UNIVERSIDADE METODISTA DE PIRACICABA

FACULDADE DE ENGENHARIA ARQUITETURA E URBANISMO

PROGRAMA DE PÓS GRADUAÇÃO EM ENGENHARIA DE PRODUÇÃO

PROPOSTA DE SISTEMA DE INFORMAÇÃO PARA GESTÃO DE DEMANDA E DE INVENTÁRIOS NAS CADEIAS DE SUPRIMENTOS

PEDRO DOMINGOS ANTONIOLLI

ORIENTADOR: PROF. DR. JOSÉ A. A. SALLES

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia de Produção, da Faculdade de Engenharia Arquitetura e Urbanismo, da Universidade Metodista de Piracicaba – UNIMEP, como requisito para obtenção do Título de Mestre em Engenharia de Produção.

SANTA BÁRBARA D'OESTE

2005

A Deus,

Por Seu infinito Amor e Misericórdia

AGRADECIMENTOS

À minha família, pelo incentivo, apoio, amor e paciência, especialmente durante o desenvolvimento deste trabalho.

Ao professor Jose Antonio Arantes Salles, pelo incentivo, orientação, compreensão e direcionamento em todos os momentos do projeto, bem como pelo conhecimento e apoio despendidos.

Aos demais professores da Unimep, que contribuíram decisivamente, com conhecimento valioso para que este trabalho pudesse ser realizado.

A todos aqueles que direta ou indiretamente, contribuíram para que este projeto fosse concluído.

“O temor do Senhor é o princípio da sabedoria; revelam prudência todos os que a praticam.”

Bíblia, Salmos 111:10a

ANTONIOLLI, Pedro Domingos. **Proposta de Sistema de Informações para Gestão de Demanda e de Inventários nas Cadeias de Suprimentos**. 2005. Dissertação (Mestrado em Engenharia de Produção) – Faculdade de Engenharia, Arquitetura e Urbanismo, Universidade Metodista de Piracicaba, Santa Bárbara D'Oeste (SP).

RESUMO

A globalização exige das empresas uma maior flexibilidade e adaptabilidade de seus processos internos às exigências de mercado. Esta flexibilidade resulta em novas formas de relacionamento entre parceiros, as cadeias de suprimentos. Para que estas cadeias possam se diferenciar de seus concorrentes, a agilidade se torna fator competitivo fundamental. Neste sentido, as empresas participantes destas configurações produtivas devem constantemente rever seus processos, e redesenhá-los. Porém, para que isto seja possível, há a necessidade de se utilizar sistemas de informação que espelhem tais processos, e que sejam facilmente adaptados sempre que houver modificações nos processos que eles representam.

Este trabalho tem como objetivo apresentar um histórico destes sistemas, salientando as principais necessidades de negócio dentro das cadeias de suprimentos. Em seguida, com base nestes requisitos, identificar alternativas tecnológicas dentro do paradigma de orientação a objetos distribuídos. E, por fim, propor um sistema de informações para as cadeias de suprimentos, com base nesta tecnologia. Como conclusão são apresentadas as recomendações para futuros desenvolvimentos em aplicações de gestão empresarial em ambientes distribuídos.

PALAVRAS-CHAVE: Cadeias de Suprimentos, Gestão de Demanda, Sistema de Informação.

ANTONIOLLI, Pedro Domingos. **Proposta de Sistema de Informação para Gestão de Demanda e de Inventários nas Cadeias de Suprimentos**. 2005. Dissertação (Mestrado em Engenharia de Produção) – Faculdade de Engenharia, Arquitetura e Urbanismo, Universidade Metodista de Piracicaba, Santa Bárbara D'Oeste (SP).

INFORMATION SYSTEM PROPOSAL TO SUPPORT SUPPLY CHAIN FULFILMENT AND INVENTORY MANAGEMENT

ABSTRACT

The globalisation enforces organizations to be more flexible and align their internal processes to the market needs. This flexibility brings new types of partnership, like Supply Chains. In order to differentiate itself from their competitors, agility becomes a fundamental competitive issue. Thus, this productive organization participant enterprises should continually review and redesign their business processes. So, to do it possible, it is necessary to use information systems that represent those processes and be easily adapted whenever modifications on processes occur.

This work is aimed to present an evolution of enterprise systems, pinpointing the main business needs regarding Supply Chains. After this, based on these requirements, identify technology alternatives using distributed object oriented paradigm. At the end, an Supply Chain Information System application proposal is presented. At conclusion, recommendations of future development to support Supply Chain processes, based on object orientation paradigm are done.

KEYWORDS: *Supply Chains, Fulfilment Management, Information System.*

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

API – *Application Program Interface*
ANSI – *American National Standards Institute*
APICS – *American Production and Inventory Control Society*
ATP – *Available to Promisse*
BOCA – *Business Object Common Architecture*
BOM – *Bill of Materials*
BPR – *Business Process Redesign*
BSC – *Based Stock Control*
CCM – *Corba Component Model*
CIDL – *Component Interface Definition Language*
CMT – *Container Managed Transaction*
COM – *Component Object Model*
CORBA – *Common Object Request Broker Architecture*
DBMS – *Data Base Management System*
DCE – *Distributed Computing Environment*
DCOM – *Distributed Component Object Model*
DLL – *Dynamic Library Link*
DRP – *Distributed Resource Planning*
ECR – *Efficient Customer Response*
EDI – *Electronic Data Interchange*
EJB – *Enterprise JavaBeans*
ER – Entidade-Relacionamento
ERP – *Enterprise Resource Planning*
ESI – *Early Supplier Involvement*
ETO – *Engineered to Order*
GSCF – *Global Supply Chain Fórum*
IDL – *Interface Definition Language*
IPR – *In Plant Representatives*
JIT - *Just in time*

QFD – *Quality Function Deployment*
LRP – *Line Requisitions Planning*
MES – *Manufacturing Executing Systems*
MRP – *Materials Requirement Planning*
MRP-II – *Manufacturing Resource Planning*
MTO – *Make to Order*
MTS – *Make to Stock*
OEM - *Original Equipament Manufacturer*
OLE – *Object Linking and Embeding*
OMA – *Object Management Architecture*
OMG – *Object Management Group*
ORB – *Object Request Broker*
OSF – *Open Software Foundation*
PC – *Personal Computer*
PCP – Planejamento e Controle da Produção
PMBOK – *Project Management Body of Knowledge*
PMI – *Project Management Institute*
POA – *Persistent Object Architecture*
PSDL – *Persistent State Definition Language*
RCCP – *Rough Cut Capacity Planning*
RFI – *Request for Information*
RFP – *Request for Proposals*
RPC – *Remote Procedure Call*
SC – *Supply Chain*
SCC - *Supply Chain Council*
SCM – *Supply Chain Management*
SCRP – *Supply Chain Resource Planning*
SI – Sistema de Informação
SPC – *Statistical Process Control*
TI – Tecnologia da Informação
TOC – *Theory of Constraints*
TTM – *Time to Market*
VMI – *Vendor Managed Inventory*
XML – *Extended Markup Language*

WIP – *Work in Process*

LISTA DE FIGURAS

LISTA DE QUADROS

1. INTRODUÇÃO:

1.1. CONTEXTUALIZAÇÃO:

As transformações sociais, econômicas e políticas ocorridas na segunda metade do século XX colocaram o mundo em processo de globalização crescente de seus mercados. Segundo Hill (1998) a globalização pode ser entendida como um conjunto de mudanças convergentes no sentido de um mundo mais integrado e interdependente, onde comércio, finanças, mercados, e produção não apresentam um escopo unicamente local. Ela tem sido impulsionada por vários fatores, como a crescente desregulamentação dos mercados, a queda de barreiras comerciais, o desenvolvimento de novas modalidades de transportes e a mudança do perfil do consumidor, que exige maior valor agregado. No setor automobilístico, por exemplo, esta globalização é impulsionada pela saturação dos mercados europeu, norte-americano e asiático, e pelo crescimento potencial dos mercados em países emergentes (HUMPHREY *et al.,* 2000; LUNG, 2000; SCAVARDA *et al.,* 2001). A globalização, em manufatura, é caracterizada pela integração do setor produtivo da fábrica, com logística e operações, tendo como base de apoio a tecnologia da informação.

Este cenário obriga as empresas a competirem de forma intensa para sobreviver em ambientes caracterizados por alta instabilidade. Com isso as organizações buscam alianças estratégicas com o objetivo de obter desempenho superior por meio da construção e gestão de estruturas virtuais, chamadas Cadeias de Suprimentos (*SC - Supply Chains*).

Não existe um consenso com relação à definição do conceito de Cadeia de Suprimentos (SC). Para Beamon (1999) trata-se de um processo integrado em que matérias primas são transformadas em produtos finais e entregues aos clientes com a participação de mais de uma organização neste processo. Para Pires (2004) é um conjunto de empresas autônomas ou semi-autônomas, que

são responsáveis pela obtenção, produção e entrega de um determinado produto ou serviço ao cliente final, razão principal do termo Cadeia de Suprimentos se aplicar a todas as atividades para que o cliente final receba o produto ou serviço. Por sua vez, Lambert *et al.*(1998) considera SC como sendo uma rede de empresas com múltiplos negócios e relacionamentos. Nesta estrutura, cada elo da cadeia provê facilidades, de forma que o produto obtenha valor ao longo da cadeia. Por se tratar de uma estrutura virtual, ela implica em socialização dos custos e alta integração horizontal dos processos de negócio e da gestão, exigindo o desenvolvimento de competências chave (VOLLMAN & CORDON, 1996). Para Kaplan & Norton (1997), a efetividade das empresas atuais está baseada em um conjunto de premissas operacionais para competir e operar com processos integrados, associando a especialização funcional com a agilidade, eficiência e qualidade da integração destes processos. Dessa forma, a efetividade das empresas está atrelada ao grau de integração e aderência dos seus processos de negócio dentro da SC nas quais estão inseridas. Salles (2003) argumenta que medir conjuntamente a efetividade destes processos é o primeiro passo para se ajustar estruturas e sistemas dentro das SC na busca de desempenho superior, bem como para se realizar a gestão efetiva destas SC.

A gestão das cadeias de Suprimentos (*Supply Chain Management* – SCM) é definida pelo GSCF (*Global Supply Chain Fórum*) como a integração dos principais processos de negócio, desde o cliente final até o fornecedor primário, para prover produtos, serviços e informação que agreguem valor para os clientes e outros *stakeholders* (LAMBERT *et al.,* 1998). Um dos fatores críticos no SCM é o desenvolvimento de competências que possibilitem a integração das atividades e processos das SC (SCAVARDA & HAMACHER, 2004 *apud* LUMMUS *et al.,* 1998).

Segundo Diaz & Pires (2004) a gestão da demanda é um fator importante para o SCM pois se for bem realizada permitirá a minimização de erros de previsão, reduzindo incertezas na gestão da capacidade produtiva e diminuindo, conseqüentemente, estoques. Por outro lado, caso não seja gerenciada de

forma efetiva tenderá a gerar amplificação desta demanda ao longo da cadeia (LEE *et al.,* 1997), fenômeno conhecido como “efeito chicote” (*bullwhip effect*). O efeito chicote ocorre quando as ordens do fornecedor apresentam maior variação que as do varejista, e esta distorção é propagada a montante ao longo da SC de uma maneira amplificada (DIAZ & PIRES, 2004 *apud* METTERS, 1997; Van LANDEGHEM & VANMAELE, 2002). Lee *et al.*(1997) estima que a amplificação da demanda pode resultar em aumento de custos em uma faixa de 12,5% a 25% ao longo da SC.

Dentre as diversas causas da amplificação da demanda, Diaz & Pires (2004), com base em Lee *et al.*(1997) e Metters (1997), destacam: atualização da previsão de demanda, periodicidade na reposição das ordens de compras, flutuação dos preços e excesso de demanda.

Segundo os autores, estas causas podem ser minimizadas se forem adotadas práticas que facilitem a troca e compartilhamento de informações, que permitam a adoção de políticas coordenadas de preços, o planejamento integrado de inventários e o aumento da eficiência operacional (LEE *et al.,* 1997; METTERS, 1997). A minimização da variabilidade de demanda, segundo Ballou (1993), gera otimização da gestão de inventários.

Entre as práticas citadas acima, com exceção da troca e compartilhamento de informações, as demais estão fora do escopo da dissertação pois implicam em outras abordagens dentro do SCM, relacionadas a alinhamento de estratégias, de estruturas, de processos e de operações, bem como em sinergia entre as diferentes culturas das empresas participantes das SC.

Neste sentido, o uso de sistemas de informações (SI) facilita e agiliza o compartilhamento de dados nas SC (ex.: ordens de compras), diminuindo o custo das transações e mantendo as informações atualizadas e disponíveis a todos. Como resultado da utilização de informações precisas para o processo decisório, há diminuição das incertezas com relação à variabilidade na demanda causada, entre outros fatores, pela colocação aleatória de pedidos

(MACKAY & ROSIER, 1996; WALTON & MARUCHECK, 1997; MURPHY & DALEY, 1999).

Conclui-se, portanto, que um SI pode auxiliar na SCM, mais especificamente no gerenciamento de demanda e de inventários ao longo destas SC.

1.2. OBJETIVOS

O objetivo principal desta dissertação é o de propor um sistema de informação para apoiar a gestão de demanda e de inventários ao longo das SC, que seja baseado no modelo de componentes.

O objetivo principal se desdobra nos seguintes objetivos específicos:

- Realizar um levantamento bibliográfico sobre as SC e suas operações, bem como as informações necessárias no tocante à gestão de inventários e previsibilidade de demanda.
- Realizar um levantamento bibliográfico sobre o modelo de componentes CORBA (*Common Object Request Broker Architecture*), da OMG (*Object Management Group*) e sua aplicabilidade como infra-estrutura para a especificação de um SI que apóie as operações de gestão de inventários e de demanda, nas SC.
- Elaborar a especificação funcional de um SI, baseado no modelo CORBA, que apóie a gestão de demanda e de inventários nas SC, e que possua funcionalidade de conexão aos sistemas legados de cada empresa participante de uma SC.

1.3. JUSTIFICATIVA DO TEMA

Embora as SC sejam configurações produtivas recentes, elas apresentam tendências de consolidação como prática de negócios, seja por suas vantagens com relação à flexibilidade e agilidade que apresentam, seja como uma alternativa economicamente viável em mercados altamente competitivos (por exemplo, pela transformação de custos fixos em variáveis por meio do compartilhamento das atividades entre os parceiros da cadeia).

Para haver uma gestão adequada destas cadeias, há a necessidade de SIs que apóiem os principais processos de negócio destas configurações produtivas. Os SI atuais são baseados na metodologia ERP (*Enterprise Resource Planning*), ou seja, consideram apenas os processos de negócio dentro do escopo da empresa, e não nos processos de toda a SC. Dessa forma, justifica-se a relevância do tema face às contribuições que esta dissertação possa trazer para a gestão de demanda e de inventários ao longo das SC, apoiados pela utilização de um SI integrado.

1.4. METODOLOGIA

Segundo Marconi & Lacatos (2002), um problema deve ser formulado com clareza e objetividade. Dessa forma, o problema desta pesquisa é:

Como deve ser um sistema de informações que apóie as operações das SC de forma que possibilite a diminuição da incerteza de demanda e da variabilidade de inventários?

Dentro do contexto desta pesquisa, os pressupostos básicos deste trabalho são:

- O desempenho de uma empresa participante da SC não se traduz em desempenho de toda a cadeia, este pode ser um elo eficaz de uma cadeia ineficaz (PIRES, 2004).
- O sucesso das operações das SC se baseia, em grande parte, na forma como é feita a gestão conjunta de demanda e de inventários dos parceiros de negócio no SCM.
- Falta informação adequada para se realizar a gestão de demanda e de inventários porque não existem SI aderentes às operações das SC.

De acordo com Silva e Menezes (2000), uma pesquisa pode ser classificada de quatro formas: quanto a natureza, quanto a forma de abordagem, quanto aos objetivos e quanto aos procedimentos técnicos.

Quanto à natureza da pesquisa, ela pode ser **básica** ou **aplicada** (SILVA E MENEZES, 2000). Esta dissertação é uma **pesquisa aplicada** porque objetiva discutir teoricamente as SC e os modelos de componentes e, a partir deste ponto, elaborar a especificação funcional de um SI que apóie as operações das SC.

Quanto à forma de abordagem do problema, uma pesquisa pode ser classificada em **quantitativa** ou **qualitativa** (SILVA E MENEZES, 2000) Esta dissertação não se enquadra dentro do escopo de uma pesquisa quantitativa por não serem utilizados recursos e técnicas estatísticas. Por outro lado, de acordo com Silva e Menezes (2000) possui as características de **pesquisa qualitativa** em função do método de interpretação indutiva dos dados e o foco principal como sendo o processo e seu significado, além da atuação do pesquisador como elemento-chave.

Quanto aos objetivos uma pesquisa pode ser classificada em **exploratória**, **descritiva** ou **explicativa** (SILVA E MENEZES, 2000). Esta pesquisa tem características de uma pesquisa predominantemente **exploratória** porque envolve: pesquisa bibliográfica e formulação de um modelo a ser aplicado nas SC reais.

Quanto aos procedimentos técnicos, a pesquisa pode ser **bibliográfica, documental, experimental, levantamento, estudo de caso, pesquisa *ex-post-facto*, pesquisa ação ou pesquisa participante** (SILVA E MENEZES, 2000). Esta dissertação consiste numa **pesquisa bibliográfica** que culmina na formulação de um modelo de sistemas de informações para apoiar as operações de gestão de inventários e de demanda na SC.

1.5. ESTRUTURA DA DISSERTAÇÃO

A dissertação é composta de 6 capítulos (ver figura 1) que descrevem as seguintes etapas:

O capítulo 1 apresenta de forma geral a contextualização do tema, os objetivos, justificativa, metodologia e a própria estrutura do trabalho.

O capítulo 2 descreve as características gerais das SC e do SCM, os seus processos de negócio e sua relevância como configuração produtiva, abordando objetivos, características e métodos aplicados.

O capítulo 3 descreve a evolução da tecnologia de informação (TI) e dos sistemas de informação (SI) dentro das operações da SC, bem como a convergência da tecnologia e as limitações dos sistemas ERP para suporte ao SCM.

O capítulo 4 trata dos principais conceitos do paradigma de orientação a objetos e do modelo de componentes. São analisadas as características desta infra-estrutura e sua aplicabilidade às operações das SC, em ambientes heterogêneos e distribuídos. São apresentados os modelos de componentes baseados na arquitetura CORBA (*Common Object Request Broker*), da OMG (*Object Management Group*).

O capítulo 5 descreve a especificação funcional de um modelo de SI para apoiar a gestão de demanda e de inventários ao longo das SC, baseado no

conceito de objetos de negócio, e construído sobre a infra-estrutura do modelo de componentes CORBA.

O capítulo 6 é destinado às conclusões e aspectos finais da dissertação, englobando os principais pontos observados, bem como sugestões para futuras pesquisas na área.

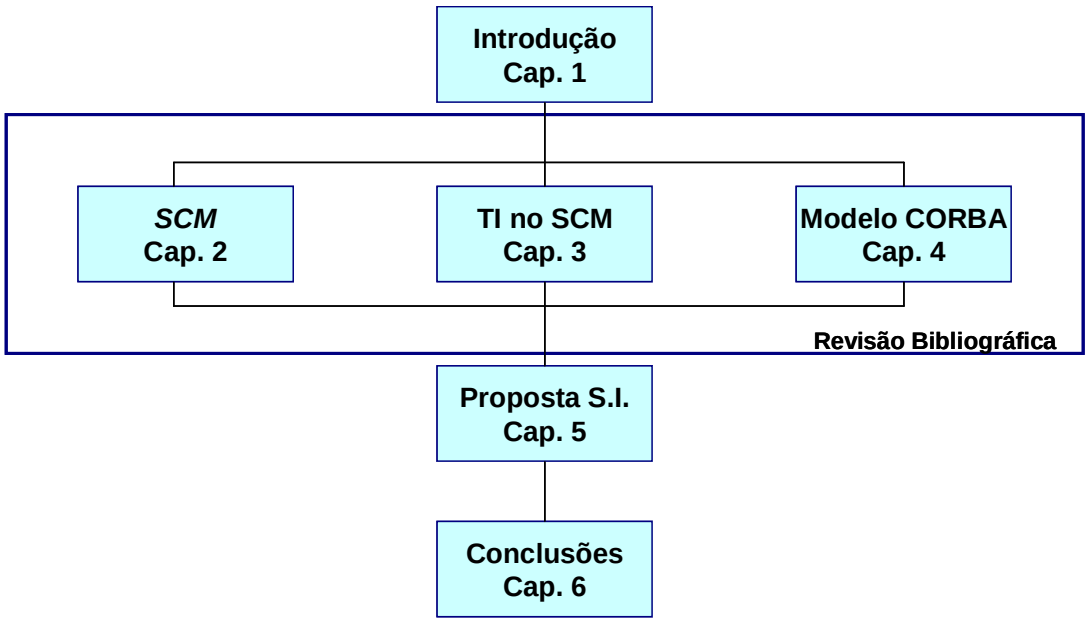

FIGURA 1 – ESTRUTURA DA DISSERTAÇÃO

2. GESTÃO DA CADEIA DE SUPRIMENTOS (SCM)

Estima-se que, na média, de sessenta a setenta por cento do custo final dos produtos é proveniente da matéria prima e de componentes comprados (BOWERSOX, 1996). A medida em que as empresas caminham para um mercado globalizado, no qual as companhias passam a focar sua produção em componentes de alto valor agregado, bem como se concentrarem em competências centrais, esta influência tende a aumentar (PIRES, 2004). A importância dos fornecedores como uma porcentagem dos custos dos produtos manufaturados e a interdependência entre compradores e fornecedores vêm aumentando o desempenho de tais empresas pela redução de estoques, melhoria da qualidade, agilidade nos processos de transformação e distribuição de seus produtos. É importante ressaltar que esta integração está em concordância com as prioridades competitivas de manufatura, descritas no trabalho de Ferdows & De Meyer (1990): custo, qualidade, desempenho de entregas e flexibilidade. Dessa forma, de acordo com Morgan & Monczka (1995), esta integração tem como objetivo melhoria no desenvolvimento de produtos, qualidade e desempenho nas entregas por meio da utilização de competências e tecnologia dos parceiros envolvidos na SC. Estes fatores, em conjunto com as pressões para a diminuição do tempo para desenvolvimento dos produtos, obrigam com que as empresas:

- Estabeleçam rapidamente a eficiência da SC para novos produtos/projetos;
- Coordenem eficientemente o fluxo dos produtos e informações ao longo das SC;
- Identifiquem contextos ambientais que requeiram modificações nos arranjos cliente-fornecedor.

Conseqüentemente, o Gerenciamento da Cadeia de Suprimentos (SCM – *Supply Chain Management*) tem despertado maior atenção como alternativa estratégica para se competir em ambiente globalizado (SALLES, 2003).

Existe muita controvérsia com relação ao termo SCM pois vários autores têm atribuído seu significado à logística que extrapola os limites da empresa (COPPER, LAMBERT & PACH, 1998). No entanto, o termo é bem mais abrangente, e engloba não somente logística, mas também integração de processos, estruturas, sistemas e estratégias, no sentido de se obter sinergia entre os parceiros, para a geração de valor agregado nos produtos entregues ao cliente final (ARAVECHIA, 2001, *apud* VOLLMAN & CORDON, 1996). Adicionalmente, o termo tem sido expandido para englobar também a reciclagem, ou seja, a utilização de logística reversa com o objetivo de se dar um destino final aos produtos que não estejam mais em utilização (BAATZ, 1995).

O SCM tem como foco a utilização dos processos, tecnologia e competências no intuito de se aumentar a competitividade (FARLEY, 1998), bem como para a coordenação de manufatura, logística, distribuição e transporte de materiais dentro de uma SC (LEE & BILLINGTON, 1992).

Dessa forma, SCM tem como meta alcançar a satisfação do cliente, tornar os serviços mais ágeis, melhorar os índices de desempenho e aumentar a competitividade (PIRES, 2004). Entre as razões que tornam o SCM crítico, destacam-se:

- A expectativa dos clientes, com relação aos serviços aumentou;
- A competição é muito mais intensa;
- A aceitação do conceito de parceria entre clientes e fornecedores se estabeleceu;

- A reengenharia acelerou os processos de terceirização e colocou em foco os sistemas de custos, contribuindo para a diminuição do número de fornecedores; e,
- A adoção das melhores práticas (*benchmarking*) entre indústrias fez com que os processos existentes fossem reavaliados e otimizados.

O SCM é difícil de ser alcançado em virtude da diversidade existente nas empresas participantes. Segundo Rice & Hoppe (2001), uma das questões críticas no SCM é o desenvolvimento de competências que possibilitem a integração de processos e atividades ao longo das SC.

Neste sentido, se uma empresa participante da SC tem um desempenho superior que os demais parceiros isto não se traduz em benefício para toda a SC, já que o cliente final percebe somente o valor agregado contido no produto ou serviço final que recebe, e não nos processos isolados envolvidos em cada etapa da transformação. Dessa forma, a competição passa a ser entre SCs, e não mais entre empresas isoladas (PIRES, 2004). Este fato exige com que os seguintes elementos estejam presentes:

- Cada nível dentro da cadeia de suprimentos deve usar ferramentas coerentes para planejamento e gestão de seus processos;
- Deve ser possível integrar a demanda, do mais baixo ao alto nível;
- O *lead time* total (de fluxo de informação e de transporte de bens) deve ser baixo;
- O desempenho global da SC deve ser medido.

Muitas práticas têm sido utilizadas em SCM para simplificar e tornar as SC mais eficazes (ARAVECHIA, 2001 *apud* PIRES, 1998):

- Reestruturação e diminuição do número de fornecedores, e conseqüente aprofundamento no relacionamento e parceria com aqueles cujos produtos apresentam maior importância dentro da cadeia de valor,

dando origem a conceitos como *outsourcing*, terceirizações, parcerias estratégicas e *core competence*. *Outsourcing*, segundo Pires (1998), envolve parceria e cumplicidade com um ou mais fornecedores na SC. Esta tendência é observada com grande intensidade no setor automobilístico, onde grande parte das atividades são transferidas para os fornecedores de primeiro nível (*first tier suppliers*) (COLLINS *et al.,* 1997)

- O compartilhamento de informações e integração de infra-estrutura com clientes e fornecedores, permitindo entregas *just-in-time* e diminuição dos níveis de estoques. Para esta finalidade são normalmente utilizadas integrações entre os sistemas de informação (SI) dos parceiros, tais como *Electronic Data Interchange* (EDI), que se trata de compartilhamento de documentos eletronicamente; e *Efficient Customer Response* (ECR), que é utilizado para que haja uma forma automática de reabastecimento, ou seja, o controle do estoque do cliente fica sob responsabilidade do fornecedor, que efetua o re-suprimento sempre que necessário. Esta prática é mais utilizada nos elos finais (*downstrains*) da SC, ou seja, mais próximos do cliente final.

- O envolvimento maior entre parceiros, tanto no desenvolvimento de produtos quanto na resolução de problemas. Nesta categoria de práticas se destacam duas: o da participação de fornecedores no estágio de desenvolvimento do produto do cliente, o *Early Supplier Involvement* (ESI), cujo objetivo é o de se antecipar possíveis problemas de projeto e diminuição do tempo de desenvolvimento do produto, o *Time to Market* (TTM). Dessa forma há maior possibilidade de que a parceria se consolide ao longo do ciclo de vida do produto (PIRES, 2004). Há também o conceito de *In Plant Representatives* (IPR) cujo objetivo é o de se alocar pessoas no parceiro para garantir com que seus processos estejam adequados aos requeridos pela empresa.

- A concepção de produtos que permitam maior facilidade e flexibilidade no manuseio durante os processos logísticos;

- A gestão de estoques de forma simplificada, conjunta e menos burocrática ao longo da SC, chamada de *Vendor Managed Inventory* (VMI), cujo objetivo é permitir agilidade e também otimizar os fluxos financeiros, já que o produto normalmente é pago quando for efetivamente utilizado. Dessa forma, o material em poder do cliente fica sob regime de consignação;

- A postergação da configuração final do produto para o ponto mais próximo do final da cadeia (ponto de desacoplamento), e consequentemente, do cliente final. Esta prática recebe o nome de *Postponed Manufacturing* e tem como objetivo dar uma maior flexibilidade à gestão de estoques de material em processo e, conseqüentemente, maior redução em seus níveis. Esta prática permite com que haja maior desempenho dentro das SC pois favorece a customização em massa, ou seja, a possibilidade de que um produto atenda a distintos clientes. Este ítem contribui para que o *trade-off* existente entre a estratégia baseada em custo e em diferenciação diminua. Segundo Porter (1999), um *trade-off* é uma correlação negativa, ou seja, quanto mais se busca a produção em escala (custo), menos se obtém a diferenciação. Este *trade-off* é também citado no trabalho de Skiner (1969), com relação às prioridades de manufatura no tocante à estratégia e operações, e que foi posteriormente revisto no clássico trabalho de Ferdows & De Meyer (1990) que apresenta o modelo do “cone de areia”, indicando que ações direcionadas para uma prioridade (ex.: qualidade) podem acarretar benefícios em outras (ex: desempenho de entregas).

A formação de uma SC envolve características diferenciadas. Primeiramente deve ficar claro o que cada empresa lucrará com a cooperação. Não havendo isto, a colaboração e sinergia dentro da cooperação não se desenvolvem. Em segundo deve-se estabelecer a forma de organização da cooperação, principalmente no tocante à questão das responsabilidades. Uma das características de uma SC é que ela se baseia na confiança do relacionamento

e não necessariamente em instrumentos legais. Na formação da SC deve estar estabelecida, inclusive, a forma da dissolução da cooperação. As fases e processos do ciclo de vida das SC são apresentados a seguir (Figura 2). Estas fases, em geral, estão alinhadas ao ciclo de vida dos produtos (Figura 3), conforme apresentado por Kotler (2000):

- Identificação de oportunidades dentro do processo produtivo para aqueles itens que têm um alto grau de importância na cadeia de valor.
- Busca de parceiros, como modelo de integração das melhores competências e recursos.
- Formação, que é a criação da SC em si, sem, porém, implicar no surgimento de uma nova organização formal.
- Operação, que é o gerenciamento da SC e;
- Dissolução/re-configuração, que é a encerramento da cooperação ou então o rearranjo da rede ou integração de novos parceiros.

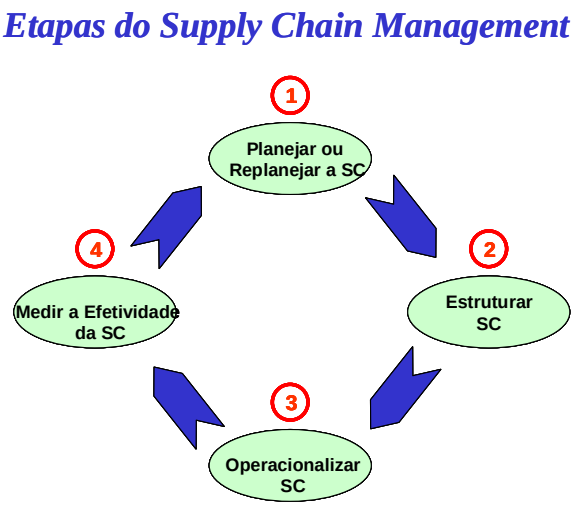

FIGURA 2 – CICLO DE VIDA DAS SC

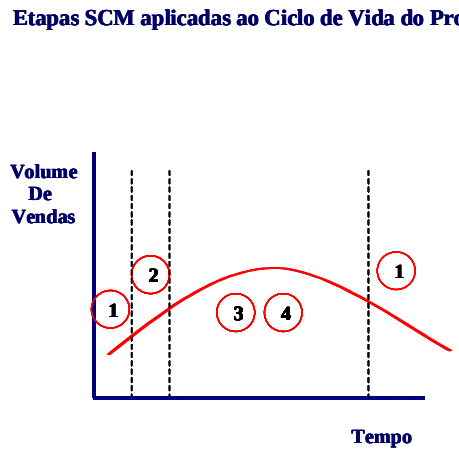

FIGURA 3 – CICLO DE VIDA DOS PRODUTOS (KOTLER, 2000)

Embora as SC sejam configurações produtivas ideais para se competir em ambientes globalizados, alguns obstáculos dificultam o SCM. Um dos maiores desafios está relacionado à gestão integrada de estoques em virtude da variabilidade e imprevisibilidade da demanda, e ao mesmo tempo, obtenção de maximização do desempenho da SC (PIRES, 1998).

Segundo Davis (1993), alguns fatores influenciam o SCM, conforme descrito no quadro 1.

Fator / Objetivo	*Numero de Entidades*	*Estrutura da rede*	*Incertezas*	*Atrasos*	*Qualidade da Informação*
Desenho da SC	*X*	*X*			
Melhor conhecimento dos clientes				*X*	*X*
Melhor desenvolvimento de produtos			*X*	*X*	*X*
Alto nível de satisfação	*X*		*X*	*X*	*X*
Crescimento de toda a SC	*X*	*X*	*X*		
Alto % de Market Share	*X*	*X*	*X*		
Operação mais eficiente	*X*	*X*	*X*	*X*	
Operação globalmente otimizada	*X*	*X*			
Resposta rápida ao mercado	*X*			*X*	*X*
Controle mais preciso de estoque	*X*		*X*	*X*	
Alta taxa de utilizacao de recursos	*X*				

QUADRO 1 – FATORES QUE INFLUENCIAM O SCM (DAVIS, 1993)

Segundo Davis (1993), o controle mais preciso de estoques influencia diretamente a qualidade de informação, as incertezas e os atrasos no SCM.

2.3. O ESTOQUE E O *LEAD TIME* AO LONGO DAS SC

Segundo Disney (2003), em uma SC tradicional, cada participante é responsável pelo seu controle de estoques, atividades de colocação de pedidos e distribuição. O SCM possibilita que a SC aja como uma unidade logística, ao invés de se administrar e “empurrar” inventários isoladamente ao longo desta SC. Dessa forma, a SCM possibilita os benefícios da integração vertical pela coordenação das funções logísticas das empresas participantes da SC (LA LONDE & MASTERS, 1994).

A Cadeia de Suprimentos é uma rede de facilidades que representa a função de procura e transformação do material em produtos intermediários e acabados, e distribuição destes produtos para os clientes. Como se trata de uma rede de fornecimento, com várias empresas participantes, existe grande probabilidade de interferências. Estoques são freqüentemente usados para proteger a cadeia dessas incertezas. Estoques armazenados em diversos pontos da cadeia de suprimento têm diferentes impactos nos custos e no desempenho do serviço da cadeia. Por exemplo, estoques em vários pontos possuem diferentes graus de flexibilidade, pois são mais flexíveis na forma de matéria prima, pois podem transformar-se em diferentes produtos acabados, sem significar um prejuízo muito grande. Finalmente, estoques em vários pontos têm diferentes níveis de resposta, os produtos acabados podem ser embarcados para o cliente sem atraso, ao passo que para produtos em processo ou matérias-prima, o cliente deve esperar até que o produto tenha sua configuração final definida. Um dos grandes desafios do SCM é o de concomitante alinhamento dos níveis de inventários (e respectivos custos) e da busca de maximização do desempenho de entregas no atendimento das expectativas dos clientes (PIRES, 1998). Conceitualmente, este nível de inventário ideal compreende a inexistência de estoques de segurança. Dessa

forma, quantificar a produção e sua correspondente necessidade de materiais se torna um ponto crucial.

Como o inventário projetado é algo não realizado, o aumento dos níveis de estoques de segurança ao longo das SC é utilizado como mecanismo de proteção contra eventuais oscilações de demanda. Porém, esta ação acaba por influenciar negativamente o custo do material ao longo do processo. Em cada elo da cadeia este "coeficiente de estoques de segurança" é acrescido na determinação dos níveis de estoque. A conseqüência desta prática é o aumento gradativo da distorção ao longo da cadeia. A este efeito dá-se o nome de "efeito chicote", ou "*bullwhip effect*" (figura 4).

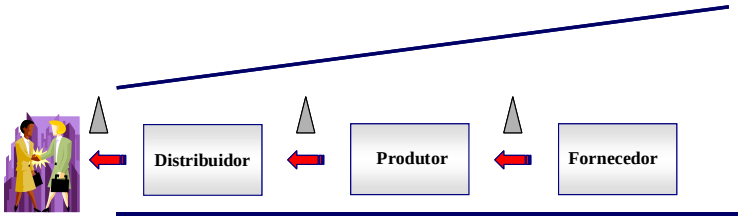

FIGURA 4 – O EFEITO CHICOTE (ADAPTADO PIRES, 2004)

Primeiramente é importante identificar que o "efeito chicote" ocorre principalmente por má visibilidade da demanda e por desconhecimento dos *lead times* (tempos de suprimento) empregados ao longo desta cadeia.

De acordo com Lee *et al.*(1997), a amplificação da demanda é causada por quatro fatores:

1. Atualização da previsão de demanda: a principal informação utilizada para que um fabricante coloque ordens de compra para seus fornecedores é a demanda de seus clientes. Quando tal demanda sofre forte oscilação ela é interpretada pelo fabricante como um sinal de demanda futura, fazendo com que ele atualize sua necessidade de compra para este novo patamar, considerando não somente a

nova demanda, mas também a proporção relativa ao estoque de segurança. Este efeito é então propagado para os demais parceiros da cadeia. Quanto maior for o *lead time* e a dimensão da SC maior tende a ser a amplificação desta demanda.

2. Periodicidade das ordens de compra: as empresas atualizam suas previsões de demanda em períodos pré-definidos (semanas, quinzenas, meses), ao invés de fazê-lo à medida que seus produtos são consumidos. Quanto mais aleatória for a freqüência para a colocação das ordens, maior será a dificuldade para que os fornecedores identifiquem a demanda futura. Uma das principais causas de se estabelecer uma periodicidade para a colocação das ordens é o custo para o processamento destas ordens e para o transporte dos produtos.

3. Flutuação dos preços: quando são adotadas políticas de desconto, condições de pagamento ou outras formas de negociação que acarretam em uma diminuição dos preços dos produtos, a tendência é de que os clientes acabem comprando além de suas necessidades imediatas, causando uma distorção na demanda. Assim que os produtos voltam aos seus preços normais a queda na demanda é imediata.

4. Excesso de demanda: quando a demanda excede a capacidade, normalmente o fabricante limita a quantidade de produtos a serem fornecidos a estes clientes. Por outro lado, o cliente sabendo deste fato, solicita uma quantidade maior que a necessária, gerando distorção na demanda. Quando a demanda voltar a se equiparar à capacidade, a tendência é que haja cancelamento de pedidos e/ou diminuição do volume de compras por parte dos clientes.

Conforme Bowersox (1996), o *lead time* total é composto pela soma do *lead time* de informações com o *lead time* de trânsito de bens e serviços (figura 5).

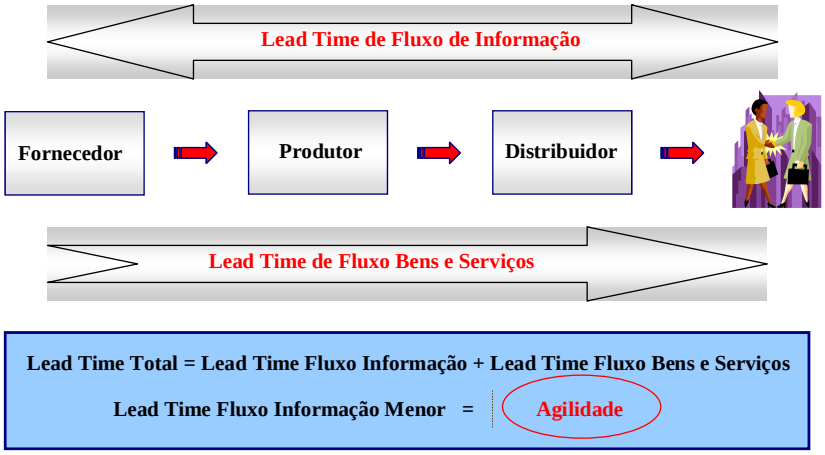

FIGURA 5 – LEAD TIME DAS SC (ADAPTADO BOWERSOX, 1996)

O *lead time* de informação é o principal candidato a ser reduzido drasticamente por possuir pouco valor agregado, porém, a troca de informações de contratos, produtos, produção e a respectiva falta de padrões dificultam sua operação.

Os *lead times* sofrem influência de acordo com o tipo de processo de transformação (de matérias primas até produtos acabados) utilizado.

Segundo Slack (1997), os tipos de processos produtivos são divididos em:

- Processos de projeto (ETO - *Engineered to Order*)
- Processos de "*jobbing*";
- Processos em lotes ou bateladas;
- Processos de produção em massa;
- Processos contínuos.

Com base em Lucero (2001), os tipos de processos produtivos apresentam-se acima em ordem crescente de volume de produção e decrescente de variedade de produtos. Neste sentido, os processos produtivos "*jobbing*" e em lote, pelas suas características de alta variabilidade e baixos volumes com

compartilhamento de recursos, apresenta maior dificuldade para programação e controle. Os processos em lote ou batelada, por outro lado, embora se pareçam com os de "*jobbing*", não possuem o mesmo grau de variedade. Como o próprio nome indica, o lote corresponde à produção de mais de um único item de um determinado produto.

Dentro do escopo de manufatura em pequenos lotes, encontram-se os ambientes de "*jobbing*" e "*batch*" na língua inglesa e os tipos de demanda "*purchase-to-order*" para produtos sob encomenda e "*make-to-order*" para produtos que não devem permanecer em estoque. (LUCERO (2001) *apud* SLACK, 1997).

De acordo com Skiner (1969), para ser considerada arma competitiva, as decisões de operações devem estar cuidadosamente alinhadas à estratégia de negócios, e estas decisões influenciam os *lead times*. Neste sentido, a gestão de inventários e de demanda sofre impactos em função da forma como estas operações são executadas. Esta correlação fica evidenciada no estudo realizado por Safizadeh & Ritzman (1997), onde os autores investigam como a escolha do processo de manufatura se relaciona com decisões sobre planejamento de produção e controle de inventário, e como tais decisões afetam o desempenho das operações sob as seguintes perspectivas:

- Plano agregado, horizonte de planejamento e *backlog*;
- Gestão de materiais na determinação do tamanho dos lotes;
- Gestão de materiais no tocante a níveis de inventário; e
- Indicadores de Desempenho.

A pesquisa demonstra que plantas com produção em linha (produção em lotes pequenos, com centros de trabalho na seqüência em que os produtos são manufaturados) e de processo contínuo (grandes lotes em fluxo contínuo) utilizam mais o nível estratégico de produção, e possuem menores níveis de inventário em matérias primas e material em processo (WIP – *Work in*

Process). Para estas plantas, o desempenho superior é proveniente da utilização efetiva de equipamentos, dos níveis reduzidos de produtos acabados em estoque, e de tempos reduzidos de *setup* e *downtime*. Para obter visibilidade de demanda futura e economias de produção, plantas baseadas em produção por *batch* (produção em lotes relativamente grandes, com equipamentos similares agrupados) e por *jobs* (pequenos lotes e equipamentos similares agrupados) se apóiam mais em *backlogs* de pedidos em seu processo de planejamento. Estas plantas utilizam mais estratégias de alteração de prioridade de produção e inventários posicionados no início do processo de fabricação. Ainda segundo os autores quatro fatores presentes em plantas de alto desempenho são:

- Antecipação de pedidos de clientes;
- Estratégia responsiva de mudança de produção;
- Níveis mais baixos de inventário de matérias primas, e;
- Diminuição do horizonte de planejamento de produção, baseando-se menos em *backlogs* de pedidos.

Para monitorar e continuamente melhorar as operações, *scorecards* de avaliação devem contemplar indicadores de desempenho que se adaptem às escolhas de processo da fábrica.

Hax e Golovin (1978) salientam que as decisões em gestão de manufatura afetam as organizações nas dimensões estratégica, tática e operacional.

- Estratégica: decisões sobre investimentos (especialmente em infra-estrutura e aquisição de recursos), estabelecimento de alianças, desenvolvimento de produto e outras com foco de longo prazo (de um a cinco anos).

- Táticas: decisões sobre produção agregada, planejamento, distribuição e alocação de recursos, cujo objetivo é a efetividade operacional de médio prazo (menos que um ano).

- Operacionais: decisões relativas à produção detalhada, com foco na minimização do custo local, no curto prazo (menor que um mês).

Dessa forma, tais decisões têm impacto na gestão de estoques.

Os sistemas de estoque utilizados nas SC são classificados, segundo Kehoe & Boughton (2001) *apud* Verwijmeren *et al.*(1996), em dois tipos:

- BSC (Controle da Base do Estoque): sistemas deste tipo operam em cada um dos pontos de estoque da SC trabalhando com base na demanda real do cliente ao invés de se apoiar na demanda gerada pelo próximo ponto na SC. Em vez de gerenciar o nível de estoque local, sistemas BSC gerenciam o nível de estoque integral de um ponto de estoque.

- LRP (Planejamento de Requisições de Linha): sistemas LRP também fazem uso do estoque integral, porém utilizam níveis de estoque baseados em tempo, como é o caso com sistemas MRP. Em contrapartida, o LRP explode não somente informação nas requisições esperadas, mas também informações nos níveis de estoque nos pontos *downstream* e *upstream*.

Segundo Kehoe & Boughton (2001), há uma crescente necessidade de que as empresas manufatureiras busquem mecanismos alternativos de gestão das suas operações em rede. Neste sentido, os sistemas de planejamento e controle têm implicações no sucesso das operações ao longo das SC e precisam ser reavaliados constantemente em função dos desenvolvimentos em tecnologias de comunicação e de informação.

Dessa forma, não somente o compartilhamento de informações precisas é importante, mas também a velocidade com que estas informações estejam

disponíveis, fato convergente com a necessidade de que o *lead time* de informação seja minimizado por não agregar valor ao produto final a ser entregue ao cliente (LEE *et al.,* 1997).

A utilização de Sistemas de Informação (SI) que possam auxiliar a gestão integrada de inventários e de demanda nas SC é, portanto, fundamental para a efetividade das operações no SCM.

3. OS SISTEMAS DE INFORMAÇÃO PARA APOIO ÀS SC

A partir de metade do século XX, a indústria ingressou numa era de automação sem precedentes. Grande parte da força de trabalho está empenhada em atividades que não incluem produção. Com isso, os sistemas de alta tecnologia desempenham um importante papel. A evolução da tecnologia propiciou grandes avanços na realização das tarefas, inicialmente, na década de sessenta, automatizando tarefas repetitivas, como as executadas nos sistemas administrativos. Posteriormente, na década de setenta, com o desenvolvimento do *hardware* (e conseqüentemente *software*) as empresas também passaram a utilizar aplicações nas atividades de apoio, tais como automação da força de vendas, MRP-II (Planejamento de Recursos de Manufatura) e outros.

Especificamente na gestão de produção e operações ocorreram contribuições significativas durante este período, tais como *Just-in-time* (JIT), Controle Estatístico de Processo (CEP), *Quality Function Deployment* (QFD), Kanban, engenharia simultânea e outros. No aspecto gerencial, dois novos conceitos também contribuíram para a gestão de manufatura: a logística integrada e o *Supply Chain Management* (SCM) (FLEURY, 2000).

A partir dos anos oitenta, e em decorrência da automação, houve um excedente de mão-de-obra e uma maior competitividade surgiu nos diversos setores da economia, fazendo com que houvesse uma tendência de reorganização empresarial como resposta a esta maior competitividade. Este processo recebeu o nome de BPR (Reengenharia de Processos de Negócio), ou simplesmente Reengenharia. O BPR tem por objetivo a reavaliação e modificação dos processos internos da empresa para dar à organização a agilidade e produtividade necessária (RIGBY, 1983). A grande contribuição do BPR é o de se analisar os processos no sentido de se buscar a efetividade

operacional, evitando-se, com isso, a execução de atividades que não agreguem valor à cadeia de transformação (PORTER, 1999).

Um outro fator que contribuiu para que, nesta fase, houvesse modificações no ambiente empresarial foi a utilização da *Internet* de forma comercial, pois propiciou uma nova forma de globalização, a da informação, dando origem a novas formas de se realizar negócios, em ambientes pautados por grande competição, em que os conceitos e valores se ajustam e se modificam constantemente, gerando expectativas e demandas diferenciadas. Na chamada nova economia a tecnologia de comunicação e de informação começa a assumir um papel importante na integração de parceiros de negócio, tendo como infra-estrutura a internet, e conseqüentemente, novas abordagens de sistemas e processos são necessários.

Fleury (2000) classifica a Tecnologia de Informação (TI) para apoio ao SCM em dois tipos:

- TI transacional: que possui foco na aquisição, processamento e comunicação dos dados. Dentro desta categoria se encontram, por exemplo: ERP (*Enterprise Resource Planning*), MRP (*Materials Requirements Planning*), DRP (*Distribution Resource Planning*).

- TI analítica: que tem por objetivo a análise dos problemas de planejamento das SC pelo uso de modelos de simulação e otimização, para suporte à decisão.

Para efeito desta dissertação, serão considerados os SI pertencentes ao grupo de TI transacional.

No significado léxico, tecnologia é apresentada como sendo a ciência ou tratado acerca dos ofícios e das artes em geral, ou a aplicação dos conhecimentos científicos à produção em geral (DIC, 1996). Em muitas circunstâncias, a interpretação do termo tecnologia está associada à execução de atividades técnicas e a geração de produtos técnicos, com o enfoque da

prática de habilidades específicas. Estas não são visões completas do termo tecnologia (FERREIRA, 2002).

Abreu (1999) apresenta uma evolução do conceito de tecnologia, onde defende que é “um conjunto de ferramentas ou um sistema de ferramentas pelas quais nós transformamos parte de nosso ambiente, derivado de conhecimento humano, para ser usada para propósitos humanos".

Neste sentido Ferreira (2002) salienta que este novo conceito representa um avanço sobre a visão tradicional de tecnologia, que tem como marco principal a revolução industrial, onde apresenta-a como uma invenção para melhorar a qualidade de vida dos homens, passando para uma visão de combinação de processo físico e conhecimento para a transformação de material em produto final.

O desenvolvimento da tecnologia sofre um processo contínuo de avanço cujo ciclo de vida pode ser estabelecido (FERREIRA 2002 *apud* SEBRAE-SP, 2001):

- Fase embrionária: Nesta fase há um grande número de possíveis alternativas para a resolução de problemas, que favorecem o surgimento de diversos modelos até a configuração de um modelo dominante.

- Fase de crescimento: Há a aplicação da tecnologia dominante que contempla o início da padronização da configuração básica aplicada.

- Fase madura: Existe uma tecnologia básica bem divulgada e os processos utilizados tornam-se mais sofisticados e especializados.

- Fase do envelhecimento: Atinge seu máximo desempenho e estagnação. Nesta fase o desempenho incremental é raro.

A Tecnologia da Informação (TI) não tem sido considerada, por muitos autores, tecnologia. Embora exista tal controvérsia, os argumentos acima permitem com

que ela seja utilizada como tal, além do fato deste termo ser amplamente aceito e aplicado pela comunidade empresarial.

Os SI para a gestão de produção e operações sofreram grande evolução desde sua criação e é necessário entender como tais transformações propiciaram a convergência da tecnologia de comunicação e de informação no apoio à logística integrada e ao SCM.

Um sistema é um conjunto de elementos dinamicamente inter-relacionados desenvolvendo uma atividade ou função para atingir um ou mais objetivos ou propósitos (CHIAVENATO,1995).

Um sistema pode compor-se, sucessivamente, de subsistemas que se relacionam, compondo o sistema maior. O SI é um dos subsistemas do sistema empresa, assim como subsistemas de vendas, manufatura e finanças como componentes do SI da empresa (BIO, 1991).

Um sistema de Informação (SI) pode ser conceituado como um *software* desenhado para apoiar, com informações, um processo de negócio. (PAGE-JONES, 1988).

Os SIs têm por objetivo coletar, armazenar e processar informações para apoiar o processo decisório. Davenport (1998) apresenta no quadro 2 uma distinção clara entre dado, informação e conhecimento, já que na prática estes conceitos são muitas vezes utilizados de forma equivocada.

Dados	Informação	Conhecimento
Simples Observação sobre o estado do mundo	*Dados dotados de relevância e propósito*	*Informação valiosa da mente humana inclui reflexão, síntese e contexto*
Facilmente estruturado	*Requer unidade de análise*	*De difícil estruturação*
Facilmente obtido por máquinas	*Exige consenso em relação ao significado*	*De difícil captura por máquinas*
Frequentemente quantificado	*Exige necessariamente a intermediação humana*	*Frequentemente tácito*
Facilmente transferido		*De difícil transferência*

QUADRO 2: DADOS, INFORMAÇÃO E CONHECIMENTO (DAVENPORT, 1998)

Antes do MRP, havia uma técnica baseada em solicitações trimestrais, que foi especificada por George Plossl e Oliver Wight, em 1967. Basicamente, a preocupação com a previsibilidade da demanda começa a se acirrar por ocasião do final da segunda guerra mundial, ocasião em que muitas empresas tinham competência para desenvolver seus planos de produção baseados somente em pedidos firmes. Como a demanda era grande, principalmente devido a recessão existente durante a guerra, houve grande quantidade de pedidos sem atendimento, ou com atendimento parcial. Era comum que pedidos ficassem em *backorder,* em alguns casos, por dezoito meses. Com isso, deu-se início a previsão de vendas em bases trimestrais (WIGHT, 1993).

Esta demanda observada no Ocidente não ocorreu na mesma proporção no Japão que, pela necessidade de se reerguer frente às sanções econômicas e políticas impostas no período pós-guerra, e com dificuldades infra-estruturais básicas, teve que adotar métodos e técnicas que trouxessem resultados adequados e alinhados com baixo custo e alta qualidade. Para tanto, alguns cientistas foram convidados para trabalhar no Japão para disseminar os conceitos de qualidade e, entre eles William Edwards Deming, que a partir de 1950, começou a ministrar seu curso de estatística de processo para os empresários e dirigentes japoneses. O trabalho de Deming trouxe resultados que culminaram, dentro da manufatura, no surgimento da manufatura enxuta (*Lean Manufacturing*), sistema cuja montadora Toyota foi o principal precursor, batizando-o de “Sistema Toyota de Produção” (MAXIMIANO, 2002).

Paralelamente a este fato, nos EUA as previsões trimestrais de demanda eram baseadas em pedidos pendentes, que eram organizados para serem produzidos. No final da década de 1950 e início de 1960, com a escassez de pedidos, as empresas começam a produzir para estoque (*make to stock*), antecipando a demanda futura. Nesta modalidade, Magee (WIGHT, 1993) cita três elementos básicos necessários para um sistema de controle da produção efetivo:

- A previsão da demanda deve ser definida em termos de unidades, e estar associada à capacidade de produção;

- Um plano de produção, ou previsão preliminar deve existir, e;
- Devem existir procedimentos para controlar o nível de reposição do estoque, decidindo em que velocidade eles devem ser ajustados aos valores estimados quando ocorrem desvios (ou erros) de demanda, no intuito de se evitar faltas ou excessos.

Nos anos sessenta, com o desenvolvimento da Tecnologia da Informação, surgiram computadores com acesso randômico aos dados, o que possibilitou com que o MRP fosse utilizado no cálculo da determinação das necessidades de componentes e matérias primas a partir de uma determinada demanda. Com isso, a primeira empresa a desenvolver um sistema de MRP em lotes (*batches,* ou modo assíncrono) foi a *American Bosch Company*, em 1959. Em 1961 – 1962 o primeiro sistema de re-planejamento foi desenhado na empresa J. I. Case sob a direção do então diretor de produção, Dr. Joseph A. Orlicky (WIGHT, 1993).

Em 1965 G. R. Gedye (WIGHT, 1993) declarou que, para otimizar os lucros, as empresas deveriam:

- Minimizar o tempo perdido, maximizando a utilização dos recursos envolvidos no processo de produção;
- Respeitar os prazos de entregas dos pedidos aos clientes, adequando-os, se possível, ao seu processo de produção;
- Manter os níveis mínimos de estoques de produtos em processo, e baixar os inventários de produtos acabados ao mínimo, desde que não inviabilizem os dois objetivos anteriores.

O conceito se difundiu, principalmente nos Estados Unidos, e evoluções foram inseridas no modelo. Estas evoluções culminaram, na década de 70, com o surgimento do MRP II.

O MRP (*Material Requirement Planning*) foi, desta forma, o sistema de gestão da produção que mais tem sido implantado pelas empresas, desde 1970.

O principal objetivo dos sistemas de cálculo de necessidades é o de permitir o cumprimento dos prazos de entrega com a mínima formação de estoques, planejando as compras e a produção de itens componentes para que ocorram apenas nos momentos e nas quantidades necessárias (WIGHT, 1993).

Para tanto, o MRP se baseia em roteiros de produção para dimensionar as necessidades de consumo de materiais para atender uma determinada demanda ou um objetivo de produtividade (como na modalidade de *make to stock*).

Uma deficiência do MRP é que não há a previsão de outros itens que também são pertinentes ao processo produtivo, como mão de obra, utilidades e horas de máquina (não era considerada a capacidade produtiva nesta análise), além disso o MRP possuía um escopo interno dentro do universo de manufatura, pois não havia a integração deste planejamento com a demanda de clientes, nem tampouco com as previsões de entrega dos fornecedores, sem considerar no escopo questões de logística de distribuição e de fornecimento.

Do ponto de vista computacional, os sistemas de apoio a este processo eram lentos devido à limitada capacidade, tanto de memória quanto de processamento dos computadores existentes. Dessa forma, o planejamento das necessidades de produção (BOM – *Bill of Materials*, ou lista de materiais) demandava dias inteiros de processamento.

Em virtude da baixa alteração nas quantidades e na diversidade de itens a serem produzidos (decorrentes da baixa competitividade existente na ocasião), um planejamento deste tipo era válido para semanas e em algumas vezes até meses inteiros.

Com os avanços nos métodos de produção e na capacidade computacional surgiu o MRP-II (*Manufacturing Resource Planning*), cujo princípio básico é o cálculo de todas as necessidades para o processo produtivo, ou seja, das quantidades e dos momentos em que eram necessários os recursos da manufatura (materiais, pessoas, equipamentos, etc.), para que se cumprissem

os programas de entrega de produtos com o mínimo de formação de estoque. Este cálculo era feito a partir das necessidades dos produtos finais.

O MRP II é um sistema hierárquico, em que os planos de produção de longo prazo ou agregados, são sucessivamente detalhados até se chegar ao nível do planejamento de componentes e máquinas específicas (WIGHT, 1993).

O MRP-II é uma evolução do MRP frente às situações de mercado. A diferença básica é que no MRP-II há uma análise de capacidade (RCCP - *rough-cut capacity planning*) e a demanda é baseada em um planejamento estratégico apoiado na análise deste mercado (MTO - *make to order*).

Além disso, o MRP-II considera os itens não inventariados, tais como mão de obra, utilidades e capacidade para elaborar as datas e quantidades necessárias para se atender uma determinada demanda, no processamento de planejamento dos recursos de manufatura.

No MRP-II é permitido o planejamento de recursos distribuídos, que considera a possibilidade de atendimento de uma demanda pela pulverização de suas necessidades para produção em lugares geograficamente distintos. Desta forma, surgem algumas facilidades para o gerenciamento do plano de produção de fábricas satélites.

O MRP-II apóia-se no planejamento estratégico de vendas, com validade de três a seis meses (figura 6). Com base neste plano é feita a análise de capacidade e, uma vez aprovado o planejamento, é elaborado o plano mestre de produção (validade de um mês), é então gerada a programação de fábrica (em intervalos semanais) com base neste plano, fundamentada nos pedidos dos clientes. Caso um pedido não seja atendido a tempo, este fica pendente (*back order*) e em um próximo planejamento ele é atendido em primeiro lugar.

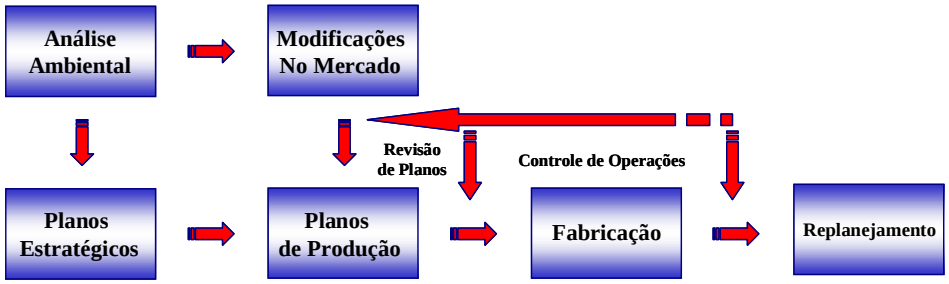

FIGURA 6 – COMPONENTES DO MRP-II (WIGHT, 1993)

Um outro avanço do MRP-II para o seu predecessor é que esta metodologia permite com que sejam definidos indicadores de desempenho, tais como número de ordens atendidas no prazo, quantidade de itens produzidos fora do especificado, nível de inventário e outros. Esta facilidade permite que sejam tomadas ações preventivas frente a alterações de mercado e/ou imprevistos no processo produtivo.

O aumento da capacidade computacional permitiu com que fossem executadas simulações de planejamento com base em variações de roteiro de materiais e de alterações na demanda, resultando em possibilidade de seleção da melhor combinação de produção, visando produtividade e atendimento de demanda.

Do ponto de vista tecnológico, porém, as aplicações baseadas no MRP-II não consideram a capacidade e disponibilidade dos fornecedores e logística de distribuição.

Uma outra deficiência é no suporte ao MRP seletivo pois as aplicações existentes realizavam apenas o MRP regenerativo. MRP regenerativo é o re-planejamento das quantidades e prazos das ordens de fabricação em virtude de uma mudança nas ordens previstas (normalmente por um gargalo na produção ou no fornecimento). Isto significa que as datas anteriormente firmadas podem ser re-planejadas e não se tem uma visibilidade do impacto desta alteração nos níveis inferiores do plano, ou seja, a alteração em uma ordem tem potencial influencia em todas as demais.

Já o MRP seletivo implica que quando há alteração em uma ordem planejada (quantidade ou data entrega), pode-se apenas re-planejar esta e as demais que têm relação com esta mudança, conservando intactas as demais ordens. O MRP seletivo é de difícil implementação, pois exige algoritmos especializados baseados em regras de seqüênciamento ou aqueles com base em inteligência artificial, como as redes neurais.

No final da década de oitenta, como uma tentativa de integração dos sistemas de informações existentes nas empresas, e também como uma nova oportunidade de negócios para as consultorias, em parte devido à globalização, e em parte devido ao aumento de produtividade em escala oriundo da própria tecnologia, surgem os sistemas ERPs (*Enterprise Resource Planning*).

Existem muitas definições do que seja um ERP, de acordo com a APICS (*American Production and Inventory Control Society*) é uma extensão do conceito de planejamento de recursos de manufatura, ou seja, sugere que seja uma nova versão do MRP-II, modificada e ajustada para apoiar as organizações diante do aumento da competitividade nos anos 90. Gumaer (1996) concorda com este princípio, acrescentando apenas o fato de que o ERP utiliza novas tecnologias de informação, como banco de dados relacionais, interfaces gráficas, sistemas abertos e arquitetura cliente/servidor. Dentro deste contexto incluem-se ferramentas para planejamento de capacidade finita e sistemas de execução de manufatura (MES). Alguns outros autores, como Farley (1998) e Stevens (1996) afirmam que o ERP vai além do planejamento centrado em materiais, mão de obra e produção, pois oferecem suporte a múltiplas subsidiárias ao mesmo tempo, consolidando e apoiando suas operações. Para Michel (1998) ERP é um termo genérico para “sistemas integrados para apoiar processos corporativos”, ou seja, automatiza e integra processos de negócio encontrados em ambiente de manufatura, incluindo aqueles existentes no chão de fábrica. Dessa forma o ERP permite com que a informação inserida no sistema seja visível e passível de acesso por todas as áreas da organização. A abrangência de atuação de um ERP pode englobar o suporte a uma rede de subsidiárias, possibilitando que cada uma utilize

métricas e procedimentos particulares, mas gerando resultados consolidados, funcionalidade aderentes com as necessidades de uma economia globalizada. Estas aplicações são compostas por módulos com funções que apóiam as principais áreas de negócio das empresas (figura 6): manufatura, comercial, logística e finanças (BIO, 1991). Por conceito, ERP é um sistema integrado no sentido de que as informações inseridas por uma área têm influência sobre as demais. Uma outra característica é que os ERP têm como espinha dorsal a metodologia MRP-II para planejamento dos recursos necessários. Algumas razões que propiciaram com que houvesse uma demanda por sistemas ERP foram:

- A idéia da utilização de um sistema integrado, propiciando a adoção de procedimentos padronizados, possibilitando maior controle gerencial e menor esforço de operação.

- Diminuição de custos em tecnologia de informação pela adoção de tecnologia mais recente, e conseqüentemente suporte do próprio fornecedor da aplicação.

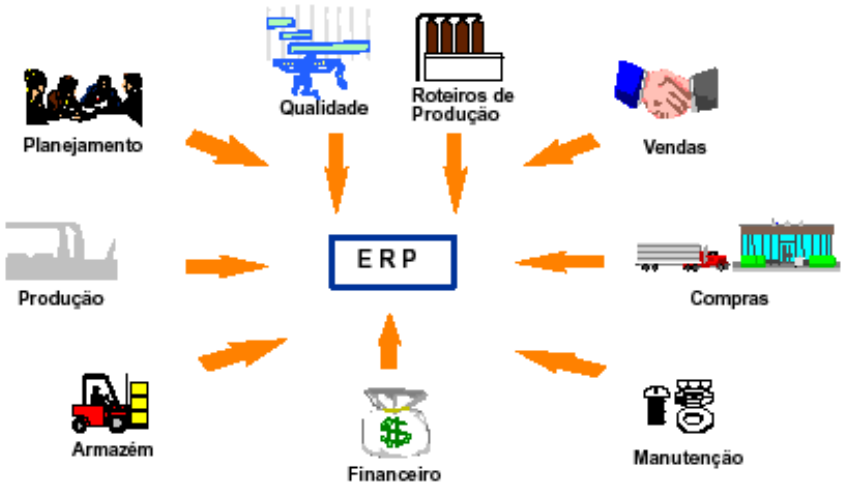

FIGURA 7 – MÓDULOS DE UM SISTEMA ERP (ADAPTADO BIO, 1991)

O principal benefício direto desta arquitetura é, portanto, a unicidade da informação, eliminando atividades de re-digitação, sem valor agregado aos negócios.

Segundo Bergamaschi (1999) a maior expansão do mercado ERP em 1997 ocorreu na América Latina (63%) e Ásia/Pacífico (68%), regiões com rápida expansão econômica na época. As estatísticas apontam que, embora as indústrias de processo representassem 40% de todas as instalações industriais, apenas 20% das vendas de sistemas ERP em 1997 foram destinadas a elas. Uma possível explicação é que o MRP, principal componente dos sistemas ERP, focavam principalmente os processos de manufatura discreta.

Ainda, de acordo com Bergamaschi (1999), os principais fornecedores de aplicações ERP são europeus, representando 45% do valor global de vendas de licença de *software* em 1997.

Uma razão do grande sucesso dos ERP é o aspecto político e organizacional, pois a prática administrativa dominante no final da década de oitenta era o da reengenharia de processos, com as conseqüentes terceirizações de atividades e pessoas. Os ERP estavam alinhados com esta tendência pois eram baseados nas melhores práticas de negócio, evitando re-trabalho com atividades sem valor agregado, tais como a inclusão dos mesmos dados em sistemas paralelos. Desta forma, muitos gestores viram nos ERP a solução ideal para integrar seus processos internos, deixando em segundo plano os aspectos organizacionais, tais como cultura, estratégia e estrutura da empresa.

Alguns fatores que dificultam as implementações de sistemas ERP decorrem do fato de que tais aplicações, assim como a maioria dos SI "*on the shelf*" possuírem funções genéricas e padronizadas. Dessa forma, para que tais sistemas possam ser aderentes aos processos de empresas pertencentes a setores diversificados da economia, desde indústrias de processo a empresas de serviços, há a necessidade de um alto grau de flexibilidade e profundidade em características específicas de cada empresa/setor, o que poderia

descaracterizar a base conceitual de processos sobre o qual ele é construído (que os produtores de aplicações ERP chamam de “melhores práticas”), além de dificultar o processo de confecção do *software*.

Atualmente várias alternativas tecnológicas têm sido utilizadas para que haja a comunicação física entre diferentes SI, tais como os *drivers* que atuam como “pontes” para integração de SI dispersos entre diferentes atores, ou mesmo modificações na própria estrutura do ERP para que ele possa ser aderente aos processos específicos destas empresas. Esta prática gera, no entanto, custos adicionais de adaptação e manutenção dos SI, além de transtornos para as organizações pois as funções principais do sistema mantêm uma harmonia de atividades entre si, que não é atingida quando se é necessário utilizar código adicional para agregar novas funções.

Uma outra alternativa é a de se construir SI para apoio à decisão, tendo como infra-estrutura de informação os ERP. Neste sentido, aplicações de BI (*Business Intelligence*) se baseiam fortemente na agregação e transformação de dados transacionais oriundos destes ERP, sendo estes transformados e organizados de forma que possam fornecer subsídios para o processo decisório dentro da gerência média.

A segunda limitação é na sua base para o planejamento das necessidades de materiais, que é baseada no MRP e não em BSC / LRP, como é o caso aplicado às SC (KEHOE & BOUGHTON, 2001).

A terceira razão da deficiência dos ERP como ferramenta de apoio aos negócios empresariais ocorre porque eles são construídos com base na estrutura organizacional de unidade de negócios e não em SC (KEHOE & BOUGHTON, 2001).

Na década de 1990 havia uma limitação técnica com relação aos ERP pelo fato deles não terem sido concebidos para ambiente distribuído. Atualmente, porém, há diversas alternativas para se vencer a barreira da plataforma. O que se observa, no entanto, é que embora haja custo adicional para se integrar SI

de empresas diferentes, somente com o avanço da utilização das SC como alternativas produtivas viáveis é que os produtores de ERP começam a desenvolver soluções de integração abertas.

Para que um SI possa suportar adequadamente os processos de uma SC, e auxiliar o SCM, ele deve ser construído sob uma arquitetura distribuída, ou seja, que não esteja fisicamente localizado em um servidor ou rede local.

Para se compreender todos os aspectos envolvidos no SI distribuído é importante que alguns conceitos sejam fornecidos.

Um sistema distribuído pode ser compreendido como coleção de computadores (nós) autônomos, conectados através de rede física e lógica, equipados com *software* projetado para facilitar a integração computacional. Tais sistemas são construídos sobre plataformas de "*hardware*" geralmente variantes no número e, muitas vezes heterogêneas (MASTELARI, 2004).

Um SI para ambiente distribuído é construído com base em objetos distribuídos, utilizando o paradigma de orientação a objetos. Este conceito será melhor detalhado posteriormente.

O conceito de objetos distribuídos é baseado nos princípios do modelo cliente-servidor e do paradigma orientado a objetos, permitindo: a abstração, que reduz a complexidade no desenvolvimento de SI, e concentração de dados comuns às diversas aplicações em um único lugar, o que evita replicações desnecessárias e leva à especialização de serviços (COELHO, 1998).

Estes sistemas devem atuar em diferentes ambientes (equipamentos, sistemas operacionais) e permitir o uso de serviços desenvolvidos em diferentes linguagens, ou seja, compostos de elementos heterogêneos. Pelo fato de ser um ambiente heterogêneo, há a necessidade de se utilizar especificações abertas, com interfaces padronizadas e públicas, levando ao desenvolvimento de "*middlewares*" abertos. Um "*middleware*" é uma camada de *software*, residente sobre o sistema operacional e do mecanismo de comunicação, que oferece abstrações de alto nível, com objetivo de facilitar a programação

distribuída (MASTELARI, 2004). As abstrações oferecidas fornecem uma visão uniforme na utilização de recursos heterogêneos existentes nas camadas de sistema operacional e redes (MONTEZ, 1997).

Como solução proposta para a necessidade de agilidade e flexibilidade no espelhamento dos processos de negócio nos sistemas de informações deve-se considerar o conceito de engenharia de *software* da orientação a objetos, com o apoio de linguagem interpretada (ou seja, transcrita para a linguagem de máquina no momento da execução, com o objetivo de manter a flexibilidade), e uma arquitetura de aplicações voltada a processos de negócio, tanto na modelagem quanto na construção e funcionalidade.

Segundo Bowersox (1996), a previsão de vendas, é um item importante dentro do Gerenciamento da Cadeia de Suprimentos para otimizar a gestão de estoques. Estoques muitas vezes causam comodidade, ou seja, ambientes do tipo *just-in-time* a dependência de estoques é mais crítica. Desta forma a Tecnologia da Informação, aplicada em ambientes distribuídos, e as boas práticas de gestão podem minimizar os impactos de variabilidades de estoque.

O próximo capítulo trata da evolução dos sistemas de informação (SI), bem como da convergência da Tecnologia da Informação (TI) para apoio às atividades de SCM. Em seguida será apresentado um modelo de arquitetura para infra-estrutura de comunicação e, por fim, uma proposta de um SI para apoio à gestão de demanda e de inventários, tendo por base esta arquitetura e os requisitos de negócios de SCM.

4. INFRA-ESTRUTURA DE TI PARA AMBIENTE DISTRIBUÍDO

O paradigma de orientação a objetos é a metodologia que atualmente atende os requisitos de processos de negócio em um ambiente de constante instabilidade.

Para se conceituar adequadamente objeto, é importante compreender o significado de modelo de dados.

Conforme Kern (1997), modelo de dados é utilizado para estabelecer o fundamento da arquitetura do banco de dados. Cada modelo de dados tem por objetivo “modelar o mundo deste sistema” tão bem quanto possível. Os modelos de dados podem ser classificados em:

- Hierárquicos: este modelo apresenta o banco de dados como uma estrutura de árvore na qual cada registro tem apenas um ascendente, com a exceção do registro-raiz, que não tem ascendente. O modelo hierárquico deu a fundamentação para os primeiros sistemas gerenciadores de banco de dados (DBMS).

- Rede: é uma generalização do modelo hierárquico. O modelo de rede possibilita com que cada registro de dados tenha vários ascendentes e descendentes.

- Relacional: seus fundamentos foram definidos por CODD (1970). O modelo relacional suporta a abstração de sistemas de banco de dados tendo por base coleções de tabelas e relacionamentos entre elas.

- Objeto-orientado: este paradigma tem os tipos abstratos de dados, a herança, e a identidade como seus aspectos mais fundamentais (KHOSHAFIAN 1993). Apóia o modelo de dados por meio de linguagens de programação orientadas a objetos.

Além dos modelos básicos, há os semânticos, que são conceituados sobre abstrações mais altas de um tipo básico. Sua aplicação não está restrita a um dos quatro modelos anteriores (KERN, 1997). O modelo semântico mais conhecido é o modelo Entidade-Relacionamento (ER), de CHEN (1976).

Os objetos de negócio surgiram da necessidade de se expressar conceitos de negócios como conjunto de objetos em *software*, isto é, como componentes de *software*. Dentro do modelo de negócios, um objeto é semelhante a uma entidade no modelo de dados Entidade-Relacionamento, que pode ser conceituada como "a representação de alguma coisa existente no mundo real que tem significado para o sistema a ser modelado" (CHEN, 1976).

Um objeto pode ser entendido como: Objeto = dados (privados) + operações (públicas). Neste conceito há a unificação de dois elementos que na programação de SI tradicional são tratados de forma separada, os dados e as operações (MASTELARI, 2004). Portanto, o termo objeto significa a combinação de dados e programas unidos para representar uma entidade do mundo real ou de um mundo imaginário.

Para que estes objetos de negócio possam ser representados de forma satisfatória nos sistemas distribuídos, dentro da tecnologia de orientação a objetos, e mediante os requisitos de negócio, eles devem possuir os seguintes atributos (OMG, 1995):

- **Identidade**: Um objeto de negócio possui uma identidade que o associa com a entidade que ele representa no domínio de negócio. Esta chave identifica o objeto de negócio e o relaciona ao SI.

- **Transacional**: Em geral, os objetos de negócio são acessados em um contexto transacional. Devido ao fato deles serem compartilhados em um ambiente distribuído e acesso simultâneo (vários usuários ao mesmo tempo), deve haver controle de concorrência e serialização de transações (execução em fila única) para se manter a integridade do modelo que eles representam. Estas funcionalidades devem ser

construídas nos objetos de negócio de tal forma que os programadores necessitem apenas definir o início e termino da transação.

- **Persistência**: A maioria dos objetos de negócio é persistente, ou seja, permanece ativo mesmo em modo assíncrono. Isto é necessário para se manter o *status* dos objetos (ex. os dados) quando há uma falha no sistema ou ele é desligado. Persistência não é requerida para todos os objetos de negócio, mas se eles representam informações atuais sobre o negócio, eles devem ser persistentes para permitir recuperação.

- **Atributos:** Os objetos de negócio podem ter atributos que o descrevam. Eles são associados com elementos de dados ou objetos que provêm informações sobre eles.

- **Estados:** Um objeto de negócio pode ter um estado (*status*) definido. Enquanto estados e transições de estados podem ser parte da especificação do objeto, as *interfaces* ou atributos de estado não são diferentes de outros atributos exceto pelo fato deles requererem valores enumerados. O estado é representado pelos valores dos atributos dos objetos.

- **Relacionamentos**: Os relacionamentos de um objeto de negócio representam associações com outros objetos. Os relacionamentos podem ser 1-1 (um-para-um) ou 1-m (um-para-muitos) e podem ser uni ou bidirecionais.

- **Operações**: Os objetos de negócio executam operações (ex.: eles possuem métodos para os quais suas interfaces definem assinaturas). Operações são executadas em um contexto transacional.

- **Notificações:** A maioria dos objetos de negócio suporta notificações de evento. Uma mensagem pode ser enviada para um objeto requisitando que certos eventos sejam notificados ao solicitante designado. Dessa forma, com base em um protocolo padrão, cada objeto de negócio pode prover esta notificação.

- **Eventos**: São mudanças de status dos objetos. Estes eventos podem ser: 1) Intrínsecos, ou seja, ocorrem dentro do próprio objeto (por exemplo: por uma alteração no estado do ambiente operacional; 2) implícitos, quando atributos ou relacionamentos são modificados ou quando operações são invocadas, completadas ou quando há falha; e 3) programados, são aqueles que são declarados na especificação das interfaces e são gerados pela lógica do negócio. O comportamento é definido pelos programas que atuam sobre o estado do objeto. Estes programas são denominados de métodos (MASTELARI, 2004).

- **Tipos Dependentes**: Os objetos de negócio dependentes têm sua existência (persistência) vinculada ao objeto de negócio principal. A maioria dos atributos é passada por valor durante uma interação.

- **Tabela ou Seqüência**: Tabelas podem conter objetos elementares ou valores primitivos. Seqüências podem conter, além dos anteriores, objetos compostos.

Como salienta Mastelari (2004), os objetos podem ser agrupados de acordo com suas características comuns, o que se denomina classificação. Por exemplo, duas pessoas João e Paulo podem ser classificadas como pessoas. Por sua vez João e Paulo podem ser compreendidos como instâncias da classe pessoa, a instanciação é o processo inverso à classificação. A partir da abstração classificação/instanciação surge o conceito de classe, uma classe é a descrição de um conjunto de objetos que compartilham os mesmos atributos, operações, relacionamentos e significado (BOOCH *et al.*,1999). O termo objeto pode, dessa forma, se referir tanto à classe como à instâncias de uma classe.

Todos os aspectos apresentados devem ser considerados quando se define uma arquitetura de referência que provê uma estrutura concisa e efetiva onde suas dependências entre domínios individuais devem ser entendidas e conciliadas.

Graham (1994), enfatiza que o termo “Métodos Orientados a Objetos” pode se referir à programação orientada a objetos, projeto orientado a objetos, análise orientada a objetos, banco de dados orientados a objetos, interface gráfica orientada a objetos, ou seja, de forma geral se refere a toda uma filosofia de desenvolvimento de sistemas.

Mastelari (2004) argumenta que a busca pelo desenvolvimento de *softwares* com métodos orientados a objetos começou com a programação orientada a objetos, dentro do contexto da “crise do *software*”, no final da década de 60. Nesta ocasião concluiu-se que os métodos de produção de software não eram adequados em função da crescente necessidade de qualidade, com menores custos, e maior eficiência e reutilização de código, além de melhor previsibilidade de prazos e custos (BUZATO & RUBIRA, 1998). A orientação a objetos oferece abordagem diferente para o desenvolvimento de *software*, sendo mais uma abordagem evolucionária do que revolucionária.

A técnica da orientação a objetos permite com que o *software* seja construído a partir de entidades abstratas chamadas de objetos, que têm características e comportamento específicos. Estes objetos, por sua vez, podem ser construídos a partir de outros objetos e assim sucessivamente. Desta forma o uso do modelo de objetos permite com que um *software* seja mais facilmente codificado a partir de componentes já existentes, que são formados por conjuntos de objetos. Estes conceitos permitem com que haja re-usabilidade (MASTELARI, 2004).

Um componente pode ser definido como uma unidade de *software* que possui *interfaces* e dependências de contexto bem definidas. Um componente pode ser instalado independentemente ou formar composições com outros componentes (SZYPERSKI, 1998).

Pelo fato de obedecerem a um padrão bem definido, os componentes se tornam reutilizáveis pelas aplicações, possibilitando o desenvolvimento rápido de aplicações complexas. Para isso, basta escolher um conjunto de

componentes, configurar as suas propriedades e efetuar as conexões entre eles.

Dessa forma, os benefícios advindos do modelo de componentes de objetos são vários: abstração de dados/encapsulamento, modularidade, hierarquia, reutilização de código, facilidade de extensão e manutenção (BUZATO & RUBIRA, 1998), aumento da produtividade e qualidade do desenvolvimento de *software* através da re-usabilidade de classes, maior flexibilidade e menor custo de manutenção devido a extensibilidade e herança, facilidade na descrição e construção de interfaces gráficas e sistemas distribuídos devido à comunicação entre objetos com o uso de mensagens (GRAHAN, 1994).

Barros (2003) apresenta algumas outras vantagens do uso de componentes de *software*, como:

- SI baseados em componentes são compactos pois possuem apenas o código relacionado ao negócio, já que os elementos de infra-estrutura e comunicação se encontram encapsulados na própria tecnologia de componentes.

- O desenvolvimento baseado em componentes possibilita com que o SI seja modular e com *interface* bem definida.

- A escalabilidade, ou seja, a facilidade de se implementar melhorias, é maior, pois é possível substituir um módulo completo de uma aplicação, por outro mais eficiente, sem modificar linhas de código.

Apesar da orientação a objetos ser uma condição necessária para espelhar os processos de negócio, ela não é suficiente. É necessário que exista uma infra-estrutura que suporte conexões de componentes de objetos para que eles tenham mobilidade dentro de um ambiente heterogêneo e distribuído.

No intuito de dar uma resposta à necessidade de se obter uma infra-estrutura de serviços e comunicação padronizada para o desenvolvimento de aplicações

distribuídas, o OMG (*Object Management Group*) começou a trabalhar nesta solução, a partir de 1993.

O OMG é um consórcio internacional formado por empresas, universidades e instituições de pesquisa da área de computação, para definição de padrões na área de objetos distribuídos, que permitam a interoperabilidade e portabilidade de aplicações distribuídas, formado em 1989 (BARROS, 2003).

O OMG produz apenas especificações (não *software*), baseadas em tópicos de interesse. O processo para a geração de uma nova especificação segue os seguintes passos: 1) O OMG lança uma requisição por informação (*Request for Information* - RFI) e uma requisição por propostas (*Request for Proposals* - RFP). 2) Os consorciados enviam idéias e tecnologia como resposta aos RFI e RFP. 3)A especificação será montada com base nestes dados e enviada aos membros para discussão e aprovação, por consenso. 4) Esta especificação é então publicada e pode ser usada comercialmente.

Entre estas especificações, o OMG desenvolveu um modelo padronizado para interoperabilidade, chamado de CORBA (*Common Object Request Broker Architecture*), com base no OMA (*Object Management Arquitecture*), em 1995 (SIEGEL, 2000).

A estrutura do CORBA reside no Modelo de Referência, ilustrado na figura 8 e definido na Arquitetura OMA (*Object Management Architecture*) da OMG (SOLEY & STONE, 1995), que consiste de:

- Camada de Chamadas a Objetos (ORB): Habilita os objetos a fazerem chamadas a outros objetos e receberem, de forma transparente, respostas em um ambiente distribuído.

- Serviços de Objeto: Coleção de serviços de apoio (*interfaces* e objetos) que suportam funções básicas para a utilização e a implementação de objetos. Os serviços são necessários para se construir aplicações em ambiente distribuído, e são independentes do domínio destas aplicações.

- Facilidades Comuns: Conjunto de serviços que oferecem capacidades de uso geral e que são úteis para muitas aplicações (por exemplo: *e-mail*, impressão).

- Objetos de Aplicação: São objetos específicos de aplicações particulares de usuários. Não são padronizados pela OMG.

A arquitetura CORBA permite com que as aplicações façam solicitações a objetos, de uma forma transparente e independente de linguagem, sistema operacional, *hardware*, ou considerações de localização (MASTELARI, 2004).

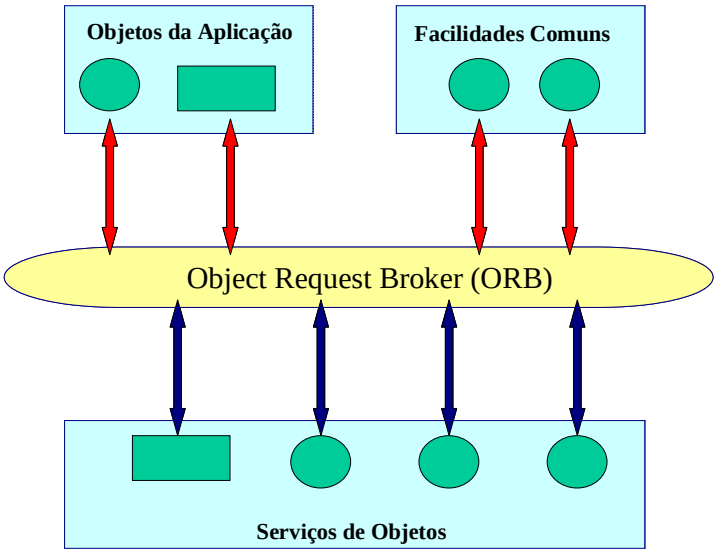

FIGURA 8 – MODELO DE REFERÊNCIA OMA (SOLEY & STONE, 1995)

O ORB é o elemento responsável por realizar a *interface* entre os diversos componentes da arquitetura OMA e é o principal componente do modelo CORBA (OMG, 1995).

O ORB provê “os mecanismos básicos para fazer chamadas de modo transparente e receber respostas de objetos localizados local ou remotamente, sem que o cliente necessite ter conhecimento sobre os mecanismos usados

para representar, comunicar-se com, ativar, ou armazenar os objetos” (OMG, 1993).

Sua funcionalidade é explicada:

> "Usando um ORB, um cliente pode chamar de modo transparente e iniciar uma ação em um servidor de objetos, que pode estar na mesma máquina ou em outro ponto de uma rede. O ORB intercepta a chamada e localiza um objeto capaz de implementar a chamada. A seguir, passa os parâmetros ao objeto, invoca seu método, e retorna os resultados. O cliente não precisa estar ciente sobre onde o objeto está localizado, sua linguagem de programação, sistema operacional, ou qualquer outro aspecto do sistema que não é parte da interface do objeto. Desta forma, o ORB implementa a interoperabilidade entre aplicações e interconecta vários sistemas de objetos de modo imperceptível”. (OMG, 1993)

Os objetos disponibilizados por um ORB publicam suas *interfaces* usando a linguagem IDL de CORBA. Em IDL, os atributos e operações de um objeto são especificados de modo independente de linguagem de programação. A gramática de IDL é um subconjunto do padrão proposto ANSI (*American National Standards Institute*) C++, com construções adicionais para suportar objetos distribuídos (OMG 1995). A herança múltipla é permitida.

A IDL é uma linguagem descritiva, obrigando com que as implementações sejam construídas com mapeamentos de IDL para linguagens de programação. VINOSKI (1993) salienta que pelo fato da IDL ser declarativa a separação entre *interface* e implementação é enfatizada (KERN, 1997).

Além do CORBA, outras especificações têm sido elaboradas para padronizar e apoiar a interoperabilidade entre aplicações:

- DCE (*Distributed Computing Environment*): é um conjunto de serviços projetado para suportar aplicações distribuídas com base em RPC (*Remote Procedure Calls*), mecanismo responsável por acionar aplicações e sistemas operacionais para a realização de tarefas, uma de

cada vez (KERN, 1997). Ele foi desenvolvido pela OSF (*Open Software Fundation*).

- OLE2: é uma tecnologia de objetos (serviços de aplicação) desenvolvida pela Microsoft, dedicada originalmente a suportar documentos compostos, baseada em COM. O COM foi desenvolvido em conjunto pela Microsoft e Digital Equipment Corporation para prover suporte a objetos distribuídos (KERN, 1997). O COM define uma interface de programação de Aplicação (API) para permitir a criação de componentes para uso na integração de aplicações customizadas ou para permitir que diversos componentes se interconectem. Porém, para que isto seja feito, é necessário que os componentes sejam aderentes ao padrão da Microsoft. Se há esta aderência, componentes escritos em linguagens diferentes podem se comunicar, caso contrário, a integração não é possível. Os objetos e interfaces COM são especificados utilizando-se Microsoft *Interface Definition Language* (IDL), uma extensão da DCE. Todo objeto COM é processado dentro de um servidor. Um único servidor pode suportar múltiplos objetos COM. Existem três formas pelas quais um cliente pode acessar objetos COM providos por um servidor: 1) Ligando-se diretamente à biblioteca no servidor, dentro do mesmo processo (*in-process server*). Neste caso são utilizadas funções de chamadas; 2) Dentro de um mesmo servidor, mas em processos diferentes (*local object proxy*), com a utilização de RPC (mecanismos de chamada remotos); 3) Acesso a um servidor remoto, em outra máquina, utilizando-se para tal DCE RPC (*remote object proxy)*. O mecanismo que suporta acesso a servidores remotos é o DCOM.

- DCOM: É uma especificação e implementação desenvolvida pela Microsoft que provê uma arquitetura para integrar componentes. Foi totalmente baseado no modelo ORB da OMG, porém com limitações visando maior funcionalidade nos softwares da própria Microsoft, em detrimento de uma maior portabilidade. O DCOM (*Distributed* COM) é uma extensão do COM que permite interação entre componentes

distribuídos. Os processos COM podem ser processados na mesma máquina, porém em espaços de memória diferentes (*address spaces*). Por outro lado, as extensões DCOM permitem processos distribuídos em uma rede. Com o DCOM, componentes operando em uma variedade de plataformas podem se comunicar.

- *JavaBeans: JavaBeans* (BARROS, 2003) compreende um modelo de componentes de *software* para Java, que foi criado em 1996, pela Sun Microsystems. Os componentes *Bean* podem ser combinados para formar: outros componentes, *applets* (que são carregados em tempo de execução), aplicações Java *stand-alone*, ou *servlets* (usados para processar comandos e retornar páginas *web*). A comunicação entre os JavaBeans é feita através do envio/recepção de eventos e o mecanismo de persistência possibilita serializar os *JavaBeans*. Cada *Bean* deve ser capaz de ser processado em diversos ambientes, facilitando a portabilidade. Além disso, é possível a implementação de *interface* entre o modelo de componentes *JavaBeans* e outros, como COM, facilitando a interoperabilidade.

- *Enterprise JavaBeans*: *JavaBeans* é um modelo de componentes orientado para aplicações clientes (*frontend*). Em 1998 foi lançado o *Enterprise JavaBeans* (BARROS, 2003 *apud* ORFALI & HARKEY, 1998), que incorporou funcionalidades para tornar o modelo *JavaBeans* mais apropriado para aplicações servidoras (*back-end*). O *Enterprise JavaBeans* (EJB) suporta dois modelos básicos para transações: *bean* de sessão e *bean* de entidade. O *bean* de sessão representa um objeto transiente, que é criado pelo cliente e eliminado após sua utilização ao passo que o *bean* de entidade representa objetos transacionais, com estado e tempo de vida longo. O modelo EJB provê uma série de serviços, entre eles: gerenciamento de ciclo de vida, segurança, transações, persistência e gerenciamento de estado.

Ao passo que CORBA oferece completo suporte ao paradigma de orientação a objetos, DCE não (BRANDO 1996). Vinoski (1993) dá um exemplo das limitações de DCE no suporte à programação orientada a objetos:

> "Como despachar um objeto C++ através de uma rede de um processo a outro? ... O fato de que objetos C++ podem conter ponteiros ocultos para tabelas de funções virtuais acaba por criar uma pedra no caminho, mesmo para o programador mais determinado, especialmente quando redes de computadores heterogêneos estão envolvidas."

Como a maior parte dos ambientes orientados a objetos, o CORBA suporta (BRANDO 1996):

- Encapsulamento: O encapsulamento permite com que informações pertencentes ao objeto possam estar ocultas, ou seja, os dados internos ao objeto só podem ser alterados por meio de operações apropriadas, isto permite que a alteração no objeto não cause alteração em outras partes do sistema.

- Abstração: É o processo de se criar tipos de dados abstatos, que incluem, além dos tradicionais texto e numero, figuras, datas, campos de tamanho variável, etc.

- Herança: É o mecanismo onde uma classe de objetos pode ser definida como um caso especial de uma classe mais geral, incluindo automaticamente nesta os métodos e variáveis definidos na classe geral.

- Polimorfismo: Trata-se do processo de se ocultar procedimentos por meio de uma *interface* comum.

- Ligação tardia de acionamentos de operações para chamadas de funções

- Reutilização de classes e objetos

A criação de novas classes e objetos em tempo de execução não é suportada diretamente, mas o CORBA inclui mecanismos suficientes para que um programador em ORB possa disponibilizar esta capacidade (BRANDO, 1996).

De acordo com Tibbits (1995), tanto CORBA quanto OLE2/COM utilizam os princípios da programação orientada a objetos e possuem um ORB para fazer chamadas a objetos distribuídos. OLE 2.0, no entanto, não aborda o problema de integrar aplicações em plataformas não-PC. A especificação COM estipula como a funcionalidade OLE pode ser implementada em plataformas não-PC onde CORBA está sendo usada. Além disso, a OMG emitiu uma chamada de propostas (RFP) para estabelecer a interoperabilidade COM-CORBA.

Maybee *et al.*(1995) faz o seguinte comentário sobre OLE2/COM:

> "Até que COM ofereça suporte completo para objetos distribuídos, é prematuro fazer comparações diretas. Recentemente, a Microsoft começou a atacar estas questões usando DCE para oferecer o suporte subjacente para nomeação e ordenamento (*marshaling*). É adequado dizer que isto significa que COM terá os mesmos méritos e deméritos de DCE."

Se o cliente e o servidor estão no mesmo processo, o compartilhamento de dados entre os dois é simples. Porém, quando o processo do servidor (elemento que recebe a solicitação de serviço) está separado do processo do cliente (quem solicita o serviço), o COM deve formatar e preparar os dados para compartilhá-los. A este processo dá-se o nome de *marshalling*.

O objetivo de se construir uma arquitetura para o desenvolvimento de aplicações orientadas a objetos é o de ocultar do programador todas as particularidades dos ambientes de implementação. Para que isso ocorra, eles devem utilizar ferramentas de analise e projeto em Orientação a Objetos (OO) que gerem subprodutos (especificações e códigos) de forma padronizada.

Entre os requisitos técnicos destas ferramentas, se encontram: habilidade de importar modelos legados, armazenamento em um repositório de objetos,

serem executadas independentemente do ambiente operacional, com a utilização de métodos de comunicação padronizada entre componentes.

O Modelo CCM do OMG foi uma especificação elaborada no sentido de prover este ambiente padronizado para o desenvolvimento de SI.

4.1. O MODELO DE COMPONENTES CORBA (CCM)

Posteriormente, no final de 1995, a OMG deu origem à especificação BOCA (*Business Object Common Archicteture*), que tem por objetivo a construção de uma infra-estrutura de camadas sobre o modelo CORBA, permitindo assim a utilização de um ambiente parametrizável para a reutilização de código, conforme ilustrado na figura 8 (OMG, 1995).

Além do modelo de componentes baseados em objetos de negócio, havia a necessidade do desenvolvimento de uma linguagem para análise e projeto das aplicações. Em 1997, a OMG e um grupo de fabricantes de *software* homologaram a IDL (*Interface Definition Language*) como padrão para o desenvolvimento de aplicações distribuídas. As definições IDL permitem uma transição para código com maior facilidade, pois se assemelham ao inglês estruturado (nas técnicas de análise de sistemas orientados a procedimentos) (OMG, 2002a).

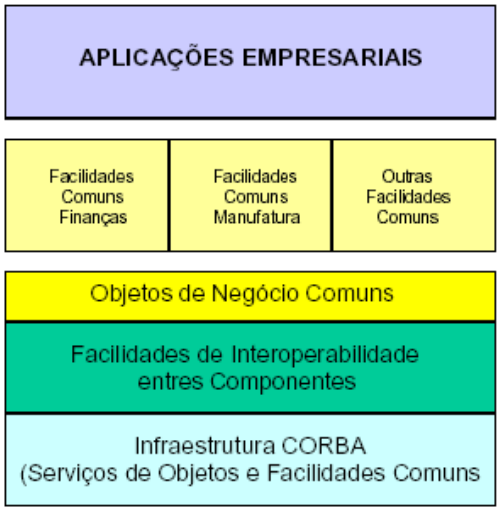

FIGURA 9: A ARQUITETURA BOCA (OMG, 1995)

Com base no BOCA, e com o objetivo de oferecer funcionalidade adicional para o desenvolvimento de SI em ambientes distribuídos, o OMG lançou o CCM (CORBA *Component Model*).

Segundo Barros (2003), o CCM complementa o modelo de objetos CORBA por meio da definição de serviços para a implementação, gerenciamento e configuração de componentes.

O modelo de componentes CORBA é uma extensão e especialização (meta-tipo) do modelo de objetos. Os tipos de componentes podem ser especificados em IDL e representados no repositório de *interfaces*. Um componente é definido por uma referência de componentes, que é representada por uma referência de objeto (OMG, 2002a).

Correspondentemente a isto, uma definição de componente é a especialização e extensão de uma definição de *interface*. Um tipo de componente é uma coleção especifica, que contém características similares e pode ser descrito em

definição de um componente IDL ou em uma estrutura correspondente no repositório de *interfaces*.

O modelo de Componentes CORBA (OMG, 2002; BARROS, 2003) é composto pelos seguintes elementos integrados:

- Modelo de componentes abstrato: extensões em IDL e o modelo de objetos.
- A arquitetura de implementação de componentes: linguagem de definição de implementação de componente (CIDL).
- A Arquitetura do Container de componentes (modelo de programação). O *container* é o *framework* onde o componente será executado.
- Integração com persistência, transações e eventos.
- Desenvolvimento e empacotamento de componentes.
- Interconexão com o EJB
- O modelo de metadados de componentes – um repositório de *interfaces* e extensões

As operações executadas sobre os componentes, dentro do CCM, são (BARROS, 2003; OMG, 2002):

- **Declaração dos Componentes**: O componente CORBA é especificado pela IDL (*Interface Data Language*). A linguagem IDL é compilada pelo compilador do fabricante, gerando os *stubs* e metadados. Estas extensões em IDL representam a visão de componentes do lado do cliente. Os subprodutos resultantes da compilação são *stubs* no cliente e descritores de metadados e de entradas no repositório de *interfaces*. A saída é, portanto, código compilado (bibliotecas compartilhadas, *bytecodes* em Java, DLL, etc), bem como a descrição XML (*Extended Markup Language*) destes componentes.

- **Implementação dos Componentes**: Tanto a independência de plataforma quanto de linguagem são requeridos para facilitar a geração automática de código para os componentes. Para isso, a OMG definiu o CIDL (*Component Implementation Definition Language*), que é uma evolução do IDL. Esta linguagem permite uma especificação independente, porém endereçando facilidades como integração com transações, persistência, eventos, etc. O componente é implementado pela complementação da lógica de negócio nos *skeletons* e *servants*. O código é escrito em uma linguagem de programação concreta, possivelmente utilizando-se um editor de texto básico e posterior compilação, ou uma facilidade em CORBA.

- **Ativação de Instâncias de Componentes**: Uma vez desenvolvidas e instaladas, as instâncias deste componente estão disponíveis para serem ativadas e usadas via mecanismos ORB padrão.

Existem dois níveis de componentes: os básicos e os ampliados. Embora eles sejam gerenciados pelo *home* de componentes, eles provêm essencialmente um mecanismo simples para gerar componentes de objetos CORBA sem adicionar uma grande quantidade de programação ao modelo. Os componentes ampliados provêm um conjunto muito mais rico de funcionalidades que o modelo CORBA existente (OMG, 2002b).

Os componentes possuem diferentes formas de conexão por meio de seus elementos primários, chamados de portas (figura 10) (BARROS, 2003). As portas são:

- **Facetas**: *Interfaces* providas pelos componentes que permitem a navegação entre eles, para interação com clientes.

- **Receptáculos**: Pontos de conexão que permitem a utilização de uma referência fornecida por outro agente externo.

- **Eventos base**: Pontos de conexão que emitem eventos de um tipo específico para um ou mais requisitantes interessados, ou para um canal de evento.
- **Eventos de *sink***: Pontos de conexão pelos quais eventos específicos podem ser empurrados.

Outras funcionalidades do modelo incluem:

- **Chaves primárias**: Valores expostos para clientes para auxiliar na identificação de componentes particulares.
- **Atributos e configuração de componentes**: Valores expostos pelos mecanismos de consulta e alteração, primariamente utilizados para configuração de componentes.
- ***Interfaces home* padrão**: Executam operações padrões de *factory* e *finder*.

Um componente pode prover múltiplas referências a objetos, chamadas facetas, que são capazes de ter *interfaces* CORBA distintas. O componente possui uma referência com a qual a *interface* se relaciona. Esta referência suporta a *interface* equivalente de componentes, que apresenta algumas facilidades para o cliente. A *interface* equivalente permite com que os clientes naveguem entre as facetas dos componentes e se conectem a portas de outros componentes (OMG, 2002b). É importante salientar que as implementações das *interfaces* das facetas são encapsuladas pelo componente, sendo consideradas parte dele. O relacionamento entre o componente e suas facetas é caracterizado por:

- Os componentes são instanciados e vivem em um *container* de componentes CORBA. Diferentes categorias de componentes e seus *containers* correspondentes são definidos.

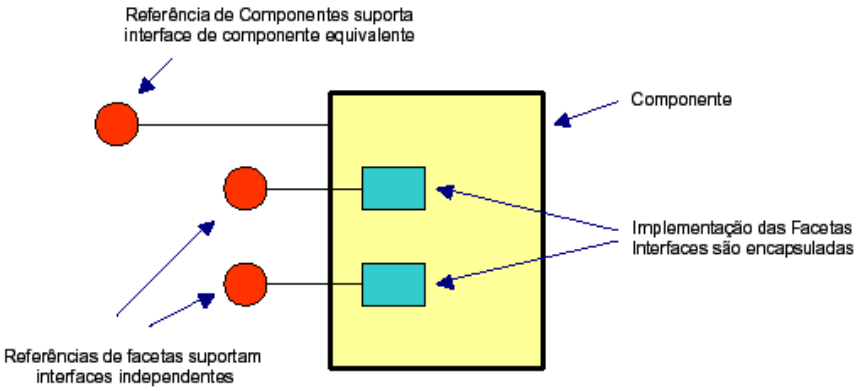

FIGURA 10: A ESTRUTURA DE UM COMPONENTE (OMG, 2002B)

- Um componente *home* é um novo meta-tipo que atua como um gerente para instâncias de um tipo de componente especificado. Um *home* pode ser imaginado como um gerente para as extensões de seus tipos de componentes (dentro do escopo do *container*). Os tipos de componentes são definidos isoladamente, independentemente do tipo *home*, uma definição *home*, por sua vez , deve especificar exatamente o tipo de componente que ela gerencia. Múltiplos tipos *home* podem gerenciar o mesmo tipo de componente, embora eles não possam gerenciar o mesmo conjunto de instâncias de componentes. Em tempo de execução, uma instância de componentes é gerenciada por um objeto *home* de um tipo particular. As operações no *home* são equivalentes a métodos estáticos ou de classes em linguagem orientada a objetos (OMG, 2002b).

- As chaves primárias podem estar associadas a outros componentes por um componente *home*. No modelo de componentes CORBA, uma chave primária é um valor de dados que é exposto para um componente de um cliente, que pode ser utilizado para identificar instâncias de componentes e obter referências a eles (dento do escopo de um *home*). Interfaces de componentes *home* provêm operações para gerenciar o ciclo de vida de componentes e associações entre chaves primárias e instâncias destes componentes. Embora múltiplos tipos *home* possam gerenciar o mesmo tipo de componente, uma instância é gerenciada por um objeto *home* simples, em tempo de execução.

Uma definição de componentes pode descrever a habilidade deste aceitar somente referências de objetos dos quais o componente pode invocar operações. Quando isto é feito, há uma conexão entre estes componentes. O ponto conceitual de conexão é o receptáculo.

O receptáculo é uma abstração que é concretamente manifestada em um componente como um conjunto de operações para que ele estabeleça e gerencie conexões (OMG, 2002b).

Estes receptáculos provêm o modelo para descrever as conexões entre os componentes e podem ser *simplex* (gerenciam uma referência, conectando-se a um objeto simples) e *multiplex* (gerenciam referências, conectando-se a múltiplos objetos).

Nas operações de conexão, o receptáculo armazena uma cópia da referência do objeto que é passada como parâmetro até que ele seja explicitamente desconectado.

Por sua vez, as operações de desconexão encerram o relacionamento entre os componentes e a referência ao objeto conectado.

No modo *multiplex* (vários objetos conectados), é responsabilidade do cliente associar *cookies* com referências a objeto para que no momento do

encerramento da conexão, eles possam ser reconhecidos pela *interface* do receptáculo.

Quando uma implementação de um tipo de componente é colocada no container, um objeto, chamado *home*, é criado. O *home* é como um gerente das instâncias do tipo de componente associado (OMG, 2002b).

O *home* executa operações de *factory* e *finder* necessárias para se criar e localizar instâncias de componentes para o tipo que ele gerencia.

Os programadores podem definir outras operações arbitrárias como parte da interface *home*, utilizadas para agregar facilidades aos clientes, tais como as operações de *query*.

O modelo de componentes CORBA possui funções que permitem com que os analistas de sistemas diferenciem *interfaces* para uso na configuração dos componentes das interfaces que são utilizadas por aplicações no momento da execução. Além disso, o CCM (*Corba Component Model*) suporta a divisão do ciclo de vida dos componentes em duas fases mutuamente exclusivas, a fase de configuração e a de operação. Durante a fase de configuração, o objeto vai normalmente invocar operações de mudança de atributos para a instância do componente que ele esta configurando. O termino desta fase é assinalado pela invocação da operação *configuration_complete* (OMG, 2002b).

O CCM define a noção de um objeto configurador que encapsula um atributo de configuração – a descrição de um conjunto de invocações no método de mudança do componente.

A arquitetura de implementação do componente (CIF – *Component Implementation Framework*) define o modelo de programação para a construção de componentes. O CCM inclui uma linguagem declarativa, o CIDL, para descrever implementações de componentes e componentes *home*, bem como seus estados abstratos. Os compiladores CIDL utilizam estas descrições para gerar *skeletons* de implementação que automatizam muitos comportamentos básicos dos componentes, tais como navegação, identidade,

consultas, ativação, gerenciamento de estado, etc. Os construtores de componentes ampliam estes *skeletons* para criar implementações completas (OMG, 2002b).

O CIDL é um superconjunto da linguagem PSDL (*Persistent State Definition Language*). Uma definição de implementação CIDL pode opcionalmente associar um tipo abstrato de memória a implementação do componente de tal forma que o tipo abstrato de memória defina a forma do estado interno, encapsulado pelo componente (OMG, 2002a).

Um outro componente da estrutura é o *container*, que é o ambiente de execução para a implementação de um componente CORBA no servidor. O modelo de programação do *container* define uma estrutura API (*Application Program Interface*). Este ambiente foi projetado para suportar um grande número de usuários simultaneamente. Além disso, esta plataforma permite alterações em componentes CORBA nativos. Basicamente, o modelo de programação do container é uma arquitetura de API desenhada para simplificar a tarefa de aplicações CORBA. Embora esta arquitetura não evite a necessidade de o programador utilizar funções nativas CORBA, ele é completo o suficiente para suportar uma grande variedade de aplicações (OMG, 2002).

O modelo é constituído de vários elementos: os tipos de API externas, que definem as *interfaces* disponíveis para um componente do cliente, e os tipos de API do container, que definem as estruturas de API usadas pelo programador de componentes.

Os tipos de API externas de um componente são contratos entre o programador de componentes e o cliente dos componentes. Existem duas formas de API externas: *interfaces home* e *interfaces* de aplicação. As *interfaces home* suportam operações que permitem ao cliente obter referências para uma das *interfaces* da aplicação que o componente implementa. Do ponto de vista do cliente, dois modelos de desenho são suportados – *factories* para criar novos objetos e *finders* para objetos existentes (OMG, 2002b).

Os tipos de API do *container* definem, por exemplo, o contrato entre um componente específico e o seu *container*. Esta especificação apresenta dois tipos básicos que definem as API comuns e um conjunto de tipos derivados que provêm funcionalidade adicional. O *container* de sessão API define uma estrutura para componentes que utilizam referências a objetos transientes. Já o *container* de entidade API define uma estrutura para componentes que usam referências a objetos persistentes.

Uma outra parte da arquitetura é o modelo de utilização CORBA, que especifica os modos de interação entre o *container*, o POA (*Persistent Object Architecture*) e serviços CORBA (OMG, 2002b). Existem três modelos de utilização CORBA como parte da especificação, que são:

- ***Stateless:*** Utiliza referências de objetos transientes em conjunção com *servant* POA, que podem suportar quaisquer *ObjectId*.
- **Conversacional:** Utiliza referências transientes em conjunção com *servant* POA dedicado a um *ObjectId* específico.
- **Durável:** Utiliza referências persistentes em conjunto com *servant* POA dedicado a um *ObjectId* específico.

O modelo de utilização do CORBA define as interações entre o *container* e o restante do CORBA (serviços, ORB e outros).

A categoria do componente é as combinações entre os tipos de API externos e os tipos de API do container com o modelo de utilização CORBA.

Os tipos de API externas são definidos pelo componente IDL e a especificação *home*. Estas *interfaces* são objetos CORBA armazenados no repositório de *interfaces* para utilização pelo cliente (OMG, 2002b).

O tipo de API de *container* é uma arquitetura feita de *interfaces* internas e interfaces de *callbacks* usadas pelo programador de componentes. Eles são definidos utilizando-se as declarações IDL e especificando-se as *interfaces home* (figura 11).

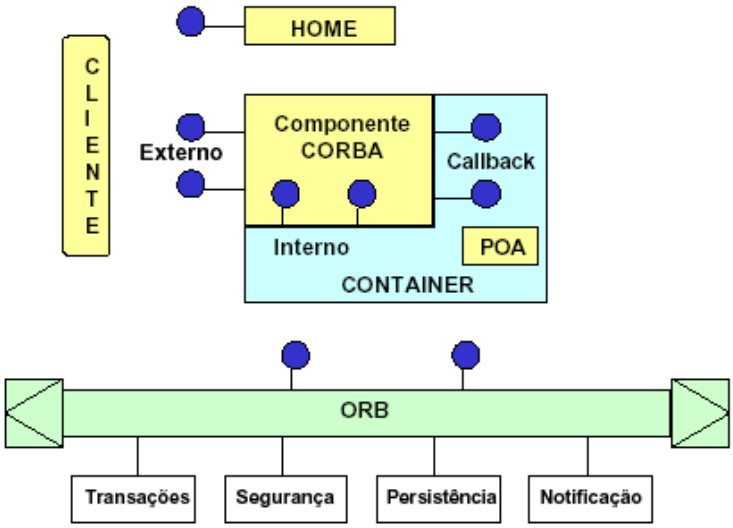

FIGURA 11: A ESTRUTURA DO MODELO CCM (OMG, 2002B)

Basicamente, o *container* de componentes é a estrutura do servidor, construído sobre o ORB e um conjunto de serviços CORBA, que estabelece o ambiente de execução para um componente CORBA.

Os componentes CORBA podem suportar transações *self-managed* (SMT) ou transações gerenciadas pelo container (CMT - *Container-managed Transactions*). Um componente que usa transações *self-managed* não possui políticas de transação definidas pelo seu descritor e é responsável pela demarcação da transação pela utilização da interface de transação ou do serviço de transações do CORBA. Por outro lado, quando uma transação gerenciada pelo container é selecionada, políticas adicionais de transação (ex.:

seguranças, persistência, etc) podem ser definidas no seu descritor (*deployment descriptor*).

O *container* utiliza esta descrição para efetuar as chamadas aos serviços de transação CORBA. A política de transação definida no descritor do componente é aplicada pelo *container* antes de invocar a operação. Diferentes declarações de política de transação podem ser feitas para operações em quaisquer portas dos componentes, bem como para a interface *home* do componente (OMG, 2002b).

A política de segurança é aplicada consistentemente para todas as categorias dos componentes. O *container* se baseia na segurança do CORBA para aceitar as declarações de segurança do descritor e verificar as credenciais ativas para invocar operações. A política de segurança permanece ativa até que uma invocação subseqüente ocorra (OMG, 2002b).

As permissões de acesso também são definidas pelo descritor. As permissões deve ser coerentes com os mecanismos de segurança CORBA instalados, pois este será utilizado no momento da invocação. As permissões de acesso podem ser definidas para quaisquer portas dos componentes, bem como para quaisquer interfaces *home* deste componente.

O servidor de componentes (figura 11) é o processo que estabelece um número arbitrário de *containers* de componentes. Um *container* suporta um outro associado do tipo API (descreve interações com um componente) e gerencia uma categoria especifica de componente. Ele possui um modelo CORBA associado, o qual descreve suas interações com o POA, ORB e um conjunto de gerenciamento de referências e de *servants*.

Um *container* (escrito em Java) pode ser um *container* EJB pelo suporte de um dos tipos de API para *containers* EJB.

Um *container* é criado como resultado da preparação (*deployment*) do componente.

Um gerenciador de *container*, por meio de uma especificação, determina o conjunto apropriado de políticas POA, o tipo de *container* de API, e o conjunto de serviços CORBA para ser usado neste *container*, e então age como um *factory* para criar o *container.* Os gerenciadores de *containers* são eles próprios criados como resultado do processo de instalação e preparação (OMG, 2002b).

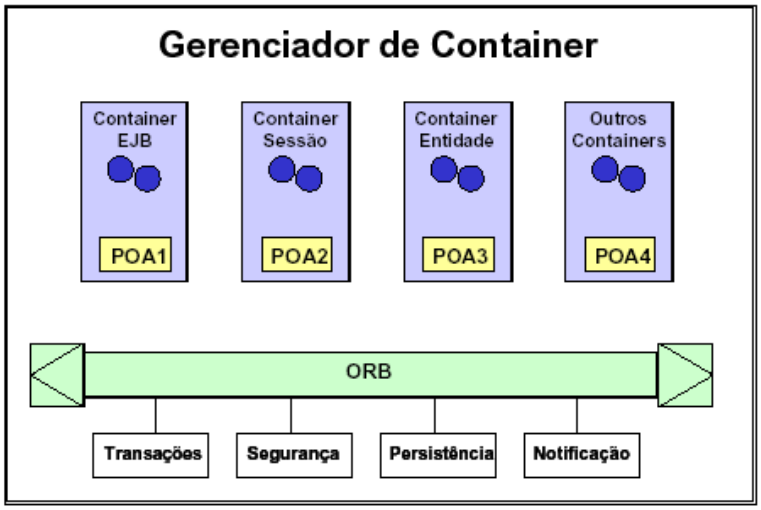

FIGURA 12: A ESTRUTURA DO GERENCIADOR DE CONTAINERS (OMG, 2002B)

O CCM define dois níveis de *containers* de componentes: básico e ampliado. A distinção não afeta o gerenciamento de referência de objeto e a disponibilidade dos serviços CORBA.

Os *containers* CORBA básicos usam os serviços de segurança, transação, e nomeação. Já os *containers* ampliados usam também o serviço de persistência de estado e o de notificação. A criação de *containers* envolve a obtenção da referência inicial destes serviços, bem como os dados de configuração de um *container* específico (ex: canal para emitir e consumir eventos e conexões a banco de dados para persistência).

A arquitetura do *container* possui sete diferentes categorias: quatro que correspondem às quatro categorias de componentes (sessão, entidade, processo, serviço), dois para os tipos de API para *containers* EJB, e um container vazio para apoiar estruturas definidas pelo usuário (OMG, 2002b):

- A categoria de serviços gerencia o acesso a componentes *stateless*.
- A categoria de sessão gerencia a sessão para componentes CORBA *stateful* como se este fosse transiente.
- A categoria de processo gerencia componentes em processos *stateful* que encapsulam mecanismos de acesso a dados no servidor.
- A categoria de entidade gerencia componentes de entidades *stateful* que compartilham responsabilidade de acesso a dados entre o cliente e o servidor.
- A categoria *EJBSession* gerencia sessões *beans* EJB.
- A categoria *EJBEntity* gerencia entidades *beans* EJB.
- A categoria vazia não provê gerenciamento automático, porém torna as *interfaces* padrão CORBA 3.0 disponíveis para a implementação do componente.

A persistência do componente é suportada pelos *containers* de processo, entidade, e *EJBEntity*.

Duas formas de persistência são suportadas para cada categoria:

- ***Container-managed***: Neste tipo de persistência, o *container-provider* coopera com o *persistence-provider*. A declaração do estado abstrato é associada com um componente ou suas facetas, utilizando-se definições PSDL. O código pode ser automaticamente gerado para carregar e armazenar o estado em tempos apropriados, bem como para implementar os requisitos das operações de *factory* e de *finder*.

- ***Self-managed***: Neste tipo, o *component-developer* coopera com o *persistence-provider* diretamente. Para os *containers* de processo e de entidade, o *self-managed* é assumido se não há definições PSDL.

Com relação ao empacotamento de componentes, este modelo utiliza uma especialização de um esquema de empacotamento de *software* para descrever e empacotar componentes. Ele utiliza um vocabulário XML para descrever pacotes de dados e suas dependências. Componentes são empacotados em arquivos compactados, cada qual com seus descritores XML que descrevem seus conteúdos (OMG, 2002).

Um pacote de componentes é o veículo para se desenvolver a implementação de um simples componente. Um pacote de montagem de componentes é o veiculo para desenvolver um conjunto de implementações de componentes inter-relacionados. Ele é um modelo ou padrão para se instanciar um conjunto de componentes e *homes*, e relacioná-los entre si.

Um pacote de componentes embute uma ou mais implementações de um componente. Cada implementação define o mesmo componente, porém com características diferentes, tais como linguagem de implementação, sistema operacional, ou mesmo comportamento em tempo de execução. Em geral, ele consiste de um conjunto de arquivos e um ou mais descritores que descrevem suas características do pacote e ponteiros para seus vários arquivos. A coleção de arquivos e os descritores podem ser agrupados em um arquivo ou mantidos separadamente (OMG, 2002b).

Um componente ou um pacote de montagem de componentes pode ser instalado em um computador ou pode ser agrupado com outros componentes para ser montado posteriormente.

Uma montagem de componente é um grupo de componentes e *homes* interconectados, representados por um pacote montado.

A preparação de um componente ou montador de componentes é carregada pela aplicação de disponibilização (*deployment*), em conjunto como um grupo

de objetos “assistentes”, dentre os quais inclui-se um objeto de instalação, um *factory* de montagem e um objeto representando a própria montagem. Adicionalmente, há a necessidade de algum tipo de repositório de componente (lógico) que o processo de instalação consulta, entre eles:

- ***Factory* de Montagem**: localizado no servidor onde os objetos montados são criados.
- **Montagem:** representa uma instanciação de montagem e coordena a criação e destruição de componentes montados, bem como a criação de componentes e a garantia de que eles estão conectados.
- **Instalação:** instala implementações de componentes no servidor de destino.

A partir das características técnicas do modelo CCM da OMG, conclui-se que pode ser utilizado como infra-estrutura para a construção de um SI para apoiar as operações de gestão de demanda e de inventários no SCM.

As características funcionais da proposta de um SI serão descritas no próximo capítulo.

5. PROPOSTA DE UM SI PARA APOIO AO SCM

Para se definir a proposta de SI para a gestão de demanda e de inventários no SCM é necessário conceituar corretamente as características do SI a ser desenvolvido, bem como o processo de desenvolvimento de *software* a ser utilizado, pois influenciam na escolha do tipo de documento a ser gerado, bem como a forma de implementação.

5.1. O PROCESSO DE DESENVOLVIMENTO DE SI

O termo *software* pode ser conceituado, segundo Dic (1996), como:

> "Conjunto de todos os recursos humanos, lógicos e mesmo de instalação e de organização, com os quais se explora uma máquina, equipamento ou sistema."
>
> "Qualquer programa ou grupo de programas que instrui o *hardware* sobre a maneira como ele deve executar uma tarefa, inclusive sistemas operacionais, processadores de texto e programas de aplicação."
>
> "Conjunto de programas para uma determinada espécie de computador, incluindo documentação tal como manuais, diagramas e instruções de operação."

Pressman (1995), conceitua *software* como:

> "(1) Instruções (programas de computador) que, quando executadas, produzem a função e o desempenho desejados; (2) Estruturas de dados que possibilitam que os programas manipulem adequadamente a informação; (3) Documentos que descrevem a operação e o uso dos programas."

Ainda, segundo a norma NBR ISO 9000-3 (1993), a seguinte definição de *software* é apresentada:

> "Criação intelectual compreendendo os programas, procedimentos, regras e qualquer documentação correlata à operação de um sistema de processamento de dados."

O processo de desenvolvimento de *software* possui algumas características particulares (PRESMANN, 1995):

- O produto resultante do desenvolvimento de *software* é algo lógico, não palpável;
- O *software* é em grande parte feito sob medida, não é algo que seja pré definido ou que seja executado através de atividades padronizadas;
- Os custos de *software* estão concentrados no trabalho de engenharia e não no processo de manufatura (ETO - *engineered to order*);
- O produto de *software* se deteriora em função de sua manutenção e do distanciamento em relação aos processos de negócio (entropia);

Alguns outros fatores são abordados por Basili *et al.*(1994) e Wangenheim *et al.*(1998):

- Existem similaridades e diferenças causadas por determinados fatores de projeto que tornam os modelos definidos não aplicáveis a todos os projetos;
- Existe uma relação entre o processo de desenvolvimento e manutenção e o produto de *software*, ou seja, a qualidade do produto depende em grande parte do processo selecionado para o seu desenvolvimento;
- É necessário que se definam métricas para que o processo de desenvolvimento de *software* possa ser visível e comparável aos objetivos estabelecidos.

- O *software* segue um paradigma experimental, sendo que a realimentação e ajuste do processo dependem em grande parte do aprendizado que é realimentado durante o seu desenvolvimento;

- Novas tecnologias devem ser incorporadas ao nos processos e produtos de *software* para a permanente evolução;

Com relação à aplicação do software, Ferreira (2002) salienta que ele pode ser utilizado em qualquer situação em que um conjunto previamente especificado de passos procedimentais (algorítmo) é definido. Dessa forma, as áreas de aplicação apresentam larga amplitude. Frick (1996), no entanto, classifica a aplicabilidade de *software* com relação a sua entrada no mercado, podendo ser produto do segmento horizontal ou vertical.

Dentro da categoria do segmento horizontal se enquadram os produtos vendidos sob a forma de pacotes, tais como sistemas operacionais, banco de dados, planilhas eletrônicas, etc. Neste segmento, de acordo com Porter (1999), são produtos cuja estratégia é pautada por economia de escala (baixo custo).

Já os produtos do segmento vertical, segundo Frick (1996), incorporam o conhecimento da informática e de um ou mais campos adicionais. Neste sentido, o *software* aplicado a este mercado é desenvolvido para uma área especifica, como por exemplo, a gestão de operações de uma empresa. Estes produtos são comercializados sob a estratégia de diferenciação (PORTER, 1999).

Para efeito desta dissertação, o desenvolvimento da solução proposta se baseia em uma aplicação de *software* orientada ao segmento vertical.

Desenvolvimento de sistemas pode ser compreendido como a atividade de criar novos sistemas empresariais, ou modificar os existentes. Neste contexto estão incluídos não somente os sistemas de informação, mas também todos os que de alguma forma são utilizados dentro do contexto empresarial.

Para efeito desta dissertação os conceitos são aplicados exclusivamente ao desenvolvimento de sistemas de informação (SI).

Embora existam diferentes metodologias para seu desenvolvimento, um projeto de SI passa pelas seguintes etapas: concepção e viabilidade, análise, projeto, implementação, manutenção e revisão (BUZATO & RUBIRA, 1998).

O ciclo de vida de um SI termina quando é iniciado um novo projeto para substituí-lo devido a um fator ou à combinação de vários, tais como: obsolescência, mudanças tecnológicas que inviabilizem o SI, ou a necessidade de uma revisão tão profunda que o projeto de um novo SI se torna mais viável (MASTELARI, 2004).

Existem diferentes processos de desenvolvimento de *software*, sendo que cada um traz consigo um conjunto de atividade, métodos e recursos específicos (SWEBOK, 2001).

O desenvolvimento de *software* pode ser representado através de modelos de ciclo de vida, que são representações do processo de desenvolvimento que contém informações sobre sua execução (FERREIRA, 2002). As atividades para o desenvolvimento de *software* são organizadas sob a forma de etapas, também denominadas fases, que associam os métodos, ferramentas e procedimentos necessários à execução destas atividades (PRESSMAN, 1995).

Existem vários modelos de desenvolvimento, sendo os mais comuns: modelo clássico, modelo de prototipação, desenvolvimento incremental/iterativo, modelo espiral, modelo de reutilização, e síntese de *software* automatizado (SWEBOK, 2001).

5.1.1. O MODELO CLÁSSICO DE DESENVOLVIMENTO DE *SOFTWARE*

O modelo clássico é representado pelas etapas de desenvolvimento descritas na figura 13.

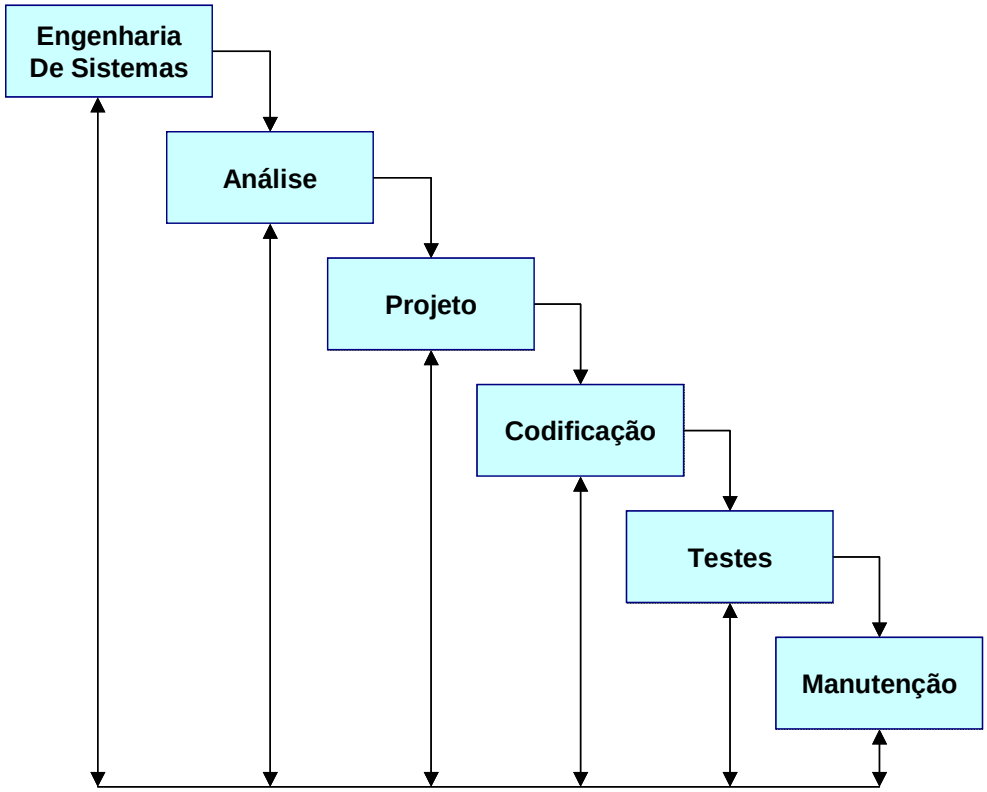

FIGURA 13: CICLO DE DESENVOLVIMENTO CLÁSSICO (PRESSMAN, 1995)

O ciclo de desenvolvimento clássico compreende, de acordo com Pressman (1995), as seguintes atividades:

- Engenharia de Sistemas: Identificação dos requisitos necessários no SI, bem como o ambiente computacional para o qual ele será desenvolvido.
- Análise: compreende o detalhamento funcional da especificação do software a ser construído, englobando funcionalidade, desempenho e *interfaces* desejados.
- Projeto: é composto de quatro principais grupos de atividades a serem especificadas: estrutura de dados, arquitetura de software, detalhes de procedimentos e a qualidade desejada.

- Codificação: É a tradução da especificação em uma linguagem que possa ser interpretada pela máquina, de forma com que o *software* possa ser construído.

- Testes: Processo de depuração dos possíveis erros e verificação dos resultados obtidos em função do esperado.

- Manutenção: Processo de ajuste do *software* à mudanças ambientais. Na manutenção são reaplicadas as etapas precedentes do ciclo de vida.

5.1.2. PROTOTIPAÇÃO

Um protótipo de um SI é algo que simula as principais funções e interações que o *software* a ser desenvolvido vai contemplar. A proposta do protótipo é tornar real o modelo especificado pelo cliente antes de se começar a codificação. Somente após o cliente testar sua consistência e usabilidade é que o sistema será construído. (FERREIRA, 2002)

O objetivo da prototipação é o de se antecipar potenciais problemas de desenho funcional do SI a ser construído com o mínimo de esforço, objetivando desta forma a qualidade e a minimização do custo de desenvolvimento de SI pela diminuição do número de erros de concepção.

5.1.3. MODELO DE CICLO DE VIDA ESPIRAL

O modelo espiral tem por objetivo possibilitar constante análise de riscos e iteração com os clientes. Ele foi proposto visando uma combinação das melhores características do modelo clássico quanto da prototipação. O modelo possui as seguintes atividades (PRESSMAN, 1995):

- Planejamento: elaboração dos objetivos, das alternativas e limitações.

- Análise de Riscos: Análise das alternativas, identificação e minimização dos riscos.
- Engenharia: desenvolvimento do produto, com conseqüentes iterações evolutivas com relação às alternativas, restrições e riscos encontrados no protótipo.
- Avaliação do Cliente: Avaliação do produto concebido.

5.1.4. MODELO "BASEADO EM TRANSFORMAÇÕES"

Outra visão do moderno ciclo de vida em cascata é apresentada por Yourdon (1995). Na verdade se trata de uma automatização do ciclo tradicional, apoiada pela utilização de ferramentas automatizadas vinculadas a geradores de código. Estas ferramentas (*softwares*), também classificadas como técnicas de quarta geração – 4GL (PRESSMAN, 1995) procuram tornar a codificação mais simples e próxima da linguagem natural, diminuindo desta forma o tempo de desenvolvimento dos SI.

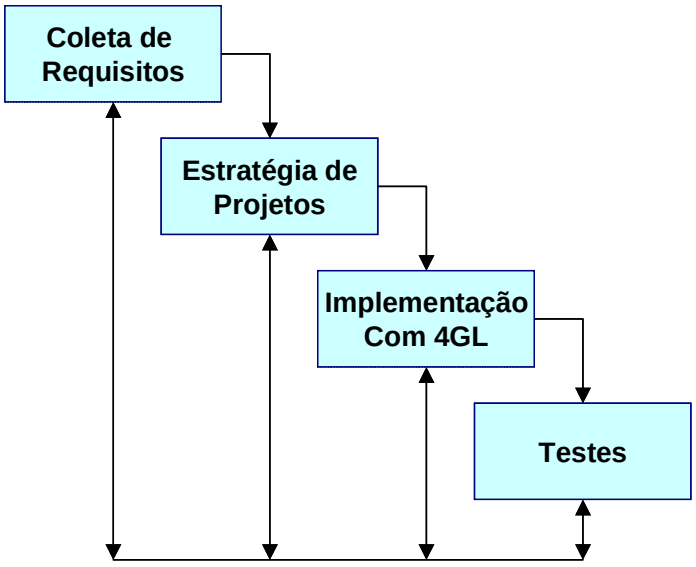

FIGURA 14: TÉCNICAS DE QUARTA GERAÇÃO (PRESSMAN, 1995)

Não existe um consenso sobre qual seja a melhor alternativa para o desenvolvimento de um *software* para a elaboração de um SI pois vários fatores devem ser considerados, tais como o ambiente onde o SI irá operar, as funcionalidades que este contemplará, bem como as características de interação com o usuário final e outros aspectos contingênciais, inclusive políticos e de motivacionais. Neste sentido, o software se enquadra no tipo de manufatura ETO (*Engineered to Order*), ou seja, cada desenvolvimento de SI pode ser considerado um projeto pois engloba etapas bem definidas, com objetivos e resultados claros, englobando desde a identificação das necessidades até sua implementação.

Como um projeto é “um empreendimento temporário com o objetivo de criar um produto ou serviço único” (PMI, 2000), sua execução deve ser eficaz. Neste sentido, o PMI (2000) apresenta nove áreas de conhecimento e gestão, cujas atividades devem ser cuidadosamente monitoradas: Integração, Escopo, Tempo, Custo, Qualidade, Recursos Humanos, Comunicações.

Um projeto de desenvolvimento de SI deve contemplar estas áreas, por mais diversidade que haja em as diferentes técnicas de projeto de *software*.

Em seguida serão apresentados os aspectos funcionais da proposta do SI.

5.2. ESPECIFICAÇÃO FUNCIONAL DA PROPOSTA DO SI

Para efeito desta dissertação o modelo de desenvolvimento a ser utilizado é o clássico em função da permitir uma divisão clara entre a proposta e o desenvolvimento, já que esta dissertação engloba a proposta da especificação funcional, sem considerar aspectos relativos à implementação, fato que pode modificar a escolha do modelo de processo de desenvolvimento.

A proposta do SI para apoio à gestão de demanda e de inventários ao longo da cadeia de suprimentos deve ser aderente ao método de gestão de inventários

de cada empresa participante da SC e permitir, ao mesmo tempo, informação útil para o processo de SCM.

Neste sentido, independentemente do método de planejamento das necessidades de inventários (demanda dependente) ser baseada em LRP, BSC ou MRP (KEHOE & BOUGHTON, 2001), o SI deve contemplar as particularidades de cada empresa.

Para tanto, a arquitetura de infra-estrutura proposta a ser utilizada como base para o SI é o modelo CCM, tanto pela sua portabilidade quanto pelo suporte e serviços de interoperabilidade (OMG, 2002b). Dessa forma, aspectos particulares de ambiente computacional ficam encapsulados dentro da definição dos componentes de *software*, criando independência entre os processos de negócio e o SI que "espelha" e suporta estes processos.

A comunicação com cada sistema legado será baseada em *interfaces*, conforme apresentado na figura 15 (OMG, 2002).

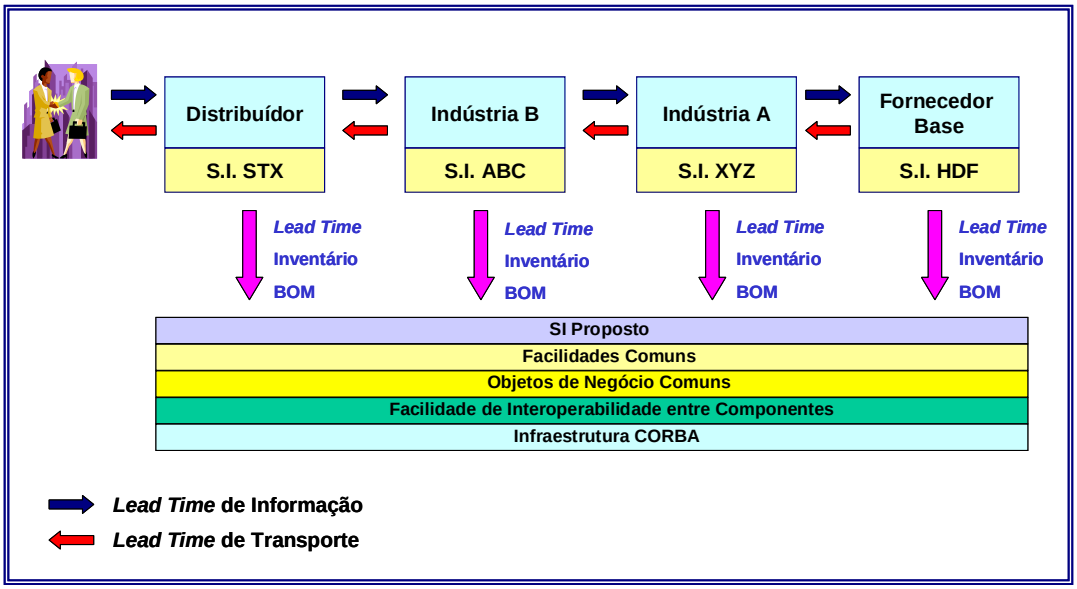

FIGURA 15: ARQUITETURA PROPOSTA PARA O SI (ADAPTADO OMG, 2002)

O sistema será formado por objetos distribuídos, ou seja, a comunicação entre os diversos participantes se dá por meio de conexões padronizadas baseadas em CORBA.

Dentro das camadas descritas no modelo, a de infra-estrutura CORBA é responsável por prover os serviços, tais como descreve Barros (2003):

- Serviço de Nome: permite com que objetos sejam localizados pelos nomes ou pela associação entre nomes e referencias de objetos;
- Serviço de *Trader*: responsável por descrever as funcionalidades e os tipos de serviço oferecidos por este objeto, agindo como tipo de "páginas amarelas";
- Serviço de Evento: fornece um conjunto de serviços para coletar e distribuir eventos entre objetos;

A camada de facilidade de interoperabilidade entre componentes é baseada em ORB, ou seja, fornece o adaptador de objetos (O A – *Object Adapter*) , que é responsável pela implementação do objeto, e pode ser básica (BOA – *Basic Object Adapter*) ou POA (*Portable Object Adapter*).

A camada que contém os objetos de negócio comuns é responsável por armazenar os principais objetos de negócio, ou seja, o correspondente às entidades no modelo ER. São elementos presentes nos SI que estão inseridos em um contexto. Como exemplo pode ser citado o objeto *Produtos*, que corresponde aos atributos, métodos que descrevem os produtos que são comercializados entre as diferentes empresas. Estes objetos de negócio comuns podem ser organizados de acordo com o contexto, como por exemplo: manufatura, finanças, comercial, etc.

A camada que contém as facilidades comuns é composta de módulos (componentes de *software*) que podem ser implementados dentro de qualquer aplicação, tais como impressão de relatórios, seleção de tipo de página, seleção de opção em uma lista de alternativas, etc.

E por fim, a camada onde será desenvolvida a aplicação, que é a de mais alto nível, sendo mais próxima do ambiente de processos do SI que do ambiente operacional da plataforma.

A linguagem adotada segue o conceito de orientação a objetos, tal como a especificação utilizada na linguagem Java e Enterprise JavaBeans. Isto devido ao suporte que ela oferece às funções de comunicação com o ambiente, bem como pela facilidade de reutilização de código (construção de módulos funcionais) e pelo fato de ser interpretada, não exigindo compilação prévia, fato que possibilita maior flexibilidade.

A arquitetura de desenvolvimento, diferentemente de uma classe de objetos que possuem apenas desenho e código, deve possuir os conceitos de classes que herdam propriedades, de objetos que encapsulam comportamentos, de interações entre objetos que visam reutilização. Existe uma estrutura particular para cada domínio que funciona como um repositório, sendo dinâmico, constituído de processos, arquitetura, desenho e implementação. Estas estruturas (*frameworks*) são como aplicações miniatura que incluem fluxos de aplicação baseados em uma arquitetura predeterminados. O modelo é composto de três camadas de código reutilizável para uso pelos programadores de aplicação (OMG, 2002b).

A arquitetura é composta de camadas, sendo que a fundamental fornece a infra-estrutura e serviços que são requeridos para a construção de Objetos Comuns de negócio e Objetos base de negócio. A camada fundamental oferece ainda uma interface e estrutura de programação para apoio aos programadores. Muitas das definições de serviços são baseados na OMG (*Object Management Group*) como, por exemplo, as relativas a serviços de transações, coleções, comunicação entre objetos distribuídos e gerenciamento de persistência. Porém, a arquitetura não provê um ORB (*Object Request Broker*) compatível com CORBA. O que existe é uma combinação de funções da OMG com implementações em Java. Também são simplificadas muitas definições da OMG para facilitar o processo (OMG, 2002).

A camada Fundamental também possui extensões que são necessárias para a arquitetura Java como, por exemplo, RMI (*Remote Method Invocation*) como base para a infra-estrutura de comunicação.

A camada de processos de negócio base é onde são implementadas as funções do SI proposto.

5.2.1. FUNCIONALIDADE DO SI PROPOSTO (SCRP)

Embora o SI proposto apresente funções executadas de forma assíncrona (as *interfaces* com os sistemas legados, por exemplo), ele é predominantemente *on-line*, ou seja, com atualizações automáticas. A camada de apresentação ao usuário será baseada em planilhas excel, tanto para facilitar a integração com ferramentas de *back-office* (suporte às atividades de automação de escritório, como o MS-Office, por exemplo), quanto para aproveitar o potencial da funcionalidade do excel como ferramenta de cálculo e para facilitar a utilização do SI por parte do usuário (o pressuposto é que a funcionalidade do excel é de domínio público).

Um outro ponto a ser considerado é de que as telas do sistema contenham apenas os dados relativos à funcionalidade principal da operação, sem considerar, para efeito desta dissertação, os serviços padronizados oferecidos pelos SI em ambiente gráfico, tais como serviço de impressão, telas de ajuda (*help* de tela e de campo), mecanismos de busca e outros.

As principais funções a serem contempladas pela aplicação são apresentadas na figura 16.

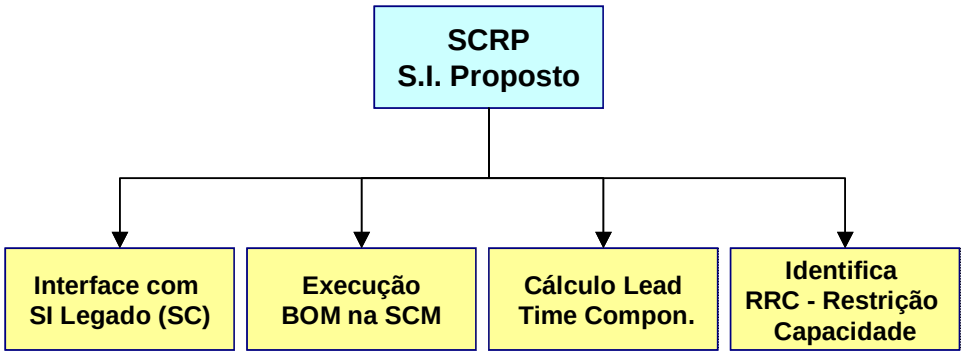

FIGURA 16: ESTRUTURA FUNCIONAL DO SCRP

Para efeito desta dissertação, o SI proposto receberá o nome de SCRP (*Supply Chain Resource Planning*).

O SCRP deverá ser processado a partir de um ícone instalado diretamente na tela do usuário, conforme apresentado na figura 17.

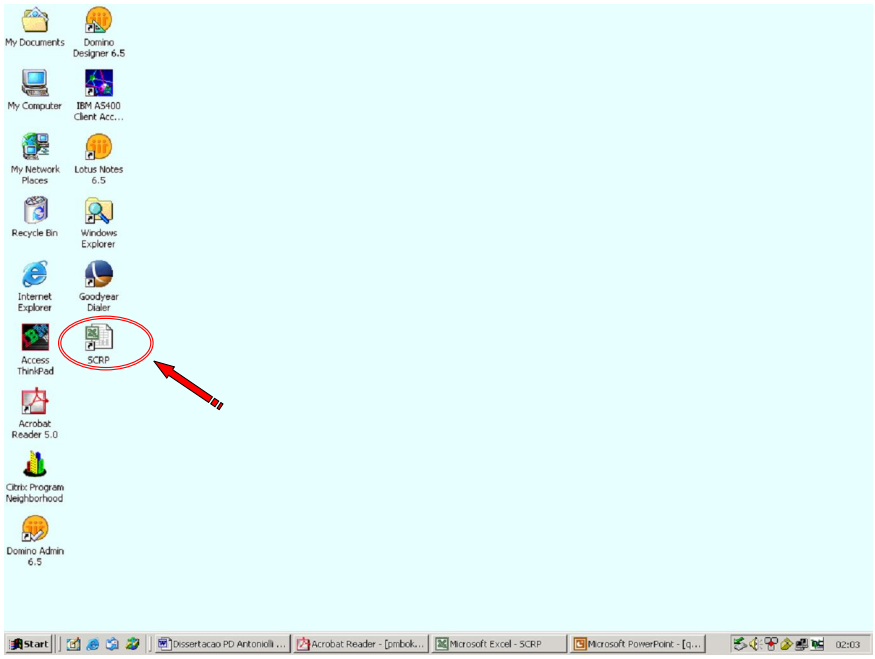

FIGURA 17: ÍCONE PARA INÍCIO DO PROCESSAMENTO DO SCRP

Com o acionamento do ícone, o sistema enviará a tela inicial, a partir da qual poderão ser processadas as diversas funções, conforme figura 18.

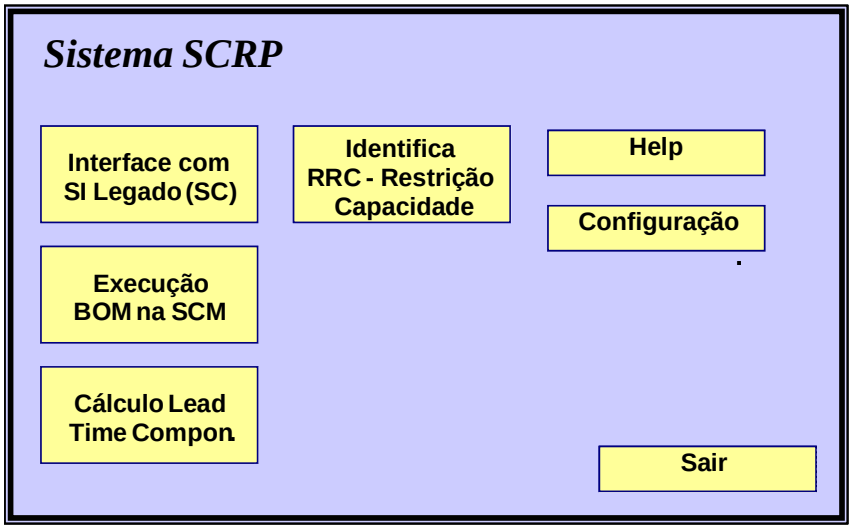

FIGURA 18: TELA PRINCIPAL DO SCRP

Função 1*: Interface* do SCRP com SI Legado:

A *interface* compreende a comunicação assíncrona entre o SI proposto e o sistema legado da empresa participante da SC. Neste sentido não é necessário que a plataforma do cliente seja padronizada (tanto em termos de SI quanto em infra-estrutura), basta que exista um SI, de onde os dados poderão ser extraídos para que o SCRP possa calcular *lead times* e identificar recursos críticos dentro do SCM.

Um aspecto a ser considerado dentro do SCRP é com relação ao compartilhamento de dados pois em toda SC existe um elo que "governa" o fluxo de operações e de demanda. Neste sentido deverão ser criados mecanismos adicionais de segurança para permitir com que o nível de acesso aos dados seja configurado de acordo com a posição da empresa participante

na SC. Esta prática pode resultar em dificuldades para que a visibilidade da demanda seja completa, mas é necessária pois os dados das empresas participantes são protegidos por políticas específicas destas empresas. Dessa forma, o SCRP pressupõe que exista uma entidade (que pode ser externa à SC), responsável pelo manuseio de acesso e custodia dos dados existentes no SCRP.

Caso a empresa participante do SI não possua um SI formal para suporte à gestão de manufatura e operações o SCRP poderá obter os dados diretamente a partir de uma planilha excel.

Esta função do sistema é composta de diversas atividades, que podem ser classificadas em: 1) Dados relativos à preparação e configuração da SC, com seus respectivos componentes; 2) Dados relativos ao processamento da *interface* entre o SCRP e o sistema de origem dos dados, podendo esta ser executada para um ou para todos os elos da SC.

Na função de configuração, deverão ser informados os elementos referentes ao endereço físico da origem de dados (endereço IP, por exemplo), os objetos principais dos sistemas legados, associados aos principais componentes de negócio utilizados no SCRP, bem como outros aspectos técnicos relevantes para o funcionamento da aplicação (como autorizações de segurança de acesso e mecanismos de proteção contra acessos não autorizados).

As telas de configuração são compostas de:

- Dados de configuração do elo da SC (a empresa e sua posição dentro da estrutura), bem como *lead times* genéricos (ex: transporte) associados a todos os produtos fornecidos por esta empresa, que serão incorporados aos demais existentes (figura 19).

- Dados de configuração dos elementos relacionados aos objetos de negócio, onde o SCRP irá buscar os dados (figura 20).

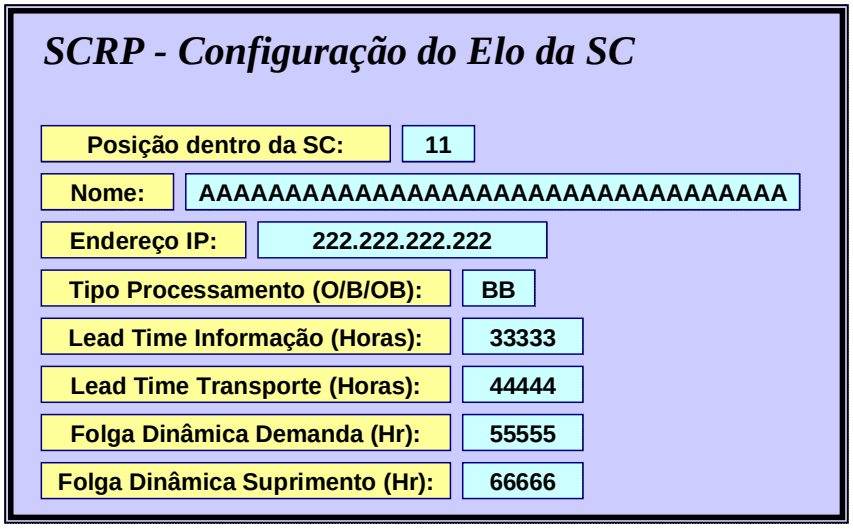

FIGURA 19: TELA DE CONFIGURAÇÃO DO ELO DA SC NO SCRP

Nesta tela os seguintes campos são observados:

- Posição dentro da SC (11): Indica a posição desta empresa na composição da SC, para posterior utilização no cálculo do *lead time*.

- Nome da Empresa (AA): Nome da empresa, que será utilizado nas telas e relatórios do SCRP.

- Endereço IP (22): Neste caso poderá ser utilizado o nome do servidor (se este estiver dentro do mesmo contexto lógico (rede), ou endereço IP (*Internet Protocol*) fixo do servidor da empresa participante do SC, onde serão obtidos os dados para operação do sistema). Para resolver este processo poder-se-ia utilizar o serviço de comunicação do ORB, dentro do modelo CCM, no SCRP proposto.

- Tipo de Processamento (BB): B = *Batch*, em lotes. Quando esta opção é selecionada uma segunda tela é mostrada para indicar se este processamento vai ser agendado ou imediato. Se for agendado, campos

adicionais serão apresentados, para que seja informado freqüência e data (para processamentos freqüentes), ou então data e hora (para processamento eventual). O = *Online*. Se a opção selecionada for o processamento em linha, no momento da solicitação, então o SCRP irá consultar o conteúdo deste campo para executar ou não a interface. Caso o conteúdo seja igual a OB = *Online* e *Batch* indica que o SCRP poderá processar a *interface* em ambos os modos. Caso seja selecionada esta opção, uma segunda tela será apresentada, conforme a opção *Online*.

- *Lead Time* de Informação (33): Este é o tempo (em horas) que o SCRP considera adicionalmente a cada item, desde o momento que a informação de demanda sai o elo anterior da SC (*upstream*) e chega para este elo, para considerar o *lead time* de informação. Por exemplo, se o *lead time* de informação do item Z é 36 horas e existe, neste campo, o conteúdo de 10 horas, o SCRP considera como *lead time* total de informação para o item Z, 46 horas.

- *Lead Time* de Transporte (44): Este é o tempo (em horas) que o SCRP considera adicionalmente a cada item, desde o momento que ele é produzido no elo em questão, até chegar ao elo anterior da SC (*upstream*). Por exemplo, se o *lead time* de transporte do item Y é 22 horas e existe, neste campo, o conteúdo de 5 horas, o SCRP considera como *lead time* total de transporte para o item Y, 27 horas.

- Folga Dinâmica Demanda (55): Este é o tempo em horas, que o SCRP considera adicionalmente ao *lead time* de informação, desde o momento que esta deixa o elo anterior da SC (*upstream*) e chega a empresa em questão. A aplicabilidade deste campo se refere a situações específicas (por exemplo, sazonalidade) onde deve ser considerado um tempo adicional de folga.

- Folga Dinâmica de Suprimento (66): Este é o tempo em horas, que o SCRP considera adicionalmente ao *lead time* de transporte, desde o

momento que o item produzido sai da empresa e chega ao elo anterior da SC. A aplicabilidade deste campo se refere à situações específicas (por exemplo, sazonalidade) onde deve ser considerado um tempo adicional de folga.

FIGURA 20: TELA DE CONFIGURAÇÃO DA ORIGEM DOS DADOS

Esta tela tem como objetivo indicar para o SCRP quais são os objetos a serem considerados para que o SI possa obter os dados relevantes para compor a demanda, sua origem (cliente) e fazer a conciliação com inventário, *lead times* (ATP – *Available to Promisse*) e necessidades de compras e respectivos fornecedores assinalados.

Os principais elementos são:

- Item (CC): Indica o endereço dentro do ambiente do elo (empresa) da SC onde se encontra a definição do repositório de dados relativo ao objeto item (dentro do escopo do SCRP o mapeamento será efetuado através de XML, para facilitar a portabilidade).

- Inventário (DD): Indica o endereço dentro do ambiente do elo (empresa) da SC onde se encontra a definição do repositório de dados relativo ao objeto Inventário.

- Roteiro (EE): Indica o endereço dentro do ambiente do elo (empresa) da SC onde se encontra a definição do repositório de dados relativo ao objeto (BOM – *Bill of Materials*).

- Item Cliente (FF): Indica uma tabela de conversão que será utilizada caso os códigos dos itens sejam diferentes ente o fornecedor e a empresa que é o elo da SC, bem como entre o código do cliente e o da empresa elo.

Após a configuração da empresa e da origem de dados o SCRP pode ser executado (nas diferentes modalidades explicadas anteriormente (O/B/OB)) para carregar os respectivos conteúdos dos dados nos seus repositórios.

Função 2*:* Execução do BOM no SCM:

As funções a serem implementadas incluem: tabela de referência entre um código padrão (que identifica o produto final) e os diferentes códigos de seus componentes intermediários, bem como suas origens (nas diferentes empresas componentes da cadeia de suprimentos), seu código de nível dentro do MRP distribuído (que identifica o número do processo na árvore global), bem como os diferentes *lead times* componentes de cada fase do processo, sendo estes divididos em *lead time* de informação e de transporte. O *lead time* de informação é o tempo de processamento da ordem dentro deste elo produtivo, e o de transporte corresponde ao tempo de transformação (produção) e logística.

Cada código intermediário tem seu mapeamento atrelado à fonte de dados no sistema ERP do parceiro (ou sistema legado), na SC, de forma que qualquer

alteração nestes parâmetros seja automaticamente refletida (por meio de agentes instalados no ERP cliente) no SCRP.

O objetivo desta aplicação é permitir com que sejam simulados os *lead times* globais a partir da árvore do produto (BOM) para que, por meio de seus resultados, se tenha a visibilidade de todos os processos ao longo da cadeia, bem como de seus respectivos *lead times*. A figura 21 representa o funcionamento desta aplicação.

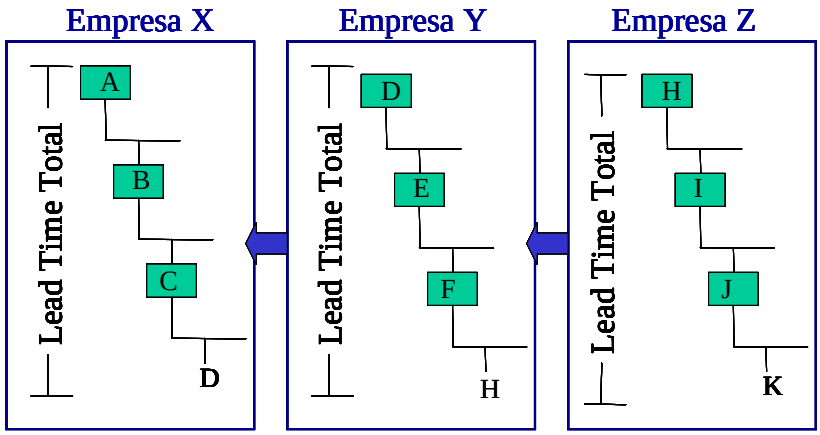

FIGURA 21: O MRP DENTRO DO SCRP

O SCRP considera, para efeito de MRP, o *lead time* total de cada item em cada elo (informação + processo), ou seja, o tempo utilizado para o processamento do pedido acrescido do tempo de transporte em cada nível dentro de cada empresa. Por exemplo, considerando-se os seguintes *lead times* para a empresa X: tempo de processamento do pedido de 2 horas; tempo de produção do componente C (incluindo-se o *lead time* de compras de D) de 3 horas; tempo de produção do item B de 2 horas; tempo de produção do item A de 1 hora, totalizando-se 7 horas. Para a estrutura MRP da aplicação SCRP a árvore (BOM - *Bill of Materials*) seria composta do item A, consumindo o item D, com um *lead time* de 7 horas. Este processo seria replicado para os demais

níveis, permitindo com que se pudesse estimar o *lead time* completo da SC como um todo.

A informação dos inventários seria aplicada somente aos itens de integração (no caso A, D, G, H e K), e mesmo assim somente à parcela de itens a serem utilizados para a SC (caso o mesmo item seja compartilhado para a cadeia ou outro fim, como no caso de demanda dependente).

Função 3*:* Cálculo do *Lead Time* dos Componentes:

Conforme salientado anteriormente, o *lead time* dentro do processo de fabricação compreende todos os tempos de produção, espera e movimentação de todos os componentes dentro da árvore do produto. Um fato a ser considerado é que nem todos os itens deverão ser controlados, apenas aqueles que são críticos para o processo produtivo e, em geral, para o SCM.

Além disso, um outro ponto a ser considerado é que pode haver mais de um fornecedor para o mesmo produto. Para efeito de cálculo de demanda e de *lead time* o SCRP vai se apoiar no fornecedor ativo para aquele item (ou o que estiver sendo utilizado com maior freqüência nos últimos 3 meses), bem como no lote econômico estabelecido para o item.

O inventário a ser considerado é aquele que não esteja comprometido, ou seja, que não esteja vinculado a uma demanda.

O objetivo desta função é o de fornecimento de parâmetros para o processo decisório com relação a operações de produção e suprimento dentro da SC.

A forma de apresentação dos resultados será através de uma planilha de cálculo, onde serão apresentados os *lead times* totais para cada item com demanda independente, no primeiro nível da SC (figura 22).

ATP - Available to Promisse - SCM										
Início:	Data/hora:			Termino:	Data/Hora:					
Item	Depend.	Empresa	Sd Inicial (Qtde)	Entradas (Qtde)	Saídas (Qtde)	Inventário (Qtde)	Demanda (Qtde)	Saldo (Qtde)	Lt Total (Hr)	Atraso (Hr)
A		1	10	20	25	5	10	-5	75	50
B	A	2	30	30	10	50	30	20	10	
D	A	3	50	10	70	10	30	-30	5	10
C	B	4	30	20	10	10	20	30	10	

FIGURA 22: ANÁLISE GLOBAL DO LEAD TIME NO SCM

Caso se necessite mais detalhes sobre o item em questão basta selecioná-lo, dentro da matriz. Dessa forma, será apresentada uma planilha para análise semelhante ao modelo utilizado dentro da programação e controle da produção (figura 23).

Item A							
Semana		**1**	**2**	**3**	**4**	**5**	**6**
Previsao		350	210	245	**400**	315	350
Estoque Projetado	150	**200**	**190**	**345**	**145**	**230**	**80**
MPS		**400**	**200**	**400**	**200**	**400**	**200**

FIGURA 23: ANÁLISE INDIVIDUAL DE CADA ITEM NO SCM

Função 4*:* Identificação do RRC – Recurso com Restrição de Capacidade:

Para identificar os pontos críticos dentro da SC, no tocante à gestão de inventários e de demanda será utilizada a abordagem da Teoria das Restrições (*Theory of Constraints - TOC*) sobre os resultados dos cálculos de *lead times*. A forma de visualização dos resultados será através de um gráfico CPM (*Critical Path Method*), uma otimização das técnicas PERT (*Program Evaluation and Review Technique*).

A técnica PERT surgiu por volta de 1957, quando os EUA planejavam construir um foguete e, em função da complexidade envolvida no empreendimento, era necessário um mecanismo de controle de. Para tanto, o grupo de pesquisa

desenvolveu um sistema denominado PERT (*Program Evaluation and Review Technique*). Como resultado, o tempo de execução do projeto caiu de 5 para 3 anos (ROCHA, 1987).

Paralelamente, a DuPont desenvolvia projetos para introdução de produtos químicos, nos quais o tempo deveria ser estimado com máximo precisão. Uma equipe de assessoria criou o método semelhante ao PERT, que recebeu o nome de CPM (*Critical Method Path*) (ROCHA, 1987).

Segundo Rocha (1987), as principais diferenças se referem a como ambas tratam as estimativas de tempo. O PERT utiliza três estimativas de tempo, que são:

- Tempo Otimista: Correspondente ao menor tempo de realização da atividade, supondo-se que ocorram todas as condições favoráveis.
- Tempo Pessimista: Corresponde ao tempo de duração da atividade, caso ocorram todas as condições desfavoráveis.
- Tempo Médio (mais provável): corresponde ao tempo que a atividade levaria, no caso de esta repetir-se diversas vezes.

O PERT possui características probabilísticas e variáveis aleatórias. O CPM, por outro lado, utiliza apenas uma estimativa de tempo, o caminho crítico.

Ambos utilizam determinados princípios:

- Lista de atividades (mais completa possível).
- Interdependência destas atividades.
- Diagrama ou rede.
- Cálculo das datas menos e mais críticas (“mais cedo” e “mais tarde”).
- Folgas – possibilidades e correções.

- Determinação do caminho crítico.
- Probabilidade de conclusão.
- Acompanhamento e respectivo controle.

Para efeito de técnica a ser contemplada como ferramenta utilizada nesta função do SCRP encontra-se o CPM por ser adequado para a identificação do RRC dentro da gestão dos *lead times*. O objetivo é fornecer subsídios para que o planejador de inventários e de demanda ao longo da SC possa tomar decisões no tocante à minimização e/ou eliminação destes gargalos.

Além do CPM como ferramenta de visualização, a força motriz desta função reside em conceitos da TOC (*Theory of Constraints*).

Segundo Cox III & Spencer (2002), a TOC é uma filosofia de gestão que surgiu a partir do desenvolvimento de um SI chamado OPT (*Optimize Production Technology*) desenvolvido no final dos anos 70 e início dos anos 80, pelo Dr. Eliahu Goldratt, cujo objetivo era o de fornecer uma visão de capacidade finita para a programação de produção nas indústrias, ou seja, considerar as restrições de manufatura na adequação das necessidades geradas pelo MRP dentro da capacidade produtiva, inclusive para sequenciamento das ordens de produção no chão de fábrica.

A TOC visa o planejamento e controle da produção, como também, a venda de produtos e serviços, enfocando principalmente a importância da restrição dentro deste contexto. Como restrição (RRC), deve ser entendido como o recurso que mais limita a produtividade de uma empresa, ou um sistema produtivo (COX III & SPENCER, 2002).

Ainda segundo os mesmos autores, os três princípios básicos da TOC podem ser citados como:

- Uma configuração produtiva tem uma meta que deve ser alcançada.
- Uma configuração produtiva é mais que a soma de suas partes.

- O desempenho de uma organização está limitado por poucas variáveis.

Segundo GIUNTINI *et al.*(2002), o gerenciamento da restrição é simples e lógico se for seguido em cinco passos que podem ser representados na forma de algoritmo pela seguinte seqüência:

1. Identificar o recurso restritivo da configuração produtiva.
2. Explorar ao máximo esta restrição.
3. Subordinar qualquer outro recurso à decisão acima.
4. Elevar a restrição.
5. Se a restrição for elevada, deve-se voltar ao item 1.

Segundo Umble & Umble (2002), no modelo de distribuição das SC, a restrição é o seu próprio sistema logístico, o qual consiste de todas as suas regras de produção, ordenamento de políticas, e estratégias de inventário. Esta é a maneira que o inventario é suprido e distribuído através do sistema, e é o que limita seu desempenho. Qualquer inventário em excesso além do requerido para manter os pulmões deve ser evitado no processo.

Dessa forma, a lógica de funcionamento do módulo de identificação do recurso crítico deverá estar baseada nos conceitos da TOC, apresentados anteriormente.

Como funcionalidade básica do SCRP foram definidas, de forma conceitual, as principais atividades que o sistema irá compreender sem considerar, no entanto, aspectos relativos ao desenvolvimento e implementação técnica da solução.

Esta especificação, no entanto, pode ser utilizada como base funcional para o desenvolvimento técnico de um SI a ser aplicado a SC reais de diferentes setores e amplitudes, sem estar atrelada ao ambiente operacional ou à proximidade física dos parceiros envolvidos na SC.

Como apresentado no capítulo 1, foi cumprido o objetivo de se propor um SI para apoio à gestão de demanda e de inventários ao longo das SC, baseado no modelo de componentes.

6. CONCLUSÕES

Um dos objetivos de um sistema de informações é que ele possa ser utilizado para agilizar o processo decisório e as operações de uma determinada organização ou estrutura, pelo fornecimento de informações precisas e que tragam conhecimento útil (PORTER, 1999). Por outro lado, uma das questões críticas nas operações das SC é a falta de visibilidade da demanda e o efeito que isto causa ao longo desta cadeia, em especial no *lead time* e níveis de inventário, causando o "efeito chicote", que é um mecanismo de proteção contra eventuais oscilações na demanda, mas que amplifica a distorção a medida que a demanda se espalha pela SC (PIRES, 2004).

Este mecanismo causa, no entanto, um *trade-off* significativo, pois o custo de transformação do produto, ao longo da SC, sofre aumento. Além disso, o armazenamento de inventário é uma atividade que não agrega valor à cadeia (PORTER, 1999).

A dissertação procurou primeiramente fazer uma revisão bibliográfica dos elementos relacionados à gestão de inventário e de demanda no SCM, bem como características operacionais das SC. Concernente a isto, buscou-se apresentar os principais aspectos do SCM e os fatores que amplificam a demanda ao longo das SC ((MACKAY & ROSIER, 1996; WALTON & MARUCHECK, 1997; MURPHY & DALEY, 1999).

Em seguida foi demonstrada a evolução da TI no apoio às operações de manufatura, tanto dentro do escopo interno das fábricas (atividades de transformação apoiadas pelo MRP e MRP-II) quanto como suporte às novas configurações produtivas (ERP). Neste tópico ficou evidenciada a limitação dos ERP no suporte ao SCM pelo fato de não considerarem a empresa expandida, ou seja, toda sua rede de valor (KEHOE & BOUGHTON, 2001).

Adicionalmente, foram discutidos os principais fundamentos do paradigma de orientação a objetos, bem como sua aderência às necessidades de SCM no

tocante a utilização de SI cujas funções estejam distribuídas em ambiente heterogêneo, como é o caso das SC e sua aplicabilidade a esta necessidade.

Posteriormente foi apresentado o modelo CCM do OMG, adicionando ao ambiente computacional o elemento de infra-estrutura, ou seja, a espinha dorsal do modelo de SI para suporte a SCM (OMG, 2002). Foram apresentados os fundamentos técnicos do ambiente. O modelo CCM CORBA possui arquitetura aberta e pode, desta forma, ser utilizado em ambientes heterogêneos, facilitando a portabilidade e integração. Por outro lado, o suporte dos serviços padronizados permite com que os aspectos de conectividade, ambiente computacional e outras questões tecnológicas sejam tratados pelo próprio ambiente, possibilitando com que o desenvolvedor tenha o foco no processo de negócio. Além da portabilidade, a infra-estrutura demonstra a possibilidade de reutilização de código e padronização no desenvolvimento do SI, gerando maior valor agregado ao processo de desenvolvimento de SI e entregando soluções mais estáveis, com menor custo de manutenção. Estes aspectos favorecem a utilização do modelo CCM como base de apoio para aplicações SI em SCM, especificamente na gestão de demanda e de inventários.

O objetivo de apresentar um modelo de SI para apoio à gestão de inventários e demanda dentro da SCM foi atingido, pois o SI proposto situa-se dentro do paradigma de orientação a objetos, com base na arquitetura de infra-estrutura CCM, atendendo integralmente ao objetivo geral pretendido. Evidentemente a proposta considera apenas a especificação conceitual, porém engloba todos os requisitos apresentados no capítulo 2. A proposta fornece uma base para a continuidade do desenvolvimento futuro do SI no tocante à elaboração da especificação técnica, codificação, implementação e testes. Nesta proposta não foi escolhido um processo de desenvolvimento de *software* em função de a especificação compreender as características funcionais do SI.

Por fim, conclui-se, pelo exposto que a solução é viável para suporte à gestão de demanda e de inventários no SCM, não sendo, porém, a única opção em termos de TI em função das crescentes transformações pelas quais ela passa.

Dentro deste contexto, tradicionais empresas produtoras de *softwares* ERP estão trabalhando em soluções de integração entre processos inter-empresarias, tendo como base o modelo de componentes distribuídos CCM. Caso esta solução se consolide no mercado poderá gerar um padrão de aplicação integradora entre aplicações distintas.

Uma outra alternativa é que este modelo possa ser a infra-estrutura básica de um grande portal Extranet, através do qual as empresas clientes possam se integrar temporariamente a seus parceiros, pelo uso de conexões intermitentes.

Como recomendação para futuras pesquisas, se encontra a elaboração de soluções para apoio à SCM que englobem aspectos de simulação, planejamento e gestão da SCM, bem como a incorporação de um sistema de gestão de desempenho que esteja atrelado à estratégia e que possa obter de sistemas transacionais os dados necessários para fornecer tais informações para o processo decisório.

Como sugestão, recomenda-se o desenvolvimento do SCRP baseado no modelo apresentado, e que possa ser testado em SC real, como a do setor automobilístico, que reúne as características apresentadas no capítulo 2.

Como as SC são configurações que têm continuamente se estabelecido no ambiente empresarial, a tendência é que a velocidade com que estes relacionamentos ocorram seja maior, gerando novas necessidades em termos de SI, que podem ser resolvidas com a utilização do paradigma de orientação a objetos distribuídos, de uma forma aberta (sem dependência computacional).

REFERÊNCIAS BIBLIOGRÁFICAS

ABREU, A. F. (1999) **Gestão da Inovação - Uma Abordagem Orientada à Gestão Corporativa**. 1a Edição. Florianópolis: Editora IGTI, 1999.

ARAVECHIA, C. H. M. (2001) **Avaliação de Desempenho na Gestão da Cadeia de Suprimentos**. Dissertação de Mestrado. Faculdade de Engenharia Mecânica e de Produção, Universidade Metodista de Piracicaba, 2001.

BAATZ, E. B. (1995) ***CIO 100 – Best Practices: The Chain Gang.*** CIO, vol.8, n.19, 1995, pp. 46-52.

BALLOU, R. H. (1993) **Logística Empresarial – transportes, administração de materiais, distribuição física**. SP: Atlas, 1993.

BARROS, M. C. B. (2003) – **Um Modelo para *Deployment* de Componentes em CORBA.** Dissertação de Mestrado. Instituto de Computação, Unicamp, 2003.

BASILI, V. R., CALDIERA, G., ROMBACH, H. D. (1994) ***The Goal/Question/Metric Paradigm.*** *Encyclopaedia of Software Engineering*, volume 1. John Wiley & Sons, 1994.

BEAMON, B. M. (1999) ***Measuring Supply Chain Performance.*** *International Journal of Operations & Production Management,* v. 19, no. 3, 1999

BERGAMASCHI, S. (1999) **Um Estudo sobre Projetos de Implementação de Sistemas para Gestão Empresarial** – Dissertação de Mestrado. Faculdade de Economia, Administração e Contabilidade, USP, 1999

BIO, S. R. (1991) **Sistemas de Informação – Um enfoque empresarial**, São Paulo: Editora Atlas, 1991, 183p.

BOOCH, G., RUMBAUGH, J., JACOBSON, I. (1999) ***The Unified Modelling Language User Guide***, *Reading Massachusetts*: Addison-Wesley, 1999, 482p.

BOWERSOX, D. J. (1996) ***Logistical Management***. McGraw Hill, 1996.

BRANDO, T. (1996) ***Comparing CORBA and DCE***. *Object Magazine* 6 (1), pp. 52-57, March 1996.

BUZATO, L. E., RUBIRA C. M. F. (1998) **Construção de Sistemas Orientados a Objetos Confiáveis**, Rio de Janeiro: DCC/IM 11, Escola de Computação, 1998, 160p.

CHEN, P. (1976) ***The Entity-Relationship Model - Toward a Unified View of Data***. *ACM Transactions on Database Systems* 1 (1), 1976, pp. 9-36.

CHIAVENATO, I. (1995) **Administração de Empresas Uma Abordagem Contingencial**, São Paulo: McGraw-Hill, 1995, pp. 41-104.

CODD, E. F. (1970) ***A Relational Model of Data for Large Shared Data Banks***. *Communications of the ACM* 13 (6), pp. 377-387, 1970.

COELHO, A. L. V. (1998) **Caracterização de um serviço de gerencia distribuído para objetos multimídia persistentes.** Dissertação de Mestrado. Faculdade de Engenharia Elétrica, Unicamp, 1998.

COLLINS, R., BECHLER, K. & PIRES, S. R. I. (1997) ***Outsourcing in the Automotive Industry: from JIT to Modular Consortia.*** *European Management Journal*, vol.15 (5), pp. 93-101, 1997.

COOPER, M. C., LAMBERT, D. M., PACH, J. D. (1998) ***Supply Chain Management*****: Mais do que um novo nome para a logística –** Parte I. Logística Moderna, n.54, pp. 17-19, janeiro/fevereiro, 1998.

COX III, J. F., SPENCER, M. S. (2002) **Manual da Teoria das Restrições**. Porto Alegre: Bookman, 2002

DAVENPORT , T. H. (1998) **Ecologia da Informação** ,São Paulo: Editora Futura, 1998, 316p.

DAVIS, T. (1993) ***Effective Supply Chain Management***. *Sloan Management Review/ Summer*, 1993.

DIAZ, C. P., PIRES, S. R. I. (2004) ***Variação da Demanda ao longo da Cadeia de Suprimentos: o efeito da amplificação da demanda***, *Proceedings* of XXII Encontro Nacional de Engenharia de Produção, ENEGEP, Ouro Preto, MG, Brasil.

DIC (1996) ***Dicionário Eletrônico Michaelis***. Versão 4.00. DTS Software Ltda., novembro/1996.

DISNEY, S. M. (2003) ***Vendor-managed inventory and bullwhip reduction in a two-level supply chain.*** *International Journal of Operations & Production Management.* Vol 23 – 2003

FARLEY, G. A. (1997) ***Discovering Supply Chain Management: A Roundable Discussion.*** *APICS – The Performance Advantage.* vol.7, n.1, pp. 38-39, 1997.

FARLEY, G. A. (1998) ***Defining Enterprise Resource Planning***. *APICS – The Performance Advantage. March*,1998. Disponível na internet: www.apics.org/OtherServices/Articles/defining.htm. Consulta em 07/07/1999.

FERDOWS, K., DE MEYER, A. (1990) Lasting Improvements in Manufacturing Performance: In Search of a New Theory. Journal of Operations Management, vol. 9, no 2, April, 1990.

FERREIRA, M. P. (2002) **Desenvolvimento de Software Alinhado aos Objetivos Estratégicos de Negócio: Proposta de Uma Metodologia.** Dissertação de Mestrado. Programa de Pós Graduação em Engenharia de Produção. Universidade Federal de Santa Catarina, Florianópolis, 2002.

FLEURY, P. F. (2000) ***Supply Chain Management***: **Conceitos, Oportunidades e Desafios da Implementação.** Rio de Janeiro, Centro de Estudos em Logística – COPPEAD – UFRJ, 2000.

FOWLER, M., SCOTT, K. (1997) ***UML Distilled***. Addison-Wesley, 1997.

FRICK, S. (1996) **Produtos, Estruturas de Mercado e Estratégias Competitivas no Setor de Software**. Economia & Empresa, vol. 3, n. 1 , pp. 34-43, 1996.

GORANSON, H. T. (1999) ***Agile Virtual Enterprise: Cases, Metrics, Tools***. Greenwood Pub Group, 1999.

GRAHAN, I. (1994) ***Object Oriented Methods***, London, Addison-Wesley Publishing Company, 1994,471p.

GUMAER, R. (1996) ***Beyond ERP and MRPII: Optimized planning and synchronized manufacturing.*** *IEEE Solutions*, v. 28, n. 9, pp. 32-35, *Sept.*1996

GIUNTINI, N. *et al.*(2002). **Teoria das Restrições : Uma Nova Forma de "Ver e Pensar" o Gerenciamento Empresarial**. Publicado no 2º Seminário USP de Contabilidade em 2002 e disponível no site: (http:// www.eac.fea.usp.br/congressousp/seminario2/trabalhos/b68.pdf) acessado em 07/11/2003 às 17:00 hrs.

HAEFEL, R. M. (1999) ***Enterprise JavaBeans***. New York : O'Reilly, pp. 1-76.

HARDWICK, M. S., RANDO D., TOM E MORRIS, K. C. (1996) ***Sharing Manufacturing Information in Virtual Enterprises.*** *Communications of the ACM. February*, 1996. Vol 39, No. 2.

HAX A. C., GOLOVIN J. (1978) **Hierarchical Production Planning Systems**. Studies on Operations Management, 1978.

HILL, C. W. (1998) ***International Business: Competing in the Global Marketplace***. Ch, IL: Irwin, 1998.

HUMPHREY, J., LECLER, Y. & SALERNO, M. (2000), ***Global Strategies and Local Realities: The Auto Industry in Emerging Markets***, GB, Macmillan Press LTD, 2000.

KAPLAN, R. S., NORTON, D. P. (1997) **A Estratégia em Ação: *Balanced Scorecard***. 4.ed. RJ:Campus, 1997.

KEHOE, D., BOUGHTON, N. (2001) ***Internet Based Supply Chain Management.*** *International Journal of Operations & Production Management***.** Vol. 21, n. 4, 2001, pp. 516-524.

KERN, V. M. (1997) **Manutenibilidade da Semântica de Modelo de Dados de Produtos Compartilhados em Rede Interoperável**. Tese de Doutorado. Programa de Pós Graduação em Engenharia de Produção. Universidade Federal de Santa Catarina, 1997

KHOSHAFIAN, S. (1993) ***Object-Oriented Databases***. John Wiley & Sons, 1993, 362 pp.

KOTLER, P. (2000) Administração de Marketing. Prentice-Hall, 2000. 764 pp.

LA LONDE, B. J., MASTERS, J. M. (1994) ***Emerging Logistics Strategies: Blue Print for the Next Century***. *International Journal of Physical Distribution & Logistics Management***.** Vol. 24, n. 7, p. 35-47.

LAMBERT, D. M., COOPER, M. C., PAGH J. D. (1998) ***Supply Chain Management: Implementation Issues and Research Opportunities***. *International Journal of Logistics Management***.** Vol. 9, n. 2, 1998, pp. 1-19.

Van LANDEGHEM, H., VANMAELE, H. (2002) ***Robust planning: a new paradigm for demand chain planning***. *Journal of Operations Management*, v. 20, n. 6, 2002, pp. 769-783.

LEE, H. L., BILLINGTON, C. (1992) ***Managing Supply Chain Inventory: Pitfalls and Opportunities***. Sloan Management Review, Vol. 33, n. 3, 1992, pp. 65-73.

LEE, H. L., PADMANABHAN, V., WHANG, S. (1997) ***The bullwhip effect in supply chains.*** *Sloan Management Review*, Spring, 1997a, pp. 93-102.

LUCERO, A. G. R. (2001) **Um Método de Otimização para a Programação da Manufatura em Pequenos Lotes**. Dissertação de Mestrado. Programa de Pós Graduação em Engenharia Mecânica. Universidade Federal de Santa Catarina, 2001.

LUMMUS, R. R.; VOKURKA, R. & ALBER, K. L. (1998) ***Strategic Supply Chain Planning***, *Production and Inventory Management Journal*, Third Quarter 1998, pp. 49-58.

LUNG, Y. (2000) ***Is the Rise of Emerging Countries as Automobile Producers an Irreversible Phenomenon?*** *Global Strategies and Local Realities: The Auto Industry in Emerging Markets*, GB, Macmillan Press LTD, 2000.

MACKAY, D., ROSIER, M. (1996) ***Measuring organizational benefits of EDI diffusion: A case of the Australian automotive industry***. *International Journal of Physical Distribution & Logistics Management,* v. 26, n. 10, 1996, pp. 60-78.

MARCONI, M. A. , LAKATOS, E. M. (2002) **Técnicas de Pesquisa**. São Paulo:Editora Atlas. 5ª edição, 2002.

MASTELARI, N. (2004) **Contribuição ao Processo de Integração de Informações de Manufatura para Empresas de Pequeno e Médio Porte** – Tese de Doutorado. Faculdade de Engenharia Mecanica, Unicamp, 2004.

MAXIMIANO, A. C. A. (2002) **Teoria Geral da Administração**. 3ª edição – Ed. Atlas – São Paulo-SP, 2002.

MAYBEE, D. H., OSTERWEIL, L. (1995) ***Multilanguage Interoperability in Distributed Systems: Experience Report***. *University of Massachusetts Amherst,* Report UM-CS-1995-075, August 1995.

METTERS, R. (1997) ***Quantifying the bullwhip effect in supply chains***. *Journal of Operations Management*, v. 15, n. 2, 1997, pp. 89-100.

MICHEL, R. (1998) ***Model citizens: ERP's implementation tools provide process framework.*** *Manufacturing Systems Magazine*, v.16, n.2, pp. 29-44, Feb 1998.

MONTEZ, C. (1997) **Um Modelo de Programação para Aplicações de Tempo Real em Sistemas Abertos**. Monografia do Exame de Qualificação de Doutorado, Departamento de Automação e Sistemas, Universidade Federal de Santa Catarina, Julho de 1997.

MORGAN, J., MONCZKA, R. M. (1995) ***Alliances for New Products****. Purchasing*, v.118, n.1, 1995, pp. 103-109.

MURPHY, P. R., DALEY, J. M. (1999) ***EDI benefits and barriers – comparing international freight forwarders and their customers***. *International Journal of Physical Distribution & Logistics Management*, v. 29, n. 3, 1999, pp. 207-216.

NBR ISO 9000-3 (1993) **Normas de gestão da qualidade e garantia da qualidade.** ABNT – Associação Brasileira de Normas Técnicas, Novembro/1993.

OMG *Object Management Group* (1995) ***The Common Object Request Broker: Architecture and Specification***. *Revision* 2.0, *July* 1995.

OMG. *Object Management Group* (1993) ***Object Request Broker Architecture***. OMG TC *Document* 93.07.02, Framingham MA, 1993.

OMG. Object Management Group (2002) ***CORBA Component Model.*** *Specification* 02-06-69. *Revision June* 2002.

OMG. *Object Management Group* (2002a) ***OMG CIDL Sintax and Semantics.*** *Specification* 02-06-70. *Revision June* 2002.

OMG. *Object Management Group* (2002b) ***CCM Implementation Framework*** . *Specification* 02-06-71. *Revision June* 2002.

ORFALI, R., HARKEY, D. (1998) ***Java and Corba***. 2nd Edition – John Wiley and Sons Inc., 1998.

PAGE-JONE, M. (1988) **Projeto Estruturado de Sistemas**. São Paulo-SP, McGrawHill, 1988.

PIRES, S. R. I. (1998) **Gestão da cadeia de suprimentos e o modelo de consorcio modular**. Revista de Administração – USP, v.33, n.3, 1998, pp. 5-15.

PIRES, S. R. I. (2004) **Gestão da Cadeia de Suprimentos: conceitos, estratégias, práticas e casos**. São Paulo-SP, Editora Atlas, 2004.

PMI (2000) ***PMBOK*: *Project Management Body of Knowledge***. *Project Management Institute*, 2000.

PORTER, M.E. (1999) ***On Competition***. São Paulo-SP, Ed. Campus, 1999.

PRESSMAN, R. S. (1995) **Engenharia de Software**. São Paulo-SP, Makron Books do Brasil, 1995.

RICE, J. B. & HOPPE, R. M. (2001) ***Supply Chain versus Supply Chain: The Hype and the reality.*** *Supply Chain Management Review*, pp. 46-54, *September/October*, 2001.

RIGBY, D. (1983) ***The Secret History of Process Reengineering****. Planning Review. 1983.*

ROCHA, L. O. L. (1987) **Organização e Métodos: uma abordagem prática**. 6ª edição – São Paulo: Atlas, 1987

SAFISZADECH, M. H., RITZMAN, L. P. (1997) ***Linking Performance Drivers in Production Planning and Inventory Control to Process Choice****. Journal of Operations Management***, 1997.**

SALLES, J. A. A. (2003) **Avaliação de Sistemas de Desempenho**. Notas de Aula. Programa de Pós Graduação em Engenharia de Produção, Unimep, 2003.

SCAVARDA, L. F. , FREESE, J., HAMACHER, S., PIRES, S. R. I. & SIHN, W. (2001) ***The Transition from Multidomestic to Global Supply Chain Operations in the Automotive Industry of Emerging Countries***, Proceedings of XII *Conference of the Production and Operations Management Society*, Orlando, U.S.A., 2001.

SCAVARDA, L. F., HAMACHER, S. (2004) ***Trends in the automotive industry's Supply Chain Management***, Proceedings of XXII Encontro Nacional de Engenharia de Produção, ENEGEP, Ouro Preto, MG, Brasil, 2004.

SEBRAE-SP (2001) **Apresenta MPEs de Base Tecnológica: conceituação, formas de financiamento e análise de casos brasileiros**, julho de 2001. Disponível em: http://www.sebraesp.com.br/sebrae/sebraenovo/pesquisa/pdf_pesquisa/EMBATEC.pdf Acesso em 21/11/01.

SIEGEL, J. (2000) ***CORBA 3 – Fundamentals and Programming***, 2nd edition, OMG Press, 2000.

SILVA, E. L., MENEZES, E. M. (2000) **Metodologia da pesquisa e elaboração de dissertação**. Programa de Pós Graduação em Engenharia de Produção, Universidade Federal de Santa Catarina, Florianópolis, 2000, 118p.

SLACK, N., CHAMERS, S. *et al.*(1997) **Administração da Produção**. 1.ed. São Paulo: Atlas, 1997. 726p.

SKINNER, W. (1969) ***Manufacturing: Missing link in Corporate Strategy***. *Harvard Business* Ver. 47(1), 1969, pp. 136-145.

SOLEY, R., STONE, C. (1995) ***Object Management Architecture Guide***. Third edition. John Wiley & Sons, Framingham MA, 164 p., 1995.

STEVENS, T. (1997) ***Kodak focuses on ERP****. Industry Week*, v. 246, n.15, pp. 130-134, Aug 1997.

SWEBOK. (2001) ***Guide to the Software Engineering Body of Knowledge. A Stone Man Version (version 0.95)***. *A project of the Software Engineering Coordinating Committee joint IEEE Computer Society* - ACM committee, May, 2001.

SZYPERSKI, C. (1998) ***Component Software: Beyond Object-Oriented Programming***. Addison-Wesley, 1998.

TIBBITS, F. (1995) ***CORBA: A Common Touch for Distributed Applications****. Data Communications 24 (7), pp. 71-75, 1995.*

UMBLE, E. J., UMBLE, M. (2002) *Integrating the Theory of Constraints into Supply Chain Management*. *Decision Sciences Institute*, 2002 - Annual Meeting Proceedings.

VERWIJMEREN, M., van der VLIST, P., DONSELAAR, K. (1996) ***Networked Inventory Management Information Systems: Materializing Supply Chain Management***. *International Journal of Operations & Production Management.* Vol. 26, n. 6, 1996, pp. 16-31.

VINOSKI, S. (1993) ***Distributed Object Computing with CORBA***. C++ Report July/August 1993.

VOLLMAN, T. E., CORDON, C. (1996) ***Making Supply Chain Relationships Work****. M2000 Business Briefing*, n.8, Lausanne, IMD, 1996.

WALTON, S. V., MARUCHECK, A. S. (1997) *The relationship between EDI and supplier reliability. International Journal of Purchasing and Materials Management*, v. 33, n. 3, August 1997, pp. 30-35.

WANGENHEIM, C. G., ROMBACH, H. D., RUHE, G. (1998) **Tutorial on Melhoramento de Software Baseado em Mensuração – Como Aplicar GQM**

na Pratica? *Proceedings of the IX CITS* – Conferência International de Tecnologia de Software: Qualidade de Software, Curitiba, Brazil, 1998.

WIGHT, O. (1993) ***Executive`s Guide to Succesfull MRPII***. APICS, 1993.

YOURDON, E. (1995) **Declínio e queda dos analistas e dos programadores**. São Paulo: Makron Books, 1995.