirements, SEI Curriculum Module SEI-CM-19-1.2, January 1990. Reprinted. In M. Dorfman & R. H. Thayer (Eds.), *Standards, Guidelines, and Examples on System and Software Requirements Engineering*. Los Alamitos, CA: IEEE Computer Society Press.

Britton, L. C., Wright, M., & Ball, D. F. (2000). The use of co-ordination theory to improve service quality in executive search. *Service Industries Journal*.

Bui, T. (2000). Building agent-based corporate information systems: An application to telemedicine. *European Journal Of Operational Research, 122*.

Bui, T., & Lee, J. (1999). An agent-based framework for building decision support systems. *Decision Support Systems*.

Chandler, A. D., Jr. (1962). *Strategy and Structure: Chapters in the History of the American Industrial Enterprise*. Cambridge, MA: MIT Press.

Checkland, P. B. (1981). *Systems Thinking, Systems Practice*. New York: Wiley.

Checkland, P. B., & Scholes, J. (1990). *Soft Systems Methodology in Action*. Chichester: Wiley.

Clancey, W. J., Sachs, P., Sierhuis, M., & van Hoof, R. (1998). Brahms: simulating practice for work systems design. *International Journal Of Human Computer Studies*.

Collofello, J. S. (1988). *The Software Technical Review Process* (Curriculum Module No. SEI-CM-3-1.5): Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.

Crowston, K. (1994). *A Taxonomy of Organizational Dependencies and Coordination Mechanisms* (Working paper No. 3718-94): Massachusetts Institute of Technology, Sloan School of Management.

Crowston, K. (1997). A coordination theory approach to organizational process design. *Organization Science, 8*(2), 157–175.

Crowston, K. (2003). A taxonomy of organizational dependencies and coordination mechanisms. In T. W. Malone, K. Crowston & G. Herman (Eds.), *The Process Handbook* (pp. 85–108). Cambridge, MA: MIT Press.

Crowston, K., & Kammerer, E. (1998). Coordination and collective mind in software requirements development. *IBM Systems Journal, 37*(2), 227–245.

Crowston, K., & Osborn, C. S. (2003). A coordination theory approach to process description and redesign. In T. W. Malone, K. Crowston & G. Herman (Eds.), *Organizing Business Knowledge: The MIT Process Handbook*. Cambridge, MA: MIT Press.

Davenport, T. H., & Short, J. E. (1990). The new industrial engineering: Information technology and business process redesign. *Sloan Management Review, 31*(4), 11–27.

Davis, F. D. (1989). Perceived usefulness, perceived ease of use and user acceptance of information technology. *MISQ, 13*, 319–340.

Davis, W. S. (1983). *Systems Analysis and Design*. Reading, MA: Addison-Wesley.

den Hengst, M., & Sol, H. G. (2002). The impact of electronic commerce on interorganizational coordination: A framework from theory applied to the container-transport industry. *International Journal Of Electronic Commerce, 6*(4), 73–91.

Englert, J., Eymann, T., Gold, S., Hummel, T., & Schoder, D. (1996). *Beyond automation: A framework for supporting cooperation* (Working paper). Freiburg, Germany: Institut für Informatik und Gesellschaft der Albert-Ludwigs-Universität.

Etcheverry, P., Lopisteguy, P., & Dagorret, P. (2001a). Pattern-based guidelines for coordination engineering. In *Database and Expert Systems Applications, Lecture Notes in Computer Science* (Vol. 2113, pp. 155-164).

Etcheverry, P., Lopisteguy, P., & Dagorret, P. (2001b). Specifying contexts for coordination patterns. In *Proceedings of Modeling and Using Context, Lecture Notes in Artificial Intelligence* (Vol. 2116, pp. 437-440).

Faraj, S., & Sproull, L. (2000). Coordinating Expertise in Software Development Teams. *Management Science, 46*(12), 1554–1568.

Freedman, D. P., & Weinberg, G. M. (1982). *Handbook of Walkthroughs, Inspections, and Technical Reviews: Evaluating Programs, Projects, and Products* (3rd ed.). Boston, MA: Little, Brown.

Frozza, R., & Alvares, L. O. (2002). Criteria for the analysis of coordination in multi-agent applications. In *Coordination Models and Languages, Proceedings, Lecture Notes in Computer Science* (Vol. 2315, pp. 158-165).

Gittell, J. (2000). Organizing work to support relational coordination. *International Journal of Human Resource Management, 11*(3), 517–534.

Grant, R. M. (1996). Toward a knowledge-based theory of the firm. *Strategic Management Journal, 17*(Winter), 109–122.

Hammer, M. (1990). Reengineering work: Don't automate, obliterate. *Harvard Business Review, 68*(July-August), 104–112.

Harrington, H. J. (1991). *Business Process Improvement: The Breakthrough Strategy for Total Quality, Productivity, and Competitiveness*. New York: McGraw-Hill.

Harrison, D. B., & Pratt, M. D. (1993). A methodology for reengineering business. *Planning Review, 21*(2), 6–11.

Holsapple, C. W., & Joshi, K. D. (2000). An investigation of factors that influence the management of knowledge in organizations. *Journal Of Strategic Information Systems*.

Holsapple, C. W., & Joshi, K. D. (2002). Knowledge management: A threefold framework. *Information Society*.

Holt, A. W. (1988). Diplans: A new language for the study and implementation of coordination. *ACM Transactions on Office Information Systems, 6*(2), 109–125.

Kaplan, A. (1964/1998). *The Conduct of Inquiry: Methodology for Behavioural Science*. New Brunswick, NJ: Transaction Publishers.

Kim, H. W. (2000). Business process versus coordination process in organizational change. *International Journal Of Flexible Manufacturing Systems*.

Klein, M., & Bernstein, A. (2004). Toward high-precision service retrieval. *IEEE Internet Computing*.

Klusch, M., Bergamaschi, S., & Petta, P. (2003). European research and development of intelligent information agents: The AgentLink perspective. In *Intelligent Information Agents, Lecture Notes in Artificial Intelligence* (Vol. 2586, pp. 1-21).

Kobayashi, T., Tamaki, M., & Komoda, N. (2003). Business process integration as a solution to the implementation of supply chain management systems. *Information & Management*.

Krieger, M., Vigder, M., Dean, J. C., & Siddiqui, M. (2003). Coordination in COTS-based development. In *Cots-Based Software Systems, Proceedings, Lecture Notes in Computer Science* (Vol. 2580, pp. 123-133).

Lawrence, P., & Lorsch, J. (1967). *Organization and Environment*. Boston, MA: Division of Research, Harvard Business School.

Lee, W. J., & Lee, K. C. (1999). PROMISE: A distributed DSS approach to coordinating production and marketing decisions. *Computers & Operations Research*.

Lizotte, S., & Chaib-draa, B. (1997). Coordination in CE systems: An approach based on the management of dependences between activities. *Concurrent Engineering—Research and Applications, 5*(4), 367-377.

Malone, T. W. (1988). *What is coordination theory?* (Working paper No. #2051-88). Cambridge, MA: MIT Sloan School of Management.

Malone, T. W., & Crowston, K. (1990). What is coordination theory and how can it help design cooperative work systems? In D. Tatar (Ed.), *Proceeding of the Third Conference on Computer-supported Cooperative Work* (pp. 357–370). Los Angeles, CA: ACM Press.

Malone, T. W., & Crowston, K. (1991). *Toward an interdisciplinary theory of coordination* (Working paper No. 120): MIT Centre for Coordination Science.

Malone, T. W., & Crowston, K. (1994). The interdisciplinary study of coordination. *Computing Surveys, 26*(1), 87–119.

Malone, T. W., Crowston, K., & Herman, G. (Eds.). (2003). *Organizing Business Knowledge: The MIT Process Handbook*. Cambridge, MA: MIT Press.

Malone, T. W., Crowston, K., Lee, J., Pentland, B., Dellarocas, C., Wyner, G., et al. (1999). Tools for inventing organizations: Toward a handbook of organizational processes. *Management Science, 43*(3), 425–443.

Marca, D. A., & McGowan, C. L. (1988). *SADT™: Structured Analysis and Design Technique*. New York: McGraw-Hill.

Martens, B. V. d. V. (2003). *Theories at Work: An Exploratory Study.* Unpublished PhD Thesis, Syracuse University, School of Information Studies, Syracuse, NY.

McChesney, I. R. (1997). Effective coordination in the software process—historical perspectives and future directions. *Software Quality Journal*.

Medema, S. G. (1996). Coase, costs and coordination. *Journal of Economic Issues, 30*(2), 571–578.

Mehring, J. S. (2000). A practical setting for experiential learning about supply chains: Siemens brief case game supply chain simulator. *Production And Operations Management*.

Ossowski, S., & Omicini, A. (2002). Coordination knowledge engineering. *Knowledge Engineering Review*.

Powers, M., Adams, D., & Mills, H. (1984). *Computer Information Systems Development: Analysis and Design*. Cincinnati, OH: South-Western.

Reezigt, C. (1995). *Zicht op interne communicatie, ontwerp van een bedrijfseconomisch georiënteerd diagnose-instrument.* Unpublished Ph.D. Thesis, University of Groningen, Groningen, Netherlands.

Ricci, A., Omicini, A., & Denti, E. (2003). Activity theory as a framework for MAS coordination. In *Engineering Societies in the Agents World III, Lecture Notes in Artificial Intelligence* (Vol. 2577, pp. 96-110).

Scheel, I. B., Hagen, K. B., & Oxman, A. D. (2002). Active sick leave for patients with back pain—All the players onside, but still no action. *Spine*.

Schmidt, K., & Simone, C. (1996). Coordination mechanisms: Towards a conceptual foundation of CSCW systems design. *Computer Supported Cooperative Work: The Journal of Collaborative Computing, 5*(2–3), 155–200.

Schnase, J. L., Kama, D. L., Tomlinson, K. L., Sanchez, J. A., Cunnius, E. L., & Morin, N. R. (1997). The Flora of North America digital library: A case study in biodiversity database publishing. *Journal Of Network And Computer Applications*.

Sikora, R., & Shaw, M. J. (1998). A multi-agent framework for the coordination and integration of information systems. *Management Science, 44*(11), 565–578.

Singh, B., & Rein, G. L. (1992). *Role Interaction Nets (RINs): A Process Definition Formalism* (Technical Report No. CT–083–92): MCC.

Thompson, J. D. (1967). *Organizations in Action: Social Science Bases of Administrative Theory*. New York: McGraw-Hill.

Tolksdorf, R. (2000). Coordination technology for workflows on the Web: Workspaces. In *Coordination Languages and Models, Proceedings, Lecture Notes in Computer Science* (Vol. 1906, pp. 36-50).

Tolksdorf, R. (2003). A dependency markup language for web services. In *Web, Web-Services, and Database Systems, Lecture Notes in Computer Science* (Vol. 2593, pp. 129-140).

van Breemen, A. J. N., & de Vries, T. J. A. (2001). Design and implementation of a room thermostat using an agent- based approach. *Control Eng. Practice, 9*(3), 233-248.

Walz, D. B., Elam, J. J., & Curtis, B. (1993). Inside a software design team: knowledge acquisition, sharing, and integration. *Communications of the ACM, 36*(10), 63–77.

Weick, K. E., & Roberts, K. (1993). Collective mind in organizations: Heedful interrelating on flight decks. *Administrative Science Quarterly, 38*(3), 357–381.

Weigand, H., van der Poll, F., & de Moor, A. (2003). Coordination through communication. In *Proceedings of the 8th International Working Conference on the Language-Action Perspective on Communication Modelling*. Tilburg, The Netherlands.

Whyte, W. F. (1948). *Human Relations in the Restaurant Industry*. New York: McGraw-Hill.

Yoshioka, T., Herman, G., Yates, J., & Orlikowski, W. (2001). Genre taxonomy: A knowledge repository of communicative actions. *ACM Trans. Inf. Syst., 19*(4), 431-456.

Figures and Tables

Table 1. Examples of common dependencies between activities and alternative coordination processes for managing them (indentations in the left column indicate more specialized versions of general dependency types). (From Malone & Crowston, 1994)

Dependency	***Examples of coordination processes for managing dependency***
Shared resources	"First come/first serve", priority order, budgets, managerial decision, market-like bidding
Task assignments	(same as for "Shared resources")
Producer / consumer relationships	
Prerequisite constraints	Notification, sequencing, tracking
Transfer	Inventory management (e.g., "Just In Time", "Economic Order Quantity")
Usability	Standardization, ask users, participatory design
Design for manufacturability	Concurrent engineering
Simultaneity constraints	Scheduling, synchronization
Task / subtask	Goal selection, task decomposition

Figure 1. The restaurant service process. Actors are shown down the left side, activities performed by each are shown in order across the page. Activities performed jointly are connected with dotted lines.

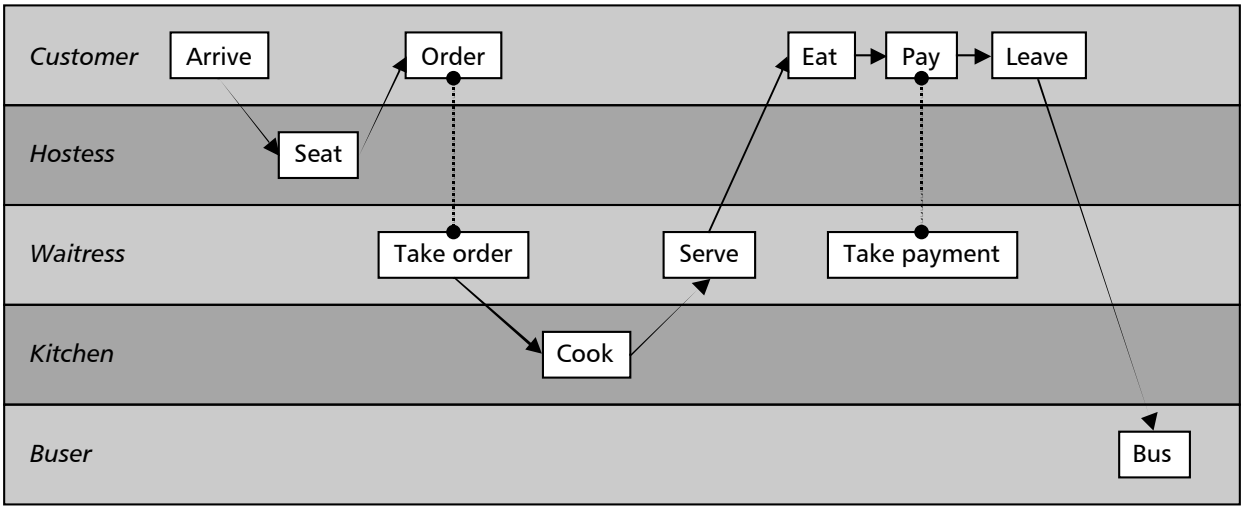

Figure 2. Flow of resources between activities and resulting dependencies in the restaurant service process.

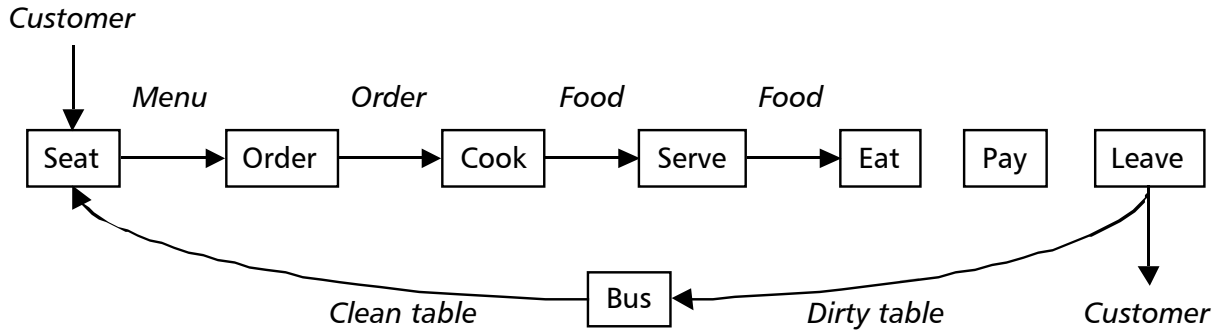

Figure 3. Model of high-level dependencies and coordination mechanisms for restaurant service.

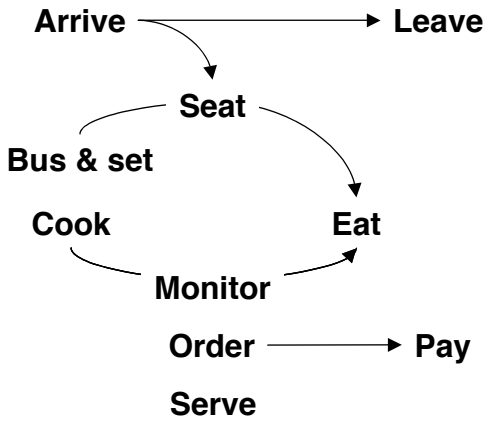

Figure 4. Dependencies between high-level tasks in software development. Solid line indicate decomposition of a task into subtasks. Dotted lines indicate a flow of resources and therefore a producer-consumer dependency.

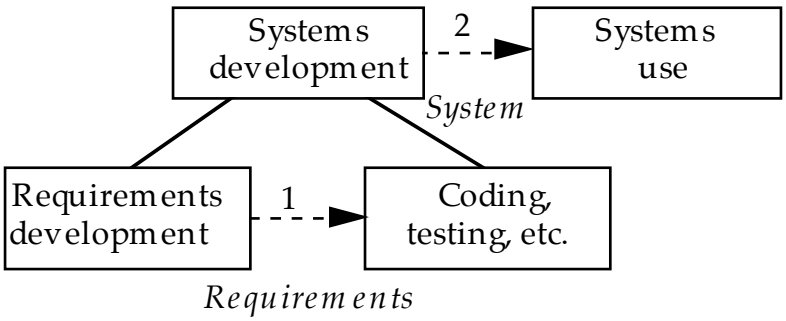

Figure 5. Counts of use of articles over time.