Multi Agent Systems in Logistics:
A Literature and State-of-the-art Review

Niels Lang, Hans M. Moonen, F. Jordan Srour and Rob A. Zuidwijk

ERIM Report Series *Research in Management*	
ERIM Report Series reference number	ERS-2008-043-LIS
Publication	July 2008
Number of pages	69
Persistent paper URL	http://hdl.handle.net/1765/12902
Email address corresponding author	rzuidwijk@rsm.nl
Address	Erasmus Research Institute of Management (ERIM) RSM Erasmus University / Erasmus School of Economics Erasmus Universiteit Rotterdam P.O.Box 1738 3000 DR Rotterdam, The Netherlands Phone: + 31 10 408 1182 Fax: + 31 10 408 9640 Email: info@erim.eur.nl Internet: www.erim.eur.nl

ERASMUS RESEARCH INSTITUTE OF MANAGEMENT

REPORT SERIES
RESEARCH IN MANAGEMENT

ABSTRACT AND KEYWORDS	
Abstract	Based on a literature survey, we aim to answer our main question: "How should we plan and execute logistics in supply chains that aim to meet today's requirements, and how can we support such planning and execution using IT?" Today's requirements in supply chains include inter-organizational collaboration and more responsive and tailored supply to meet specific demand. Enterprise systems fall short in meeting these requirements The focus of planning and execution systems should move towards an inter-enterprise and event-driven mode. Inter-organizational systems may support planning going from supporting information exchange and henceforth enable synchronized planning within the organizations towards the capability to do network planning based on available information throughout the network. We provide a framework for planning systems, constituting a rich landscape of possible configurations, where the centralized and fully decentralized approaches are two extremes. We define and discuss agent based systems and in particular multi agent systems (MAS). We emphasize the issue of the role of MAS coordination architectures, and then explain that transportation is, next to production, an important domain in which MAS can and actually are applied. However, implementation is not widespread and some implementation issues are explored. In this manner, we conclude that planning problems in transportation have characteristics that comply with the specific capabilities of agent systems. In particular, these systems are capable to deal with inter-organizational and event-driven planning settings, hence meeting today's requirements in supply chain planning and execution.
Free Keywords	supply chain, multi agent systems, MAS
Availability	
Classifications	

Multi Agent Systems in Logistics

A literature and state-of-the-art review

Niels Lang, Hans M. Moonen, F. Jordan Srour, Rob A. Zuidwijk[*]
RSM Erasmus University, Rotterdam, The Netherlands

Abstract

Based on a literature survey, we aim to answer our main question: “How should we plan and execute logistics in supply chains that aim to meet today’s requirements, and how can we support such planning and execution using IT?” Today’s requirements in supply chains include inter-organizational collaboration and more responsive and tailored supply to meet specific demand. Enterprise systems fall short in meeting these requirements The focus of planning and execution systems should move towards an inter-enterprise and event-driven mode. Inter-organizational systems may support planning going from supporting information exchange and henceforth enable synchronized planning within the organizations towards the capability to do network planning based on available information throughout the network. We provide a framework for planning systems, constituting a rich landscape of possible configurations, where the centralized and fully decentralized approaches are two extremes. We define and discuss agent based systems and in particular multi agent systems (MAS). We emphasize the issue of the role of MAS coordination architectures, and then explain that transportation is, next to production, an important domain in which MAS can and actually are applied. However, implementation is not widespread and some implementation issues are explored. In this manner, we conclude that planning problems in transportation have characteristics that comply with the specific capabilities of agent systems. In particular, these systems are capable to deal with inter-organizational and event-driven planning settings, hence meeting today’s requirements in supply chain planning and execution.

---

[*] Corresponding author: rzuidwijk@rsm.nl, RSM Erasmus University, room T9-20, PO Box 1738, 3000 DR Rotterdam, Tel. +31 10 408 2235, Fax. +31 10 408 9010

1 Introduction

This paper reviews contributions from the scientific literature, professional journals, and other sources such as the internet, and provides a discussion on the state of the art in the use of agent technologies in logistics planning and execution.

The review is triggered by our main question: “How should we plan and execute logistics in supply chains that aim to meet today’s requirements, and how can we support such planning and execution using IT?” To answer this question, we first need to understand what we mean by logistics planning and execution, which challenges supply chains are facing today, and how we could design planning and execution processes to meet the challenges. These issues are considered in Chapter 2 and this chapter provides a motivation for our research.

Secondly, we need to understand how IT can support planning and execution that meets the challenges in modern supply chains. We first explore in Chapter 3 how IT has been used to provide such support the last few decades, introduce new developments that challenge the traditional approach, and then discuss how advanced types of inter-organizational system could meet the challenge. We provide a detailed analysis of inter-organizational systems to achieve this.

In Chapter 4, we focus on a particular aspect of planning and execution, namely the distribution of information and decision capabilities. As such, we focus on some design characteristics of how to support planning and execution using IT, i.e., some design characteristics of inter-organizational systems. In particular, we contrast centralized versus decentralized (or distributed) decision making. We classify planning systems according to information generation and use (availability and scope), and decision making aspects (distribution and objectives) and discuss in more detail relevant aspects of distributed decision making.

In Chapter 5 we bring forward agent based systems as an important class of IT systems that support distributed decision making. We explain what software agents are, discuss Multi Agent Systems (MAS), and elaborate on relevant coordination structures: organizational structuring, contracting, and multi-agent planning.

Our discussion in Chapter 6 provides a state of the art in applications of multi agent systems in transport. Next to production, transportation is a supply chain process in which agent technologies have actually been implemented. We explain why the application of agent systems in transportation is promising, and provide some successful implementations. We do observe that adoption of MAS in supply chains is limited and discuss some factors that hinder further adoption.

Finally, we draw conclusions from the discussion outlined above. We provide an answer to the main question where we focus on the role of agent technologies.

2 Planning and Execution in Logistics

Companies are continuously making logistical decisions at different organizational levels and with different time horizons as Table 2-1 illustrates. The table shows the different decision levels that are present, objectives that drive decisions, the corresponding time horizon, and typical decisions. Planning takes place at the strategic, tactical, and operational level and basically involves decisions that prepare the logistics system for execution. At the execution level, replanning may be necessary in order to deal with specific outcomes of previously unknown factors or unexpected events, in short, both planning and execution needs to deal with task uncertainty.

Decision level	Objectives	Time horizon	Example decisions
Strategic	Costs of building and owning assets; match of (global) demand and supply; meeting mission requirements	Less than once a year	Design of distribution network; determining which plants, distribution centers, and lanes to open or close; layout of facilities; equipment purchasing
Tactical	Customer service; supply-chain costs and performance	Yearly	Planning procurement, manufacturing, and transportation; operational strategy under expected conditions with current facilities
Operational	Customer service; equipment utilization; transportation costs	Monthly, Weekly	Aggregated production plan combined with demand forecasts to derive master production plans; deployment of inventories; transportation planning
Execution	Fulfillment	Daily or real-time	Detailed decision making about timing and sequencing of activities; real-time (re)scheduling of production and transportation

Table 2-1: Overview of decision levels of logistics decision making (adapted from Raman, 1995; Sodhi, 2001)

2.1 Uncertainty in supply chains

Task uncertainty is one the main factors that complicates planning. In the early 1970s, Galbraith (1974) identified two strategies to reduce task uncertainty in business processes: one could either reduce the need for information processing, or increase the capacity to process information. Raman (1995) illustrated well that most logistical systems over the years have been focused on reducing the need for information processing, by listing the following mechanisms as instruments to deal with uncertainty in logistics decision making: (1) forecasting, (2) elimination, (3) mitigation, and (4) recovery.

A specific cause of uncertainty in supply chains is lack of information sharing between supply chain partners. Moreover, suboptimal decisions arise when decisions in the supply chain are not coordinated. A textbook example that clarifies why decisions in supply chains should not solely be taken in isolation is the “beergame”, a management game which illustrates the so-called “bullwhip effect”. The bullwhip effect is the effect of information distortions and delays between links in a supply chain. The true market needs are not communicated upstream throughout the supply chain, and as a matter of fact – due to ordering policies and reviews – distortions and therewith amplifications of distortions move upstream too; parties upstream have no clue of what is happening in the end market. This is nicely documented by Lee (1997). Playing the MIT beergame can be recommended to all, for a better understanding of the effects – as they are naturally to occur, even with a group of people knowing its principles.

The alternative as Galbraith (1974) identified is increased information processing through information coupling within supply chains; which is a way to reduce uncertainty and to react in real-time on actual events instead of sole anticipation on possible events one could forecast or expect. Sriram (2000) made a similar observation by stating that businesses operate in highly competitive environments, and therefore need a great deal of up-to-date information. As Sriram makes clear, a single corporation can neither collect all this data nor operate without cooperation of its trading partners. Investing in inter-organizational linkages is therefore a recommended next step for many organizations and their supply chains.

2.2 Developments in SCM to meet the challenge

The field of logistics and SCM is rapidly changing. The inter-organizational chain management of supply chains increases in importance. However, that is not all as Kopczak (2003) and Lee (2004) illustrate.

SHIFT	OLD QUESTION	NEW QUESTION
Shift No. 1: From cross-functional integration to cross-enterprise, too	How do we get the various functional areas of our company to work together to supply product to our immediate customers?	How do we coordinate activities across companies, as well as across internal functions, to supply product to the market?
Shift No. 2: From Physical Efficiency to Market Mediation	How do we minimize the costs our company incurs in production and distribution of our products?	How do we minimize the costs of matching supply and demand while continuing to reduce the costs of production and distribution?
Shift No. 3: From Supply Focus to Demand Focus	How can we improve the way we supply product in order to match supply and demand better, given the demand pattern?	How can we get earlier demand information or affect the demand pattern to match supply and demand?
Shift No. 4: From Single-Company Product Design to Collaborative, Concurrent Product, Process and Supply-Chain Design	How should our company design products to minimize product cost (our cost of materials, production and distribution)?	How should collaborators design the product, process and supply chain to minimize costs?
Shift No. 5: From Cost Reduction to Breakthrough Business Models	How can we reduce our company's production and distribution costs?	What new supply-chain and marketing approach would lead to a breakthrough in customer value?
Shift No. 6: From Mass-Market Supply to Tailored Offerings	How should we organize our company's operations to serve the mass market efficiently while offering customized products?	How should we organize the supply chain to serve each customer or segment uniquely and provide a tailored customer experience?

Table 2-2: Shifts in SCM and logistics thinking (Kopczak, 2003)

Kopczak (2003) discusses six major shifts in business focus brought about by supply chain management thinking over the past decade which we listed in Table 2-2. The changing shifts are well in line with the “Triple A” that drives developments in supply chains as introduced in (Lee, 2004): Agility (respond to short-term changes), Adaptability (adjust SC design to structural shifts), and Alignment (create incentives for better performance). The major differences as Kopczak identifies them are increased utilization of information throughout decision making processes, an inter-enterprise focus, and a true customer focus. She states that SCM moves to the core of today’s enterprising: it is no longer just some logistics decisions to be made. It has far more reaching impact: it truly deals about the design of one’s own organization, wider supply chain and inter-organizational operations. Lee (2004) illustrates this when he discusses the impact VMI (vendor managed inventory) has on supply chain structuring. He states that VMI is not just about information sharing, or even redistributed responsibilities, but also concerns the ownership question: when do products change owner?

As combinatorial problems associated with logistics planning are very complex, one may not expect to obtain solutions that are optimal within reasonable time, especially when planning is required real-time. Therefore, one utilizes different types of heuristics, which are expected to approach optimality. Sodhi (2001) demonstrated that most of the Advanced Planning and Scheduling (APS) software packages (in the earlier 2000s) relied on (combinations of) smart heuristics.

Figure 2-1 gives an overview of different perspectives on logistics decision models which can be found in the literature. The left-hand side of the picture shows the control system perspective (Churchman et al. 1957; Leeuw 1974), which allows independent analysis of management systems (generating task related decisions) and transformation systems (actually performing logistic tasks). The center figure presents an example of such a logistic control system for Automated Guided Vehicles (AGVs) operating on a grid infrastructure which deliver manufactured products from multiple points of production (labeled ‘P’) to multiple points of consumption (labeled ‘C’). In this example, the management system (here: an AGV controller) performs detailed control, which means that it not only decides on job allocation, but also on AGV movements. In such a setting, the AGV controller decides on direction and speed properties for all AGVs involved. Its information input would include state information of all AGVs and, of course, information on transport jobs to be performed by the system.

The right-hand side of Figure 2-1 illustrates the perspective of model-based control. The decision model is derived from the control problem, especially its objective, decision variables and relevant constraints. Based on any given complete decision set, it predicts outcomes in terms of the objective function. A decision model can be associated with a solution procedure in order to locate an optimal decision set, i.e. a decision set optimizing the model's objective function. Alternatively, solution procedures improving rather than optimizing decisions may be employed, e.g. because an optimization procedure is time consuming or simply unknown. Finally, a decision maker (DM) specifies the model, initiates the solution procedure and decides based on the outcomes of the model.

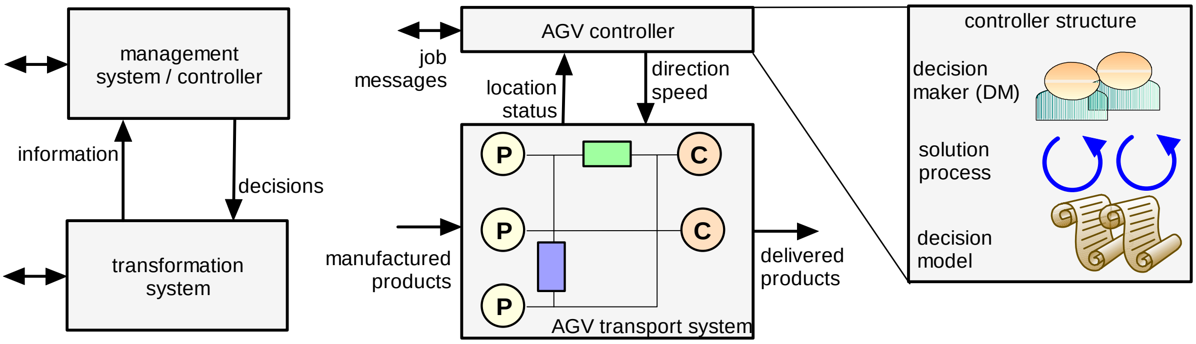


Figure 2-1: Different system perspectives on logistic control. Control system perspective (left), an Instantiated logistic system (center), and a Controller structure (right).

Is the automation of a logistical process an optimization problem as is the focus in the example above, or a communication problem? And what is the role of technology herein? There is no straightforward answer to these questions. Orman (2002) describes the different impacts communication technologies and information processing technologies have on organizational structures: *communication technologies* may lead to decentralization of decision making because they reduce the cost of communication and coordination, and allow decisions to be delegated to better informed and better monitored lower level managers. Conversely, *information processing technologies* may lead to more centralization of decision making because they increase the information processing capacity of managers, extend their reach, and allow them to usurp more power at the top. As operational decision tasks are usually of a recurring kind, the decision making processes are often amenable to automation. Figure 2-1 emphasizes the possibility of a management system being composed of multiple models, procedures and decision makers. In the example, two Decision

Makers (DMs) may control different kind of AGVs. In such cases, coordination between DMs and decision models becomes a relevant design issue. Therefore, let us now look at how IT applications support planning and execution in supply chains in more detail.

3 IT support in planning and execution

The role of IT in supply chain management is that of an enabler (Sridharan, 2005). The primary goal of IT in the supply chain is to link the point of production seamlessly with the point of delivery and/or purchase. The idea is to have an information trail that follows (or better: precedes) the product's physical trail. This allows planning, tracking and estimating lead times based on real-time data.

3.1 Exploitative and explorative use of IT

Enterprise software can be deployed in two fundamentally different ways: *exploitative* or *explorative* – as Subramani (2004) and McFarlane (2003) make clear. Exploitation refers to the class of actions to improve operational efficiency, basically through the automation of existing processes – for example software that helps in (Buxmann, 2004) reduction of production costs; reduction of lead time; reduction of inventory and shortfall; reduction of transportation costs; improvement of supplier evaluation and selection; improvement of service levels; improvement of cooperation. Exploration refers to the pursuit of new possibilities, or entire new SCM practices. Basic focus herein is not automating, but informating (of personnel responsible for the decision making). The same difference is made by Singh et al. (2007): in their description of the SCM software field this is one of the axes in their framework: (1) nature of use, i.e., informational versus decision making, (2) nature of process, i.e., less formalized versus highly formalized, and (3) nature of supply chain technology, i.e., highly specific versus less specific. With respect to exploitative actions, Davenport (2005) states that automated decision making has come of age in organizations – especially for decisions that must be made frequently and rapidly, using information that is available in an electronic form. The knowledge and decision criteria need to be highly structured and the factors taken into account must be well understood. Automated decision making has impacted several fields: solution configuration, yield optimization, routing decisions, corporate compliance, fraud detection, dynamic forecasting and operational control. An important factor that drives adoption: widespread availability of data throughout industries – the more data that exists, the greater potential for automating. Developments such as the internet, but also the widespread adoption of tracking and tracing technologies such as RFID contribute to the latter.

3.2 Historical development of IT

The Second World War aggressively accelerated the development of computing and computing technology. Computers were first developed and deployed for military purposes, such as calculations on artillery firing tables and the design of the bomb. Computing hit the business stage a decade later, in the early 1950s. This was also the time the Operations Research (OR) field emerged (Mahoney, 1988). The power computers revealed made it possible to solve the complex mathematics OR brought about. Over the years, governmental spending kept on dominating and driving developments in computer systems: microelectronics, interactive real-time systems, artificial intelligence and modern software engineering would not have been where they are today without large (e.g. US) government spending on military applications and the space program (Mahoney, 1988).

The pace of change in enterprise information systems application in industry is a paradoxical one. On the one hand developments in hardware and software in general seem to go at rocketspeed – see for example Coltman's (2001) description of Internet adoption pace; technology matures, prices go down, new more user-friendly tools are introduced continuously. What is state-of-art today is ready for the museum tomorrow. As Milojicic (2004) put it: "*What you have on your desk now, is more powerful than all power of the world's supercomputers together some 30 years ago. Imagine what another 30 years will bring us?*" Thus, the general impression many have is that there is continuous change, and new technologies may appear overnight. On the other hand, when one looks at the true underlying processes, and the systems that support these, one could come to an opposite observation: fundamental change, even in (enterprise) information systems develops slowly; it takes long to grow visions into realities, and establish enterprise systems that can do what was envisioned before. Real-time systems, for example, by many perceived as something from the last few years, have been reported about as early as 1970 already (Zani, 1970). He concludes that not every industry has a need for real-time systems. Second, it was found that the added value of real-time systems is only tapped into when meaningfully integrated into a well-designed management steering process. Haigh (2001) observes that the Management Information Systems as envisioned in the 1960s were only first realized to a (still) limited extend in the early 1990s with the introduction of enterprise-wide ERP systems. ERP systems are software packages that integrate financial operations, human resources, sales and logistics on a global scale. Even more extreme, the LEO, which is recognized (Glass, 2005) as the first business software application, first booted in 1951. LEO was deployed for the elaboration of daily orders which were phoned in every afternoon by the shops and used to calculate the overnight production requirements, assembly instructions, delivery schedules, invoices, costing and management reports. This, arguably, could be the first instance of an integrated management information system, or decision support systems, plus a computerized call center. Although technology might have accelerated at rocket speed ever since, many of today's system implementations still aim at achieving similar objectives as LEO first delivered in 1951. The observation we do is supported by Glass (2003), who describes that although hardware has developed in a fast pace over the past 50 years, software (and especially also the way we build software) is still similar to the things we had in the 1950s. Languages have changed, but the underlying structures have not; which of course does not mean it is bad, but does tell something about the pace of change in systems development.

Undeniable has IT fundamentally changed the way we conduct business. Farrell (2003) explains the scaling effect in IT investments. Once you have installed software for transaction processing, the marginal costs of processing additional transactions falls rapidly towards zero. The same holds for software: once it is developed, it can be sold over and over again. Porter (2001) and Carr (2003) reasoned that IT does no longer matter for organizations: it has become nothing than a commodity. A must-have commodity, and exactly therefore the competitive extra value of it has decreased: best practices are easy to copy – you simply buy the same system. We however do think that this situation has not yet been reached. We find support in the observation by Clemons and Row (1991): the strategic advantage of well-developed and implemented IT lays in its ability to tap into "unique resources" of the innovating firm, so that competitors do not fully benefit from imitation. Siau et al. (2003) shows that that observation still holds, they conclude that although late(r) adopters can benefit from copying technologies, they generally do lack the innovative mindsets that would keep them in front of the pack. Ordanini (2005) similarly states that the strategic benefits of experimenting with innovative B2B technologies is especially in the innovative atmosphere it creates in a company, that is something which is very hard to copy.

The first computing applications in business were introduced in the 1950s and 1960s, and were mainly used for simple calculations and data storage. When hard- and software capabilities evolved Material Requirements Planning (MRP) and Manufacturing Resource Planning (MRP-II) applications became available in the 1970s (Busschbach, 2002). These applications helped the business need for better-coordinated (internal) material flows. Systems evolution continued and resulted in the late 1980s in the first Enterprise Resource Planning (ERP) packages. These systems were mainly designed to solve the fragmentation of information in large business organizations, as Davenport (1998) explains. ERP’s are generic systems: designed with “best practices” in mind. Best is however (software) vendor defined. Customizing an ERP (changing internal code, or interface legacy system) adds complexity, costs, and complicates future upgrades and integration with business partners. Hagel (2001) states that ERP systems brought solutions to some problems, but they also created new problems such as a lock-in to rigid business processes.

Nowadays, the enterprise software landscape is more-and-more changing towards inter-organizational supply chain systems. Hagel (2001) explains that that is a logical change, since that is where the limitations of existing IT architectures are most apparent and onerous; applications on the edge of one’s enterprise can benefit by definition from sharing. Globalization has under while led to networked organizations (Heng, 2003), which in turn need electronic linkages and electronic forms to conduct business. Wortmann et al. (2001) state that the monolithic nature of current ERP systems shows that these systems are in essence designed for individual enterprises, and not for supply chains – see also (Davenport, 2004). The most examples of successful integration efforts are related to collaborative systems within one single enterprise; integration between enterprises lags behind (Wortmann, 2001). Davenport (2004) expects a complete transformation of enterprise systems into inter-enterprise systems to take a long path; it is a transformation which “*has to be measured in years or even decades*”.

Another important dimension which is changing, next to the “scope” as discussed above, is the factor “time”. The trend is to move enterprise system support more into the real-time domain. Where ERP systems were designed around an optimization engine that typically ran once a day, companies now start requiring real-time support solutions. Most ERP systems typically have their optimization run during the night, since that hinders daily operations the least. With the advent of information nowadays being available everywhere at everytime, this is something new system architectures have to incorporate. Consider for example the changes brought about by RFID (or RFIT – see (Bose, 2005)) technology: sensors are everywhere, instant updates follow, and chains now can be controlled intelligently in real-time. The example of Continental Airlines (Anderson-Lehman, 2004) describes the use of real-time business intelligence techniques. Continental continuously monitors what is happening and adjusts its systems accordingly – huge savings have been achieved in areas such as marketing, fraud detection, demand forecasting and tracking, and improved data center management. Sridharan (2005) points out that nowadays “clear communications and quick responses to those communications are key elements of successful supply chain management”. Worley et al. (2002) pinpoint at the need for a new smart generation of workflow systems that provide real-time decision support. Less of a scientific proof, but nevertheless an important signal, is that the technology evangelists from the Gartner Group have been speaking about nothing else than the real-time enterprise, business activity monitoring, supply chain event management, dynamic applications, et cetera since the beginning of the third millennium.

Following the discussion above, we think we should give a clustering of different generations of enterprise systems. Evernden (2003) presented a division in three generations of enterprise systems. Following the research we did, and visions we developed, we think we have ground to add a fourth generation – in which the above discussed factors inter-organizational and real-time are of great importance. The generations are presented in Table 3-1.

Generation	Focus	Driven By	Content
1st Generation 1970s and 1980s	Systems as standalone applications within individual organizations.	Increasing functionality and sophistication of standalone applications.	Explanation of the need for an architectural approach; Analogies with building architecture; Simple 2D diagrams or frameworks providing overviews of the architecture.

2nd Generation 1990s	Systems as integrated sets of components within individual organizations.	Growth in system complexity and interdependence; Demand for software reuse.	Extension and adaptation of diagrams from 1st-generation architectures; Population of frameworks with industry reference models.
3rd Generation late 1990s and 2000s	Information as corporate resource with supporting IT tools and techniques.	Emergence of the Internet, e-commerce, and an increase in business-to-business applications; Growing interdependence among organizations; Adoption of knowledge management, systems thinking, and a more holistic view of information as a resource.	Explicit definition of principles and background theory Development of multidimensional architectures; Customization of information frameworks to the needs of individual organizations; Generic information patterns and maps.
4th Generation late 2000s and 2010s	Information is everywhere, and the true business lays in connectivity and tapping in real-time into distributed sources.	Enterprises have their in-house systems in place, and search for the next frontier (often: inter-organizational); New Internet enabled standards, such as WebServices enable instant low-cost connectivity.	Architectures for connectivity. Information systems have to overcome inter-organizational hurdles such as trust, gain sharing, but also different technological legacies. Sense and response technologies become central parts of solutions.

Table 3-1: Four generations of enterprise information systems (adapted from Evernden, 2003)

3.3 Two changing dimensions in enterprise systems

Following our discussion above we recognize two major shifts in current thinking about enterprise software systems, and the supply chain operations they are supporting. One dimension of change is the shift in systems thinking from a pure intra-enterprise perspective to a inter-enterprise perspective. Where the first decades of automation took the enterprise at its center – systems were primarily build to support one company’s operations – whereas today we recognize the inter-enterprise perspective as core domain of extension. A domain where automation can help smooth out deficiencies, enable smoother supply chains, and cut out large amounts of unnecessary costs – such as for example unnecessary inventory cost, unsynchronized activities, low performance, et cetera. As Dedrick (2005) illustrated it: although companies over the past decade have been focusing on their core competences and invested heavily in internal information systems, they now realize that just the inter-organizational perspective that IOS offer are (future) crucial parts of their systems: an IOS can signal earlier what is happening, and help in synchronizing supply and demand, and therewith help in better managing demand in response to production capacity.

The other dimension of change we find is the planning domain; which has for long been centered on batch-wise planning approaches: a large complex optimization engine which takes a long time to run. ERP systems are a typical example of these: during daytime all transactions are inserted, and in most instances, the optimization run (known as the “ERP run”) happens at night – when no one is accessing the system. Runtime planning capabilities are generally missing in ERP’s. APS (Advanced Planning & Scheduling) systems partially solve this problem, but are generally still not capable of real-time planning support: their architectures are still very much batch-wise focused. In many domains we recognize an increased need to handle real-time events – and many of those domains are domains where “traditional automation” never gained ground, since these systems were simply not capable of handling the dynamics in such domains. For example in the transportation domain, still many smaller firms do their planning manually in real-time, for the simple reason that they were not yet able to purchase a system which totally fits their need. As Gendreau et al. (2004) show, this has gained attention from the OR community now. Real-time fleet management is an exemplary domain which is increasingly studied, for its dynamic behavior and the solutions which are needed in that market domain. Especially with all data which is nowadays available, about current vehicle locations, order statuses, customer requirements and traffic conditions one could consider bringing this in real-time into decision making processes. The paper of Fleischmann et al. (2004) is an example of this, in which they show how to utilize real-time traffic information in an order assignment process. Thus, increasingly scholars and business community are adopting Galbraith’s advice from the early 70s (Galbraith, 1974) to increase the capacity to process information, instead of reducing the need to process information (through the means of forecasting models, et cetera).

These two factors combined plead for a totally different breed of enterprise systems. We expect enterprise systems to become both real-time as well as inter-organizational in the decades that follow – this situation is depicted in Figure 3-1. The lines and numbers present three different growth paths system evolution could follow. It is of course hard to foresee how change will really evolve, but we do not expect trajectory (1) to be a very logical one. In this scenario a batch-wise intra-enterprise system transforms itself without immediate steps into a real-time inter-organizational system; basically meaning a scenario of radical change, in which one architecture is to be replaced by another one. In practice, hardly any company dares to make such a transformation, since it would be far too radical, and as the experiences from large enterprise software implementations have learned us, enterprises are not willing to run the risk of shutting down properly running systems before knowing that all will work out fine for them. Trajectory (2) is one that first brings in the inter-organizational aspect, and then moves on over the real-time axis. Although practice has shown several good examples of inter-organizational systems working with batch-wise engines – think about EDI and XML integrations for example – we are not aware of any system that transformed from that state to a state to real-time.

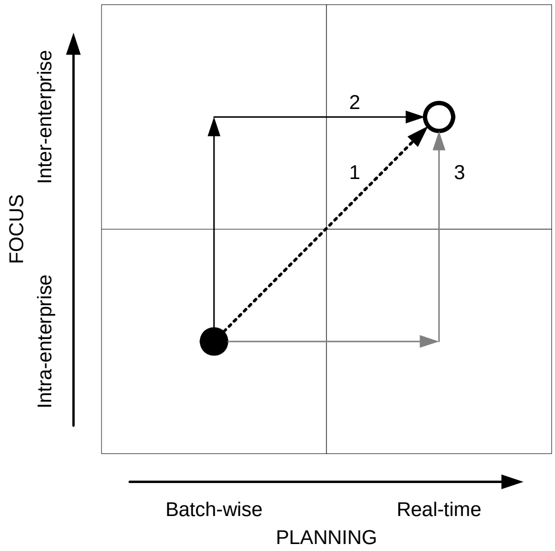


Figure 3-1: Framework of change in Enterprise Systems

Therefore trajectory (3) is perhaps most logical: first increase a system’s real-time character, and then bring in the inter-organizational component. This type of systems handle more about (real-time) communication than pure optimization, and this communication can be extended relatively easily to elements outside of one’s own enterprise system.

3.4 Economics of Inter-organizational systems

Uncertainty reduction in supply chains can be achieved by information sharing throughout the supply chain: the entire chain could than for example work with real (end-) market demands, is signaled when deviations occur elsewhere in the chain, and can save enormously on internal uncertainty reducing activities (such as extra safety stock keeping, quick response activities, et cetera). Inter-organizational systems can help in realizing this potential (Sheombar, 1997). "*A typical inter-organizational system (IOS) is an information system that links one or more firms to their customers or their suppliers and facilitates the exchange of products and services*" (Bakos, 1991). As this definition tells us, the field of IOS is wider than only information exchange between supply chain partners – it can also mean the deployment of information systems that help in collaborative product development, process control or other knowledge sharing. As Kulmala (2005) found out, most of the larger firms have some type of inter-organizational process and system in place nowadays. Bowersox (2003) explains that it is easiest for companies that start working with their chain partners to first start with the sharing of operational information – IOS' with this purpose can be found throughout industries in large numbers – tactical or strategic integration is way more complex to achieve, since often processes need to change dramatically, and trust between partners becomes a true issue. As Gosain (2005) however makes clear, the focus should never solely be on information sharing persé – pure focusing on the quantity of information exchanged – often improving the quality of the information would be a greater benefit. For an interesting other split in different typologies of IOS we would like to refer the reader to (Hong, 2002). Hong makes a split over two axis: role linkage (horizontal versus vertical) and system support level (operational support versus strategic support). Per typology the IOS to use has different targets and different objectives.

Back in the eighties, Malone et al. (1987) reflected on the impact information technologies were going to have on industrial structures. Analyzing a wide range of economic theory, they make a split between markets and hierarchies. They expect an overall shift towards a proportionately larger use of markets—rather than hierarchies—to coordinate economic activity. Bakos (1997) explains how electronic markets might decrease information-asymmetry by reducing search costs. This is especially negative for the sellers; their profits decline gradually as search costs are reduced. It therefore is clear that electronic markets have a different impact on parties within supply chains, and on buyer/supplier selection processes. The past two decades did however not yet show a massive market uptake of market-type technology in the business-to-business domain. On the contrary: some of the larger success stories come from hierarchical systems. Dell, for example, is a typical hierarchical supply chain master (Li, 2006). Dell shares (demand forecast) information with many of its (top) suppliers, but it also passes on data about its defect rates, engineering changes, and product enhancements. Studying this case of Dell, Li et al. (2006) wonder whether companies can truly benefit from frequent supplier changes made possible through electronic markets – these might result in lower prices, but what about the overall quality, and unique competitive features? Is not one of the large advantages of IOS that these systems imply a level of cooperation and coordination well beyond that of the traditional arms-length relationship that exists between organizations acting as free-agents in markets, as Kumar et al. (1996) states it? Rai (2006) adds to this discussion by stating that established supply chain integration with partners through an IOS is a very important strategic weapon, not at least since it is very difficult to copy. And it is not solely about information systems, it is foremost a human activity system which is subject to all risks and foibles of joint human endeavor (Kumar et al., 1996). In their review on transportation exchanges in the late nineties, Alt and Klein (1998) confirmed many of the theories discussed above as factors leading to the limited success. They make clear that the markets never achieved liquidity, for the simple reason that too often only cargo was presented that could not have been sold through traditional channels (the so-called “shit-loads”). The exchanges did offer loads, but little value-added and process support. An as important conclusion is that many companies, especially also the larger ones, benefited well from the information-asymmetry. The desired “network-effect” was hardly realized, except for TeleRoute in France and COMIS in Germany. Last but not least they pinpointed at the expensive proprietary technologies which were needed. France showed the largest success, which could then be claimed to the Minitel backbone which was all-around at that time. Van der Heijden et al. (1995) add to the economic discussion on the benefits of inter-organizational systems by their interesting observation which combines IOS literature with agency-theory. They state that EDI can on the one hand lead to dyads of companies controlling each other purely based on outcomes, less on internal processes. On the other hand they observe that providing perfect information in chains can also be a major struggling point. It can cause opportunistic behaviour among trading partners – which it was never intended to be.

The situation in the transportation of physical goods does however have several characteristics of a pure market: the prime decision variable for shippers is still cost, although logistics strategies emphasize the need for reliability and service.

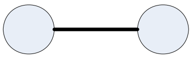
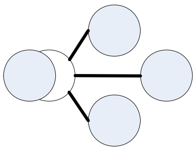
Alt and Klein (1998) identify this as one the important causes for today's inefficiencies in the transportation of physical goods, along with the lacking integral management and widely available information. Makukha (2004) comes to similar findings as they conclude that "*shippers tend to avoid close integration with logistics service providers (LSP), whereas LSP' claim to be true strategic partners but remain unable to provide the service required*". The article concludes that LSPs should thrive towards long term strategic relationships with their clients. This is currently not common practice, and as a result shippers switch LSP purely cost focused with little systems integration. Alt and Klein (1998) state that factors as diverse as time-definite customer demands, low reliability, ecological problems, congested infrastructures, and suboptimal utilization of current capacities lead to a situation where the design of information flows should be expected to dominate the design of the physical flows. This in turn also pleads for integrated systems and chain collaboration.

Despite the fact that there is huge potential for companies in coordinating activities with suppliers and customers it is very complex (Raman, 1995). Namely, a high level of understanding and trust between parties is needed. An industry survey in Sweden in the early 2000s revealed that the state-of-art in IOS for many SME's was still the phone and fax-machine (Stefansson, 2002). The author identifies expensive technology as one of the core problems, and addresses the need for less expensive methods for data-sharing in supply chains to make it possible for SMEs to participate. Bharati and Chaudhury (2006) reported similar results: their results show that SME's nowadays have little SCM software implemented. The systems they have implemented are generally internal / inhouse focused, such as finance, design and internal planning. External integration is generally lacking; in those cases where it is present, it is mainly driven from the SME's customers (which are often bigger firms).

The first decades of inter-organizational systems were mainly driven by EDI (electronic data interchange) technologies. Implementing and operating these technologies is however rather expensive: high costs for implementation, and furthermore high costs for operation, since EDI uses a dedicated phone based network, with third-party operators providing the connection. Implementation costs exist due to the fact that EDI integration often requires many (consulting) hours in order to change systems and processes, and to standardize and agree on the interaction – it is therefore a time consuming and thus costly process. With the introduction of Internet-based technologies, such as XML based integration technologies, this picture has changed to some extent. Operating costs are lower since standard network infrastructures can be used; however, implementation costs are still present. As a result, the amount of information linkages in chains is still relatively limited nowadays, and has generally been implemented solely to link large trading partners. Both Sriram (2000) and Mukhopadhyay (2002) came to the observation that forced-implementations of EDI showed better results than those that started voluntary. An example of a forced-implementation is a (large) customer forcing a supplier to establish an information link (EDI or XML based) for transactions. Fortunately, as Mukhopadhyay (2002) observes, this often results in a significant amount of additional business between those partners – which thus should plead for actively stimulating investments in information linkages when partners are motivated.

3.5 *Transactional backbone of inter-organizational systems*

With transactional backbone we refer to the technology layer which enables inter-organizational activities through connecting two or more geographically-spread applications with each other which do not belong to the same organization. Were it within one organization, we would speak about enterprise application integration, or EAI. From an architectural point a division in four different architectures can be made, as depicted in Table 3-2.

	Architectural Type	Explanation
	Bilateral (1:1)	– Point-to-point (P2P) connectivity between separate systems; – Direct connection between two trading partners; – Connectivity in its most basic form. – Works well for important partnerships. – Too costly to connect P2P with smaller (in trading) partners.
	Private hub (1:N)	– Hub structure that makes it possible to connect to partners. – Internal applications need only one connection point. – Standardized access for external partners. – Helps to reduce amount of linkages to establish.

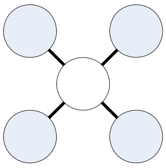
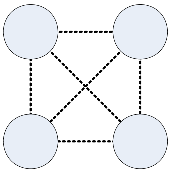
		– Generally initiated by a strong party, to link with many smaller parties
	Central orchestration hub (N:M or N:1:M)	– Like a private hub; but generally run by independent operator. – Focus on supply chain orchestration. – Process focused. – Expected to work best in industries without dominant parties.
	Modular distributed plug & play architecture (N:M)	– No permanent linkages – plug & connect capabilities. – Parties connect when interaction needed, exchange information and conduct business. – Standardization very important. – Processes are leading mechanism.

Table 3-2: Architectural types of inter-organizational systems

The first type of inter-organizational integrations that appeared were of the bilateral type; enabling one-to-one connections. This type of integration is relatively cheap – since no intermediaries are needed, and the two parties can design their own message format – and easy to implement. It works well for establishing connections between large parties, with many transaction exchanges.

However, when scaling up a problem arises. In order to connect *n* parties with each other, one needs (*n* * (*n*-1)) / 2 connections. This number of connections explodes literally: in case of 10 parties, there is a need for 45 point-to-point connections, and when 20 parties want to connect with each other there is a need for 190 connections already. Hubs appeared as the solution to this problem. Each party connects to the hub, and a connection with another party is established through the hub. This way significantly less connections have to be constructed; to fulfill the connectivity question for 10 or 20 parties, one just needs 10 or 20 connections to the hub.

In practice there are two different kinds of hubs: Independent central orchestration hubs, which are hubs that do not belong to any of the (traditional) parties in the network. This hub is thus operated through an independent third party and does purely bring connectivity. The second type concerns private hubs that are owned by a (generally large) party to connect to the outside world. The latter type is of a one-to-many type, and not many-to-many as the central orchestration hubs. Private hubs can especially been found in industries with strong supply chain players, who dominate their up- or downstream supply chain partners. A well known example is the private hub that Cisco constructed – see (Grosvenor, 2001) and (Edwards, 2001). The boom days of the Internet gave rise to many hub-typed e-marketplaces (Markus, 2000) of different categories. The technology was relatively cheap, but too often perceived as the only missing piece in the puzzle. These marketplaces often lacked process support, commitment from its members, and sometimes even a real thought-true business case.

The last architectural type discussed here is referred to as the modular distributed plug & play architecture. Not yet a truly established category, but a collection of initiatives and developments that thrive towards a new trend. This architectural type is being capable to realize fast connect (and disconnect) capabilities within a supply chain, where system integration is not a matter of months of hard work, but more the result of a single mouse-click. Web Services technologies which are nowadays heavily pushed by technology providers have an included technology named UDDI (Universal Description, Discovery, and Integration). UDDI is aimed at the creation of an online discovery system for seamless connection between two different parties – which do not yet have to know each other. In practice, however, such mechanisms are not yet included in daily business practice, since IT integration is always the result of a management decision. More to the extreme, much research is currently devoted towards semantic web technologies and mechanisms, aiming at understanding the content and context of messages. This way one enables an easy understanding between different computer systems without too much human-interaction. Despite all these developments, not much practical adoption of these technologies can be found in any industry yet. Our feeling is that although such developments will ease integration practices increasingly, pure automatic modular plug and play can be expected to remain a utopia for many years or perhaps even decades to come.

Apart from the technical architecture chosen, there is also a design choice to be made regarding the technology. A short overview is given in Table 3-3. The first systems used plain ASCII formatted messages; later on standardized in different EDI standards. For almost any industry, standard EDI formats were introduced, which are still largely in use. In the Port of Rotterdam PortInfolink offers EDI services for its customers – for a description of the EDI community building process in Rotterdam we refer the reader to (Van Baalen et al., 2000). An important element with EDI is that it not only standardized messages, but also came with a standardized infrastructure, often with external brokers making the connection through dedicated lines. As a result of this EDI technology comes with a relatively high price, also since implementations tend to take long for the two parties to connect and adapt or develop a standard message format – since often the general template did not suit all needs. Some even state that EDI comes to a too high price (Bergeron, 1997). Therefore EDI is most suited for dyadic relationships and less for chain or network applications as Markus (2000) observes.

The emerge of the Internet – which made information sharing possible to virtually no costs (Jap, 2002) – gave rise to the adoption of XML (eXtensible Markup Language) throughout industry and within many applications. XML enables a free formatting of documents and cheap submission over the web. XML therewith works around two of the disadvantages of EDI, being the expensive dedicated infrastructure and the rigid standard structures. Within a short period all different kinds of pre-specified XML standards appeared – RosettaNet and ebXML being some of the best known examples. Often those standards do not only describe the way messages should be formatted, but also include process descriptions and agreements.

Technology	Description
Plain ASCII (legacy formatted)	Non-standardized message based integration. The most basic form of integration.
EDI	Electronic Data Interchange. Structured message exchange between computer systems. In use since the beginning of the 80s. Standardized by the UN/EDIFACT consortium. Standards exist for many different industries. Generally goes hand-in-hand with a dedicated EDI infrastructure, what makes it expensive and inflexible. Often used for process focused integration.
XML	Extensible Markup Language. Became popular to the end of the 90s, in the Internet boom days. Ideal to make easy understandable messages, that can be transmitted to little costs over the Internet.
Standardized XML	Disadvantage of XML is that anything can be an XML message. Interpreting the content requires some form of

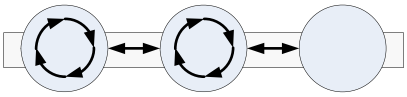
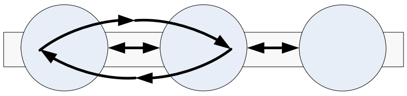
	standardization. With XML came many different standard initiatives, such as for example: RosettaNet and ebXML. New technology developments such as Service Oriented Architectures, WebServices and the Semantic Web depend heavily on XML.
WebServices orchestration languages	Languages such as BPEL en BPML are emerging to create fast interoperability and connectivity between systems. Typically those tools and languages allow modeled workflow inclusion in the XML, and process mapping.

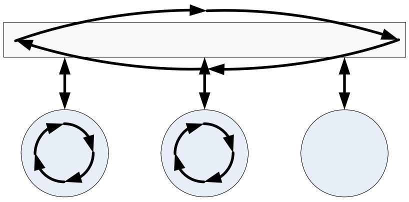
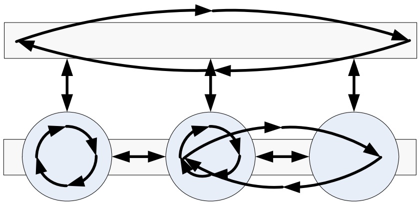
Table 3-3: Information exchange technologies.

For example, the RosettaNet PIP 3A1 (Request Quote) includes agreements on partner response times. In recent years WebServices empowered orchestration languages and systems appeared such as BPEL and BPML, which make it relatively easy to integrate two disparate systems by mapping business processes. It also makes it possible to include workflows within the XML message – easing system integration and lowering costs therefore. For a good description of BPEL we refer the reader to Peterson (2003) which gives an example of how BPEL influences inter-organizational processes. It illustrates the service oriented architecture mindset: create new applications and functionality by linking existing systems in a smart manner.

3.6 Planning functionalities of inter-organizational systems

Inter-organizational Systems are built not just to enable plain data exchange, but are intended to improve inter-organizational processes. Therefore, IOS are meant to support planning. It is either used by the individual companies in their internal operations (IOS enabled intra-enterprise planning) to adjust their own plans based upon external information, or it is utilized for inter-enterprise planning, i.e., a coordinated planning of activities between two (or more) parties working in the same supply chain. A third, not yet mentioned typology for planning is IOS enabled chain planning which refers to synchronization or optimization of the chain, which comes down to the coordination of individual activities based on system wide objectives. Most IOS’s that have been implemented successfully follow the first concept: the information exchange is a way to enable individual parties to better adjust their internal planning to the status of the outside world. The second typology is also quite common. Chain steering, on the other hand, is a very promising but ambitious concept, which is hard to put to real practice – as it often comes with many practical problems such as the fact that optimization of a chain might result in individuals not behaving optimal. Essential in this therefore are performance measurement and evaluation mechanisms that result in compensation control systems for the participating parties. Although a centralized planning mechanism for a supply chain – a sort of ERP for the Hub as Markus et al. (2000) put it – sounds like a brilliant idea, implementing a centralized system for a supply chain is not an easy thing to do. First of all, chains are not static, but do continuously change: parties come and go. Second, most enterprises nowadays participate in a number of for them important supply chains – chain planning therefore still needs to include many external events that cannot be steered upon. Third, in industries with strong chain leadership this model will be dominated by the chain leader – often resulting in sub-optimality for all but the leader. Realizing this, there is a fourth typology that combines the strengths of the others. We refer to it as IOS enabled chain synchronization and inter-enterprise planning. By letting the autonomy with the individual enterprises, and utilizing the IOS pure for information exchange, and high-level synchronization of activities, one can perhaps get the best results. Unfortunately this is still largely theory, since not that many examples are known of this type.

	Chain planning type	Characteristics
	IOS enabled intra-enterprise planning	– IOS enables information exchange between enterprises – Information used for intra-enterprise planning purposes – Own planning adjusted based on external information – Most IOS of this type
	IOS enabled inter-enterprise planning	– IOS enables information exchange between enterprises – Information used for inter-enterprise planning purposes – IOS makes it possible to interact and rearrange planning between enterprises – Some successful examples known such as CPFR

	IOS enabled chain planning	– The IOS is far more than plain information exchange: it becomes the chain orchestrator – Information is exchanged, planning takes place at a higher level, and its impacts are communicated back – Optimal for the chain does not necessarily mean optimal for all individual enterprises – Need for measurement and compensation systems – Promising but few examples
	IOS enabled chain synchronization and inter-enterprise planning	– A combination of types (2) and (3): combining their strengths – At a higher level synchronization between enterprises takes place – Autonomy for local planning however remains with individual enterprises – Promising but few examples – Key is in plan aggregation levels, responsibilities, real-time chain insight, et cetera

Planning can (and should) never been seen separate from the previously discussed planning levels or planning horizon. Long-term planning requires a different type of interaction, than last-minute adjustments to be made after constant monitoring of execution processes. See (Sodhi, 2001) for a detailed discussion on the planning levels in operations, and the type of IT support. In any design for an IOS the technology and typology (as described above) should not be seen separated from a process focus on the type of intra- or inter-organizational processes to support. For example, in environments with continuous change, decentralized decision making might be the smartest thing to do (Chen, 1999). We conclude that sharing information within a supply chain can be beneficial to many. In order to gain the biggest benefits and a significantly better performance an IOS should be more than just transactional support, and preferably designed from the ground up focusing on the processes to empower (Saeed, 2005). Enterprise systems get more robust when information is exchanged throughout the supply chain, as Chen (1999) learns us, especially when the flow of information is two-directional (meaning, there is a feedback loop involved). He furthermore states that in environments with large variance and continuous change, decentralized decision making might be a smart thing to consider. Let us therefore look centralized versus decentralized decision making in more detail.

4 Centralized and decentralized planning and execution

In logistics we see definitions of centralization and decentralization taking on a geographic dimension. For example, in the field of urban operations research much attention is given to efficiency trade-offs realized by locating one service facility centrally or multiple small or branch facilities throughout the region. As Larson and Odoni (1981) note in their textbook on urban operations research, these trade-offs may be studied through as spatially distributed queuing systems. This approach may be seen in application to decisions regarding crew location in a Hong Kong copy machine repair problem presented by Chu, Lin, and Ng (1991). While these geographically based definitions are prevalent in the literature, they do not subsume the decision making process in both centralized and decentralized contexts.

The economics literature is rife with definitions and examples of centralization versus decentralization in the context of control of complex systems. One pivotal definition in the field was proposed by Marschak. In a 1969 paper, he proposed that a system is said to be centralized if there exists an entity that can monitor all of the signals transmitted between individual actors in the system, but not necessarily the signals between the environment and the actors; and from this monitoring, can influence the performance of the actors to complete a given process. Alternately, decentralized systems are those in which the information and decision making capabilities are pushed away from this observant entity and towards the individual actors in the system. This definition is more appealing than the simple geographic definitions as it highlights the role information and decision making capabilities play in labeling a system as centralized or decentralized.

Alternately, decentralization is the movement away from a structure of command and control [wikipedia.org]. Decentralization can be best described as the movement of a system toward a dependence on lateral relationships. The movement away from a centralized structure can traverse a broad spectrum of features defining decentralization – including autonomy and flexibility. As such this section provides a framework to capture the spectrum of decentralized decision making structures described in the literature. We begin this section with a subsection describing our proposed framework. Following this are two subsections describing centralization and decentralization in greater detail within the context of planning and control.

4.1 A classification framework

The framework we espouse is not the first. The literature contains several classification schemes. Focusing on the level at which decision making is held Schneeweiss (2003a) presents a framework by which to classify systems of distributed decision making. Schneeweiss identifies a tree like structure for the categorization of distributed decision making systems. The tree is four layers deep with each layer representing, in turn, the number of decision making units (DMUs), the presence of coordination amongst the DMUs, the type of coordination, and the type of negotiations utilized by the system. The branches of the tree highlight the different options that exist within each category. This tree structure provides for a relatively comprehensive framework, however, it does not fully capture the nuances of the spectrum describing decentralized systems. Specifically, in the structure and delineation of this framework, Schneeweiss purports that hierarchical systems are one of the only forms of decentralization in decision making. In the second edition of a book on the matter, Schneeweiss explicitly states that this tree-based classification scheme pertains only to systems displaying some degree of hierarchy (2003b). Hierarchical systems do not, however, reflect full decentralization as the objectives employed at each level are often passed from a higher (possibly centralized) level. While this book (2003b) does include a discussion of MAS, it is relegated to a scant chapter on coordination and negotiation. We argue that MAS are not only a mechanism by which distributed decision makers (or agents) may communicate, they represent a decision making mechanism worthy of classification.

Our framework, depicted in Figure 4-1, avoids the rigidity of a tree-based structure for classification. This framework instead highlights the overlap between the different planning approaches. We begin by exploiting the identified roles information and decision making play in demarcating a planning system as centralized or decentralized. Information may pertain to global events and may be held globally or it may pertain to events relevant on the local level and be maintained locally. With respect to decision making, there may be a single objective or multiple objectives and the capability to act on these objectives may be held only at a central level or at a local level. Finally, while not depicted in the figure, we can imagine a third axis running along the diagonal describing the number of decision making units (DMUs), moving from a single DMU in the upper left to multiple DMUs in the lower right.

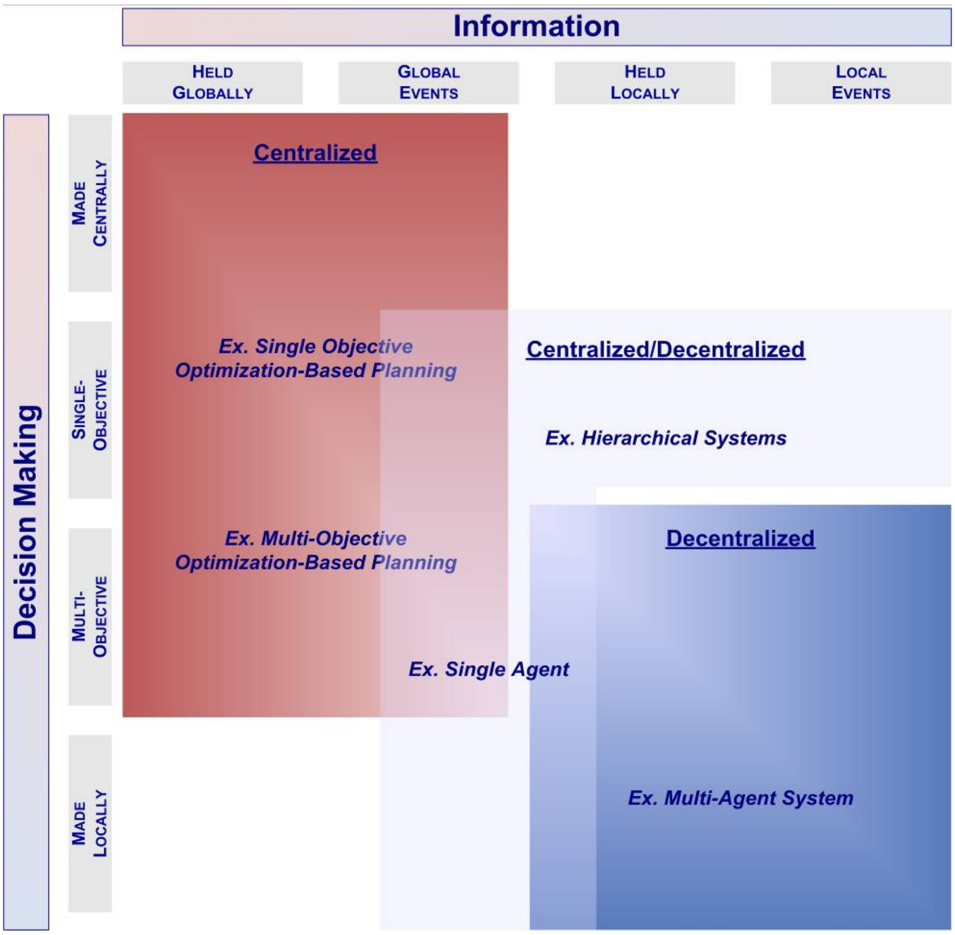


Figure 4-1: Classification of planning systems as centralized or decentralized according to the role of information and decision making capabilities.

In this way we classify a spectrum of planning systems, at one end (upper right corner in Figure 4-1) there is the fully centralized system in which information regarding global events is held centrally at the same level as the decision making capabilities; there may be only one or possibly multiple objectives guiding the decision maker, but the tradeoffs between multiple objectives are decided by one entity at a central level. These systems may be exemplified by optimization or a classical mathematical programming approach. In the middle of the spectrum are systems that garner part of their information or decision making capabilities from a central source, but other aspects are maintained locally. For example, in hierarchical systems the objectives are often passed to each layer from a central entity, but decisions may actually be made or implemented locally. The concept of holonic organization (Mathews, 1995) also occupies a middle-ground with respect to task concentration and authority structure. Holons act autonomously, but will follow directions of higher level holons, if present; hence, they recognize authority but do not depend on it. At the other end of the spectrum (lower left corner of Figure 4-1) are fully decentralized systems; systems in which there are multiple entities with information pertaining only to local events, each entity maintains its own objective (hence, the system as a whole is multi-objective), and makes decisions. These decentralized systems are typified by multi-agent systems.

4.2 Centralized versus decentralized systems

The remainder of this section is devoted to a detailed exploration of both centralized and decentralized systems used in planning and control. We begin by considering the case, depicted in Figure X, of one single decision maker (DM) that controls a full system. We assume that in such a situation, the DM has access to all information regarding the control problem's objective, decision variables and constraints. The design challenge becomes to formulate an appropriate decision model and associated solution procedure allowing for optimization at modest (computational) costs. Formulated as a mathematical program, the decision problem becomes (Papadimitriou et al. 1998):

$$\begin{array}{lll} f : \mathbb{R}^n \to \mathbb{R} & & \\ \text{minimize } f(x) & & \\ \text{subject to} & g_i(x) \leq 0 & i = 0, .., m \\ & h_j(x) = 0 & j = 0, .., n \end{array}$$

The function $f(x)$ predicts the outcome of an n-dimensional decision x in terms of a single-dimensional objective – which is to be minimized in order to be optimal. The function sets $g_i(x)$ and $h_j(x)$ denote inequality and equality constraints respectively, as functions of the decision vector. The solution procedure associated with the model should be capable of actually performing the optimization, i.e. locating a concrete vector x minimizing $f(x)$ under the given constraints. Table X summarizes the major elements that influence the exact details of such a mathematical programming formulation.

Dimension	Range (low)	Range (high)	Remarks
model description	discrete	continuous	see (Biegler et al. 2004)
search procedure	heuristic	optimization	see (Cordeau et al. 2002; Taillard 1990) for overview heuristics
time horizon	short, real-time, online	long, static, offline	see (Sabuncuoglu et al. 2000) for different horizons in job scheduling
information quality	stochastic, incomplete, incorrect	deterministic, complete, correct	see (Gendreau et al. 1996) for stochastic scheduling approaches for AGVs
information quantity	small packages, infrequent	large packages, frequent	relevant for communication and information processing capacity

Table 4-1: Controller design dimensions in a single DM setting

In a comprehensive overview of analytic mathematical formulations, Biegler et al. (2004) distinguish discrete from continuous formulations. Continuous formulations are those in which all variables may take any real valued number as part of the solution. Discrete formulations, on the other hand, are those for which some or all of the decision variables are restricted to discrete values, e.g. integer or binary. In logistic control systems discrete formulations often arise, for example with regard to resource scheduling or vehicle routing.

Finding an efficient solution procedure depends strongly on the selected problem formulation. For continuous mathematical programming formulations, many good solution procedures are known. Linear Programming (LP) formulations are routinely solved by the Simplex algorithm. Non-linear Programs (NLP) are less tractable, but efficient procedures exist for NLP problems meeting regularity conditions (e.g. differentiability). Biegler et al. reference the IPOPT solver’s ability to solve NLP instances with over 2 million variables. Derivative Free Optimization (DFO) methods are available for NLP problems not meeting regularity conditions, however, such methods tend to scale poorly (Biegler et al. 2004). Continuous solvers cannot be directly applied in such cases and many discrete problems are known to be computationally complex (Papadimitriou et al. 1998). A well known algorithm for integer programs is ‘Branch and Bound’ (BB), which selectively searches the discrete solution space based on relatively fast bound calculations (Papadimitriou et al. 1998). An overview of current developments in BB is provided by (Johnson et al. 2000). Problems that cannot be solved via known algorithms are often solved by the use of heuristics. Heuristics are procedures that are employed to generate a feasible solution to a complex problem – although the quality of this solution may not be known.

Aside from the use of discrete variables, time may also play a factor in the need to use heuristics as opposed to optimization techniques. A common distinction, along the time dimension, is between online and off-line scheduling. In online scheduling, a short scheduling horizon is used, whereas off-line scheduling looks ahead for a significant period of time. Online scheduling thus allows addressing a problem of smaller size, but only at the potential sacrifice of longer term scheduling opportunities. We note that decision rules (e.g. ‘First Come, First Served’) may be regarded as extreme forms of online scheduling.

Another factor troubling a single DM design is the often limited availability of reliable forecasts (e.g. in vehicle routing, a limited forecast for customer orders). Clearly, unavailability of such information severely limits the applicability of off-line scheduling. Moreover, information that is available, is often of a stochastic rather than a deterministic kind. Several researchers have reported different tactics to overcome constraints in size and information quality. For example, Sabuncuoglu et al. (2000) compare on- and off-line scheduling strategies in environments of different size and information availability. Gendreau et al. (1996) provide an extensive overview of research developments addressing stochastic vehicle routing. Heuristic methods relax the maximization constraint in order to reduce computational complexity. Taillard (1990) provides an extensive overview of heuristics for the flow shop scheduling problem. Parallel taboo-search is identified as an apparent near-optimal procedure. Similarly, Cordeau et al. (2002) provide an extensive overview of heuristics in vehicle routing – they confirm the good performance of taboo search algorithms.

Centralized solutions often fail because of their inability to cope with a high degree of complexity and change. These conditions require the solution to be both robust to disruption and reconfigurable when necessary. Decentralized solutions may be very suitable in situations where a classical centralized solution isn’t appropriate and where the distribution of information and decision making is necessary. More specifically, Marik (2005) describes three possible characteristics that make a centralized approach inappropriate – and hence a decentralized solution attractive. First, a centralized solution may be infeasible. At any time, each decision-making node may have only a part of the information required to make the decision. Second, a centralized solution may be impractical. Even if all information is available there may be practical constraints (time, cost, and quality) to making information centrally available or to performing synchronized, centralized decision making. Third a centralized approach may be inadvisable. Even if a centralized decision process is feasible and practical, it might still be inadvisable due to the susceptibility of a single decision-making node to disruptions. Furthermore, the complexity of making system reconfigurations and long-term changes under centralized regimes may be prohibitive. As Singh (2007) recently noted, despite the listed drawbacks, most of the SCM software in industry now is of a centralized nature.

Recognizing these troubles in operational decision making parallelizing the decision making processes is one method to alleviate the computational burden. Distributing the computational load of decision making across multiple decision makers promises significant speed ups, due to benefits of parallel processing. However, distribution of tasks introduces a need for coordination: i.e. for managing the dependencies between them (Malone et al. 1994). Obviously, the need for coordination strongly depends on the level of coupledness (dependence) of the decision processes involved. This, in turn, strongly depends on the structure of the control problem under consideration. Decision settings may be inherently distributed in spatial, temporal or functional dimensions as Durfee et al. (1989) make clear. AGVs may find themselves in different spatial regions of the system. Job delivery is performed after job pick-up (temporal dependency). Finally, routing decisions are functionally different from job allocation decisions (although highly intertwined). Distribution properties may reduce the level of decision coupledness, but will not dissolve the need for coordination altogether. Furthermore, the dependencies in distributed systems develop dynamically over time, as they result from many decision processes of varying frequency and phase, communicating over channels with varying delay (Androulakis et al. 1999). Coordination thus sets distributed designs apart from single DM designs.

An important dimension in distributed design is the method used for distributing the decision problem across decision makers. Decomposition (e.g. Lagrange multipliers) methods exist which allow partitioning of combinatorial problems into many loosely coupled sub-problems (Androulakis et al. 1999). By processing the sub-problems in parallel, the master problem may be efficiently solved. In settings strongly driven by an existing organizational partitioning, (top-level) decomposition is likely to follow organizational bounds. This need not be inefficient, as operational processes in naturally evolved business networks are not unlikely to be relatively independent (Simon 2002). Another dimension is the temporal coordination between decision makers. Full state synchronization, i.e. completed state propagation before every decision in the system, may introduce prohibitive costs of communication. Asynchronous protocols may decrease communication loads but may also introduce state inconsistencies (Androulakis et al. 1999). Table 4-2 summarizes the dimensions discussed so far. Following Table 4-2, we provide a detailed description of hierarchical systems. A description of heterarchical systems is left for Section 5.2 as MAS are one of the only examples of a truly heterarchical system.

Dimension	Range (low)	Range (high)	Remarks
structure	loosely coupled	tightly coupled	
synchronization	synchronous	asynchronous	(Androulakis et al. 1999)
level of conflict	cooperative	antagonistic	(Schneeweiss 2003)
information visibility	opaque, private information	transparent, public information	(Schneeweiss 2003)
decomposition	problem based	organization based	
network structure	heterarchical	hierarchical	Dimensions: communication, authority, task division

Table 4-2: Several dimensions for distributed controller designs

The word hierarchy is a surprisingly modern word – coined by Dionysius in 1380 [Oxford English Dictionary]. Derived from the Greek word for Bishop, hierarchy refers in origin to the three orders of angels with God at the pinnacle. In current usage, the term hierarchical refers to any organization system composed of subsystems in which every subsystem has communication only with its immediate superior subsystem or immediately subordinate subsystems. This definition conjures up visions of a vertical tree based structure.

Hierarchical systems arise as a control mechanism for complex systems when there is a need to segregate a larger system or problem into smaller subcomponents. In the literature on hierarchical systems, this segregation primarily occurs in three different ways, as identified by Libosvar (1988): 1) decomposition of the decisions to be made (multi-layer), 2) decomposition of a physical system (multi-level), and 3) decomposition of a larger mathematical model (aggregation techniques).

Of most relevance in the field of logistics is the second category of systems. As such the remainder of this section is focused on multi-level systems. These categories were first proposed and explained in the seminal book on hierarchical control for large scale processes by Mesarovic, et al (1970). Subsequently, Findeisen, et al (1979) published a book addressing the practical issues of design and implementation of hierarchical control systems. Figure X (inspired by Libosvar, 1988) presents a generic 2-level representation of a multi-level system.

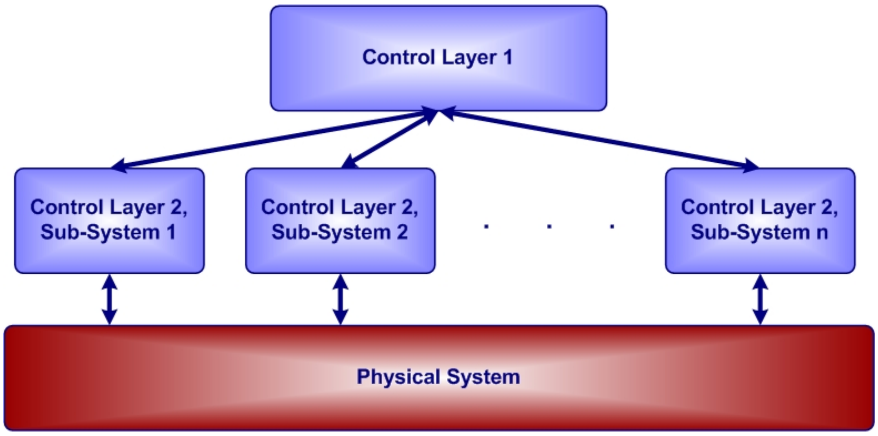


Figure 4-2: Generic Multi-Level System

The primary focus of Mesarovic (1970) and subsequent research addressing multi-level systems is on coordination. This is particularly evident in the literature on supply chain management. Schneeweiss (2004) describes the classification of “coordination-oriented supply chain management” as divisible into three categories: 1) quantity discounts, 2) inventory control, and 3) contracting. It should be noted, however, that the theory driving all three categories is surprising similar. All categories are premised on making a financial decision, dependent on the reaction of another party, but made fully in absence of information on the other party’s response to that decision.

The first category of quantity discounts is concerned with the relationship between suppliers and buyers in the supply chain. Considering Figure X, the suppliers may be seen as control level 1, while the buyers as control level 2 and the system in which prices are set and quantities ordered is the physical system. Monahan (1984) was the first to show that in a 2-echelon system, with one supplier and one buyer, the supplier can increase profits by offering quantity discounts. The second category of inventory control is generally focused on the relationship between divisions in a single company. The concern here is coordinating the production or stock levels to satisfy unknown buyer demand. The final category of supply chain contracting is similar to inventory control in that it focuses on the establishment of incentives to coordinate a stream of production. The difference, however, lies in the fact that the coordination and incentives are arranged across multiple companies in a supply chain.

As can be seen here the decision making process for all of these applications occurs at a local level. What separates these decentralized or distributed decision making methodologies from the fully decentralized category of systems is the alignment of incentives and thereby the alignment of objectives. Notice that in the hierarchical systems the objectives at each control level are premised on incentives set centrally. In fully decentralized systems each decision making unit acts in a fully independent manner based on individually set and held objectives (either single or multiple).

5 Agent based systems

5.1 Agent theory

Agent theory first began to appear in the computer science and artificial intelligence (AI) literature in the mid- to late-1980s as an outgrowth of objected oriented and distributed AI fields[1]. Despite almost twenty years of history, a definition for the term agent still remains debated. Schleiffer (2005) states that: “intelligent agent technology is the articulation of human decision making behavior in the form of a computer program”. While this definition is particularly elegant it is lacking in that it does not explicitly specify the characteristics of human behavior that agents seek to emulate. One of the most cited agent definitions was published by Wooldridge and Jennings in 1995. They put forth four distinct characteristics known as the weak notion of agency. These four characteristics are: autonomy, social ability, reactivity, and pro-activeness. These characteristics are widely accepted as they are at the heart of what agents represent – human decision making processes. This set of four properties has been expanded on significantly over the years and across multiple fields. Table 5-1: Agent characteristics and the reference citing the characteristic.Table 5-1 presents a list of agent characteristics cited in the literature.

This list of agent characteristics may at first seem to comprise only terms that are ambiguous in their application as part of a software system. This ambiguity can in part be clarified via a review of agent architectures. Agent architectures provide a formalized description of how an agent software entity perceives its environment and subsequently transforms this information into decisions (Wooldridge, 1999). In 1999, Wooldridge identified four primary types of agents and their corresponding architectures – logic based agents, reactive agents, belief-desire-intention agents, and layered architectures. In the first case, decision making is performed via logical deduction or theorem proving. The second case, reactive agents operate based on a maintained library mapping situations (or perceived situations) to actions. Belief-desire-intention (BDI) agents, developed by Rao and Georgeff (1991), are premised on practical reasoning – the process of manipulating data structures in an effort to decide what goals should be achieved and how those goals should be achieved. Finally, layered architectures as described by Wooldridge (1999) are architectures

---

[1] For a history of the field, see Jennings et al, 1998.

based on the premise that agents must be capable of both reactive and proactive behavior. As such, the architecture is layered with one layer handling the reactive behavior and a second layer handling the pro-active behavior. Luck and d'Inverno (2003), add to this list of four architectures with a description of autonomous agents, memory agents, planning agents, and sociological agents. These agents borrow much from the primary four classes described by Wooldridge in 1999, and hence are not described here.

Characteristic:	Cited in:
Autonomy	Wooldridge (1995), Schleiffer (2005), Wooldridge (1999), Jennings (2000), Franklin (1996), Luck (2001), Luck (2005), Samuelson (2006), Rudwosky (2004)
Social ability	Wooldridge (1995), Schleiffer (2005), Wooldridge (1999), Luck (2001), Rudwosky (2004)
Communication	Luck (2001), Luck (2005), Marik (2005)
Negotiation	Luck (2005), Marik (2005)
Cooperation	Schleiffer (2005), Luck (2001), Marik (2005)
Reactivity	Wooldridge (1995), Wooldridge (1999), Luck (2001), Rudwosky (2004)
Pro-activeness/Goal oriented	Wooldridge (1995), Wooldridge (1999), Luck (2001)/ Jennings (2000), Franklin (1996), Luck (2001), Rudwosky (2004)
Situatedness (both time and space)	Schleiffer (2005), Wooldridge (1999), Jennings (2000), Franklin (1996), Luck (2001)
Decision Making	Schleiffer (2005), Wooldridge (1999), Samuelson (2006)
Ability to influence environment	Wooldridge (1999), Franklin (1996)
Reasoning/Problem solving	Schleiffer (2005), Jennings (2000), Luck (2005)
Learning	Wooldridge (1999), Luck (2005)
Robustness	Schleiffer (2005), Wooldridge (1999)
Coherence in sensing environment	Schleiffer (2005), Wooldridge (1999), Franklin (1996)

Table 5-1: Agent characteristics and the reference citing the characteristic.

Examining these architectures we see that just as operations research may be used to describe centralized planning approaches, OR terminology may similarly be invoked in describing agents. For example, an agent may be viewed as an entity with an objective function that must be met (in the language of agents – a goal) subject to constraints imposed by the environment. In the language of agents, these constraints may be based on perceptions of the environment and constructed from logic (as in logic based agents) or from pre-established mappings between situations and outcomes (as in reactive agents). The process of formalizing these goals and environmental constraints into a mathematical program to be solved via optimization represents but one mechanism by which an agent may achieve its goals. Despite being able to describe agents in an OR sense, the agent-based literature tends to be premised on rather poor optimization procedures and hence may benefit from concepts in mathematical programming; a claim similarly noted by Schneewiess (2003b) – his explanation is the roots MAS has in computer science, with a solution focus rather on "coordination through communication" than the pure application of the best optimization algorithms. Therefore, much of the MAS work focuses on the implementation and communication aspect of distributed decision making. In the most recent Agentlink roadmap, computer scientist professor Michael Luck (2004) identified the same, and does a call to action to the agent field to deepen links with the operations research community, especially the researchers working on distributed decision making.

5.2 Multi Agent Systems (MAS)

Systems consisting of multiple agents interacting with each other and the environment in which they are situated are known as multi-agent systems. In such a setting the agent construct becomes more than just an entity performing local optimization tasks – the agent must also possess the ability to communicate and coordinate. The important characteristics of a multi-agent system are, according to Rudowsky (2004): (1) Each agent's information or capabilities for solving the problem is incomplete; (2) No global control system; (3) Data are decentralized; and (4) Computation is asynchronous.

The methodologies implemented to achieve this communication and coordination are among the defining features of a MAS; as Odell (2002) put it: "designing an agent based system is not just about designing the agents, it is also about designing the agent environment." The agent environment does not only include the different agents but also the principles and processes under which the agents exist and communicate. Agent communication is described by both the language and the method by which they exchange messages in that language. Agent coordination (sometimes referred to as "interaction") refers to the mechanism by which agents organize themselves to work on the problem of the full system. The following two subsections address, in turn, MAS communication and coordination.

Agent communication is a field of study unto itself; situated at the crossroads of linguistics, cognitive science, artificial intelligence, formal logic, and computer science. This field of communication is dominated by both language semantics and dialogue protocol. Language semantics refer to the meaning that is expressed in a language or code [www.wikipedia.org]. A dialogue protocol, additionally, specifies a set of rules that regulate the dialogue between two or more communicating agents (Endriss et al, 2003). The remainder of this section presents a brief (and by no means comprehensive) review of the multi-agent work being carried out in both areas of communication.

There are a multitude of pre-cursors to formalized agent communication languages[2]. These languages arose on a predominately ad hoc basis and afforded a low level of inter-operability across systems. One largely used language pioneered by the United States Defense Advanced Research Programs Agency (DARPA) was Knowledge Query and Manipulation Language (KQML) – a language premised on restricted message sets and types (Finin et al, 1994). Throughout the 1990s, as interest in agent technology grew with the rise of the internet, efforts were made to formalize these early and fragmented languages into one Agent Communication Language (Singh, 1998).

In tracing this formalization process for agent language semantics and dialogue protocol it is necessary to examine theories of agent interaction. The 1990s saw the field of agent theory very much dominated by concepts of mental agency – as exemplified by BDI agents (Rao, 1991). At the foundation of BDI agents is the theory of rational action. This theory pioneered by Cohen and Levesque (1990a, 1990b) states that agents act and interact based on the four operators of belief, goal, happens (describing what will happen next), and done; with goal serving to represent intents and moderate the longevity of agent plans. Premised on this theory the Foundation for Intelligent Physical Agents developed an Agent Communication Language (ACL) that puts forth a formal semantics for agent interaction [www.FIPA.org].

At the end of the 1990s, concerns were raised with the concept of mental agency serving as the foundation for ACLs. These concerns were primarily focused

---

[2] For a more detailed account of early ACL evolution, the reader is directed to Singh, 1998.

on difficulties encountered with verifiability in terms of inter-operability among different agent platforms. Walton (2003) points out that while BDI agents can interact using commonly defined semantics; it is not possible to verify that these agents are acting according to these semantics as that information is only encoded as an intention within the agent's mental state – a state hidden to external observers. As such, these languages have given way to languages premised on social agency (Singh 1998, Walton 2003).

Prominent among the ACLs based on social agency, is a class of languages known as commitment based languages. In a paper published in 2000, Singh argues that an ACL based on agent interactions will meet the four criteria of formal, declarative, verifiable, and meaningful. Singh goes on to describe such a language based on social commitments. These commitments are defined by two notions – that of social context (i.e. the team with which an agent may participate and communicate with) and metacommitments (i.e. legal or social relationships). Walton (2003) and Grando and Walton (2006) have similarly used a social approach based on social norms within societies in order to develop their MAP and MAPa languages.

It should, however, be noted that these socially based agent communication languages are not without their critics. Specifically, Rovatsos et al (2004) argue that both intentions or mental states and commitments or social obligations are external to the communication itself and hence form poor foundations for ACLs. Instead they argue that the meaning of communication (i.e. semantics) lies in the consequences of the communicative actions. As such they propose a functionalist or empirical semantics based on the statistical correlation between messages passed and actions taken in a MAS. Fischer et al (2005) have extended this work to answer the following two questions that have arisen as a result of their disdain for the previously suggested mental agency and social agency based ACLs: "(1) If strict adherence to communication languages and protocols cannot be taken for granted, how can meaningful and coherent communication be ensured? And (2) Observing the course of conversations that take place in a MAS, how can agents effectively organise this kind of knowledge and relate it to existing specifications, so that they can actually benefit from it?" In answering these questions, Fischer et al (2005) propose the use of probabilistic models to facilitate agent learning of the communicative practices within a given environment. Thus, the development of ACLs has progressed from its foundations in mental agency via social agency and is now in a phase premised on autonomous agency.

In addition to a language and semantics, agents also need dialogue protocols to govern their communication. Many of the languages defined above include in their definition a dialogue protocol. These protocols are often premised on scenes (Walton, 2003) or frames (Rovatsos et al, 2004). These constructs allow for the designation of a setting that can govern the interaction. For example, in Walton (2003), an example of the MAP protocol is illustrated via a scene representing a patient visiting a doctor in order to obtain a diagnosis based on several symptoms. As such, the scene dictates that the interaction must be between doctor agents and patient agents. Furthermore, the scene construct allows for a designation of the type of actions that may be undertaken – i.e. “make an appointment” or “give a referral”. Rovatsos et al’s (2004) definition of interaction frames is similar, but formalized around four distinct information groups. These four elements are trajectories (sequences of actions and messages), roles and relationships (the types of parties and their involvement), contexts (the state of affairs before, during, and after actions and messages), and beliefs (the cognitive states of the parties involved).

Recently, Chopra and Singh (2004) and Singh (2007) have argued that these dialogue protocols tend to be rather rigid in that they don’t allow agents to step out of turn as designated by their role in the scene or frame. For example, when considering the situation where a customer requests a quote for a product, such dialogue protocols do not allow a merchant to proactively send a quote (i.e. advertising), a customer to preemptively accept a quote (i.e. customer loyalty/trust), nor the provision of goods in advance of negotiation (i.e. a trial offer) (Singh, 2007). As such, Singh (2007) recommends a protocol based on generalized commitment machines. General commitment machines model a protocol based on how commitments evolve as represented by states reachable by specified action models premised on a variety of established formalisms (i.e. causal logic).

To date, no single protocol seems to dominate in the literature. It is, however, evident in the work of Endriss et al (2003) that just as ACLs must be verifiable, dialogue protocols must also be tested for consistency. They argue that such conformance is important as MAS often engage in antagonistic or non-cooperate interactions (such as in negotiations or some auction settings). As such, Endriss et al (2003) examine different levels of conformance and then demonstrate how logic based agents may check for conformance both a priori and during runtime. Despite this robust body of knowledge on agent language development, protocols, and protocol conformance, such systems should not be developed in isolation of the coordination architecture selected. The following subsection describes several MAS coordination architectures with an emphasis on the associated communication requirements.

Coordination

Coordination among agents in a multi-agent system is a critically important process to ensure that the system acts in a coherent manner (Nwana et al, 1996). For an overview of developments in coordination schemes we refer the reader to Durfee et al. (1989) and Jennings et al. (1998). The process of coordination has four objectives as adapted from Lesser and Corkill (1987):

- All necessary components of the full problem are contained in the activity set of at least one agent
- Agents operate in a manner that allows their activities to be integrated into a solution to the full problem
- Agents act in a consistent or tractable manner barring harmful interactions
- Computation resource limitations are not exceeded by the system

These four objectives may be broadly generalized into the coordination sub-processes of task allocation and plan generation. There are multiple well established mechanisms for executing these MAS coordination processes. These techniques generally fall into three distinct categories:

- Organizational structuring
- Contracting
- Multi-agent planning

Nwana et al (1996) cite 4 categories – the three above plus negotiation. In general, most coordination mechanisms designs are based on some form of negotiation protocol, of which the contract net is a well known example (Davis et al. 1983). Protocols for example differ in the transaction properties being negotiated (e.g. price or service level) and the mechanism used to arrive at agreement (e.g. by deadline or by bidding). We, however, focus on only three as negotiation may be seen as a dialogue protocol useable in all other coordination frameworks (it is for example an important part of most Multi-agent Planning approaches). It is, therefore, not a coordination framework in its own right. The following subsections describe each of these coordination techniques in greater detail.

Organization structuring

If we imagine each agent as a node within a multi-agent system, then we can describe an organizational structuring as a method of control by which authority and connectivity for the flow of information and control is pre-determined (Durfee et al, 1989). The origin of this coordination technique is very much grounded in the theory of human organizations. Specifically, the seminal work of Galbraith (1977) introduces the concept of bounded rationality – directly addressing limitations on the amount of sensory information and amount of control that can be effectively used and exercised. Given these limitations several topologies have emerged as being effective organizational structures. These structures, ranging in their level of centralization, include hierarchical, heterarchical, lateral, matrix, and group or team structures (Durfee et al, 1989). Technology-aided decision making is nowadays also an important determinant to empower decision makers, and overcome some of the hurdles of bounded rationality, as is clearly pointed out by Buchanan (2006) in his work on the history of decision making.

One of the most common means by which these systems may be administered is via a blackboard architecture (Hayes-Roth, 1985). In computing, hierarchical organizational structures most often fall into a master/slave arrangement. In such a scheme the blackboard architecture may apply as follows: a master agent or scheduling agent serves to schedule the activities of the slave agents based on centrally posted information. This knowledge is gathered locally by the slave agents (or knowledge sources) and is posted centrally (i.e. on a blackboard). Notice that this structure imposes a level of centralization on the agent system that counteracts the benefits of decentralization (Nwana, 1996). Davidsson (2003) identifies four different types of agent architectural models – from fully decentralized to more-or-less centralized approaches; split in a matrix with two main axis: synchronous versus asynchronous coordination, and centralized versus decentralized. Davidsson makes clear that the best solution is context dependent.

The blackboard coordination mechanism may, however, also be applied to more lateral organizational structures based on interest areas or groups as demonstrated by Lesser and Corkill (1983). In their scheme for a Distributed Vehicle

Monitoring Testbed (DVMT) each node (or agent) acts as a blackboard-based problem solver in order to track vehicles moving among the spatially distributed nodes. Each node has the capacity to decide which node's information will be applied to the partial results posted on the blackboard. In order to fulfill the coordination requirements that plans must integrate and not result in harmful conflicts of interest, the nodes are assigned to interest areas in order to prioritize the goals. While blackboard coordination mechanisms can be very effective in a setting with a pre-defined organizational structure it is not without implications for the communication dialogue protocol requirements. Specifically, if a blackboard scheme for coordination is selected then the agents must be able to post and retrieve information from a public repository (Barber et al, 1999). Parunak (1999) puts a critical note to the use of "watchdog agents", which monitor the behavior of a population of physical agents – they should sense conditions and raise signals but not plan or take action. One should however be aware that not all researchers and software developers mean pure heterarchichical systems when they speak about MAS organization structures. As Caridi (2004) lines out in her review on agent application in production planning and control, there are several architectures possible – she specifically mentions five different types: heterarchical, heterarchical with coordinators, hierarchical, modified hierarchical, and holonic. She claims that the choice for a specific architecture is application dependant.

Contracting

The coordination mechanism of contracting has the contract-net protocol at its foundation. The contract net protocol was first introduced by Smith in 1980. Within this scheme, each agent may act either in the role of a manager or in the role of a contractor. A multi-agent system is then defined as the contract net where the execution of a task is described in terms of a contract between two agents. A strength of Smith's definition of the contract net protocol is that an agent may take on either role dynamically depending on the contract. Contracts are established via a process of task announcement, bid making, bid evaluation, and award. Contractors may then subdivide an awarded task and serve as a manager for other contractors. In such a way the resultant task division structure is hierarchical, but unlike a master/slave structure all agents have the same capabilities to both manage and contract. Hence decentralization is maintained in the communication and coordination. MASs enacting a contract net protocol must host a communication dialogue protocol that can support the announcing of tasks, placing of bids, and awarding of tasks (Barber et al, 1999).

De Weerdt et al (2005) formally define the multi-agent planning problem as follows:

"Given a description of the initial state, a set of global goals, a set of (at least two) agents, and for each agent a set of its capabilities and its private goals, find a plan for each agent that achieves its private goals, such that these plans together are coordinated and the global goals are met as well."

They then go on to identify six phases affiliated with the solution of this problem – global task refinement, task allocation, coordination before planning, individual planning, coordination after planning, and plan execution. The approaches used for global task refinement, task allocation, coordination before planning, and plan execution are not unique to multi-agent planning and are therefore not reviewed here. Nwana et al (1996) indicate that the phase of individual planning and coordination after planning may be loosely coupled as in centralized systems or more tightly coupled as in decentralized systems. In centralized systems, agents make their plans individually for the tasks allocated to them. These plans are then coordinated by means of a coordinating agent examines the individual agent plans for inconsistencies. This approach is implemented by Georgeff (1983, 1984). He introduces a supervisor process (1983) or process model (1984) that operates by identifying and grouping conflicting interactions within the agent plans. These interactions are then eliminated by the insertion of communication commands into agents' partial plans in such a manner that the partial plans are coordinated. A similar, but more recent approach is that of Tsamardinos et al (2000) in which the individual plans are merged based on a temporal network. In this way the coordinating agent can be seen as a constraint solver faced with a critical path problem for scheduling agent actions. Cox and Durfee (2005) develop a structurally similar approach to plan coordination, but instead of examining temporal conflict they focus on a branch and bound algorithm to remove plan flaws across agents.

Unlike centralized multi-agent planning in which a single layer or entity performs a conflict checking/plan coordination role, within decentralized multi-agent planning each agent is responsible for checking their own plans for consistency as they build them. This is most commonly done by providing each agent with a model of other agents’ plans; the agents then communicate to build and refine their plans and the models they maintain of others plans until no conflicts remain (Corkill, 1979). This approach is best exemplified by the Partial Global Planning (PGP) framework pioneered at the Multi-Agent Systems Lab at the University of Massachusetts. This technique introduced by Durfee and Lesser (1987, 1991) was designed to further the coordination techniques (described earlier) for the DVMT. Recall, the DVMT is the problem of coordinating a field of acoustic sensors to accurately describe the path that a vehicle makes through that field. As such, Durfee and Lesser developed a system in which each agent (or node) would be passed files containing the ordering, duration, importance, and expected outcomes of other agents’ goals. The agent would then use this partial information to form a partial global view of the full system – this view would form the basis for the actions performed by the individual agent in pursuit of its goals. This approach proved successful, yet highly tailored to the context of distributed sensor networks in the form of the DVMT. Thus, in 1992, Decker and Lesser proposed a generalized partial global planning algorithm This generalized approach to PGP is focused on the definition of a task-oriented framework for each agent. These frameworks may then serve as the basis for the identification of coordination schemes that may be implemented to improve the planning process. A coordination scheme then dictates what information should be exchanged among agents, when that information should be exchanged, and how that will affect the local scheduling of tasks. In this way, the PGP framework is generalized as the DVMT becomes only one instance of a coordination scheme.

6 Development and Implementation of MAS

So where have multi agent systems been implemented? Roth (2004) claims that 30 years after their first inception, the only widespread incarnation of mobile software agents is in the malware domain: computer viruses, spyware, Trojan horses, et cetera. This pessimistic conclusion can however easily be falsified. To give an example, modern computer games are a domain in which one can find widespread MAS application, it is a way to add intelligence to computer games and a means to let the system learn from the behavior of the user. Another domain which is often mentioned is that of telecommunication networks (Luck, 2004), in which agents have tasks such diverse as loadbalancing, selling & buying of network capacity, routing, self-healing, et cetera. Important in this domain are the real-time behavior and fast coordinated interactions. Multi-agent systems have been used as a simulation technique for ecological and biological systems, military applications, supply chain coordination, and resource allocation to name only a few (Luck et al, 2004); AgentLink maintains a robust and growing library of papers on agent systems and their applications (http://eprints.agentlink.org/).

Most of the early publications on agents cover work on single agent systems: agents that gather information on behalf of a user, or do specific tasks for them. An often cited reference is Maes (1994), which reports about work from the MIT Media Laboratory. Almost all the ideas and scenario's they envisioned and tested covered smart agents working for a user, on specific tasks such as e-mail scanning, filtering, searching, sorting, meeting scheduling, et cetera. Reading it more than a decade later one realizes that what was state-of-the-art in the early Nineties, is now part of anyone's daily Desktop and Office systems. Klusch (2001) gives an overview of internet agents applications. Internet agents are a type of agents which are basically applied for purposes such as information gathering and assisted decision support. Most of the examples he gives also concern single agent systems – having agents crawling the web, on their search for information, bargains and knowledge.

Application domains that are likely to benefit from multi-agent technology and concepts are domains with the following characteristics (Sierra, 2004): (1) interactions are very fast; (2) interactions are repeated with either (a) high communication overheads, or (b) a limited domain so that learning by the agent about user behavior is effective; (3) each trade is of relatively small value; (4) the process is repeated over long periods; (5) the product traded is relatively easy to specify. Reasoning along the same line several scholars pinpoint at expected future applications in domains (Maes, 1999) (Shen, 1999) (Luck, 2004) (Belecheanu, 2005), (Zimmermann, 2006) such as automated marketplace trading, defense simulation and training, planning and scheduling in logistics and SCM, industrial control systems, simulation modeling, smart sensor networks, enterprise system integration, and event management systems.

Davidsson (2005) sees specifically a large future for agent application in logistics, especially because the problem characteristics found in that domain closely match those of an ideal agent technology application, although he makes clear that

several problem areas within logistics seem to be under-studied: e.g. the applicability of agent technology to strategic decision-making within transportation logistics. Caridi (2004) explains the five basic strengths of multi-agent systems for logistical applications: (1) modularity; (2) decentralization reduces impact of local modifications on other system modules (replanning, etc.); (3) embedding multiobjective functions ; (4) allows to effectively model time-varying physical systems; (5) designing systems can be stepped wise process.
MAS in transport applications

6.1 MAS applications in transport

The applications of MAS most relevant for this research, however, are those in the field of logistics and transportation. Fischer et al (1993, 1996) succinctly identify the reasons why applying MAS to transportation problems (and specifically vehicle routing problems) is so appealing. They identify four primary reasons, which we summarize here:

- Vehicle routing is an inherently distributed task. Trucks and jobs are not only geographically distributed, but also maintain some level of autonomy in the field.
- Vehicle routing performed with any degree of realism must cope with multiple dynamic events. Fischer et al (1996) indicate that agent architectures have the capability to handle such dynamics.
- In order to use classical methods (i.e. those described in the previous subsection) for vehicle routing, a large amount of information must be maintained centrally. However, given the proprietary nature of this data, obtaining and maintaining what is needed for optimal routing may be difficult. As such, MAS provide an alternative solution method focused only on local information.
- In reality, transportation firms engage in a high-level of negotiation and cooperation in performing their daily transport tasks. MAS have the capability to include such cooperative capabilities that optimization based algorithms do not.

Premised on these four motivating factors, Fischer et al (1996) go on to describe a simulation test bed for multi-agent transport planning. This model is then tested on the vehicle routing problem with time windows; unfortunately, dynamics are ignored as all orders are known in advance. Leong and Liu (2006), similarly examine the application of MAS to the VRP with time windows in a setting where all demands are known a priori. Their findings are encouraging for MAS in vehicle routing as their solutions are competitive with the best known solutions for the Solomon benchmark datasets.

Several others also report on interesting applications of agents in transportation. See for example Spieck (1995), who was among the first to point out the promises agents may hold for transport planning, and Sataphy (1998), although this paper does unfortunately not reach much further than solely positioning an interesting architecture. Davidsson et al (2005) state that agent systems are the way to go for transport logistics. They provide a good overview of reasons why to use agents, and in which setting. They however point out that much work is still needed: till now too much work only done in simplified lab settings, or just focused on introducing new agent-platforms. Bodendorf (2005) envisions agent applications to be within the domain of Supply Chain Event Management (SCEM) systems, by adding sense & response capabilities.

Moving from the a priori or deterministic applications of MAS to the vehicle routing problem, to the best of our knowledge, the literature we studied contained two applications of MAS to the specific dynamic vehicle routing problem. Kohout and Erol (1999) present an “in-time” algorithm based on a stochastic improvement mechanism that enables an agent-based system to find solutions to a version of the “dial-a-ride” problem. Their approach is demonstrated on a case of airport shuttle routing in which they demonstrate that the MAS approach results in fewer vehicles than the centralized approach although the total time (a combination of travel time and waiting time) is higher.

The second application to dynamic vehicle routing is that of Mes et al (2006). They consider the vehicle routing problem with time windows in a setting where orders arrive during schedule execution. One strength of their exposition is that they compare their multi-agent system with a partially centralized OR-based heuristic for on-line vehicle routing. Mes et al compare their MAS to a heuristic premised on a hierarchical framework consisting of vehicle distribution to nodes and then a node-level assignment of vehicles to jobs. An interesting result of this paper is that the agent systems perform significantly better than the OR-based heuristics when compared in terms of vehicle utilization, service level (the average on-time delivery percentage), and relative costs. This result is promising; however, it may also be attributable to benchmark heuristic selected. The benchmark heuristics are based on hierarchical control concepts and thus fall short of providing a benchmark between a fully centralized and fully decentralized system. Filling this void by benchmarking MAS for dynamic vehicle routing against fully centralized systems for dynamic vehicle routing is the basis of work currently underway at RSM Erasmus University.

6.2 Developing agent systems

From a software engineering and design perspective it is good to understand where agent-based approaches differ from more traditional Object Orientation (OO) development methods. Jennings (2001) gave an overview on what he found to be the most compelling differences: (1) Objects are generally passive in nature (they need to be send a message before they become active); (2) Although objects encapsulate state and behavior realization, they do not encapsulate behavior activation (action choice) – more specific, an agent can have behaviors which are reactive, proactive, and/or social in nature; (3) OO fails to provide an adequate set of concepts and mechanisms for modeling complex systems; (4) OO approaches provide minimal support for specifying and managing organizational relationships. (5) Agents have at least one thread of control but may have more, whereas Objects have one thread of control only (Wooldridge, 1999). Nevertheless, as Jennings states, one can construct agent based systems utilizing OO techniques and environments, the value of agents primarily being in the mindset (and specific techniques) it creates; agent based approaches are an extension of the methodologies currently available, and especially suited for the development of complex distributed systems. An example of such an approach is demonstrated by Huang (2004).

In another paper Jennings (2003) writes about the adoption of agent based software development methods. He makes clear that it might be not that different from current methods, since many object-oriented analyses start from precisely this perspective: “we view the world as a set of autonomous agents that collaborate to perform some higher level function”. In short, agent-oriented techniques represent a natural progression of current software engineering thinking and, for this reason, the main concepts and tenets of the approach should be readily acceptable to software engineering practitioners. One should furthermore realize that agent-based systems are, after all, computer programs and all programs have the same set of computable functions. However, this does miss the point somewhat. The value of a paradigm is the mindset and the techniques it provides to software engineers. In this respect, agent-oriented concepts and techniques are both well suited to developing complex, distributed systems and an extension of those currently available in other paradigms.

6.3 MAS implementations elsewhere in supply chains

Agent implementation in industry is still very limited. Caridi et al. (2004) state that the few applications which are referenced in (academic) literature are all outcomes of research programs – very specific applications, that seem to have no further spin-off into other applications or get transformed into commercial on-the-shelf software. Industrial companies and software houses are not yet receptive for the agent paradigm.

Several scholars have studied this phenomenon. In Table X we give an overview of the factors which we identified in literature. "*Cost*", not surprisingly is a factor that matters – we think that this factor is related to most of the other factors, specifically for example the factor "*accuracy and correctness of the results*", which is mentioned as an important factor in several sources, relating to the fact that much of the agent work till now is only tested and evaluated to a limited extent. Another related factor is what we named "*the legacy of legacy systems*" – meaning the fact that most software architects, developers and consultants are still so much acquainted to the traditional way of system development: centralized monolithic systems. "*Security*" of the distributed objects is seen as a tread as well: specifically the issues that more points of failure create. "*Legal and ethical issues*" are perceived important as well, especially also when more "intelligence" is to be added to the system, and the outcome is not directly traceable back to a clear set of decision rules. "*Scalability*" of the system, "*acceptance by users*", and the importance of a "*central role for human decision makers*" as part of the system are some other factors mentioned. Mentioned by many is the factor we refer to as "*stuck in academic prototyping*" which seems to be the case right now. Within Academia a whole legion of researchers are working on agent based systems, concepts, languages, standards, et cetera, but there is too little interaction with industry – as for example Caridi (2004) mentioned. Petrie (2003) suggests the agent research community to reinvent itself, and integrate its concepts and ideas into the emerging WebServices domain. He specifically states that "ignoring industrial technologies leads only to published papers, while ignoring well-studied advanced distributed computing principles can lead to slow industrial progress due to the necessity for re-invention based on experience". He thus clearly pinpoints at the importance of exchange between the research and professional communities. Parunak (2000) also pleads for a clear interaction between researchers and business developers to converge agent concepts and industrial applications. Wareham (2005) identified in an extensive literature survey that most of the agent research is predominantly of a theoretic design orientation – there is little empirical or experimental research done to really apply agent concepts in practice. To list one example of this, the recent work by Lima (2006) in which the authors basically present a high-level framework without getting any concrete regarding its purpose, and plans for future testing or evaluation, et cetera. The problem is over simplified, and seems to be far too abstract for real implementation. The supply chain trading agents competing is a misleading name for a competition that basically deals with automating certain processes in a chain with technology, like for example procurement and sales. The competition – reported about by among others Arunachalam (2005) – is an example of how technologies are developed and tested in a controlled but competitive academic setting.

These systems however are not real agent systems; they do not derive to solutions through communication and negotiation. Three other factors turn out to be important. The factor "*standards*" is mentioned, just like "*misapplication*" which basically relate to the fact that agents cannot solve all problems and should primarily be used there where they could be of good use. Last, but certainly not least, since it was mentioned by many, is the factor need for "*professional development methods*". We discussed agent based development methods in the previous section, but will reflect some more on why these development methods hinder adoption. As Wooldridge and Jennings (1998) make clear, agent-based systems need different design methodologies – it is truly a different software engineering paradigm. One should really do a great deal of a-priori system level engineering beforehand, especially for large-scale systems – it is more than just throwing together a number of agents and let the system run, as they state. Pena (2006) pleads for a focus on product software, instead of the current practice of non standardized legacy from the start single MAS (prototypes). Nissen (2001) adds an important issue to this discussion by outlining how important it is to integrate the role of human decision makers, and the role and capabilities of intelligence in the agents into account within the design phase of agent systems. Hess (2000) comes to a similar conclusion: developing the intelligence and interactivity in an agent-based DSS makes development rather complex. Nissen (2006) states that "*when tasks become particularly complex, novel or risky, humans should be the decision makers, supported by smart software support systems*". Thus, design methodologies need to be further developed and made part of the education of tomorrow's software engineers as Marik (2005) and Wagner (2005) both conclude separately – it is no longer just about OO development based centralized system construction. From a traditional perspective perhaps more complex (Hess, 2000), but we expect that to change when the educational system has changed, and integrated agent development environments have been introduced (Parunak 1999, 2000). Belecheanu (2005) however makes clear that such professional development practices will be the key in selling and rolling out agent solutions to industry. Luck (2004) does a call to action to industrial researchers to start experimenting with agent prototype applications in industry, especially prototypes spanning inter-organizational boundaries. A similar call to action is made by Sandholm (1999) who calls for interaction between technicians and economic experts – for the reason that in agent based inter-organizational systems: parties which are interacting are likely to have a natural tendency towards manipulation, which pleads for a detailed understanding of economic principles.

Cost	Rudowsky (2004), Caridi (2004), Marik (2005)
Security	Rudowsky (2004), Roth (2004), Belecheanu (2005)
Legal / ethical issues (i.e. when more "intelligence" is added to a system)	Sandholm (1999), Rudowsky (2004)
Accuracy and correctness of the results / Guarantees of operational performance	Wooldridge and Jennings (1998), Rudowsky (2004), Caridi (2004), Marik (2005), Davidsson (2005), Belecheanu (2005)
Scalability	Wooldridge and Jennings (1998), Roth (2004), Marik (2005)
Acceptance by users	Sandholm (1999), Rudowsky (2004), Belecheanu (2005)
Central role human decision makers	Hess (2000), Nissen (2001), Wagner (2005), Nissen (2006)
Professional development methods	Wooldridge and Jennings (1998), Parunak (1999), Hess (2000), Parunak (2000), Belecheanu (2005), Marik (2005), Wagner (2005), Pena (2006)
Standards	Belecheanu (2005), Marik (2005)
The legacy of legacy systems	Caridi (2004), Marik (2005), Wagner (2005)

(traditional focus on centralized control systems)	
Misapplication (cannot solve all problems)	Wooldridge and Jennings (1998), Marik (2005), Wagner (2005)
Stuck in academic prototyping	Wooldridge and Jennings (1998), Parunak (1999), Parunak (2000), Petrie (2003), Caridi (2004), Davidsson (2005), Wareham (2005), Wagner (2005)

Table 6-1: Factors that hinder adoption of agent based systems in industry, with references.

AgentLink (www.agentlink.org) gives an overview of several case studies of “successful implementations” – among others in domains such as telecommunications, emergency operations management, adaptive transportation planning, insurance handling, and factory control. Most of these however also refer to solely research projects in several phases of maturity – and from several of these one could argue whether it is really agents what is applied. This brings up a bunch of interesting questions. Why do we see so little agent application in industry? Is the technology to blame, the concept wrong, or is it a question of the wrong marketing (if any)? Or are there actually many agent systems out there that have not been sold and recognized as agent based systems? What about for example the entire buzz around service oriented architectures; how different are these concepts from agent based systems? What can we learn from the development, implementation and adoption of previous generations of technologies and apply to agent-based systems? How does the inter-organizational factor come in?

Let’s look for example at the implementation of ERP packages – a topic which has been published about with a great frequency over the past years. An ERP implementation has a huge impact on an organization, which sometimes needs to “*learn function in radically different ways*” as Robey (2002) states it. An implementation is basically an organizational change process, in which the technology is an enabler. Biehl (2007) adds to this that the change process does not end when the system is implemented; it will be an ongoing process also after the implementation. Employees have to absorb knowledge, and develop new ways of working. One of the aspects which make an ERP implementation a though job is the customization which needs to be done. ERP’s are standardized software packages, with thousands of functions, features and screens, and largely based on so-called best practices: a standard representation of how a certain process should be conducted in a company. Not surprisingly, most processes in practice are not fully 100% best practice compatible. This asks for changes and customization, in the software, but often also in the way of working. Davenport (1998) refers to this as "*putting the enterprise in the system*". One of the disadvantages of large monolithic systems such as ERP's (Levy, 1998) is that, although originally by many often perceived as instruments to cope with adaptiveness and change, in practice they basically help in process automation which often results in loosing flexibility – simply since changing the system or the process is a hell-of-a-job. Sridharan et al. (2005) researched the domain of APS (advanced planning and scheduling) packages implementations and came with similar conclusions: namely that APS packages are often hard to implement, foremost for their complexity. They warn that great care is needed when one starts changing the "standard templates" – which is needed in most cases. Before clients switch to a new system, or changed templates, rigorous and adequate testing is needed to see if the system meets the client's true requirements.

From IOS literature we compiled the list of success factors for the implementation of inter-organizational systems, which is depicted in Table X. General conclusion is that companies should start from their business objectives, go through the change process together with their chain partners, and assure top-management support.

Top management support	(Ngai, 2004) (Lee, 2005) (Jones, 2006) (Li, 2006) (Zhu, 2006) (Biehl, 2007)
External pressure to implement	(Sriram, 2000) (Mukhopadhyay, 2002)
Cross-organizational implementation team	(Li, 2006) (Biehl, 2007)
Inter-organizational BPR	(Li, 1999) (Robey, 2002) (Nahm, 2003) (Wang, 2003) (Ngai, 2004) (Lee, 2005) (Li, 2006) (Biehl, 2007)
Own house in order	(Li, 1999) (Frohlich, 2002) (Wang, 2003) (Ngai, 2004) (Lee, 2005) (Li, 2006)
Strong integration with internal systems	(Wang, 2003) (Li, 2006)
Shared standards	(Lee, 2005) (Li, 2006)
Education and training	(Robey, 2002) (Nahm, 2003) (Ngai, 2004) (Jones, 2006)
Trust needed	(Li, 1999)
Project urgence	(Biehl, 2007)

Table 6-2: Critical success factors in implementing intra- and inter-organizational systems, with references.

DeLone (1992) did a thorough literature survey which explains the success factors for IT implementations. His survey results reveal that the six important factors are: (1) System quality, (2) Information quality, (3) Use, (4) User satisfaction, (5) Individual impact, (6) Organizational impact. With respect to the use of such technology Venkatesh (2003) compared a large body of implementation and adoption theories, and concludes that there are three important dimensions one should be aware of – which are in line with DeLone's findings – namely: (a) performance expectancy – meaning the expected usefulness of the system in the job (productivity increase, efficiency, etc.); (b) effort expectancy – easy to get acquainted with and utilize the system; (c) social influence – organizational support to use the system.

Not mentioned yet, but certainly related is the question why companies adopt certain software solutions. Moonen (2003) mentioned the important role that external parties, more specifically industry analysts (such as the Gartner Group, AMR Research, et cetera) and consultancy firms, have in an enterprise's buying decision. Buying therefore might be less rational than one would expect. Furthermore, one should realize that excess inertia, such as high switching costs due to the utilization of previous generations of (legacy) technologies or standards make it often very hard to switch to more open and better standards or solutions – as Zhu (2006) made clear in the case of EDI investments.

7 Conclusions

In this concluding section, we aim to answer our main question: “How should we plan and execute logistics in supply chains that aim to meet today’s requirements, and how can we support such planning and execution using IT?”

We first observe in Chapter 2 that today’s requirements in supply chains are characterized by a number of developments of which important elements are inter-organizational collaboration and more responsive and tailored towards specific demand.

We then conclude in Chapter 3 that enterprise systems fall short in meeting the requirements of these developments, and that the focus of planning and execution systems should move towards inter-enterprise (versus intra-enterprise) and event-driven (versus task-driven) mode. It is not obvious how to design the appropriate inter-organizational system (IOS) in a given business situation. We reflect on the development of IOS, architectural types, and technologies used. We point out that IOS may support planning in a progressive way: from supporting information exchange and henceforth enable synchronized planning within the organizations towards the capability to do network planning based on available information throughout the network.

One of the issues that comes up is to what extent we should centralize information and decision capabilities. IT as such supports both a centralized approach and a decentralized approach in planning. In Chapter 4, we provide a framework for planning systems, in which we contrast centralized versus decentralized decision capabilities, local and global impacts of these decisions, information available local and global, single versus multi-objective decision making, and information relevant to local versus global events. In this rich landscape of possible configurations, the centralized and fully decentralized approaches are two extremes. We then discuss some of the pros and cons of these approaches.

In Chapter 5, we define and discuss agent based systems and in particular multi agent systems (MAS). We emphasize the issue of the role of MAS coordination architectures, and then explain that transportation is, next to production, an important domain in which MAS can and actually are applied. However, implementation is not widespread and some implementation issues are explored.

In this manner, we conclude that planning problems in transportation have characteristics that comply with the specific capabilities of agent systems. In particular, these systems are capable to deal with inter-organizational and event-driven planning settings, hence meeting today’s requirements in supply chain planning and execution.

However, there are many issues that need to be dealt with. Apart from the list of factors that hinder adoption of agent based systems in industry, stated in Chapter 6, the question of the relative performance of centralized versus decentralized planning systems has not been answered conclusively.

Bibliography

1. Alt, R. and S. Klein (1998). Learning from failure: The myths and magic of electronic transportation markets. HICSS 1998, Hawai, IEEE press.
2. Anderson-Lehman, R., H. J. Watson, et al. (2004). "Continental Airlines Flies High With Real-Time Business Intelligence." MIS Quarterly Executive 3(4): 163-176.
3. Androulakis, I.P., and Reklaitis, G.V. "Approaches to asynchronous decentralized decision making," Computers and Chemical Engineering (23:3) 1999, pp 341-355.
4. Arunachalam, R. and N. M. Sadeh (2005). "The supply chain trading agent competition." Electronic Commerce Research and Applications 4(1): 66-84.
5. Bakos, J. Y. (1991). "A Strategic Analysis of Electronic Marketplaces." MIS Quarterly 15(3): 295.
6. Bakos, J. Y. (1997). "Reducing buyer search costs: Implications for electronic marketplaces." Management Science 43(12): 1676-1692.
7. Baalen, P. J. v., M. v. Oosterhout, et al. (2000). Dynamics in Setting up an EDI Community. Delft, Eburon Publishers.
8. Barber, K.S., T. H. Liu, and D. C. Han, 1999. Agent oriented design. Technical report: TR98-UT-LIPS-AGENTS-02, The university of Texas at Austin.
9. Belecheanu, R. A., S. Munroe, et al. (2006). Commercial applications of agents: lessons, experiences and challenges. AAMAS 2006, Hakodate, Japan ACM Press
10. Bergeron, F. and L. Raymond (1997). "Managing EDI for corporate advantage: A longitudinal study." Information & Management 31(6): 319-333.
11. Bharati, P. and A. Chaudhury (2006). "Studying the current status of technology adoption." Communications of the Acm 49(10): 88-93.
12. Biegler, L.T., and Grossmann, I.E. "Retrospective on optimization," Computers and Chemical Engineering (28:8) 2004, pp 1169-1192.
13. Biehl, M. (2007). "Success factors for implementing global information systems" Communications of the ACM 50(1): 52-58
14. Bodendorf, F. and R. Zimmermann (2005). "Proactive supply-chain event management with agent technology." International Journal of Electronic Commerce 9(4): 57-89.
15. Bonabeau, E. and C. Meyer (2001). "Swarm Intelligence. A whole new way to think about business." Harvard Business Review(May 2001): 107-114.
16. Bose, I. and R. Pal (2005). "Auto-ID: Managing anything, anywhere, anytime in the supply chain." Communications of the ACM 48(8): 100-106.
17. Bowersox, D. J., D. J. Closs, et al. (2003). "How to Master Cross-Enterprise Collaboration." Supply Chain Management Review 7(4): 18-27.
18. Busschbach, E. v., B. Pieterse, et al. (2002). Support of Virtual Enterprises by an Integration Infrastructure. Proceedings of PRO-VE '02, 1-3 May 2002, Sesimbra, Portugal, Kluwer Academic Publishers, Boston.
19. Buxmann, P., A. von Ahsen, et al. (2004). "Usage and evaluation of Supply Chain Management Software - results of an empirical study in the European automotive industry." Information Systems Journal 14(3): 295-309.
20. Caridi, M. and S. Cavalieri (2004). "Multi-agent systems in production planning and control: an overview." Production Planning & Control 15(2): 106-118.
21. Carr, N. G. (2003). "IT doesn't matter." Harvard Business Review 81(5): 41-+.
22. Chang, P-L, Lin, C-T., (1991) On the effect of centralization on expected costs in a multi-location newsboy problem. The Journal of Operational Research Society, Vol. 42, No. 11. pp. 1025 – 1030.
23. Chen, F. G. (1999). "Decentralized supply chains subject to information delays." Management Science 45(8): 1076-1090.
24. Chen, I. J. and A. Paulraj (2004). "Towards a theory of supply chain management: the constructs and measurements." Journal of Operations Management 22(2): 119-150.
25. Cho, Y., and Sahni, S. "Preemptive scheduling of independent jobs with release and due times on open, flow and job shops," Operations Research (29:3) 1981, pp 511-522.
26. Chopra, A. and M. P. Singh, 2004. Nonmonotonic commitment machines. AAMAS, ACL Workshop, LNAI 2922, pp. 183–200. Springer.
27. Chu, S. C. K., C. K. Y. Lin, K. Y. K. Ng, 1991. Centralized versus Decentralized Manpower Resource Planning: the case of a Hong Kong Company. Journal of the Operational Research Society. Vol. 42, No. 7, pp. 525-536.
28. Churchman, C.W., Ackoff, R.L., and Arnoff, E.L. Introduction to Operations Research John Wiley, New York, 1957.

29. Clemons, E. K. and M. C. Row (1991). "Sustaining It Advantage - the Role of Structural Differences." Mis Quarterly 15(3): 275-292.
30. Cohen, P.R. and H.J. Levesque, 1990, Rational interaction as the basis for communication, in: P.R. Cohen, J. Morgan and M.E. Pollack, eds., Intentions in Communication, MIT Press, Cambridge, MA.
31. Cohen, P.R. and H.J. Levesque, 1990. Intention is Choice with Commitment. Artificial Intelligence. 42, pp. 213-261, Elsevier Science Publishers B.V. (North-Holland).
32. Coltman, T., T. M. Devinney, et al. (2001). "E-business: Revolution, evolution, or hype?" California Management Review 44(1): 57-+.
33. Cordeau, J.F., Gendreau, M., Laporte, G., Potvin, J.Y., and Semet, F. "A guide to vehicle routing heuristics," Journal of the Operational Research Society (53:5) 2002, pp 512-522.
34. Corkill, D. 1979. Hierarchical Planning in a Distributed Problem-Solving Environment. Proceedings of the 7th International Joint Conference on AI.
35. Cox, J.S. and E. H. Durfee, 2005. An efficient algorithm for multiagent plan coordination. Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems. pp. 828-835.
36. Davenport, T. H. (1998). "Putting the enterprise into the enterprise system." Harvard Business Review 76(4): 121-+.
37. Davenport, T. H. and J. D. Brooks (2004). "Enterprise systems and the supply chain." Journal of Enterprise Information Management 17(1): 8.
38. Davenport, T. H. and J. G. Harris (2005). "Automated Decision Making Comes of Age." MIT Sloan Management Review 46(4): 83.
39. Davidsson, P., L. Henesey, et al. (2005). "An analysis of agent-based approaches to transport logistics." Transportation Research Part C-Emerging Technologies 13(4): 255-271.
40. Davidsson, P. and S. Johansson (2003). Evaluating multi-agent system architectures: A case study concerning dynamic resource allocation. Engineering Societis in the Agents World Iii. Berlin, SPRINGER-VERLAG BERLIN. 2577: 170-183.
41. Davis, R., and Smith, R.G. "NEGOTIATION AS A METAPHOR FOR DISTRIBUTED PROBLEM SOLVING," Artificial Intelligence (20:1) 1983, pp 63-109.
42. Decker, K. and V. Lesser, 1992. Generalizing the Partial Global Planning Algorithm. International Journal on Intelligent Cooperative Information Systems, Volume 1, Number 2, pp. 319-346.
43. Dedrick, J. and K. L. Kraemer (2005). "The impacts of IT on firm and industry structure: The personal computer industry." California Management Review 47(3): 122-+.
44. DeLone, W. H. and E. R. McLean (1992). "Information Systems Success: The Quest for the Dependent Variable." Information Systems Research 3(1): 60-95.
45. Desouza, K. C. (2001). "Intelligent agents for competitive intelligence: Survey of applications." Competitive Intelligence Review 12(4): 57.
46. Durfee, E. H., and V. R. Lesser, 1987. Using Partial Global Plans to Coordinate Distributed Problem Solvers. Proceedings of the Tenth International Joint Conference on Artificial Intelligence, pp. 875-883.
47. Durfee, E.H. and V.R.Lesser, 1991. Partial Global Planning: A Coordination Framework for Distributed Hypothesis Formation. IEEE Transactions on Systems, Man, and Cybernetics, Volume 21, Number 5, pp. 1167-1183.
48. Durfee, E. H., V. R. Lesser, and D. D. Corkill, 1989. Trends in Cooperative Distributed Problem Solving, IEEE Transactions on Knowledge and Data Engineering, Vol. 1, No. 1, pp. 63-83.
49. Edwards, P., M. Peters, et al. (2001). "The effectiveness of information systems in supporting the extended supply chain." Journal of Business Logistics 22(1): 1-27.
50. Endriss, U., N. Maudet, F. Sadri, and F. Toni, 2003. Protocol Conformance for Logic-based Agents. In G. Gottlob and T. Walsh, editors, Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-2003). Morgan Kaufmann Publishers.
51. Evernden, R. and E. Evernden (2003). "Third-generation information architecture." Communications of the ACM 46(3): 95-98.
52. Farrell, D. (2003). "The real new economy." Harvard Business Review 81(10): 104-+.
53. Findeisen, W., F.N., Bailey, M. Brdys, K. Malinowski, P. Tatiewski, and A. Wozniak, (1979), Control and Coordination of Hierarchical Systems. Wiley.
54. Finin, T., R. Fritzson, D. McKay and R. McEntire, 1994. KQML as an Agent Communication Language, The Proceedings of the Third International Conference on Information and Knowledge Management (CIKM'94), ACM Press.

55. Fischer, K., N. Kuhn, H. J. Muller, J.P. Muller, and M. Pischel, 1993. Sophisticated and distributed: the transportation domain. In Proceedings of MAAMAW-93, Fifth European workshop on modeling autonomous agents in a multi-agent word.
56. Fischer, K., J. P. Muller, and M. Pischel, 1996. Cooperative transportation scheduling: an application domain for DAI. Applied Artificial Intelligence. Vol 10, pp. 1-33.
57. Fischer, F., M. Rovatsos, G. Weiss, 2005. Acquiring and Adapting Probabilistic Models of Agent Conversation. In Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems, The Netherlands, pp. 106-113.
58. Fleischmann, B., S. Gnutzmann, et al. (2004). "Dynamic vehicle routing based on online traffic information." Transportation Science 38(4): 420-433.
59. Franklin, S. and A. Graaesser, 1996. Is it and agent, or just a program?: A taxonomy for autonomous agents. In Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages, Springer-Verlag.
60. Frohlich, M. T. (2002). "e-integration in the supply chain: Barriers and performance." Decision Sciences 33(4): 537-556.
61. Galbraith, J. R. (1974). "Organization design: an information processing view." Interfaces 4(3): 28-36.
62. Galbraith, J., 1977. Organization Design. Addison-Wesley, Reading, MA.
63. Gendreau, M., Laporte, G., and Séguin, R. "Stochastic vehicle routing," European Journal of Operational Research (88:1) 1996, pp 3-12.
64. Gendreau, M. and J. Y. Potvin (2004). "Issues in real-time fleet management." Transportation Science 38(4): 397-398.
65. Georgeff, M., 1983. Communication and Interaction in Multi-Agent Planning, Proceedings of the Third National Conference on Artificial Intelligence, pp. 125-129.
66. Georgeff, M., 1984. A Theory of Action for Multi Agent Planning, Proceedings of the Fourth National conference on Artificial Intelligence, pp. 121-125.
67. Glass, R. L. (2003). "One giant step backward." Communications of the ACM 46(5): 21.
68. Glass, R. L. (2005). "The first business application: A significant milestone in software history - Celebrating the pivotal role of the first business applications software." Communications of the ACM 48(3): 25-26.
69. Gosain, S., A. Malhotra, et al. (2004). "Coordinating for flexibility in e-business supply chains." Journal of Management Information Systems 21(3): 7-45.
70. Grando, A. and C. Walton, 2006. MAPa: a Language for Modelling Conversations in Agent Environments. Proceedings of the Intelligent Information Processing and Web Mining Conference 2006 (IIPWM06), Ustron, Poland.
71. Grosvenor, F. and T. Austin (2001). "Austin.“Cisco’s eHub Initiative.”" Supply Chain Management Review(July-August): 28–35.
72. Hagel III, J. and J. S. Brown (2001). "Your next IT strategy." Harvard Business Review 79(9): 105.
73. Haigh, T. (2001). "Inventing information systems: The systems men and the computer, 1950-1968." Business History Review 75(1): 15-61.
74. Hayes-Roth, B., 1985. A blackboard architecture for control. Artificial Intelligence. Vol. 26, pp. 251-321. Elsevier Science Publishers B.V., North Holland.
75. Heijden, H. van der, R. Wagenaar, et al. (1995). "Redesigning process control mechanisms using EDI. An agency-theoretic perspective." Proceedings of the 28th Hawaii International Conference on System Sciences.
76. Heng, M. S. H. (2003). "Understanding Electronic Commerce from a Historical Perspective." Communications of the Association for Information Systems 12: 104-118.
77. Hess, T. J., L. P. Rees, et al. (2000). "Using autonomous software agents to create the next generation of decision support systems." Decision Sciences 31(1): 1-31.
78. Hong, Y. B. (2002). "A new framework for interorganizational systems based on the linkage of participants' roles." Information & Management 39(4): 261-270.
79. Huang, C. C. and W. Y. Liang (2004). "Object-oriented design processes of electronic commerce in a multi-agent environment." Journal of Intelligent Manufacturing 15(5): 661-678.
80. Jap, S. D. and J. J. Mohr (2002). "Leveraging Internet technologies in B2B relationships." California Management Review 44(4): 24-+.
81. Jennings, N. R., 1993. Commitments and Conventions: The Foundation of Coordination in Multi-Agent Systems. The Knowledge Engineering Review, 8 (3). pp. 223-250.
82. Jennings, N. R., K. Sycara, and M. Wooldridge, 1998. A roadmap of agent research and development. Autonomous Agents and Multi-Agent Systems. Vol 1, pp. 7-38.

83. Jennings, N. R., 2000. On agent-based software engineering. Artificial Intelligence. Vol 117, pp. 277-296.
84. Jennings, N. R. (2001). "An agent-based approach for building complex software systems - Why agent-oriented approaches are well suited for developing complex, distributed systems." Communications of the ACM 44(4): 35-41.
85. Jennings, N. R. and S. Bussmann (2003). "Agent-based control systems: Why are they suited to engineering complex systems?" Control Systems Magazine, IEEE 23(3): 61-73.
86. Johnson, E.L., Nemhauser, G.L., and Savelsbergh, M.W.P. "Progress in Linear Programming-Based Algorithms for Integer Programming: An Exposition," INFORMS Journal on Computing (12:1) 2000, pp 2-23.
87. Jones, M. C., M. Cline, et al. (2006). "Exploring knowledge sharing in ERP implementation: an organizational culture framework." Decision Support Systems 41(2): 411-434.
88. Klusch, M. (2001). "Information agent technology for the Internet: A survey." Data & Knowledge Engineering 36(3): 337-372.
89. Kohout, R. and K. Erol, 1999. In-time agent-based vehicle routing with a stochastic improvement heuristic. Eleventh Conference on Innovative Applications of Artificial Intelligence. Orlando, FL.
90. Kopczak, L. R. and M. E. Johnson (2003). "The supply-chain management effect." Mit Sloan Management Review 44(3): 27-34.
91. Kulmala, H. I., A. Vahteristo, et al. (2005). "Interorganizational operations in value chains - Experiences from networked software firms." Production Planning & Control 16(4): 378-387.
92. Kumar, K. and H. G. v. Dissel (1996). "Sustainable collaboration: Managing conflict and cooperation in interorganizational systems." MIS Quarterly 20(3): 279-300.
93. Lageweg, B.J., Lenstra, J.K., and Kan, A.H.G.R. "Job-shop scheduling by implicit enumeration," Management Science (24:4) 1977, pp 441-450.
94. Larson, R. and Odoni, A. (1981) Urban Operations Research, Prentice-Hall, NJ – out of print – available via http://web.mit.edu/urban-or-book/www/book/index.html.
95. Lee, G. G., H. F. Lin, et al. (2005). "Influence of environmental and organizational factors on the success of internet-based interorganizationaI systems planning." Internet Research 15(5): 527-543.
96. Lee, H., A. F. Farhoomand, et al. (2004). "Innovation Through Supply Chain Reconfiguration " MIS Quarterly Executive 3(3): 131-142.
97. Lee, H. L., V. Padmanabhan, et al. (1997). "Information distortion in a supply chain: The bullwhip effect." Management Science 43(4): 546.
98. Leeuw, A.C.J.d. "Systems theory and organization theory," Technische Universiteit Eindhoven, 1974.
99. Leong H. W., Liu, M., 2006. A multi-agent algorithm for vehicle routing problem with time window. SAC ’06. April, 23-27, Dijon France. Available from the Association of Computing Machinery.
100. Lesser, V.R. and D.D. Corkill, 1983. The distributed vehicle monitoring testbed: A tool for investigating distributed problem-solving networks, AI Magazine, Vol 4., No. 3, pp 15-33.
101. Lesser, V. R. and D. D. Corkill, 1987. Distributed problem solving In Shapiro, S. C. (ed), Encyclopedia of Artificial Intelligence, pp 245-251, John Wiley and Sons.
102. Levy, M. and P. Powell (1998). "SME flexibility and the role of information systems." Small Business Economics 11(2): 183-196.
103. Li, F. and H. Williams (1999). "Interfirm collaboration through interfirm networks." Information Systems Journal 9(2): 103-115.
104. Li, J., R. Sikora, et al. (2006). "A strategic analysis of inter organizational information sharing." Decision Support Systems 42(1): 251-266.
105. Libosvar, C. M. (1988) Hierarchies in Production Management and Control: A Survey. Laboratory for Information and Decision Systems, Massachusetts Instute of Technology, Cambridge, MA.
106. Lima, R. M., R. M. Sousa, et al. (2006). "Distributed production planning and control agent-based system." International Journal of Production Research 44(18-19): 3693-3709.
107. Luck, M. and M. d’Inverno, 2001. A conceptual framework for agent definition and development. The Computer Journal. Vol 44, pp. 1-20
108. Luck, M. and M. d’Inverno, 2003. Unifying agent systems. Annals of Mathematics and Artificial Intelligence. 37: pp. 131-167.
109. Luck, M., P. McBurney, et al. (2004). "A Manifesto for Agent Technology: Towards Next Generation Computing." Autonomous Agents and Multi-agent Systems 9: 203-252.
110. Luck, M., P. McBurny, O. Shehory, S. Willmott, and the AgentLink Community, 2005. Agent Technology Roadmap: A Roadmap for Agent Based Computing. Published by the University of Southampton on behalf of AgentLink III.

111. Maes, P. (1994). "Agents that reduce work and information overload." Association for Computing Machinery. Communications of the ACM 37(7): 30.
112. Maes, P., R. H. Guttman, et al. (1999). "Agents that buy and sell." Association for Computing Machinery. Communications of the ACM 42(3): 81.
113. Mahoney, M. S. (1988). "The History of Computing in the History of Technology." Annals of the History of Computing 10(2): 113.
114. Makukha, K. and R. Gray (2004). "Logistics partnerships between shippers and logistics service providers: the relevance of strategy." International Journal of Logistics - Research and Applications 00007(00004): 361-378.
115. Malone, T. W., J. Yates, et al. (1987). "Electronic markets and electronic hierarchies." Communications of the ACM 30(6): 484-497.
116. Malone, T.W., and Crowston, K. "Interdisciplinary study of coordination," ACM Computing Surveys (26:1) 1994, pp 87-119.
117. Marik, V. and D. McFarlane (2005). "Industrial adoption of agent-based technologies." IEEE Intelligent Systems 20(1): 27-35.
118. Markus, M. L. (2000). "Paradigm Shifts - E-Business and Business/Systems Integration." Communications of the Association for Information Systems 4(10).
119. Marschak, T. (1969) On the Comparison of Centralized and Decentralized Economies. The American Economic Review. Vol 59, No 2, Papers and Proceedings of the Eighty-first Annual Meeting of the American Economic Association. pp 525-532.
120. Mathews, J. "Organizational foundations of intelligent manufacturing systems - the holonic viewpoint," Computer Integrated Manufacturing Systems (8:4) 1995, pp 237-243.
121. McFarlane, D. and Y. Sheffi (2003). "The impact of automatic identification on supply chain operations." The International Journal of Logistics Management 14(1): 1-17.
122. Mes, M., van der Heijden, M., van Harten, A., 2006. Comparison of agent-based scheduling to look-ahead heuristics for real-time transportation problems. European Journal of Operational Research, doi: 10.1016/j.ejor.2006.02.051.
123. Mesarovic, M. D., D. Macko and Y. Takahara, (1970). Theory of Multilevel Hierarchical Systems, Academic Press, New York.
124. Milojicic, D. (2004). "Microsoft's Jim Gray on Computing's Breakthoughs, Lessons, and Future." IEEE Distributed Systems Online 5(1).
125. Monahan, J.P., (1984) A quantity discount pricing model to increase vendor profits. Management Science. Vol. 30, No. 6, pp. 720-726.
126. Moonen, H. M. (2003). Third Parties' Influence on Sales in the Enterprise Software Market, RSM Erasmus University, working paper.
127. Mukhopadhyay, T. and S. Kekre (2002). "Strategic and operational benefits of electronic integration in B2B procurement processes." Management Science 48(10): 1301-1313.
128. Nahm, A. Y., M. A. Vonderembse, et al. (2003). "The impact of organizational structure on time-based manufacturing and plant performance." Journal of Operations Management 21(3): 281-306.
129. Ngai, E. W. T. and A. Gunasekaran (2004). "Implementation of EDI in Hong Kong: an empirical analysis." Industrial Management & Data Systems 104(1-2): 88-100.
130. Nissen, M. E. (2001). "Beyond electronic disintermediation through multi-agent systems." Logistics Information Management 14(4): 256-275.
131. Nissen, M. E. and K. Sengupta (2006). "Incorporating Software Agents into Supply Chains: Experimental Investigation with a Procurement Task." MIS Quarterly 30(1): 145.
132. Nwana, H.S., L. Lee, and N.R. Jennings, 1996. Co-ordination in software agent systems. BT Technology Journal, Vol. 14, No. 4, pp. 79-88.
133. Odell, J. J., H. V. Parunak, et al. (2002). Modeling agents and their environment. Agent-Oriented Software Engineering Iii. Berlin, SPRINGER-VERLAG BERLIN. 2585: 16-31.
134. Ordanini, A. (2005). "The effects of participation on B2B exchanges: A resource-based view." California Management Review 47(2): 97-+.
135. Papadimitriou, C.H., and Steiglitz, K. Combinatorial optimization Dover publications, Mineala NY, USA, 1998.
136. Parunak, H. V. D. (1999). Industrial and practical applications of DAI. Multiagent systems: a modern approach to distributed artificial intelligence MIT Press Cambridge, MA, USA: 377-421.
137. Parunak, H. V. D. (2000). "A Practitioners' Review of Industrial Agent Applications." Autonomous Agents and Multi-Agent Systems 3: 389-407.
138. Peña, J., M. G. Hinchey, et al. (2006). "Multi-agent system product lines: challenges and benefits" Communications of the ACM 49(12): 82-84.

139. Peterson, D. M. (2003). "Power to the BPEL: A technology for Web services." Business Communications Review 33(6): 54.
140. Petrie, C. and C. Bussler (2003). "Service agents and virtual enterprises: A survey." IEEE Internet Computing 7(4): 68-78.
141. Porter, M. E. (2001). "Strategy and the Internet." Harvard Business Review 79(3): 62-+.
142. Rai, A., R. Patnayakuni, et al. (2006). "Firm performance impacts of digitally enabled supply chain integration capabilities." Mis Quarterly 30(2): 225-246.
143. Raman, A. (1995). "Coordination: An overview." Harvard Business School teaching note(9-696-001).
144. Rao, A. S. and M. P. Georgeff, 1991. Modeling rational agents within a BDI-architecture. In R. Fikes and E. Sandwall, editors, Proceedings of Knowledge Representation and Reasoning (KR&R-91), pp 473-484. Morgan Kaufmann Publishers: San Mateo, CA.
145. Robey, D., J. W. Ross, et al. (2002). "Learning to implement enterprise systems: An exploratory study of the dialectics of change." Journal of Management Information Systems 19(1): 17-46.
146. Roth, V. (2004). Obstacles to the adoption of mobile agents. 2004 IEEE International Conference on Mobile Data Management.
147. Rovatsos, M., M. Nickles, G. Weiss. An Empirical Model of Communication in Multiagent Systems. In Dignum, F. (ed.), Advances in Agent Communication, LNAI 2922, Springer-Verlag, Berlin, 2004
148. Rudowsky, I. S. (2004). "Intelligent Agents." Communications of the Association for Information Systems 14: 275-290.
149. Sabuncuoglu, I., and Bayiz, M. "Analysis of reactive scheduling problems in a job shop environment," European Journal of Operational Research (126:3) 2000, pp 567-586.
150. Saeed, K. A., M. K. Malhotra, et al. (2005). "Examining the impact of interorganizational systems on process efficiency and sourcing leverage in buyer-supplier dyads." Decision Sciences 36(3): 365-396.
151. Samuelson, D.A. and C.M. Macal, 2006. Agent-based simulation comes of age. OR/MS Today. Vol. 33, Number 4, pp. 34-38.
152. Sandholm, T. (1999). Distributed Rational Decision Making. Multiagent Systems A Modern Approach to Distributed Artificial Intelligence. G. Weiss. Cambridge, MA, USA, MIT press: 201-258.
153. Satapathy, G., S. R. T. Kumara, et al. (1998). "Distributed intelligent architecture for logistics (DIAL)." Expert Systems with Applications 14(4): 409-424.
154. Schleiffer, R., 2005. An intelligent agent model. European Journal of Operational Research. Vol. 166, pp. 666-693.
155. Schneeweiss, C. (2003a) Distributed decision making – a unified approach. European Journal of Operational Research. Vol 150, pp 237-252.
156. Schneeweiss, C., (2003b). Distributed Decision Making, Second Edition. Springer-Verlag, Berlin.
157. Schneeweiss, C., K. Zimmer. (2004) Hierarchical coordination mechanisms within the supply chain. European Journal of Operational Research, No. 153, pp. 687-703.
158. Shen, W. and D. Norrie (1999). "Agent-based Systems for Intelligent Manufacturing: A state-of-the-art survey." Knowledge and Information Systems 1(2): 129-156.
159. Sheombar, H. S. (1997). "Logistics coordination in Dyads: Some theoretical foundations for EDI-induced redesign." Journal of Organizational Computing and Electronic Commerce 7(2-3): 153-184.
160. Siau, K. (2003). "Interorganizational systems and competitive advantages - Lessons from history." Journal of Computer Information Systems 44(1): 33-39.
161. Sierra, C. (2004). "Agent-Mediated Electronic Commerce." Autonomous Agents and Multi-agent Systems(9): 285-301.
162. Simon, H.A. "Near decomposability and the speed of evolution," Industrial and Corporate Change (11:3) 2002, pp 587-599.
163. Singh, M., Agent Communication Languages: Rethinking the Principles. IEEE Computer, volume 31, number 12, December 1998, pages 40-47.
164. Singh, M., 2000. A Social Semantics for Agent Communication Languages. Proceedings of the IJCAI Workshop on Agent Communication Languages, Springer-Verlag.
165. Singh, M., 2007. Formalizing Communication Protocols for Multiagent Systems. Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI), January – available at: http://www.csc.ncsu.edu/faculty/singh/papers/.
166. Singh, N., K.-h. Lai, et al. (2007). "Intra-organizational perspectives on IT-enabled supply chains." Communications of the ACM 50(1): 59-65

167. Smith, R. G., 1980. The contract net protocol: High-level communication and control in a distributed problem solver. IEEE Transactions on Computers. Vol, C-29, No. 12, pp. 1104-1113.
168. Sodhi, M. S. (2001). "Applications and opportunities for operations research in Internet-enabled supply chains and electronic marketplaces." Interfaces 31(2): 56-69.
169. Spieck, S., M. Weigelt, et al. (1995). "Decentralized problem solving in logistics and production with partly intelligent agents and comparison with alternative approaches." Proceedings of the 28th Annual Hawaii International Conference on System Sciences.
170. Sridharan, U. V., W. R. Caines, et al. (2005). "Implementation of supply chain management and its impact on the value of firms." Supply Chain Management-an International Journal 10(3-4): 313-318.
171. Sriram, R., V. Arunachalam, et al. (2000). "EDI adoption and implementation: an examination of perceived operational and strategic benefits and controls." Journal of Information Systems 14(1): 37-52.
172. Stefansson, G. (2002). "Business-to-business data sharing: A source for integration of supply chains." International Journal of Production Economics 75(1-2): 135-146.
173. Subramani, M. (2004). "How do suppliers benefit from information technology use in supply chain relationships?" MIS Quarterly 28(1): 45-73.
174. Taillard, E. "Some efficient heuristic methods for the flow shop sequencing problem," European Journal of Operational Research (47:1) 1990, pp 65-74.
175. Tsamardinos, I., M. E. Pollack, and J. F. Horty, 2000. Merging plans with quantitative temporal constraints, temporally extended actions, and conditional branches. In Proceedings of the Fifth International Conference on AI Planning Systems, pp. 264-272.
176. Walton, C., 2003. Dialogue Protocols for Multi-Agent Systems. Published as Informatics Technical Report EDI-INF-RR-0183, University of Edinburgh.
177. Wang, E. T. G. and J. C. F. Tai (2003). "Factors affecting information systems planning effectiveness: organizational contexts and planning systems dimensions." Information & Management 40(4): 287-303.
178. Wareham, J., J. G. Zheng, et al. (2005). "Critical themes in electronic commerce research: a meta-analysis." Journal of Information Technology 20(1): 1-19.
179. Weerdt, M. de, A. ter Mors, and C. Witteveen, 2005. Multi-agent Planning: An introduction to planning and coordination. In Handouts of the European Agent System Summer School, pp. 1-32.
180. Wooldridge, M., and N.R. Jennings, 1995. Intelligent agents: Theory and practice. The Knowledge Engineering Review, vol. 10 (2) pp. 115-152.
181. Wooldridge, M., 1999. Intelligent Agents. In G. Weiss, editor: Multiagent Systems, The MIT Press.
182. Worley, J. H., G. R. Castillo, et al. (2002). "Adding decision support to workflow systems by reusable standard software components." Computers in Industry 49(1): 123-140.
183. Wortmann, H. and N. Szirbik (2001). "ICT issues among collaborative enterprises: from rigid to adaptive agent-based technologies." Production Planning & Control 12(5): 452-465.
184. Zani, W. M. (1970). "Real-time information systems: A comparative economic analysis." Management Science 16(6): 350-355.
185. Zhu, K., K. L. Kraemer, et al. (2006). "Migration to open-standard interorganizational systems: Network effects, switching costs, and path dependency." Mis Quarterly 30: 515-539.
186. Zimmermann, R., S. Winkler, et al. (2006). Agent-based Supply Chain Event Management – Concept and Assessment. 39th Hawaii International Conference on System Sciences, IEEE.

Publications in the Report Series Research[*] in Management

ERIM Research Program: “Business Processes, Logistics and Information Systems”

2008

An Evolutionary Framework for Determining Heterogeneous Strategies in Multi-Agent Marketplaces
Alexander Babanov, Wolfgang Ketter and Maria Gini
ERS-2008-002-LIS
http://hdl.handle.net/1765/10972

Choosing between Auctions and Negotiations in Online B2B Markets for IT Services: The Effect of Prior Relationships and Performance
Uladzimir Radkevitch, Eric van Heck and Otto Koppius
ERS-2008-004-LIS
http://hdl.handle.net/1765/11288

Key Issues in Expansion of End-User Mobile Communication in China
Sunanda Sangwan, Guan Chong and Louis-Francois Pau
ERS-2008-011-LIS
http://hdl.handle.net/1765/11762

Some Comments on the Question Whether Co-Occurrence Data Should Be Normalized
Ludo Waltman and Nees Jan van Eck
ERS-2008-014-LIS
http://hdl.handle.net/1765/11810

Bibliometric Mapping of the Computational Intelligence Field
Nees Jan van Eck and Ludo Waltman
ERS-2008-015-LIS
http://hdl.handle.net/1765/11811

Demand Management Opportunities in E-fulfillment: What Internet Retailers Can Learn from Revenue Management
Niels Agatz, Ann Campbell, Moritz Fleischmann, Jo van Nunen and Martin Savelsbergh
ERS-2008-021-LIS
http://hdl.handle.net/1765/12244

Time Slot Management in Attended Home Delivery
Niels Agatz, Ann Campbell, Moritz Fleischmann and Martin Savelsbergh
ERS-2008-022-LIS
http://hdl.handle.net/1765/12245

Mobile Service Affordability for the Needy, Addiction, and ICT Policy Implications
L-F Pau
ERS-2008-023-LIS
http://hdl.handle.net/1765/12246

CBPRS: A City Based Parking and Routing System
J.L. Boehlé, L.J.M. Rothkrantz and M. van Wezel
ERS-2008-029-LIS
http://hdl.handle.net/1765/12467

Coping with Costly Bid Evaluation in Online Reverse Auctions for IT Services
Uladzimir Radkevitch, Eric van Heck and Otto Koppius
ERS-2008-039-LIS
http://hdl.handle.net/1765/12871

Multi Agent Systems in Logistics: A Literature and State-of-the-art Review
Niels Lang, Hans M. Moonen, F. Jordan Srour and Rob A. Zuidwijk
ERS-2008-043-LIS
http://hdl.handle.net/1765/12902
