

Component Based Software Architecture for Supply Chain Management Systems

Ryan Darby

Department of Computer Science, University of Pretoria
rdarby@cs.up.ac.za

Judith Bishop

Department of Computer Science, University of Pretoria
jbishop@cs.up.ac.za

Willem Cilliers

Graduate School of Management, University of Pretoria
willemc@za.sappi.com

Abstract

The supply chain environment is very flexible due to changing demand and pressure from competitors. This article determines that a software architecture is needed to allow information systems to be realigned with the changing supply chain without effort or delay. The characteristics and requirements of the supply chain of the future matches the advantages and characteristics of a software component architecture. The software engineering methodology of component based development, which builds systems using the software structures Components, allows the underlying software architecture to be rapidly changeable. We show how components can be optimized for inter-operability, facilitating the integration of information systems as the supply chain view extends outside of the company. The required specialization of components before they are applicable to supply chain software is explained.

Authors

Ryan Darby

Ryan Darby is a supply chain integration and applications consultant with a South African Information Technology service provider focussing on the chemical industry. Ryan's involvement focuses on the identification of emerging technologies and "killer apps" within the extended enterprise. Ryan is a student of the Department of Computer Science of the University of Pretoria and has an MSc in computer science from the University of Pretoria, completed under supervision of Prof. Bishop of Computer Science, with an application area of supply chain management, under Willem Cilliers of the Graduate School of Management of the University of Pretoria. Ryan's research interests focus on the use of Information Technology to alter the nature of business, particularly supply chain management and the extended enterprise.

Judith Bishop

Judith Bishop is professor of computer science at the University of Pretoria, South Africa. She has a PhD in computer science from the University of Southampton, UK, and has been on academic staff there and at the University of the Witwatersrand. Her research interests include distributed systems, programming languages and web-based programming. She is a founder fellow of the SA Institute of Computer Scientists, and currently chairman of IFIP WG 2.4 (System Programming Languages).

Willem Cilliers

Willem Cilliers holds the position of Chair in Logistics within the Graduate School of Management of the University of Pretoria. Willem has an MBA degree from Pretoria and extensive experience within supply chain strategy consulting. Willem's research interests include benchmarking supply chain performance and supply chain strategy.

1. Introduction

Supply chain management is increasingly being regarded as a competitive factor in business. Companies are attempting to advance their level of supply chain proficiency, and by so doing redefine the rules of engagement of the supply chain. A future model of supply chain management is being created, which companies need to adopt to stay competitive.

Similarly, and in an effort to prepare for the future, information technology is used to provide the information required to manage a company effectively, and to automate business processes. The use of information technology in supply chain management is regarded as a key differentiator. However, this article contrasts the architecture of currently available information technology and shows that the existing applications supporting supply chain management do not meet the demands of future developments of the supply chain. For this purpose a model of the dynamic supply chain is created.

This article determines that a software architecture is needed that allows the information systems to be realigned with the changing supply chain without effort or delay. The supply chain environment is very flexible due to changing demand and pressure from competitors. The characteristics and requirements of the future supply chain matches the advantages and characteristics of a software component architecture. This article discusses components as used within the domain of supply chain management. The software engineering methodology Component Based Development, which builds systems using the software structures Components, allows the underlying software architecture to be rapidly changeable. In addition, components are by definition optimized for inter-operability, facilitating the integration of information systems as the supply chain view extends outside of the company. The required specialization of components before they are applicable to supply chain software is explained.

2.1 Traditional Definition of Supply Chain Management

Supply Chain Management is not simply a new buzzword for logistics (Metz 1998). However, there is often disagreement as to what supply chain management means, depending on the individual involved. This disagreement stems from the differing viewpoints of the participants in a supply chain, which encompasses all of the organisation and the organisations' partners in a direct or indirect way. Table 1 represents a view of supply chain definitions found in the literature.

To pull the definitions presented in table 1 together allows the definition of a supply chain as being :

- ?? A process encompassing all aspects from raw material origin to final consumption
- ?? Co-operation among supply chain members
- ?? Physical networks and structures
- ?? The adding of value by each supply chain participant
- ?? All leading to the delivery of value to the end consumer

A basic definition of the supply chain is given by (Poirier et. al. 1996), who regards the supply chain as the system through which organisations deliver their products and services to their customers. As further definitions are given, it will be seen that this definition focuses on the structural aspect only, and lacks mention of the core issues in supply chain management.
The Supply Chain Council defines supply chain management as encompassing every effort involved in producing and delivering a final product, from the supplier's supplier to the customer's customer (SCCa 1998). Four basic processes, Plan, Source, Make, Deliver, broadly define these efforts, which include managing supply and demand, sourcing raw materials and parts, manufacturing and assembly, warehousing and inventory tracking, order entry and order management, distribution across all channels, and delivery to the customer.
A further definition of supply chain management is that of (Franz et. al. 1994) : Supply chain management refers to the level of co-operation between respondents, their suppliers, their customers, and their customers' customers. Ultimately, supply chain management refers to the level of co-operation between the different companies from the point of origin until the final point of consumption.
(Bowersox et. al. 1996) reinforces the concept of integration and co-operation by regarding the basic notion of supply chain management as being grounded on the belief that efficiency can be improved by sharing information and joint planning.
The above definitions can be further refined by emphasising that the supply chain is a paradigm based on a business process which is focussed on relationships outside the enterprise, and on bringing maximum value to the end consumer (LaLonde et. al 1996).
A definition of supply chain management that encompasses the above is that of the Massachusetts Institute of Technology Centre for Transportation Studies, given by (Metz 1998) : "Integrated Supply Chain Management (ISCM) is a process-oriented, integrated approach to procuring, producing, and delivering products and services to customers. ISCM has a broad scope that includes sub-suppliers, suppliers, internal operations, trade customers, retail customers, and end users. ISCM covers the management of material, information, and funds flows. Note that we add the word 'integrated' to underscore the objective of integrating the many functions into the total process."

Table 1 : Traditional Definitions of Supply Chain Management

2.2 Limitations of the Traditional Definitions

However, the existing traditional definitions of supply chain management do not fully encompass the fundamental nature of the future of supply chain management as described below. There are several limitations to the traditional definitions :

- ? An over-emphasis on the supply of inventory. Most definitions refer to the movement of goods. Issues such as relationships among companies, inter-dependency, strategy etc. are not focussed on.
- ? Although the majority of companies are not yet following supply chain management practices, and have not incorporated supply chain management into their corporate strategies, there are many companies who have leapfrogged ahead of their competition by encompassing supply chain management. These companies are now facing the challenges of the next generation of supply chain management, namely virtual corporations, networked economies etc. They are playing by new rules, and thus only following the traditional supply chain concept will not allow competition with the leaders of the field.
- ? The key elements of supply chain definitions are typically those of integration between a set of companies. The definitions characterize a “snap-shot” of the life of those companies within a supply chain. There is no support for the dynamic nature of the supply chain, as discussed below.

3 Future Requirements of the Supply Chain

It is useful to discuss the requirements being recognized of an effective supply chain, as well as future requirements of supply chain. These requirements will allow for the creation of a checklist against which software support can be compared for effectiveness.

3.1 Agility

Agility deserves more than just a mention as being one of the goals of supply chain management. The concept of agility is central to the virtual supply chain, and encompasses several key goals of supply chain management.

Agility is an emerging concept in supply chain management, and still not clearly defined. A first definition of agility is given in the World Class Logistics study by the Council of Logistics Management (CLM 1995). Agility is defined as the achievement and retention of competitiveness and customer success, by providing quality supply chain management. Agility is top down driven, responding to large forces by an overall strategy focused on thriving in an unpredictable environment. Current supply chain management strategies are aimed at minimising the time from placing the order to collecting the cash, among other things. The aim of agility is to minimise the time from making the concept to collecting the cash. Agility is thus the competency that sustains world class performance over time (CLM 1998). (Goldman 1998) emphasises the aspect of agility being a greater concept than flexibility (defined below) by defining agility as being capable of operating profitably in a competitive environment of continually, and unpredictability, changing customer opportunities. A definition of agility that more than the above definitions recognises the extended nature of the future supply chain is given by (Greis et. al. 1997). Greis et. al. define agile supply chains as networks of strategically aligned firms which replace individual companies as the unit of competition, and are focussed on capturing specific market opportunities as they arise. These enterprises are different from traditional strategic alliances in that they are dynamic and flexible, and based on collaboration rather than competition.

The key features of agility have been introduced above, and can now be more clearly defined, as by (CLM 1995) :

- ? **Relevancy.** Relevancy is the ability to maintain focus on the changing needs of customers (CLM 1995).
- ? **Accommodation.** Accommodation is the ability to respond to unique customer requests (CLM 1995).
- ? **Flexibility.** Flexibility is the ability to adapt to unexpected circumstances (CLM 1995, Fawcett 1997).

Further features are provided by (Harrison 1998) :

- ? **Virtual Organisation.** The virtual organisation is a part of agility (Harrison 1998). Internal and external co-operation are the strategies of first choice. The aim is to bring agile products to market in minimum time by leveraging resources through co-operation. The virtual organisation is a means to provide agility.
- ? **Entrepreneurial.** (Harrison 1998) recommends that companies organise to thrive on change and uncertainty. Innovative, flexible organisation structures that promote rapid decision making are essential.
- ? **Knowledge-based.** As mentioned above, intellectual capital is essential to agility. The key differentiators in tomorrows supply chain will be people and information (Harrison 1998). Thus, agility embraces the notions of distributed authority, supported by maximising personnel and information technology resources.

By looking at the components of agility, it can be seen that agility is not a new technique or list of methods, but rather an approach to business. Agility is not in competition with the kinds of innovations that businesses have been exploring to get them to their current level. Agility uses those innovations where possible, but also takes a more competitive futuristic approach, and goes beyond certain existing practices. According to (Goldman 1998), agility defines a strategic framework for dealing with change. Agility exists on the strategic level, and uses tools such as information technology. This article enables agility by assessing an information system architecture that is designed to react to change.

3.2 Responsiveness

Stemming from agility, as a requirement of agility, is responsiveness. Responsiveness is a time based principle (Bowersox 1996), dealing with the ability to react to sudden potential and opportunities in time to take advantage of the opportunities. Responsiveness is characterised by the ability to capture customer demand as close to real time as possible, and have the supply chain respond as a direct result (Christopher 1993). A lessened dependence on forecasts is then possible, while achieving greater satisfaction of demands (Christopher et. al. 1997). (Gattorna et. al 1996a) refers to responsiveness as a world class attribute, under the concept of best practice. Although responsiveness can be used as a measure of best practice, being responsive to market change will no longer be a competitive differentiator in the future of virtual supply chains.

Responsiveness can be positioned by discussing anticipatory based operating in contrast to response based operations (Bowersox et. al. 1996). The fundamental difference between

anticipatory and response based arrangements is timing. Anticipatory arrangements are traditional and reflect best practice developed during a period prior to widespread availability of information technology. In contrast, responsiveness reflects strategies to exploit the potential of time based supply chain management.

Anticipatory practices were developed during a time period when business was primarily conducted on a transactional basis. Because information was not shared companies operated on the basis of long term forecasts. The operational goal was to build and push inventory to the next level in the channel. Because of high risk and cost associated with anticipatory practices, the prevailing relationship between supply chain members was adversarial. Response based arrangements stress co-operation and information sharing. (Stank et. al. 1996) has shown that information sharing positively affects responsiveness. Because of channelwide data concerning requirements, timely point of sale experience can now be substituted for total reliance on forecasts. When all members in a marketing channel synchronise their operations, opportunities exist to reduce total supply chain inventory and eliminate duplicate practices that increase cost without generating customer value.

The reality of today's best practices is that extremes in anticipatory or response based management are not shown. Many well established practices preserve conformance to anticipatory paradigms. The greatest barrier to adopting response based practices is the need for corporations to report and project earnings to financial observers (Bowersox 1996). This accountability factor leads to loading the channel with inventory to create timely sales volume, but then having to have to deal with all of the excess inventory.

3.3 Flexibility

Flexibility is the ability to adapt to unexpected circumstances (CLM 1995, Fawcett 1997). Flexibility is viewed as leading to supply chain leadership because it permits an organisation to continuously improve customer satisfaction by leveraging routine performance to high levels of non-routine compliance. Flexibility concerns a firm's capability to encounter, resolve, and exploit the unexpected emergency or opportunity. Flexibility differs from agility in being a component of agility. Flexibility is the ability to change what is already done, whereas agility is a deliberate competitive response to constantly changing requirements (Harrison 1998). (Gattorna et. al. 1998) define flexibility as being multi-capable. The supply chain is able to, within its parameters, change to a different product with ease. The adapting to change is emphasized, as opposed to being continuously fluid, never having a set form to adapt from.

4 A Next Generation Supply Chain Definition

The above traditional definitions of supply chain management are those that are currently used by academics and practitioners. The definitions are presented to show an evolution of supply chain management from a logistics integration focus to an enterprise extension focus. However, supply chain management is still pre-occupied with the efficient and effective movement of inventory. The optimization and integration is linear, and does not completely extend to the rest of the enterprise particularly those aspects that have an indirect effect on the supply chain.

As a result, supply chain management needs to be given a more encompassing definition. The definition needs to take into account

The supply chain can thus be seen as :

- ? A system that exists within a greater whole
- ? Dependent on other enterprise functions for total efficiency
- ? A set of companies with mutual dependency making up the system
- ? Existing at a point in time, to respond to a particular customer demand
- ? Existing on a higher, more abstract level than currently defined in operational terms. This allows for the implementation of processes across enterprises that are evolving at a slower rate than the virtual supply chain to which they belong.

The management of the supply chain can thus be regarded as comprising :

- ? A strategic approach, not only operational logistics.
- ? Relationship management.
- ? The management of processes, not single entities, across relationships and companies.
- ? A total management view, of the raw material to final consumer.
- ? Demand recognition and the agile re-organization of companies into a supply chain to meet that demand.
- ? A responsive system to enabling competitive advantage.

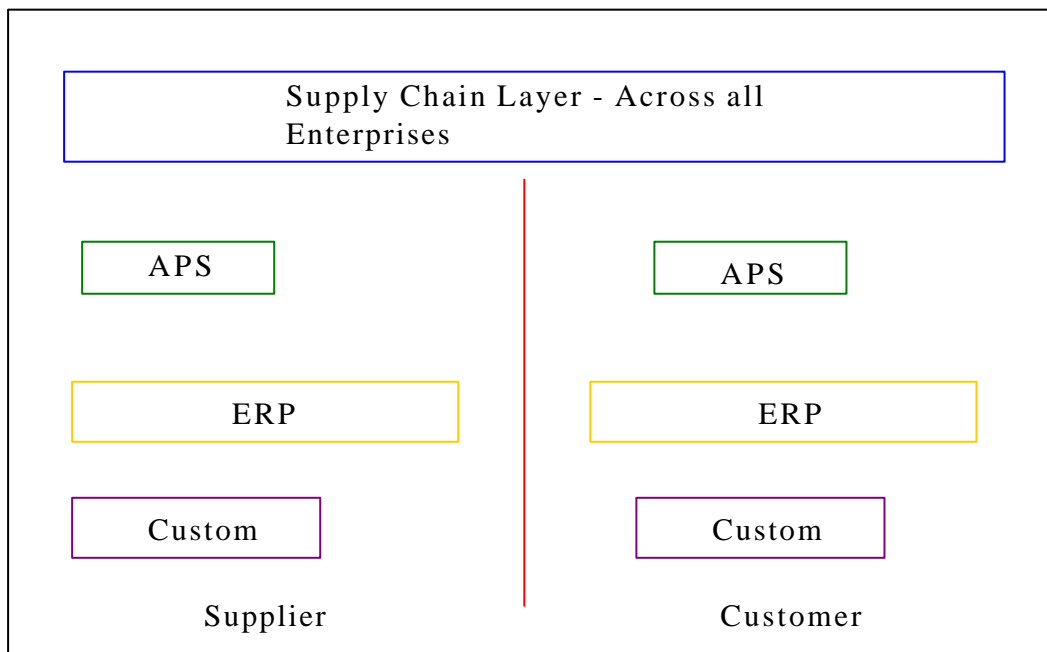


Figure 1. The Supply Chain exists on a higher, enterprise wide level.

5 Future Supply Chain Model

Above the concepts of supply chain management that are driving a different form of business, such as agility, flexibility, and core competency have been discussed. These concepts define the characteristics of a competitive supply chain, but emphasise dynamic concepts resulting in supply chain structure change.

In this section the concepts discussed so far are melded into a single future supply chain model. The above two paragraphs have provided a basic classification for a supply chain model, by highlighting the static component of the supply chain, namely structure; and the aspect of the supply chain that provides for the concept of agility, namely dynamic behavior. Consequently, the model of this study is described in two components, Static Structural Aspect, and Dynamic Behavior Aspect.

? Static Structural Aspect

- ? The scope of the future supply chain is global. Management efforts recognise the influences of all aspects of the chain, from raw materials to final consumer.
- ? Elements of the supply chain (nodes or companies) are formed by adopting the concepts of outsourcing and core competency. The result is nodes forming that are specialised and to a large extent "reusable" service providers.
- ? The nodes forming a supply chain are all proficient at supply chain management. Companies not able to participate in a way that optimises the total supply chain are excluded. By maintaining common standards, relationship establishment between companies reforming a supply chain is facilitated.
- ? The supply base is rationalized, while the remaining participants become more flexible. Fewer are partnered with at any given time, but more are available at any given time. The preferred pool is becoming smaller, while the choices are increasing.
- ? Any size company is able to participate in the supply chain. The emphasis is on adding value and being able to be an efficient supply chain partner. The use of information technology results in company size no longer being as important, as all that is required are the goods and information.
- ? Similarly to size, location is not important. Time and space is collapsed by information technology and modern transport.
- ? As supply chains form, certain companies are excluded due not to being unable to provide the required level of service, but due to not being able to provide value. Such companies are disintermediated.
- ? Market knowledge on a very advanced level is required. This knowledge is used to determine the structure of the supply chain to meet customer requirements. Information Technology is required to measure customer requirements at a greater speed and level of accuracy than before. requirements need to be measured directly from the consumer, and the supply chain reconfigured as the requirements change at the consumer.

? Dynamic Behavior Aspect

- ? All actions taken in the supply chain are holistically accessed. Although the supply chain may not remain in a fixed form for a length of time, the dynamic supply chain should not lose optimized performance due to flexibility. The entire supply chain should share common goals and objectives.

- ? The relationships between supply chain members is described by (Poirier 1998) as global interfaces. The essence of the interface concept allows for relationships to be formed, but for the relationships to be directed to another company who is better suited to provide the required services, when required.
- ? When a changed supply chain structure is required, the entire supply chain needs to be agile. Agility will allow a reorganization of the supply chain with minimal negative effect to the supply chain. Consequently, individual companies need to be internally agile to be able to adapt to changing demands, but individual companies also need to project agility outward in the form of their relationships and interfaces with partners to allow the establishment of new relationships as needed.
- ? All action taken by the supply chain needs to be rapid. Factors creating change should be immediately detected. Companies need to be optimised for change and evolution to meet changing circumstances immediately.

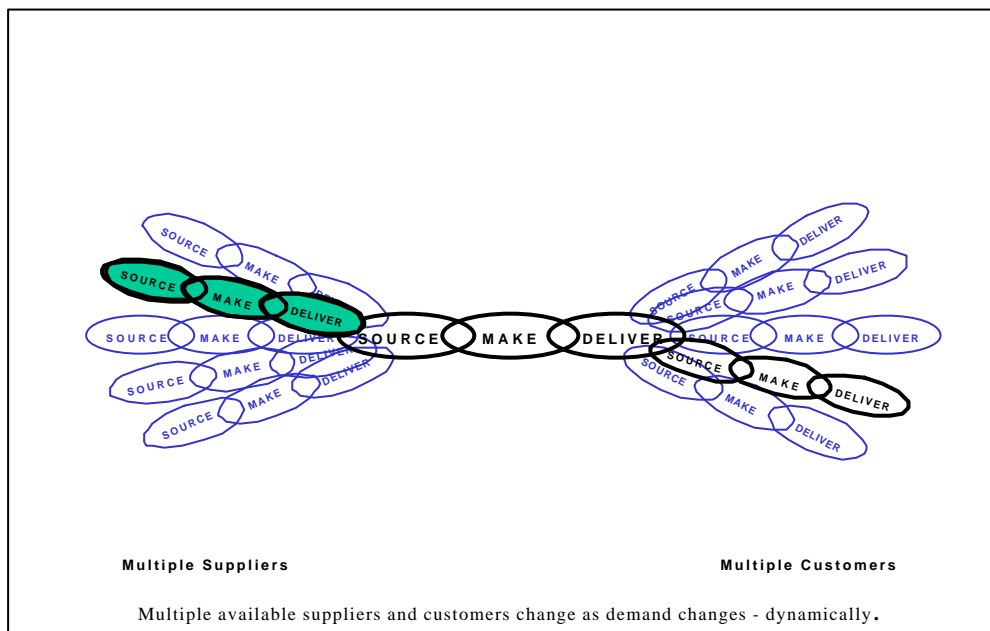


Figure 2. The agile, dynamic, virtual supply chain (Adapted from Supply Chain Council).

6 Information Technology support for the Next Generation Supply Chain

The support of information technology is critical to the above model. Such information technology needs to be :

- ? Reusable (an equivalent of core competency)
- ? Rapidly changeable (to provide agility to the supply chain)

Following on from above, it can be seen that the underlying software architecture of a supply chain solution also needs to be :

- ? Agile
- ? Flexible
- ? Deployable over a multi-enterprise scope
- ? Multi-function support when used as a suite
- ? Handle complexity
- ? Enable collaboration
- ? Enable co-ordination

In addition, a new method of thinking is required within supply chain management projects. It is necessary to be very rapid in decision making, and this approach must be conveyed to the software project. The software must therefore be rapidly implementable.

It is proposed that software components provide an appropriate architecture capable of meeting the above requirements.

7 The Component Approach

7.1 Definition of Components

Component software is an object-based software model aimed at efficient and incremental software development (Leeb 1996). The main idea is to break monolithic applications into reusable components that can be developed, distributed and upgraded independently. Component based development is defined as the process of building systems by way of combination, aggregation, and integration of pre-engineered and pre-tested software objects (Kara 1998). Component software allows complete applications to be created out of small pieces of software, or components. Each component has well defined functionality and will blend with existing pieces to form a completely integrated application. (Stevens et. al. 1997). However, a focus should be on designing new applications in a component manner. In essence, such design is to providing standard mechanisms for interoperability between applications and components. If components can inter-operate, they can be combined to build larger applications in a flexible and incremental way.

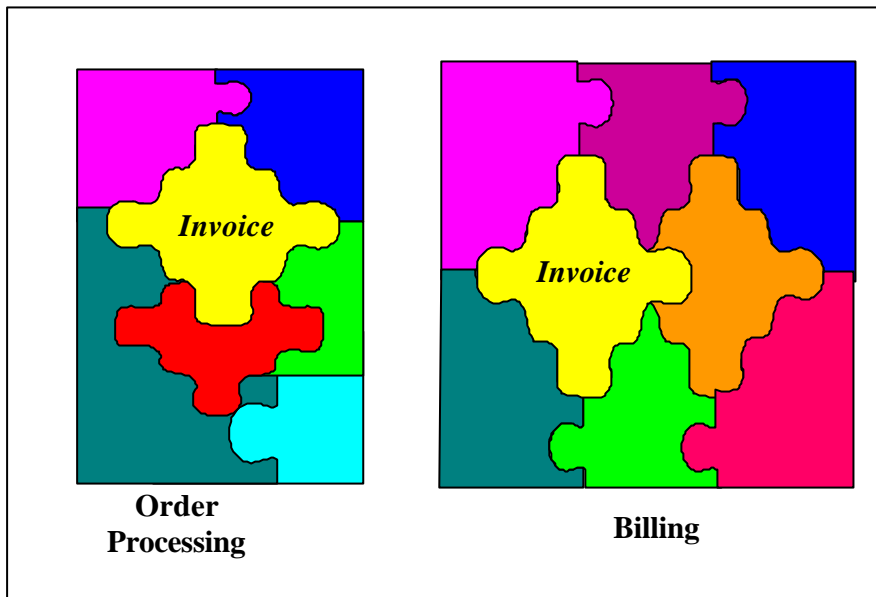


Figure 3. The component concept (Cameron 1998).

Components can be developed and loaded to respond to specific tasks, such as data collection, materials accounting, scheduling, or machine control. The key point is that each component is a self-contained unit of functionality. This aids in software design and testing since the complexity of these units is hidden from the rest of the program and each unit can be independently tested with assurance that problems in each component will be completely isolated from any others. Benefits of this include an overall reduction in risk, fewer (or at least manageable) maintenance problems in the future, and an increased development rate.

Once developed and tested, these components cooperate to form a synergistic, supply chain wide, distributed software framework. The key to effective component-based application design is the efficient design and utilization of an underlying software framework which incorporates and integrates these individual applications by providing the communication layer and database accessibility (Stevens et. al. 1997).

Conceptually, components can be seen as an important extension of traditional software. Configuration of traditional software is achieved by changing configuration files associated with the application. These are often text files stored on the computer hard disk, or can be manipulated by graphical configuration tools. While the sophistication of these files may be great, the flexibility of the software is fundamentally limited by the hard-coded implementation of the application. For example, it is impossible to add a new parameter to the configuration that was not part of the initial design. While component software might use something similar to these configuration files, these component files are much more sophisticated and may contain executable code that can be used to extend or modify the existing program. This executable code will become part of the running applications and so new parameters and capabilities may be added and the behavior of existing features can be modified and changed (Leeb 1996, Stevens et. al. 1997).

8 Reason for Component Software in Supply Chain Management

For many organizations, the delivery of quality software applications at the right time has been a critical inhibitor to business success. The difficulties of software development has led to many organizations acquiring packaged application software. But the switch to packaged applications has not been a satisfactory alternative. The packaged applications have been difficult to integrate into an established information system architecture, and have resulted in automated functional silos. Even client-server systems have not solved the problems of systems being sluggish and monolithic. Table 2 depicts the advantages to using a component approach to solving the difficulties of monolithic applications.

	Component Apps	Monolithic Suites
Rapid response to business change	Medium	Low
Support for Best practices	High	Medium
Support for easy customization	Medium	Low
Lower cost of Ownership	High	Medium
Comments	Components blend an ability to change with ease of use and multi-vendor interoperability	Big, tightly integrated systems change slowly

Table 2. Comparison between components and traditional monolithic suites (Forrester Research Inc.)

The future of supply chain management also poses problems to applications (Butler 1998) :

- ? It is becoming increasingly difficult to **own and control** an application – as supply chain integration occurs, your applications become part of the applications of your customers and suppliers.
- ? Applications must be **delivered synchronously** with the supply chain change cycle time.
- ? It is no longer competitive to select a packaged application based on current needs. The ability to adapt to new requirements is crucial.
- ? Replacement of **legacy systems** cannot be justified. New functionality must be integrated with existing packages.
- ? Applications must **align with the business**. Although it may be necessary to align to a packaged application so as to advance the business, this should be unacceptable.
- ? Improvements to an existing function must be possible **without impacting other functions**.

The influence of a component based approach on these issues is discussed in table 4.

	Monolithic Suites	Components
Ownership and control	Difficult	Simple. You own the components in you're your company only, and they integrate with other components in other companies to form a whole
Delivery	Slow, no synchronization with business needs	Only applicable components delivered as the business demands
Legacy replacement	Difficult to replace and replicate as suites not designed for it.	By nature plug-and-play replaceable
Business alignment	Not modeled on the modern business	Each component represents and aligns with a business element
Non-impact improvements	Change impact the entire system	Components separate the business logic from the interface. Therefore changes are not seen outside of the component

Table 4. The effect of components on applications management as opposed to monolithic suites.

When we consider the future of supply chain management as set out in the model of section 6 further requirements of a solution are found :

Static Components

1. Globally accessible
2. Reusable and focussed on a core competency
3. Have a low entry point to integrate to the system
4. Be location independent

Behavior of the Component System

1. Ability to easily change relationships between components
2. Ability to re-deploy components in other environments/other companies
3. Rapid change to component behind interface
4. Rapid removal and replacement of a component in a system

Componentization addresses these core business issues directly :

Static Components

1. Component standards tend to be web enabled. This allows a global reach. Different systems in different areas are no longer an issue as all that needs to be standardized is an interface.
2. Components are founded on the simple principle that if business services are designed as components, they are inherently reusable, replaceable, and upgradeable. Components support the service based approach. A virtual organization does not want to get involved in the implementation of a service. It just needs it to be performed. Components provide functionality with no insight into the details of implementation.
3. By using the interface approach, all a company needs to do is create a standard interface. What is behind the interface is irrelevant to the rest of the system.
4. Web enabled components inherit the location independent characteristics of the Internet.

Behavior of the Component System

1. Components do not, by nature, contain logic that is tied into the logic of other components. It is thus not an issue to switch a component to interact with an appropriate component that is similar to the previous partner, but possibly in another company.
2. Java is increasingly becoming accepted for server side components (e.g. Enterprise JavaBeans). This allows for platform independence. In addition, as a component is only tied to a system by its interface, there is no difficulty in moving it to another system.
3. New components can be implemented which extend the functionality of the existing application in a controlled and progressive manner.
4. Componentized applications are inherently and explicitly designed to adapt to change. Similarly, legacy and packaged applications are wrapped with an interface and reused in the same way. The componentization task does not require replacement of previous software investment.

An example of the component approach is given to further illustrate the advantages of a component based architecture. Figure 4 illustrates a typical monolithic software solution, in this case Enterprise Resource Planning.

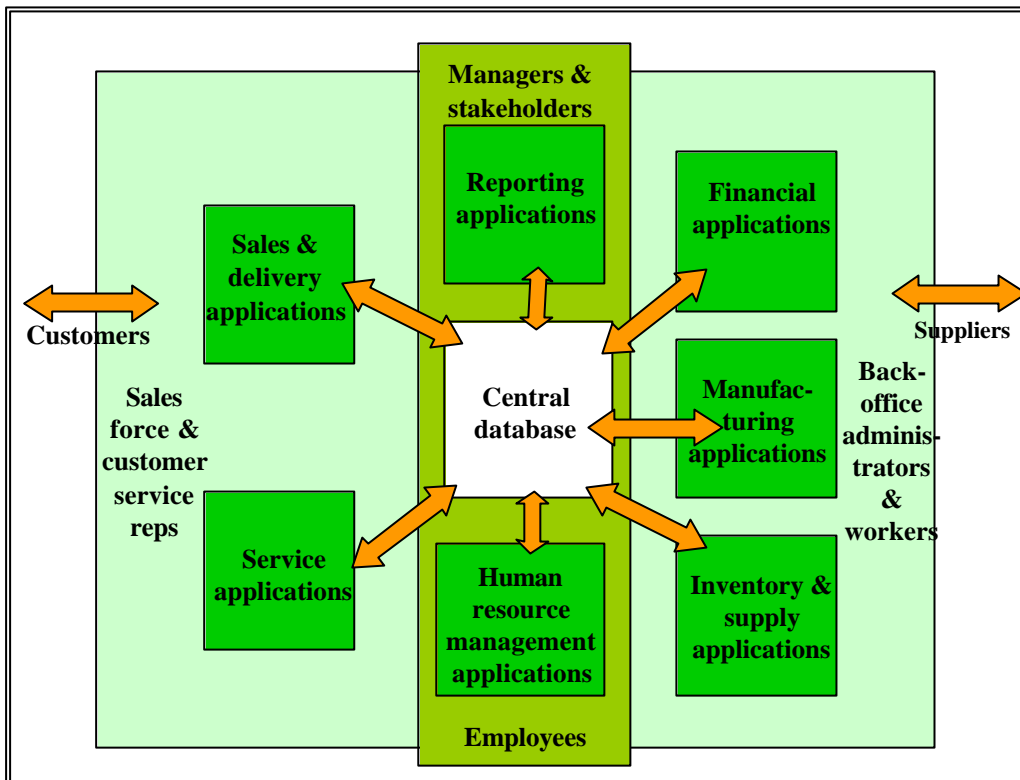


Figure 4. An Monolithic ERP Suite (Chaudhry 1998).

The emergence of component based software is creating less expensive ways to create enterprise solutions. Although software architectures such as depicted in figure 4 have served well in the past, these architectures will not allow a supply chain to survive in a competitive arena such as depicted by the future supply chain model of section 6 :

- ? Integration through a common database proves impossible in the supply chain, as no one vendor has all the required modules and no vendors or partners agree on a common data model. The lack of a common data model means high integration costs.
- ? This centralized approach is inflexible. The company is forced to use a single vendor fit to their business, as opposed to gaining advantage by specializing key areas.
- ? The central system does not allow for extending the system over geographical areas, and the trend in supply chain management is to focus globally.
- ? Many modules of ERP such as the manufacturing applications have proved to be a market area of their own, and are largely replaced by best-of-breed systems or Advanced Planning and Scheduling systems, requiring integration with a suite not designed for such integration.
- ? This monolithic architecture emphasizes direct activities at the expense of cross-functional supply chain processes, and does not provide the ability to build processes.

To a large degree, supply chains have been disadvantaged by the monolithic applications installed several years ago. By using component based architectures as depicted in figure 5 :

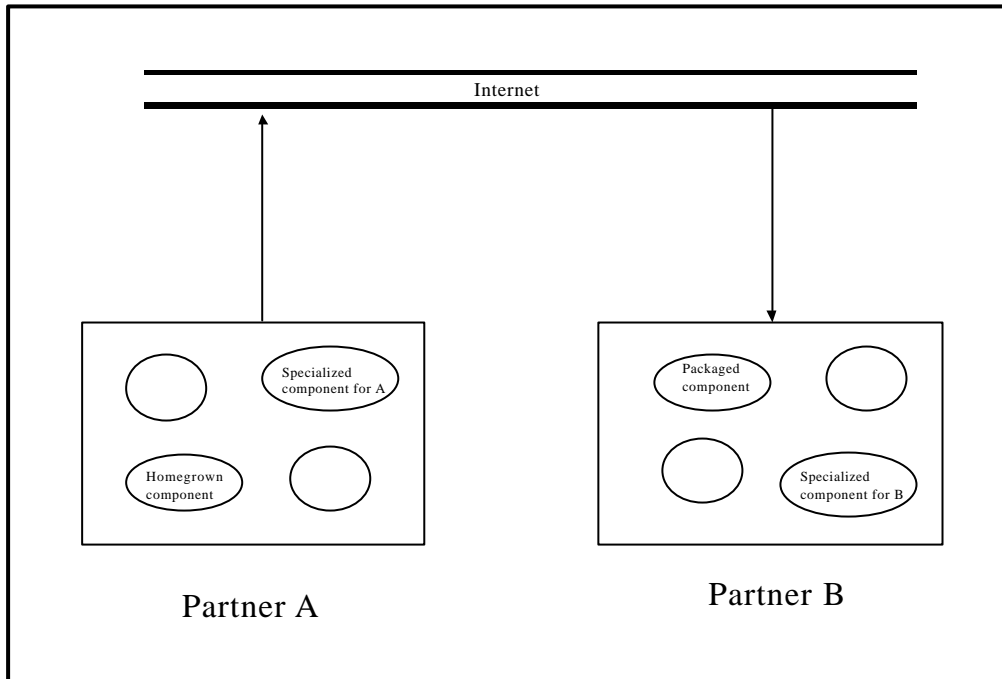


Figure 5. Component based software architecture.

- ? Suites can be selected to achieve a 90% fit to a specific industry (AMR 1998).
- ? Data models are no longer an issue - each supply chain entity deals with the best model for it, and exposes a standard enabled by the component interface for data transfer.
- ? Suites are composed of the necessary components, with special attention paid to critical components. Equal weight is not enforced on what forms a component system.
- ? Components are aligned with and modeled on business entities, allowing for a direct representation of the business model, including processes.

9 Capabilities Required By Supply Chain Component Software

(Cottman 1998) has identified six basic capabilities required by supply chain application component software :

1. Environment Independence: Components can interoperate across languages, operating systems, networks, development or production environments. (Schmidt 1998) furthers the notion of environmental independence to include not only platform independence, but also business domain independence.
2. Location Transparency: You can transparently access a component from within a single process, multiple process running within the same machine, or multiple processes running across networks and operating systems. From an architecture or implementation view, you are one level above transport, server location, process activation, datatype representation on different platforms.
3. Interface separate from Implementation: The interface is a contract that specifies a component's functionality. The interface exposes no implementation or platform or environment dependencies.

4. Self-describing Interface: A component provides the ability to describe the interface specification. At a minimum the specification must include method and property specifications. This is an essential feature for component software to be able to bind together at run-time and an essential capability for reusability.
5. Platform Compiled Independence: The ability to install and use a component out of the box on any platform.
6. Universal Application Component Framework: A component must be able to integrate with one or more components to form a new well-behaved and well-defined application.

However, component software needs a ubiquitous set of rules of engagement across different component interaction boundaries, as provided by a distributed object management framework.

(Hurwitz 1998) discusses further requirements :

- ? Coordination Logic. This rationalizes and controls the activities of all the components that an application requires to perform its work. A coordination logic service needs to be included in the component specification which reflects the coordination logic of the higher level supply chain model. Using any of a number of approaches — 3GL, 4GL, business rules, or, more likely, a workflow facility — the coordination logic service will manage long sequences of events. Hurwitz believes workflow facilities offer the strongest potential for coordination because they can best handle the very long (and often convoluted) sequences of events that characterize business processes. However, the key to the success of this approach is large-grained, abstract components with interfaces defined in terms of business functions (e.g., methods like "Create Order" or "Add Customer"), not IT functions.
- ? Component Wrapping. Legacy systems are a central feature of any supply chain component project. The component specification needs to be designed to ease the integration of the components in the system with existing systems. This capability allows developers to incorporate legacy code, existing client/server applications, and even packaged applications as components. Wrapping turns existing code into a black box. Its functionality is invoked only through the defined interface. Developers expose the application's functionality through the component interface. A variety of techniques are available to wrap legacy applications and expose an interface, including using an existing API, using "screen scraper" technology, or interfacing to legacy middleware.

(Schmidt 1998) adds the following requirements :

- ? Support changes of the underlying process model. Business processes are subject to frequent changes triggered by changing business needs, e.g., reorganizations, process optimization efforts, outsourcing. The realization of a business process by a set of components must be done in a way that makes it easy to reflect those changes.
- ? Enable composition of reusable business components. A business process defines the flow of work and relies on existing application components to perform the actual work. Supply chain components must interact with other application components and support changes of the overall process without affecting these components.

- ? Allow for monitoring of process execution. Business process models are often developed as the result of a business process re-engineering effort with the objective to optimize business performance; as a result, an important requirement on the implementation of business process models is the capability to monitor performance of process execution in terms of the underlying process model.
- ? Enable distribution of a process across business domains. Business processes can span multiple business domains or enterprise boundaries. For example, in a supply chain scenario, an order process initiated by company A might require parts of the order to be processed by another company B; in addition, the distribution of the process might change, e.g., due to an outsourcing decision. A realization of a business process must allow for the flexible distribution of process components and support the interaction of these components.
- ? Support assignment of process steps to supply chain participants. An important aspect of business process automation is the capability to manage assignment of work to the resources required to perform the work. Process steps can, for example, be assigned to people (or other units in an organization model) that contribute to the execution of that process step; other examples of resources include machines or documents required to perform a business task. The realization of a business process must support association of entities in a resource model to process steps.

(Sebes et. al. 1997) describe further functionality that is applicable to the collaborative nature of the supply chain component environment. The set of components making up the supply chain community instance at that time need to be a secure enclave. The components should preferably provide security built into the specification or architecture for the creation of secure subsets of components with security levels differing from the outside and inside of the organization.

The above requirements are of a technical nature, dealing with the component itself. (Wills 1998) and (Schmidt 1998) reinforce the need for components to be based on a model of the business. The computing function of each department should be componentized, based on business boundaries, not technical computing requirements.

Basing supply chain components on a model of the supply chain leads to a further requirement. It is necessary that the fundamentals of supply chain management be reflected within the software components representing aspects of the supply chain.

- ? The supply chain software components need to represent supply chain processes. Current supply chain software solutions provide isolated tools for planning and execution within functional areas. The output of a tool does not form the input to another tool. If (Davenport et. al. 1990)'s definition of a business process, being a set of logically related tasks performed to achieve a defined business outcome, it can be seen that current supply chain solutions do not provide support for the fundamentals of supply chain management.
- ? To effectively implements processes, relationships between supply chain entities need to be represented. Relationships are caused by :
 - ? Business rules.
 - ? Architectural constraints in complex systems (Lea 1999).
 - ? ERP/APS integration consisting of easy data and difficult process integration. The problem occurs when the APS system replaces functionality within the ERP system, such as MRP (Bermudez 1998).

- ? **Granularity.** The granularity of the components is determined by the level of encapsulation the component displays (Ambler et. al. 1998). Today's applications are not packaged well as usable components. There is no separation of the functionality that an enterprise wishes to use to differentiate itself with functionality standard across all enterprises. In addition, in an attempt to provide solutions for different functional areas as standalone applications, vendors have bundled together data, process control, and calculations that are highly redundant across different applications. Data and process control exchanges are made difficult.

The above problems can be remedied by effective component creation (requiring appropriate encapsulation of process control, data and calculations) allowing the purchasing of components of the granularity that allows supply chain functions and processes to be easily changed. There are two criteria defined by (Ambler et. al. 1998) for this :

- ? **Stability :** Stability deals with the possibility of change of a component. Change is managed by the inclusion or exclusion of certain functionality, allowing us to consider stability a granularity of function. It is desirable to separate stable elements, which are unlikely to change, from elements which are more dynamic. Ambler et. al. differentiate the concepts of Interface, Control, and Entity object categories. Interface components will change as business functionality changes. Client side components such as JavaBeans are already available to deal with these changes. Control components are those that implement supply chain processes, and would be implemented with Session beans. Control components will be dynamic within the supply chain community environment. Entity components contain the basic data and functions/calculations that provide information to the business. Entity components are less likely to change. Entity components are the most likely candidates for purchase. Another view of granularity of function is provided by Platinum Technology (Platinum 1998), who differentiate between horizontal and vertical components. Horizontal components can be used across the enterprise. Vertical components can only be used within specific domains. It is important to encapsulate functionality that is unique in the vertical domain into vertical components as this will solve a major problem in supply chain management, namely that systems are too general for a specific company.
- ? **Scope :** The second criteria for encapsulation into a component is referred to as scope by Ambler et. al., but is what we consider granularity of size. It is desirable to have large components that solve significant business problems without intervention. However, whether the functionality and data encapsulated in one component is determined by the functionality and data encapsulated in the other components that make up the system. When components in the same environment contain the same (i.e. redundant) data and functionality the design suffers the consequences due to broken encapsulation. Smaller components allow more flexibility to change features through replacement and are less likely to have overlapping functionality. But, when components get too small, integration and assembly get tedious.

The impression that this article conveys is that supply chain processes are represented by components. Component technology of this massive grain does not yet exist, but the Java Enterprise JavaBeans are a beginning. (Raber 1998) reports that medium granularity is known to be most appropriate for beans, particularly session beans. With the approach of session beans representing processes, and entity beans representing supply chain entities, a difference in optimal granularity is expected between entity and session beans. The granularity of the supply chain processes and entities set an upper limit on bean size, as the entity or process represents a real life object, and is made up of several traditional objects or components. Component granularity can

then be anything smaller, down to objects. To allow end users to use builder tools granularity of components should be that of the entities or processes. The entity or processes sized component can in turn consist of components of a size more realistic to application developers, but these details are hidden from the user.

The Enterprise JavaBeans (EJB) specification aims at an EJB relating to a relational database table. Components representing a RDBMS table are a limiting concept for object oriented design. A more distributed object approach would be to map data from possibly several tables to the data required by a business unit supply chain process, add the methods, and take the granularity. Supply chain management is dominated by central relational databases, being the lowest layer of ERP software. The existing EJB specification aims at satisfying the use of EJB and the existing databases. An evolution to object oriented databases and the evolution to process oriented systems will change the mismatch in granularity. (Singh 1998b) suggests a granularity that facilitates generic reuse. A granularity should be determined that represents business entities that are common throughout the enterprise, such as a Purchase Order.

The above should not be taken as a rule for supply chain component granularity. Granularity should still take optimization into account, and weight optimization against ease of development when using large components. Optimal granularity can be found by having invocations do a meaningful amount of work, as distributed calls are expensive. However, very large methods should also be avoided. The typical TP invocation should return quickly, not be delayed by a large method. Figure 3 is a comparative scale of component granularity in relation to reuse and productivity.

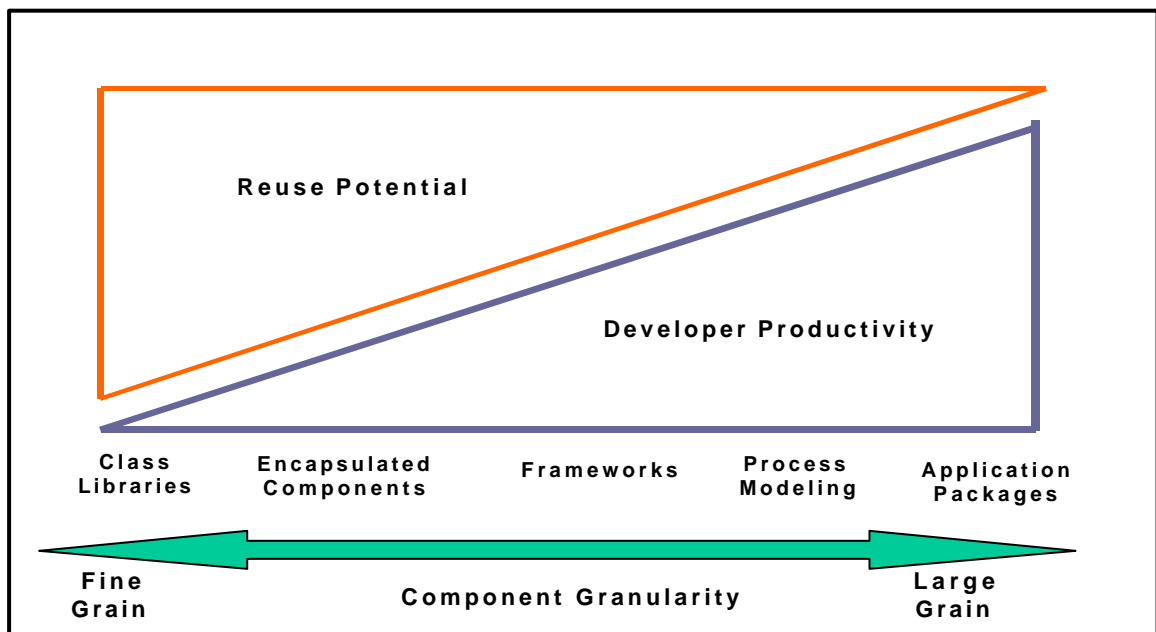


Figure 4. Component Granularity Comparison (Adapted from Software Productivity Group).

It can be argued that although process modeling is halfway into the productivity side, and does not, according to the diagram, facilitate reuse, the diagram is not applicable for the supply chain. The diagram is based on the view of software engineering, not business. This study has attempted to recommend that the supply chain manager become involved in the systems development process. For this reason business objects were introduced. Figure 4 does not take business objects into account, and values reuse over understanding between developer and user. In addition, by rating the componentization of application packages as higher than processes, Figure 4 does not take into account that current application packages are based on functions, and are inflexible. By emphasizing process components, greater productivity is achieved when the supply chain changes. In addition, very small grained components are not resilient to change.

10 Conclusion

We have shown that it is possible to align the requirements of supply chain management of the future and the underlying information systems. In conclusion, we should however mention further work that is required before information systems begin to benefit by the component concept :

- ? Standards among vendors to achieve component inter-operability
- ? Further understanding of large-grained software components
- ? Architectural structures in place in company's to hold components

On the business side, it should be emphasized that component development is not a technological issue - it is a concept implemented by technology currently, but great benefit can be derived by managing processes by enabling for reuse and leveraging of investment.

References

- AMBLER B., VENKAT S. Large Grained Components and Standards, Perfect Together. Lucent Technologies. 15 June 1998.
- AMR 1998. ARM Research Report on Manufacturing. Do We Need a New Model For Plant Systems? Oct. 1998.
- BERMUDEZ 1998.
- BOWERSOX D.J, CLOSS D.J. Logistical Management - The Integrated Supply Chain Process. McGraw Hill. 1996.
- BUTLER. Component-Based Development – Application Delivery and Integration Using Componentised Software. Butler Group. Management Report. 1998
- CHRISTOPHER M. Logistics and Supply Chain Management - Strategies for Reducing Costs and Improving Services. Financial Times Pitman Publishing. 1993.
- CHRISTOPHER M., PECK H. Managing Logistics in Fashion Markets. The International Journal of Logistics Management. vol 8, no 2. pp. 63-74. 1997.
- CLM. World Class Logistics - The Challenge of Managing Continuous Change. The Global Logistics Research Team at Michigan State University. Council of Logistics Management. 1995.

- COTTRILL K. The Supply Chain of the Future. *Distribution*. vol. 96., no. 11. pp. 52-54. Oct. 1997.
- DAVENPORT, T., H., SHORT, J.E. (1990). The New Industrial Engineering: Information Technology and Business Process Redesign. *Sloan Management Review*, Summer, 11-27.
- FAWCETT P., McLEISH R., ODGEN I. *Logistics Management*. Pitman. 1997.
- FRANZ P., CILLIERS W.W., ANDREWS A. *Logistics Excellence in South Africa*. Chair in Logistics, University of Pretoria & Anderson Consulting. 1994.
- GATTORNA J., KERR A. The Impact of Technology on Delivered Costs. In GATTORNA J.L. ed. *Handbook of Logistics and Distribution Management*. 4th. ed. Gower. 1996.
- GATTORNA J.L., WALTERS D.W. *Managing the Supply Chain : A Strategic Perspective*. MacMillan Business Press. 1996.
- GOLDMAN S. Agility Tutorial. Agility Forum.
<http://www.agilityforum.org/Tutorial/whatisagility.html>. Date unknown, aquired 1998.
- GRIES N.P., KASARDA J.D. Enterprise Logistics in the Information Era. *California Management Review*. vol 39. no. 3. Spring. pp. 55-78.
- HARRISON A. An Investigation of the Impact of Schedule Stability on Supplier Responsiveness. *The International Journal of Logistics Management*. vol 7, no 1. pp. 83-91. 1997.
- HARRISON A. Agility : Myth or Substance? Cranfield School of Management. Date unknown, aquired 1998.
- HURWITZ. Hurwitz Group. Component Road Map - A Hurwitz Group White paper. <http://hurwitz.harvard.net/component.html>. 1998.
- KARA D. Build vs. Buy : Maximizing the Potential of Components. *Component Strategies Online*. <http://www.sigs.com/cso/frompages/9807/buildvsbuy.kara.html>. July 1998.
- LALONDE B.J., POHLEN T.L. Issues in Supply Chain Costing. *The International Journal of Logistics Management*. vol. 7, no. 1. pp. 1-12. 1996. Sept. 1998.
- LEEB A. A Flexible Object Architecture For Component Software. Massachusetts Institute Of Technology. Master of Science in Computer Science Thesis. May 10. 1996.
- METZ P. Demystifying Supply Chain Management. *Supply Chain Management Review*. Winter 1998. pp. 46-55. 1998.
- PORIER CC, REITER SE. *Supply Chain Optimization : Building the Strongest Total Business Network*. Berrett-Koehler. 1996.
- RABER C. EJB Design Wisdom. Email from EJB-INTEREST mailing list. 14 Sept. 1998a.
- RABER C. EJB Design Wisdom. Email from EJB-INTEREST mailing list. 14 Sept. 1998b.

- SCC. Supply Chain Operations Reference Model Version 2.0. Supply Chain Council. 1998.
- SCHMIDT M. Building Workflow Business Objects. OOPSLA'98 Business Object workshop. <http://jeffsutherland.org/oopsla98/mts.html>. 1998
- SEBES E.J., VICKERS BENZEL T.C. Collaborative Computing, Virtual Enterprises, and Component Software. Trusted Information Systems, Inc. Nov. 20. 1997.
- SINGH C. Email from the EJB-Intersert Listserv. Entities and Relationships. 20 Aug 1998a.
- SINGH C. Email from the EJB-Intersert Listserv. EJB Design Wisdom. 14 Sept. 1998b
- STANK T.P., DAUGHERTY P.J., ELLINGER A.E. Information Exchange, Responsiveness and Logistics Provider Performance. The International Journal of Logistics Management. vol 7., no 2. pp. 43-57. 1996.
- STEVENS R., REDMAN J., COOPER E. Plug And Play Component Software For Manufacturing And Control. ErgoTech Component Software White Paper. <http://www.ergotech.com/component.html>. 1997.
- WILLS A. Objects and the Net. Trireme Internet Development. <http://www.trireme.com/trireme/papers/onet.htm>. Acquired 1998.