FACULDADE DE TECNOLOGIA DE SÃO PAULO

FELIPE DE ALCANTARA PIO

Tecnologias Middleware na implementação de sistemas ERP SAP R/3

São Paulo
2011

FACULDADE DE TECNOLOGIA DE SÃO PAULO

FELIPE DE ALCANTARA PIO

Tecnologias Middleware na implementação de sistemas ERP SAP R/3

Monografia submetida como exigência
Parcial para a obtenção do Grau de
Tecnólogo em Processamento de Dados

Orientador: Professor Paulo Roberto Bernice

São Paulo
2011

Dedico este trabalho aos meus amigos e família, que sempre me incentivam a progredir e a me esforçar mais para alcançar meus objetivos.

Agradeço, primeiramente, aos meus amigos que me ajudaram a concluir esse trabalho, ao Professor Paulo Roberto Bernice pela orientação, ao meu companheiro Daniel que sempre se mostrou solícito me ajudando com material de pesquisa e a minha mãe, que mesmo indiretamente sempre me deu forças e me cobrou quando necessário.

RESUMO

Vários fornecedores de ERP trabalham com tecnologias diferentes. O ERP SAP R/3 possui várias formas de *Middleware*, ou seja, interfaces que permitem a interação entre diferentes aplicações de softwares, geralmente também sobre diferentes plataformas de hardware e infraestrutura para troca de dados. Desta forma a escolha correta da tecnologia é de extrema importância, pois, conhecendo-se antecipadamente as diversas características de cada uma, permitirá a correta decisão.

Este trabalho traz uma abordagem sobre as implementações de sistemas ERP, utilizando principalmente o ERP de Mercado SAP R/3. São abordados neste trabalho os principais fatores pelos quais as empresas não alcançam os resultados esperados e apresenta um conjunto de boas práticas de planejamento, dentre as quais recebe destaque o *Middleware* do ERP, mostrando inicialmente os principais mecanismos existentes para esta integração e indicando quais as melhores alternativas dentro do contexto tecnológico atual.

Palavras-chave: ERP, SAP, tecnologias de *middleware*, eficiência operacional.

ABSTRACT

ERP vendors work with different technologies. ERP SAP R/3 has various forms of middleware, i.e., interfaces that allow interaction between different software applications, usually also on different hardware platforms and infrastructure for data exchange. Thus the correct choice of technology is extremely important because, knowing in advance the various features of each, will allow the correct decision.

This paper works on the implementation of ERP systems, primarily using the market ERP SAP R/3. There is also in this paper the main factors by which the companies did not reach the expected results and a presentation of a set of best practices in planning, among which receive prominent ERP Middleware, initially showing the main mechanisms for this integration and indicating which better alternatives within the current technological context.

Keywords: ERP, SAP, middleware technologies, operational efficiency.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

BD - Banco de dados

COM - *Component Object Model*

CORBA – *Common Object Request Broker Architecture*

CSV - *Comma-Separated Values*

DAT - *Data*

DCOM - *Distributed Component Object Model*

EDI - *Electronic Data Interchange*

EDS – *Electronic Data System*

ERP – *Enterprise Resource Planning*

ESA - *Enterprise Services Architecture*

FTP – *File Transfer Protocol*

GUI – *Graphical User Interface*

HTML - *HyperText Markup Language*

HTTP – *HyperText Transfer Protocol*

HTTPS – *HyperText Transfer Protocol Secure*

IBM – *International Business Machines Corporation*

IDL - *Interface Definition Language*

J2EE – *Java 2 Platform, Enterprise Edition*

MRP – *Manufacturing Resource Planning*

NCP - *Network Control Protocol*

POS - *Persistent Object Service*

PPP - *Point-to-Point Protocol*

RFC - *Remote Function Call*

RMI - *Remote Method Invocation*

RPC - *Remote Procedure Calls*

SAML – *Security Assertion Markup Language*

SAP (Sistema) - *Systeme, Anwendungen, Produkte in der Datenverarbeitung*

SAP (Empresa) - *Systemanalyse und Programmentwicklung*

SCM – *Supply Chain Management*

SOA – *Service-Oriented Architecture*

SOAP – *Simple Object Access Protocol*

TCP/IP - *Transfer Control Protocol/Internet Protocol*

TXT – *Text*

UDDI – *Universal Description Discovery and Integration*

URL – *Uniform Resource Locator*

WSDL – *Web Services Description Language*

XML – *Extensible Markup Language*

SUMÁRIO

INTRODUÇÃO

O ERP é um Sistema de Informação Integrado, dividido por vários departamentos que se comunicam e atualizam a mesma base de dados conforme CORREA H.L. (1999). Seu objetivo é abranger a empresa como um todo para o compartilhamento de informação com outros departamentos.

Para isso, o ERP SAP R/3 utiliza diversas formas de *Middleware*, que são interfaces que fazem a interação entre diferentes aplicações de softwares, plataformas de hardware e infraestrutura de troca de dados. Sendo assim, a escolha da tecnologia mais adequada é extremamente importante, e para isso é necessário conhecer os diferentes aspectos de cada uma delas.

Com esta monografia, pretende-se discorrer sobre os sistemas ERP e suas implementações, utilizando como base o ERP de Mercado SAP R/3. Serão ainda abordados neste trabalho os grandes motivos que fazem as empresas não conseguirem alcançar os resultados planejados e será apresentado um conjunto de boas práticas de planejamento, com destaque para o *Middleware* do ERP, falando primeiramente sobre os principais meios existentes para esta integração e mostrando quais as melhores opções no contexto tecnológico atual.

1. IMPLEMENTAÇÃO DE SISTEMAS ERP

1.1. Sistemas ERP

Na década de 1960 foi criada uma nova técnica de planejamento de pedidos de matéria-prima, que foi chamada de MRP – "*Manufacturing Resource Planning*" ou Planejamento de Recursos de Manufatura –, que tinha como meta auxiliar a produzir e comprar de acordo com a demanda, segundo o livro Planejamento, Programação e Controle da Produção – MRP II / ERP (1999). Essa técnica permitia determinar, com base na informação de estoque dos materiais que compunham um produto final, qual matéria-prima comprar e em qual momento comprar. Esse controle ajudava a reduzir a quantidade de matéria-prima estocada, reduzir o tempo gasto na produção e distribuição e ainda melhorar a eficiência do processo.

Com o passar do tempo, o MRP mostrou ser um eficiente método para gestão de inventários, mas que deixava isoladas as outras áreas da empresa. Com isso, na década de 80 houve a evolução do sistema para o MRP II, que é diferente do MRP na questão da decisão, pois além das decisões do que comprar, quanto e quando, o MRP II auxiliava a tomada de decisões sobre como produzir, definindo quais recursos usar. Por consequência, não atendia mais apenas as necessidades de informação sobre o inventário de materiais, mas também atendia às necessidades de informação para tomada de decisões gerenciais.

Ainda que essa mudança tenha sido um diferencial, o MRP II só funcionaria adequadamente agregando informações de outros sistemas da empresa. Com essa necessidade de integração, foi criado o ERP – "*Enterprise Resource Planning*" ou, Planejamento de Recursos do Empreendimento. A figura 1 mostra os módulos atuais do ERP SAP R/3.

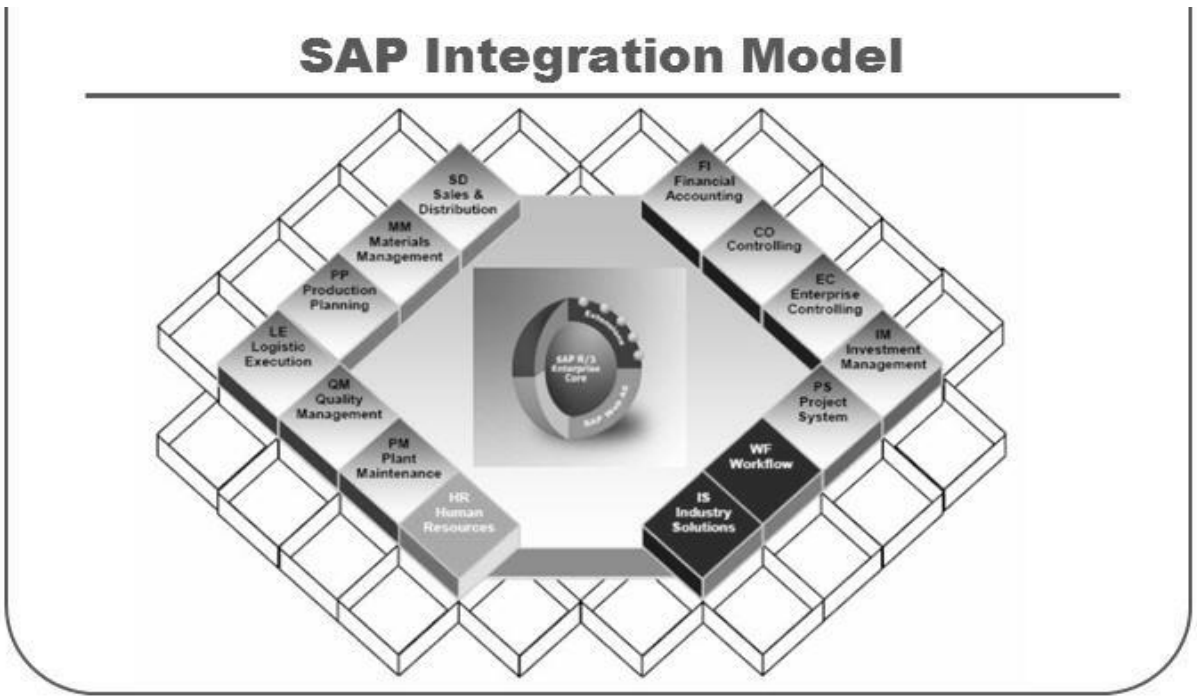

Figura 1 - Diferenças entre auditoria interna e externa

- SD (*Sales e Distribution* - Vendas e Distribuição)
- MM (*Materials Management* – Gerenciamento de Material)
- PP (*Production Planning* – Planejamento de Produção)
- LE (*Logistic Execution* – Execução de Logística)
- QM (*Quality Management* – Gerenciamento da Qualidade)
- PM (*Plant Maintenance* – Manutenção da Planta)
- HR (*Human Resources* – Recursos Humanos)
- FI (*Financial Management* – Gerenciamento Financeiro)
- CO (*Controlling* – Controles)
- EC (*Enterprise Controlling* – Controle Empresarial)
- IM (*Investment Management* – Gerenciamento de Investimentos)
- PS (*Project System* – Sistemas de Projeto)
- WF (*Workflow* – Fluxo de Material)
- IS (*Industry Solutions* – Soluções Industriais)

O ERP não serve apenas no planejamento, mas também faz o controle e suporta a todos os processos produtivos, operacionais, comerciais e administrativos da empresa, facilitando a troca de informações dentro da empresa, integrando todas as áreas da organização, como Recursos Humanos, departamento de Marketing ou departamento Financeiro. Isto é possível porque o ERP é um Sistema de Informação Integrado, dividido por vários departamentos que se comunicam entre e atualizam um mesmo repositório de dados.

O sistema ERP SAP R/3 (*Systeme, Anwendungen, Produkte in der Datenverarbeitung* ou Sistema, Aplicações e Produtos em Processamento de Dados) surgiu em 1º de abril de 1972. Cinco colaboradores da IBM fundaram a divisão SAP como "*Systemanalyse und Programmentwicklung*" (Sistemas de Análise e Desenvolvimento de Programas) na cidade de Mannheim, Alemanha. Essa divisão foi fundada para o desenvolvimento de softwares com padrão organizacional onde todos os processos de negócios seriam integrados. Atualmente, a matriz fica em Walldorf, também na Alemanha, conforme livro *Web Programming with SAP Web Application Server* (2004).

O sistema ERP SAP R/3, conforme Schneider, Neureither (2005), é baseado na arquitetura cliente-servidor de três camadas nas quais as mesmas podem estar dispostas em:

- **Um único computador:** sendo denominada de hierarquia de uma fila (*one tier hierarchy*);
- **Dois ou mais computadores:** onde a camada de apresentação ao usuário se comunica com as camadas de aplicação e de banco de dados instanciadas de forma agrupada. Este tipo de disposição é denominado de hierarquia de duas filas (*two tier hierarchy*);
- **Três ou mais computadores:** onde a camada de apresentação ao usuário se comunica com as camadas de aplicação e estas se comunicam com a camada de banco de dados instanciadas de forma separada., Este tipo de disposição é denominado de hierarquia de três filas (*three tier hierarchy*).

Segundo Cliffe, S.(1999), o auge dos sistemas ERP foi por volta de 1998-1999 devido ao *bug* do milênio, quando várias empresas adotaram o ERP como uma nova plataforma, já que seus sistemas legados precisariam ser alterados.

Algumas das características principais dos sistemas ERP são:

- **Flexibilidade**: Se adapta a mudanças no ambiente externo;
- **Adaptabilidade**: Se adapta as necessidades da empresa;
- **Integridade**: todas as transações processadas pela organização devem ser gravadas de forma centralizada para que as consultas tiradas do sistema possam refletir o melhor possível suas realidades operacionais, dando acesso às informações em tempo real
- **Modularidade**: é possível usar um módulo independentemente do outro.

Segundo Corrêa, H.L.(1999), o sistema ERP possui outras vantagens sobre o MRPII e "(...) talvez seja esta a principal motivação de grande número de empresas que optam por adotá-lo: a integração entre as várias áreas e setores funcionais da organização, todas compartilhando uma mesma base de dados única e não redundante".

1.2. Arquitetura de Sistemas SAP R/3

Davenport, Thomas (1998) divide os ERP em quatro blocos:

- **Financeiro:** dividido em contabilidade, contas a pagar, contas a receber e fluxo de caixa;
- **Recursos Humanos:** dividido em folha de pagamento, gerenciamento de recursos humanos e controle de despesas de viagem.
- **Operações e Logística:** dividido em gerenciamento de estoques, o MRP e o faturamento.
- **Vendas e Marketing:** processamento de pedidos e gerenciamento e planejamento de vendas.

1.3. Escolha e Implementação de Sistemas ERP

Segundo Souza; Zwicker (2000), as várias fases a que são submetidos um projeto de utilização e desenvolvimento de sistemas de informação é representada por um ciclo de vida de sistemas.

Na etapa de decisão e seleção a empresa decide utilizar um sistema ERP e procura um fornecedor. Para escolher o sistema a ser adquirido, a empresa deve estudar vários aspectos, tais como o custo de manutenção, licenciamentos futuros e atualização de software, correção de erros, capacidade de adequação às mudanças (do negócio e da legislação), assim como a própria longevidade da aplicação adquirida devem ser considerados. Também deve ser feita uma análise do ROI (*return of investment*) e de uma forma mais ampla a implementação de ERP em outras organizações do mesmo setor.

A implementação é a segunda etapa do ciclo de vida de sistemas ERP, embora o termo seja utilizado para representar o ciclo de vida completo. Essa etapa pode ser definida como o processo pelo qual os módulos do sistema são colocados em funcionamento em uma empresa e envolve a adaptação dos processos de negócio ao sistema, a parametrização e modificação do sistema, a configuração de hardware e software, o treinamento de usuários e a disponibilidade de suporte.

Na etapa de utilização o sistema passa a fazer parte do dia-a-dia das operações. Essa etapa realimenta a etapa de implementação, pois durante o processo de utilização surgem novas possibilidades e necessidades que podem ser resolvidas por novos módulos, pela parametrização ou pela modificação dos módulos atuais.

A implementação de um sistema ERP é um investimento que consome muitos recursos, entre eles, tempo e dinheiro conforme matéria *Living with ERP* apresentada na revista *CIO Magazine* (1998). Se não houver um rigoroso controle de custos, diversos problemas podem surgir em conseqüência disto, levando algo que deveria oferecer uma solução a se tornar um problema. Muitas vezes ocorrem problemas organizacionais durante a implementação e a utilização de sistemas ERP, sendo relativamente comum as empresas relatarem suas traumáticas experiências durante e após estes processos conforme NAH, F.F.H et al. (2001).

1.3.1. Problemas enfrentados na implementação

A forte expansão dos ERPs na indústria de software trouxe também vários casos de fracassos em relação às implementações. Nestas implementações mal sucedidas as empresas perderam não só o capital e tempo investidos, mas também prejudicaram o andamento dos processos de negócios.

Muitos casos ocorreram devido aos problemas relacionados à organização dos processos empresariais ou às perspectivas inadequadas levantadas pela implementação do ERP. Os principais problemas enfrentados pela empresa se referiam às tentativas de se integrar a funções e atividades que na organização sempre foram tratadas em separado.

Outra situação ocorre porque a empresa busca uma padronização dos procedimentos e das normas a serem seguidas pela empresa. "Esta busca por uma cultura mais disciplinada no que diz respeito a informações, processos e sistemas, conduz a conflitos pessoais e setoriais na organização e a conseqüente fortificação de cada setor ao procurarem evidenciar suas especificidades em detrimento de uma integração maior. (...) Este tipo de situação é facilmente encontrada em empresas do setor de tecnologia da informação, em ambientes de trabalhadores do conhecimento, liberais e empreendedores." conforme Davenport, Thomas (1998).

Davenport, Thomas (1998) ressalta também que a estratégia financeira sofre pelo menos dois impactos significativos em sua elaboração, decorrente da implementação de sistemas ERP. O primeiro é quando ocorrem os altos custos ocasionados pela decisão de investimento em uma determinada tecnologia como os sistemas ERP. O segundo ocorre quando os resultados decorrentes do bom ou mau uso destes sistemas geram provisões positivas ou negativas para o fluxo financeiro sendo necessário reposicionamento monetário de caixa.

Segundo autores Smith H. Jeff et al. (2001) e Keil; Robey (2001), qualquer que seja o campo de projeto, atrasos, falhas e dúvidas muitas vezes só são comunicados à administração sênior quando já é tarde demais.

Diversos estudos indicam que 70 por cento de todos os projetos de reengenharia de processos de negócios fracassam em proporcionar os benefícios prometidos. Do mesmo modo, uma alta porcentagem dos projetos de planejamento de recursos não é implementada totalmente e não atinge seus objetivos mesmo após três anos de trabalho segundo Cliffe, S.(1999).

Nem todos os aspectos do processo de implementação podem ser facilmente controlados ou planejados (Alter; Ginzberg (1978)). Usuários que lideram atividades de projeto costumam usar sua posição para favorecer interesses privados e conquistar poder, ao invés de promover objetivos organizacionais (Franz; Robey (1984)). Ou seja, nem sempre os usuários se envolvem nos projetos de sistemas de modo produtivo.

Enquanto alguns recebem bem o novo sistema porque percebem que a mudança introduzida será benéfica para si, outros podem considerá-la prejudicial aos seus interesses e resistir a ela (Joshi, Kailash (1991)).

Por conseguinte, a estratégia de implementação deve não apenas incentivar a participação e o envolvimento do usuário, como também abordar a questão da contra-implementação (Keen, Peter(1981)). Contra-implementação é uma estratégia deliberada para frustrar a implementação de um sistema de informação ou de uma inovação na organização.

Os principais fatores que podem ocasionar o insucesso da implementação do sistema ERP são:

- **Funcionalidade:** a empresa deve verificar se o sistema se adapta à maioria das práticas, políticas e regras da organização;
- **Resistência Organizacional**: os empregados podem acreditar que a mudança irá prejudicar o processo e comprometer seus empregos, uma vez que a implantação do ERP traz uma melhor distribuição da informação. Muitas vezes, inclusive, serão necessárias menos pessoas para desempenhar a atividade agora automatizada, criando o receio de possíveis demissões. Por se tratar de projeto complexo, as implementações de sistemas ERP necessitam de uma equipe de projeto dedicada, o que leva as organizações a enfrentarem outro problema: o da resistência de líderes funcionais em disponibilizar seus recursos mais valiosos ao projeto, pois isso compromete suas capacidades em executar as atividades rotineiras da organização. Alguns fatores organizacionais importantes no planejamento e implementação de sistemas:
  - Participação e envolvimento do usuário;
  - Projeto de cargos;
  - Monitoração de padrões e de desempenho;
  - Ergonomia;
  - Procedimento para resolução de queixas de usuários;
  - Saúde e segurança;
  - Cumprimento das regulamentações governamentais.
- **Tecnologia:** desenvolver a tecnologia apropriada e integrada ao ERP requer cuidado com a integração com outros sistemas (Middleware), uma vez que, em alguns casos,

pode-se chegar a uma situação de existir tantos sistemas paralelos, que o ambiente volte a apresentar os mesmos problemas que o ERP deveria solucionar, como inconsistência entre os dados, não obtenção das informações em tempo real com necessidade de atualizações não automáticas, etc. Deve-se levar em conta também que os sistemas ERP evoluem e assim podem passar a incorporar novas funções e informações que na versão anterior não eram necessárias, criando a possibilidade de que, a cada atualização de versão do sistema ERP a empresa tenha que analisar e em alguns casos reescrever, as aplicações de integração entre os sistemas. Serão abordadas mais adiante algumas das principais tecnologias de integração com o ERP, baseados em um ERP de mercado SAP R/3 e suas tecnologias envolvidas.

- **Funcionários despreparados**: Um programa de treinamento intensivo é necessário para que as pessoas entendam como utilizar o novo sistema, o que passará a ser esperado delas e como suas atitudes afetarão a organização como um todo. Sem treinamento adequado, os empregados da organização não estarão aptos a utilizar o novo sistema.

Outro problema existente durante a implementação é que, à medida que vai se aproximando o tempo determinado para o fim do projeto, a pressão aumenta e há o risco de profissionais abandonarem o processo ao meio. Para evitar isso é preciso haver um plano de contingências.

Existem alguns pontos que são igualmente importantes e que influenciam na implementação do ERP, devendo ser considerados ao adotar o sistema. Merecem destaque:

- **Alta rotatividade**: a alta rotatividade de pessoas envolvidas na implementação dificulta algumas etapas do andamento do projeto, chegando a ser necessário em um determinado ponto de uma reciclagem completa dos usuários na utilização da ferramenta e a contratação de consultores externos para oferecer o treinamento.
- **Sobrecarga de funções por parte dos usuários**: em algumas áreas os usuários ficam sobrecarregados com o novo sistema, pois antes mesmo de completar sua implementação já ocorre corte de pessoal. Isto dificulta os testes, já que as pessoas que

permanecem ficam encarregadas de fazer o seu trabalho normal e o das pessoas cortadas.

- **Falta de capacitação dos funcionários**: alguns funcionários da empresa podem apresentar limitações e não conseguir se enquadrar na nova realidade trazida com o sistema ERP. Com as atualizações de versões, também é gerado um grande problema devido à necessidade de readaptação dos funcionários às novas rotinas do sistema, bem como toda a readaptação dos módulos específicos que foram desenvolvidos para atender às necessidades de adaptações do sistema.
- **Vínculo com a empresa fornecedora do ERP**: cria-se um vínculo de dependência com a empresa fornecedora do ERP. A Empresa fica obrigada a custear o contrato de manutenção e taxas de atualização do software.
- **Necessidade constante de manutenção**: com a dinâmica dos negócios, a Empresa continua precisando de novos desenvolvimentos no sistema ERP, uma vez que nem todos os recursos necessários são contemplados pelo sistema original. Isso gera a necessidade de profissionais caros, qualificados tanto nos negócios, como na tecnologia utilizada pelo ERP. Com as atualizações de versões no próprio produto gera-se também um grande problema devido à necessidade de novos treinamentos dos funcionários às novas rotinas do sistema, bem como a de revisão de todos os módulos específicos criados pela própria empresa e que foram desenvolvidos para atender às necessidades de adaptações do sistema.

De acordo com Corrêa, H.L. (1999), ao tomar a decisão da utilização de sistemas ERP, as empresas esperam obter diversos benefícios. Entre os benefícios apontados pelas empresas fornecedoras estão: a integração, o incremento das possibilidades de controle sobre os processos da empresa, a atualização tecnológica, a redução de custos de informática e o acesso a informações de qualidade em tempo real para a tomada de decisões sobre toda a cadeia produtiva. Entretanto há também problemas a considerar. A seguir, uma síntese que relaciona dificuldades e benefícios às características desses sistemas que são:

- **São pacotes comerciais:** Para o sistema ser um pacote comercial ele deve ser fornecido e suportado por uma empresa através de meios comerciais ao qual já se encontram disponíveis no mercado empresarial;
- **Usam modelos de processos:** A empresa fornecedora utiliza padrões, certificados ou não, para o processo de implementação e manutenção do sistema;
- **São sistemas integrados:** O sistema trabalha de forma integrada para suas funcionalidades e características;
- **Usam banco de dados coorporativos:** O sistema trabalha sobre um único banco de dados coorporativo;
- **Possuem grande abrangência funcional:** O sistema possui diversos módulos funcionais nativos atendendo aos diversos departamentos da empresa.

1.3.2. Fatores críticos para o sucesso da implementação

Para a implementação de ERP dar certo se deve considerar alguns fatores críticos de sucesso:

- **Envolvimento da alta direção:** – Segundo Bancroft et al. (1998) para se obter o apoio da alta administração é necessário, antes de tudo, justificar o custo/benefício da implantação do sistema ERP, estar ciente do investimento e conseqüentemente comprometido a se envolver com todos os processos. Os benefícios incluem redução de trabalho redundante e melhoria na base de dados para estimativas.
- **Equipe de projeto com dedicação em tempo integral e capacitado em ERP:** Um dos problemas enfrentados no início das implantações de sistemas ERP é a escassez de pessoal especializado. A baixa disponibilidade pode fazer com que o cronograma da implantação se estenda muito além do prazo. Bancroft et al. (1998) sugere alguns passos para esse planejamento, entre os quais estão a definição do líder do projeto, a formação do comitê executivo, a definição do plano geral de implementação e a estruturação das equipes do projeto.
- **Compromisso de que a organização aceitará a mudança:** Bingi; Sharma; Godla (1999) afirmam que “a implementação de sistemas causam mudanças maciças nas

organizações e devem ser cuidadosamente gerenciadas para que os benefícios possam ser obtidos". O comprometimento e a motivação dos recursos humanos são cruciais neste processo de mudança. Além da necessidade de saber utilizar a tecnologia, a apreensão do novo modelo de trabalho e a cooperação com outras equipes são fatores que devem ser trabalhados. O sucesso na condução e assimilação dessas mudanças depende da cultura da organização. Organizações, em que a alta administração não possui a confiança dos empregados, em que a tomada de decisões é centralizada e em que também a moral dos empregados é baixa, tendem a ser resistentes às mudanças encarando qualquer nova implantação como algo passageiro e que não merece dedicação ou esforço pessoal. Desta forma, possuir um plano de comunicação, envolvimento e comprometimento de todos na organização é de extrema relevância.

- **Planejamento adequado, visão e objetivos claros:** o que é esperado do novo sistema deve estar bem definido, uma vez que a percepção dos resultados esperados pela utilização/ exploração do software são inferiores às expectativas e exigem prazos e custos superiores aos inicialmente estimados – o que é uma das principais causas de cancelamento dos projetos de implementação de ERP. Segundo Slater, Derek (1999), comentando a respeito da necessidade de um processo formal e extenso para a seleção do fornecedor de sistemas ERP, afirma que: "empresas compram sistemas que custam milhões de dólares para depois descobrir que não funcionam – ou pelo menos não funcionam bem – para um dos seus principais processos de negócios". Segundo o autor, "uma das razões para isso é que os sistemas ERP estão de tal maneira em alta e a imprensa e consultores insistem tanto em suas possibilidades, que muitas empresas embarcam nessa solução sem fazer o estudo necessário".
- ***Middleware*:** Heinemann F. (2004) afirma que "Os softwares de ERP trabalham muito com integração devido às necessidades dos negócios". Muitos fornecedores de ERP trabalharam com diversas tecnologias. Deve ser muito bem definido como o ERP irá se comunicar com os outros sistemas da organização e quando isso deverá ocorrer.
- **Pós-ERP:** em um grande número de projetos há uma diminuição de eficácia e de capacidade de resposta logo após a implementação. A razão mais comum é o fato da equipe ainda não estar familiarizada com a nova tecnologia, o que pode provocar situações de maior ou menor desconforto. Já a descoberta de que o sistema não atende

a uma determinada atividade não planejada na fase de aquisição do ERP é uma das principais razões de insatisfação das empresas que adquiriram e já implementaram.

1.4. Fases da implementação

Não é somente a implantação de um ERP, entretanto, que irá fazer a empresa levar vantagem sobre as outras, uma vez que a empresa não pode ser considerada eficaz apenas por seus aspectos tecnológicos, ou ainda, mesmo que uma organização implante o mesmo ERP utilizado por suas concorrentes, não necessariamente irá desenvolver a mesma capacidade competitiva delas.

Assim como acontece na maioria das atividades, o planejamento no processo de implementação de um sistema ERP é muito importante, pois as organizações só obtêm o máximo de vantagem ao entenderem que os ERPs são sistemas de suporte aos processos e que, quanto mais aderentes forem os processos ao sistema, maior será o aproveitamento e conseqüentemente levará a melhorias desses processos. Durante o planejamento, se forem discutidas e estabelecidas alternativas para eventuais contratempos, economiza-se tempo e dinheiro.

Em muitos casos, o fator competitivo de uma organização dependerá da maneira como a informação que é gerenciada por esses sistemas será administrada, pois a qualidade da informação a ser gerada pelo sistema dependerá diretamente da qualidade com que foi definida na etapa de implementação.

Assim, pode-se considerar que, para se tornar competitiva no mercado, a empresa, além de utilizar um sistema ERP, necessita ter a informação de como fazer bom uso dos dados armazenados, devendo se planejar muito bem para conduzir a implantação e com maior eficiência do que os seus concorrentes.

Além da etapa de planejamento é necessário ainda que a implementação seja monitorada com atualização freqüente do cronograma estabelecido, identificando eventuais necessidades de ajustes e revisão dos recursos alocados para permitir cumprimento dos objetivos estabelecidos.

É importante também analisar os processos internos da empresa, identificando melhorias e propondo novas soluções, envolvendo a configuração dos parâmetros do sistema para que ele contemple o novo processo. Essa identificação exige um grande conhecimento do sistema e também um grande conhecimento das características do negócio da empresa onde será

implantado o ERP. A complexidade da parametrização depende tanto do sistema (podendo chegar ao nível de se definir quais dados aparecem na tela do computador ou não), quanto da complexidade do processo em si.

Caso a empresa possua características particulares, que não sejam contempladas pelo sistema ERP, o sistema pode ser modificado com soluções específicas para adaptar o sistema aos negócios da empresa.

Após todas essas fases deve ser feita a validação do sistema, o que envolve análises críticas da implantação, confrontando-se o que foi planejado com o que foi executado e verificando se todos os objetivos foram alcançados.

Conseqüências do mau planejamento da implementação:

- Estouro de custos;
- Perda de prazos;
- Deficiências técnicas que prejudicam o desempenho;
- Fracasso na obtenção dos benefícios operados;

O processo de planejamento se compõe de um conjunto de fases, das quais se podem citar as seguintes como sendo as principais:

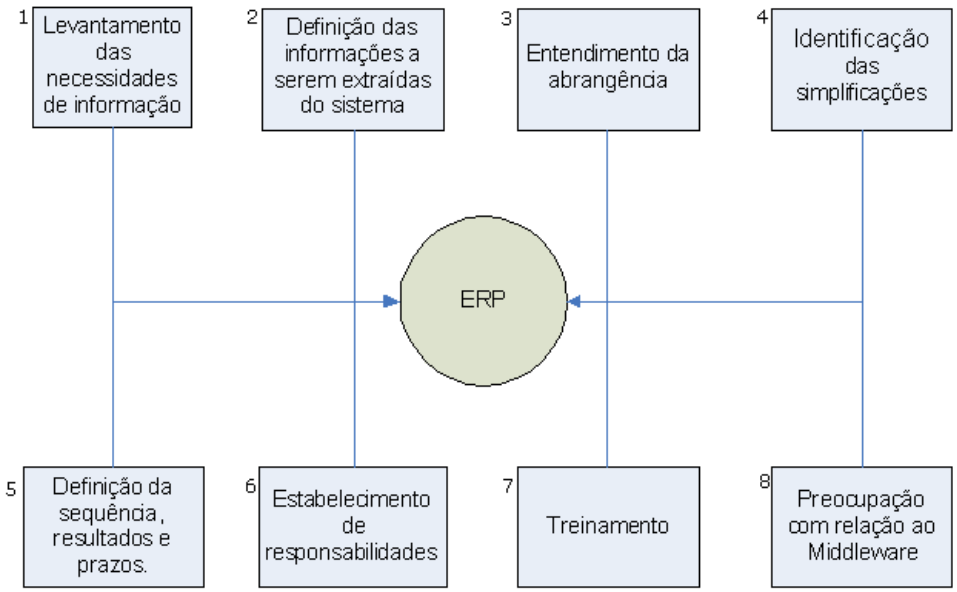

Figura 2 – ERP e suas principais fases – Laudon (2005)

- **Levantamento das necessidades de informação:** deve ser detalhado para permitir a identificação de todos os requisitos relevantes a serem atendidos pelo sistema, tanto operacionais como gerenciais, de maneira a evitar que o sistema seja modificado imediatamente após ou mesmo durante sua implementação. Neste levantamento deve-se também abordar a sua disponibilidade aos usuários. Na prática, um grande número de projetos é entregue com alguma funcionalidade faltando (prometidas, então, para versões posteriores) segundo Keil; Mann; Rai (2000).
- **Definição das informações a serem extraídas do sistema**: preparar os relatórios e consultas que serão colocados à disposição dos usuários e prever o tempo necessário para treinamento, visando uso eficiente do sistema. Submeter as definições aos usuários para aprovação para assegurar o entendimento adequado de suas necessidades.
- **Entendimento da abrangência dos vários módulos do sistema, alternativas para seu uso, necessidades de eventuais ajustes e tempo requerido:** Sistemas ERP permitem normalmente caminhos distintos para atender a uma determinada necessidade: evitar ajustes e modificações, em casos em que o sistema já oferece a solução desejada, depende de conhecimento detalhado do sistema e análise cuidadosa das alternativas disponíveis.
- **Identificação das simplificações e eliminações permitidas pelo sistema:** um sistema integrado permite simplificação e eliminação de tarefas normalmente executadas em sistemas manuais, muitas vezes repetitivos, ou com baixa integração. Identificar as principais oportunidades de simplificação faz parte da etapa de planejamento da implantação.
- **Definição da seqüência de implantação, resultados esperados e prazos envolvidos:** deve ser preparado um cronograma que estabeleça as atividades previstas, quais processos chaves e permita a identificação de pontos críticos e dos resultados bem definidos que marquem o progresso do processo de implementação. É válido ressaltar a importância desta etapa uma vez que é sabido que 30 a 40 por cento de todos os projetos de software fogem do controle; ultrapassam em muito a programação e as projeções de orçamento originais, além de não funcionarem como o especificado Keil, Mann e Rai(2000).

- **Estabelecimento de responsabilidades**: delegar responsabilidades aos usuários envolvidos no atendimento dos analistas encarregados dos aspectos técnicos da implementação e no processo para aprovação dos resultados.
- **Treinamento:** consiste no treinamento dos usuários sobre as funcionalidades do sistema pertinentes às suas atividades operacionais e gerenciais. A participação dos usuários responsáveis dos processos envolvidos é importante na questão do treinamento como um dispositivo de qualificação, desenvolvimento e adequação dos usuários ao novo sistema.
- **Seleção do *Middleware* do ERP:** *Middleware* de acordo com o livro *Web Programming with SAP Web Application Server* (2004) é o software que interliga duas ou mais aplicações permitindo que se comuniquem e transmitam dados uma para a outra.

O *Middleware* pode compor-se de software personalizado escrito pela própria organização ou de um pacote de software. No item 2.0, serão abordados as técnicas de *Middleware* com maiores detalhes.

Deve-se efetuar um levantamento de quais sistemas atuais da organização devem-se comunicar diretamente com o ERP e qual o impacto dessa comunicação no processo atual. Faz-se necessária ainda uma análise de risco, bem como a identificação de potencias, duplicidade de informações e retrabalhos que possam vir a ser gerados, caso não optar pelo *Middleware* neste momento.

A empresa necessitará desenvolver mecanismos de tratamento de dados e informações integrados, considerando o ambiente da organização e traçando suas estratégias baseadas nesses recursos. Grande parte das empresas não podem simplesmente livrarem-se de todos os seus sistemas e criar integração total a partir do zero. Muitas das aplicações de mainframe (legado) que utilizam são essenciais para suas operações diárias e é muito arriscado modificá-las. De qualquer forma, isto não impede estas empresas de tornarem-se ainda mais úteis se suas informações e sua lógica empresarial puderem ser integradas a outras aplicações, segundo W. Noffsinger et al. (1998).

Um modo de integrar diversas aplicações legadas é utilizar um software especial chamado *Middleware* para criar uma interface ou ponte entre dois sistemas diferentes.

2. TECNOLOGIAS *MIDDLEWARE* PARA ERP SAP R/3

2.1. Introdução

As empresas não podem se prender a uma tecnologia. Elas têm que se adaptar para oferecer serviços diferenciados para clientes e parceiros.

Segundo Sampaio, Cleuton (2006), não é mais possível para uma empresa manter uma interface com tecnologias ultrapassadas, prejudicando fornecedores e clientes e utilizando recursos propensos ao desuso.

Muitos fornecedores de ERP trabalham com diversas tecnologias de *Middleware*. Neste tópico serão analisados os principais métodos do mercado, focando principalmente o SAP R/3 (*Systeme, Anwendungen, Produkte in der Datenverarbeitung* ou Sistema, Aplicações e Produtos em Processamento de Dados), visto que este é um produto já consolidado e com a maior base instalada atualmente. O gráfico abaixo (figura 3) provê informações de junho de 2010.

Figura 3 – Participação dos ERPs no mercado

É importante também saber como tal tecnologia se originou. Para isso, será relatado um pouco da origem das redes de computadores - o elemento base nos mecanismos de *Middleware* atuais.

2.2. Histórico

No início da década de 70, universidades e outras instituições que faziam trabalhos relacionados à defesa tiveram permissão para se conectar à ARPANET (L. Roberts (1988)). Em meados de 1975 existiram aproximadamente 100 sites interligados.

Pesquisadores que trabalhavam na ARPANET estudaram como o crescimento da rede alterou o modo de como as pessoas a usavam. No final dos anos 70 a ARPANET tinha crescido tanto que o seu original protocolo de comutação de pacotes, chamado de *Network Control Protocol* (NCP), tornou-se inadequado. Foi aí, então, que a ARPANET começou a usar um novo protocolo chamado TCP/IP (*Transfer Control Protocol/Internet Protocol*), o qual é amplamente utilizado pela internet e redes locais atuais.

Para se utilizar eficazmente uma rede deve-se satisfazer a um determinado grupo de requisitos. Os principais deles são: segurança e velocidade da transmissão da informação.

Diversas empresas propuseram diferentes soluções de protocolos de comunicações. Os protocolos de comunicações especificam como os computadores interagem e trocam mensagens. Cada protocolo (ex. PPP, FTP, HTTP, IP, TCP...) tem diferentes responsabilidades e regras. Conseqüentemente, desenvolver aplicações usando essas regras cria-se laços específicos.

Diferentes tecnologias de integração foram criadas e utilizadas para possibilitarem essa integração podendo-se utilizar diferentes protocolos. A maioria delas faz a troca de dados entre as aplicações de uma forma ou de outra. (Os mais populares são: *Remote Procedure Calls* (RPC), *Distributed Component Object Model* (DCOM), *Remote Method Invocation* (RMI), *Common Object Request Broker Architecture* (CORBA), *Remote Function Call* (RFC), *Electronic Data Interchange* (EDI), *Web Services, Service Oriented Architecture* (SOA), *Enterprise Services Architecture* (ESA), XML, XML para EDI e *text files* (TXT)). Cada uma delas será abordada em tópicos posteriormente.

A dificuldade com a comunicação via Internet ou com qualquer outra rede é que é preciso haver uma estrutura previamente combinada da mensagem. O destinatário tem que conhecer a estrutura da mensagem para ser capaz de recebê-la e tratá-la automaticamente.

Um dos problemas com a comunicação eletrônica entre diferentes sistemas computacionais reside nas diferenças existentes entre formatos de arquivos, esquemas

relacionais, protocolos de troca de dados, entre outros, que tornam o processo de troca de dados complicado.

No desenvolvimento de padrões entre uma implementação rápida e o nível de colaboração atingido entre os atores existe um sinuoso caminho a ser percorrido. Os grupos de desenvolvimento de padrões geralmente têm visões de alto nível do grau de colaboração e querem gastar um tempo considerável para desenvolver uma solução que sirva a todas as partes. As empresas, por sua vez, querem uma implementação rápida para que sejam transmitidos os dados corretos.

Como tendência do mercado, especialistas de TI identificaram a base de tecnologias WEB, que serve de base para toda a internet. Estas bases consistem nas seguintes tecnologias:

- **TCP/IP:** Protocolo universal, entendido por todos os dispositivos de rede.
- **HTML**: Linguagem universal de textos, usada para mapeamento de informação.
- **XML**: Linguagem universal para trabalhar com qualquer tipo de dados.

São padrões abertos e independentes de fornecedor de tecnologia. Todos esses princípios fazem com que o *Web-Services* ganhe vantagem sobre os demais, tornando-o um sistema independente.

O conceito de *Web-Services* nasceu depois de várias tentativas sem sucesso de muitos grupos de analistas, arquitetos e desenvolvedores de todo o mundo para criar mecanismos imediatos de interação entre diferentes sistemas de informação. Baseado nessas necessidades surgiu a arquitetura de software chamada SOA.

Serão tratadas agora as tecnologias descritas acima, de uma forma independente, para que o leitor possa entender e, de acordo com as necessidades, conseguir optar entre uma delas e se aprofundar. Não serão discutidos aspectos específicos de uma determinada tecnologia de integração ou *Middleware*. O foco será dado nas decisões de projeto.

2.3. Diferentes Tipos de Tecnologia *Middleware*

2.3.1. Arquivos de Texto (TXT)

Uma aplicação escreve em um arquivo, o qual outra aplicação o lerá depois (Silva, Muniz (2006)). As aplicações devem ter um acordo quanto a: nome, localização do arquivo, formato do arquivo, tempo em que será escrito e lido e responsável pela exclusão do arquivo.

Os integradores têm a responsabilidade de transformar os arquivos em formatos diferentes. Os arquivos são produzidos a intervalos regulares de acordo com a natureza do negócio. Veja figura 4 como ilustração.

Figura 4 – Modelo de troca de dados via *text files*

2.3.2. *Remote Procedure Calls* (RPC)

Em RPC (Chamada de Procedimento Remoto) (Silva, Muniz (2006)), uma aplicação disponibiliza parte da funcionalidade, a qual é remotamente acessada por outras aplicações, como um procedimento remoto, conforme ilustrado na figura 5. Neste processo tipicamente ocorre comunicação síncrona.

Deve-se desenvolver cada aplicação como um componente com comportamento e dados encapsulados e prover uma interface para permitir que a outra aplicação possa interagir com a aplicação em execução.

RPC são funções de softwares hospedadas em máquinas remotas. Parâmetros para a função são passados através da rede na sua chamada (processo chamado de "*marshalling*"), em que a função remota é executada e retorna um resultado. O resultado, então, é retornado através da rede para o executor. Tudo isso ocorre de uma forma transparente.

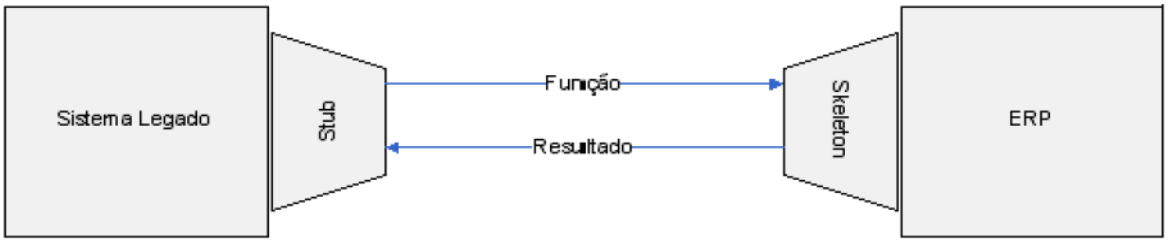

Figura 5 – Utilização do RPC

2.3.3. *Distributed COM* (DCOM)

O *Distributed Component Object Model* (DCOM) ou Modelo de Objetos de Componentes Distribuídos é um modelo de objetos distribuídos proprietário e definido pela empresa Microsoft que concorre com outras soluções na arena da computação distribuída, sendo um protocolo que permite que componentes de software se comuniquem através de uma rede.

O DCOM é uma extensão do *Component Object Model* (COM) ou Modelos de Objetos de Componentes, que é uma estrutura baseada em objetos para desenvolvimento e distribuição de componentes de software.

O COM é uma arquitetura de software orientada a objetos que permite a criação de componentes de software por diferentes fabricantes com uma variedade de linguagens e ferramentas (*C, C++, Java, JScript, VBScript, Delphi, PowerBuilder e MicroFocus COBOL*). Além de especificação de um padrão para criação de componentes interoperáveis, configuráveis e atualizáveis, o COM também é um conjunto de serviços de suporte. Este modelo corresponde a uma evolução unificada e expansível da tecnologia OLE, integrada à família de sistemas operacionais *Windows* e suporta as seguintes características:

- Um padrão independente de linguagem para um executável cliente Win32 carregar e chamar uma DLL Win32;
- Uma forma genérica de um executável controlar outro, substituindo o *Dynamic Data Exchange* (DDE);
- Uma versão 32 bits para substituição dos antigos controles VBX, chamados controles *ActiveX*, que são controles que utilizam a tecnologia COM. Projetados para facilitar

sua distribuição em redes de alta latência, provêm sua integração em navegadores *web* que podem ser incluídos em aplicações ou páginas *web*;

- Uma forma poderosa para aplicativos interagirem com o sistema operacional;
- Um suporte para criação de documentos compostos, configuração de controles, transferência de dados, scripting e outras interações entre aplicações;
- Expansão para acomodar novos protocolos como a interface de acesso para banco de dados OLEDB (OLEDB é uma especificação aberta da Microsoft, designada para acessar todos os tipos de dados).

O padrão de interoperabilidade binária do COM permite que os componentes sejam distribuídos sem código fonte e integrados no ambiente dos clientes. O COM é um protocolo que permite a conexão entre um objeto cliente e outro servidor e depois sai de cena. Após a conexão ser realizada, os objetos podem se comunicar através de um mecanismo chamado de interface. Uma interface COM define o comportamento ou as capacidades de um componente de software com um conjunto de métodos e propriedades – é um contrato que garante a consistência semântica do objeto que a suporta, sendo que cada objeto pode suportar diversas interfaces.

As interfaces são definidas com *Microsoft's Interface Definition Language* (MIDL), uma extensão da linguagem DCE RPC IDL, especificado pela *Open Software Foundation* (OSF). O compilador MIDL gera os seguintes códigos:

- *Proxy* que corresponde ao lado do cliente da API para os objetos que suportam a interface, sendo responsável pela linearização dos parâmetros (*marshaling*);
- *Stub* para decodificar as requisições recebidas dos clientes e disparar o objeto apropriado no servidor. Estes dois códigos garantem a compatibilidade binária entre as diferentes linguagens utilizadas ou, no caso do DCOM, diferentes sistemas operacionais, máquinas e protocolos de rede.

Para eliminar qualquer ambigüidade e colisão de nomes, cada interface recebe uma *Globally Unique Identifier* (GUID). Esta identificação é denominada *Interface Identifier* (IID) e é substituída a cada nova versão da interface. Um objeto pode suportar novas interfaces ou novas versões da mesma, cada uma com sua própria IID. Desta forma, clientes antigos podem invocar

métodos em paralelo com os novos. Além da identificação IID, toda a interface também possui um nome legível, mas neste caso não existe garantia de unicidade.

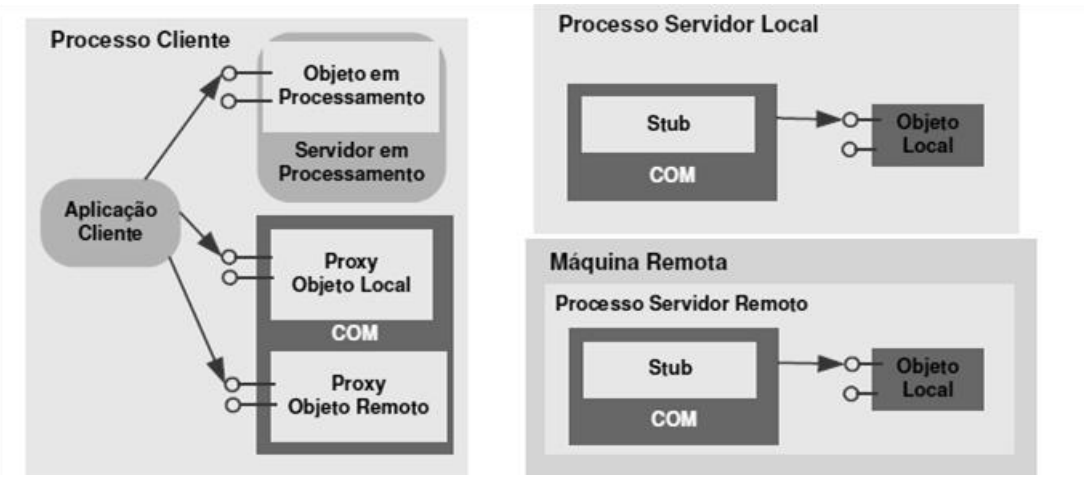

Figura 6 – Empacotamento de objetos DCOM

Uma classe COM é o código fonte que implementa uma ou mais interfaces em qualquer uma das linguagens suportadas.

Uma ou mais classes podem ser empacotadas num servidor no formato de DLL, a qual é carregada pelo processo cliente (*in-process server*), como por exemplo, um controle *ActiveX*. As classes também podem ser empacotadas num executável separado (outprocess server) que pode executar na mesma máquina que o cliente ou numa máquina remota acessada através do DCOM. Na figura 14 é apresentado um cliente COM invocando métodos nas diferentes formas. Esta transparência de empacotamento é uma característica fundamental do COM.

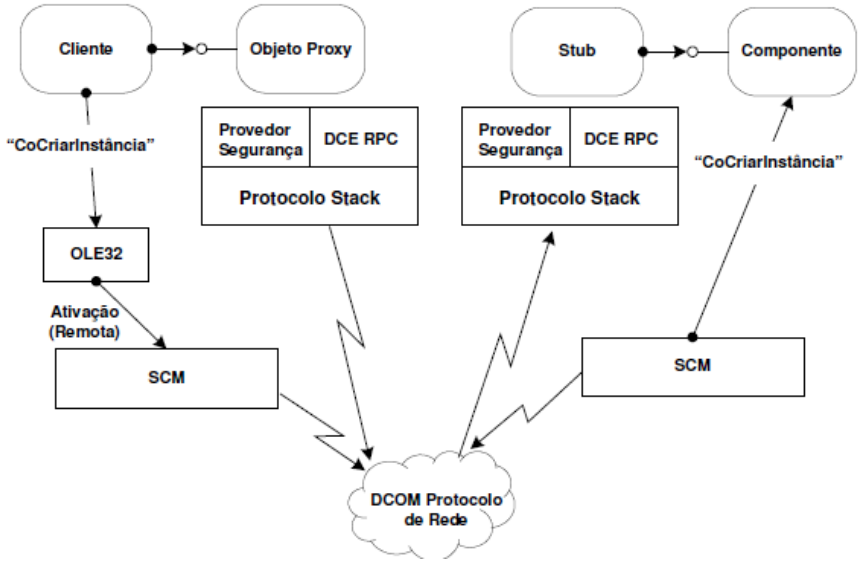

Figura 7 – Arquitetura DCOM

Para alcançar o máximo desempenho os servidores são tipicamente desenvolvidos como aplicações *multithreaded*. Veja a ilustração arquitetônica na figura 7.

Com o DCOM cada invocação pode ser tratada por um *thread* separado e um único objeto pode tratar diversas chamadas de forma concorrente. O custo disto é o aumento de complexidade da aplicação com a utilização de primitivas de sincronização dos *threads* no acesso a recursos compartilhados no processo. O DCOM provê diferentes níveis de segurança de forma transparente ao suportar *Access Control List* (ACL) para os componentes COM. Se o cliente não possui direitos de acesso para acessar ou disparar um componente, ocorre falha na requisição sem qualquer envolvimento do código da classe COM.

Um problema comum em aplicações distribuídas é a necessidade de servidores detectarem se seus clientes ainda estão vivos. O DCOM implementa um componente chamado *Object Exporter* o qual verifica referências a objetos que foram trocados com outras máquinas e mantém a consistência da contagem de referências em caso de perda da conexão. Isto é realizado de forma otimizada através de uma única mensagem keepalive por máquina, independente da quantidade de clientes ou servidores acessados.

2.3.4. *Remote Method Invocation* (RMI)

RMI (*Remote Method Invocation* ou Invocação de Métodos Remotos) (Sun Microsystems) é uma das abordagens da tecnologia Java para prover as funcionalidades de uma plataforma de objetos distribuídos. Esse sistema de objetos distribuídos faz parte do núcleo básico de Java desde a versão JDK 1.1, com sua API especificada através do pacote java.rmi e seus subpacotes.

Através da utilização da arquitetura RMI, é possível que um objeto ativo em uma máquina virtual Java possa interagir com objetos de outras máquinas virtuais Java, independentemente da localização dessas máquinas virtuais.

A arquitetura RMI oferece a transparência de localização através da organização de três camadas entre os objetos cliente e servidor:

- A camada de *stub/skeleton* oferece as interfaces que os objetos da aplicação usam para interagir entre si;
- A camada de referência remota é o *Middleware* entre a camada de stub/skeleton e o protocolo de transporte. É nesta camada que são criadas e gerenciadas as referências remotas aos objetos;
- A camada do protocolo de transporte oferece o protocolo de dados binários que envia as solicitações aos objetos remotos pela rede.

A figura 8 ilustra a organização dessas três camadas em uma aplicação RMI:

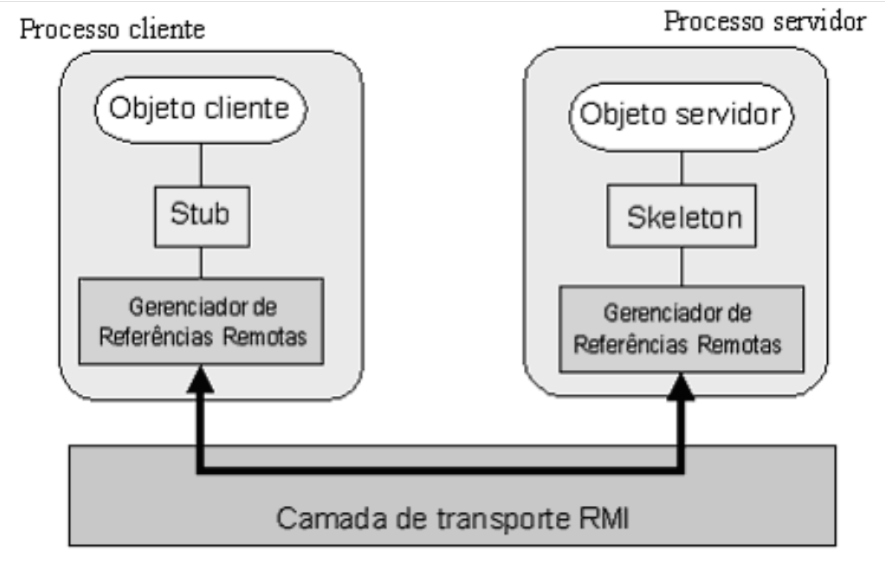

Figura 8 – Arquitetura RMI

No desenvolvimento de uma aplicação cliente-servidor usando Java RMI, como para qualquer plataforma de objetos distribuídos, é essencial que seja definida a interface de serviços a ser oferecida pelo objeto servidor.

Os serviços especificados pela interface RMI deverão ser implementados através de uma classe Java. Nessa implementação dos serviços é preciso indicar que objetos dessa classe possam ser acessados remotamente.

A implementação do serviço se dá através da definição de uma classe que implementa a interface especificada. No entanto, além de implementar a interface especificada, é necessário incluir as funcionalidades para que o objeto dessa classe possa ser acessado remotamente como um servidor.

A implementação da interface remota se dá da mesma forma que para qualquer classe implementando uma interface Java, ou seja, a classe fornece implementação para cada um dos métodos especificados na interface.

Com a interface estabelecida e o serviço implementado, é possível criar as aplicações cliente e servidor RMI.

A execução da aplicação cliente-servidor em RMI requer, além da execução da aplicação cliente e da execução da aplicação servidor, a execução do serviço de registro de RMI. Além também do princípio básico de execução de aplicações RMI, a arquitetura RMI oferece

facilidades para operação com código disponibilizado de forma distribuída e ativação dinâmica, dentre tantos outros serviços distribuídos.

É interessante observar como padrões de programação distribuída, como a fábrica de objetos remotos e o padrão de *callback*, são trabalhados em aplicações RMI.

No procedimento básico para desenvolver uma aplicação distribuída em RMI, cada servidor remoto criado é cadastrado no serviço de registro RMI. Esse tipo de estratégia, no entanto, pode ser ineficiente quando o número de objetos remotos for grande ou mesmo não previsível antes da execução.

A estratégia para lidar com esse tipo de situação é usar o conceito de fábrica de objetos remotos.

2.3.5. *Common Object Request Broker Architecture* (CORBA)

CORBA é a arquitetura padrão criada pelo *Object Management Group* (OMG) para estabelecer e simplificar a troca de dados entre sistemas distribuídos heterogêneos. A sigla CORBA vem de *Common Object Request Broker Architecture.*

Em face à diversidade de hardware e software que encontrada atualmente, CORBA atua de modo que os objetos (componentes dos softwares) possam se comunicar sem que o usuário perceba, ou seja, de forma totalmente transparente, mesmo que para isso seja necessário interoperar com outro software em outro sistema operacional e em outra ferramenta de desenvolvimento. CORBA é um dos modelos mais populares de objetos distribuídos, juntamente com o *Distributed Component Object Model* (DCOM), formato proprietário da Microsoft.

A arquitetura CORBA define o ORB (*Object Request Broker*) como um barramento de objetos que permite aos objetos fazerem requisições e receberem respostas de objetos locais ou remotos. Os clientes não têm conhecimento dos mecanismos de comunicação, ativação ou armazenamento dos objetos servidores.

O ORB deixa que os objetos possam localizar os objetos com os quais pretendem se comunicar e, assim, invocar os serviços entre si. O ORB é muito mais sofisticado que as formas alternativas de *Middleware* cliente-servidor, tais como *Remote Procedure Call* (RPC).

Usando o ORB, um cliente pode invocar um método do objeto servidor, o qual pode estar no mesmo processo, na mesma máquina ou em qualquer site do mundo. O ORB intercepta a chamada e é responsável por encontrar o objeto que pode implementar a requisição, passar os

parâmetros, invocar o método e devolver os resultados. Para o objeto cliente não importa a linguagem ou compilador usado na implementação do servidor ou sua localização, o sistema operacional, o protocolo de transporte dos dados ou o tipo de rede. Para o cliente, basta conhecer a interface que o servidor tornou pública, a qual funciona como um contrato entre cliente e servidor.

As especificações das interfaces dos objetos em CORBA são sempre escritas em *Interface Definition Language* (IDL), uma linguagem neutra que torna os componentes acessíveis por diferentes linguagens, ferramentas, sistemas operacionais e redes. Esta linguagem é puramente declarativa permitindo definir atributos, heranças entre classes, tratamento de exceções e os métodos suportados pela interface, incluindo os parâmetros de entrada, saída e seus tipos. A gramática IDL é um *subset* de C++ com palavras chaves adicionais para suportar os conceitos de distribuição, conforme o exemplo de Schmidt, Douglas (2000) apresentado na figura 9.

```
module Example {
   struct Date {
      unsigned short Day;
      unsigned short Month;
      unsigned short Year;
   }

   interface Ufo {
      readonly attribute unsigned long ID;
      readonly attribute string Name;
       readonly attribute Date FirstContact;
       unsigned long Contacts();
       void RegisterContact( Date dateOfContact );
   }
}
```

Figura 9 – Exemplo de código usando gramática IDL

Ao contrário dos objetos tradicionais, os objetos em sistemas distribuídos possuem uma característica de dualidade: um estado dinâmico, tipicamente alocado em memória volátil e em tempo de execução, e um estado persistente, que não pode ser destruído após o encerramento do programa que os criou e que pode ser usado para reconstruir o estado dinâmico, devendo ser armazenado, portanto, em memória não volátil, seja em sistema de arquivos, seja em banco de dados. A arquitetura CORBA, para prover a persistência, define o *Persistent Object Service* (POS) como sendo responsável por armazenar o estado persistente dos objetos, utilizando quatro

elementos: Objetos Persistentes (*Persistent Object* (POs)), Gerenciador de Objetos Persistentes (*Persistent Objects Manager* (POM)), Serviços de Persistência de Dados (*Persistent Data Services* (PDSs)) e Base de Dados (*Datastores*).

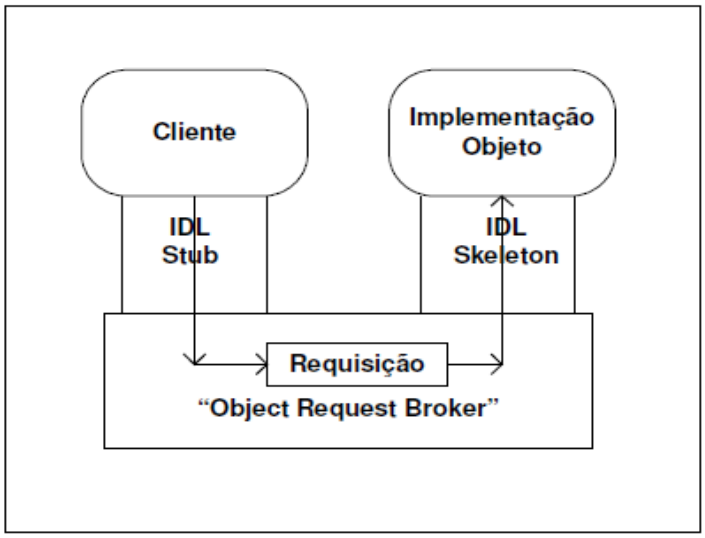

Figura 10 – Esquema do funcionamento da integração aplicações em diferentes linguagens

2.3.6. *Remote Function Call* (RFC)

RFC (Chamada de Função Remota ou Funções de Negócios Compartilhadas) apresentada por Schneider, Neureither (2005) é um protocolo de comunicação proprietário da SAP para execução de rotinas através de conexão TCP/IP ilustrada na figura 19.

Possuem parâmetros de entrada e saída com tratamento de exceção nos quais podem conter complexas estruturas e tipos de dados. Esses tipos de dados podem ser definidos livremente, possibilitando transferir tabelas, estruturas e parâmetros para ambos os lados.

RFC's suportam comunicações síncronas, assíncronas e chamadas transacionais. Também possibilitam consolidar chamadas de módulos de funções como uma transação simples, estendendo a segurança de transações para sistemas remotos.

A RFC é o protocolo preferido para transferências de dados em modo síncrono, especialmente para comunicação entre o ERP e um sistema legado.

As rotinas comuns devem ser implementadas de modo compartilhado, tornando-as serviços para os sistemas.

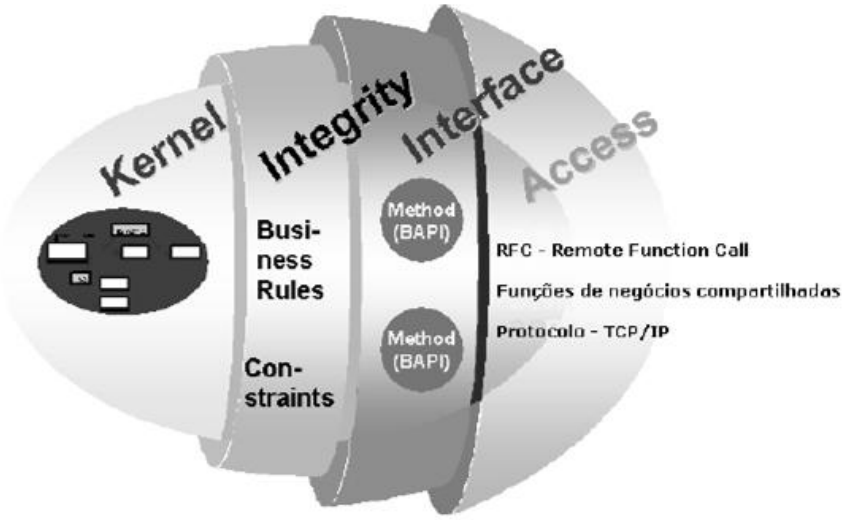

Figura 11 – Diagrama de um RFC como meio de acesso

Onde:

- *Business Rules* ou Regras de negócios que possuem suas rotinas internas;
- *Constraints* ou constantes que são utilizadas para a integridade do dado;
- *Method* (BAPI) – Métodos padrões que são executados pelas funções remotas ao qual possuem seus parâmetros de entrada e saída para efetuar mudanças ou buscas de dados.

2.3.7. *Eletronic Data Interchange* (EDI)

EDI (*Electronic Data Interchange* ou troca eletrônica de dados) é o mesmo que troca estruturada de dados através de uma rede de dados qualquer. No mundo SAP uma forma específica de implementação de EDI é chamada de IDOC (*Intermediate DOCument*).

Segundo Turban et al. (1999), o EDI pode ser definida como o padrão para movimentação eletrônica de documentos de negócio entre as empresas ou dentro das empresas. O EDI usa um formato de dados estruturado que efetua a carga automática de dados, ao qual permite que sejam transformados sem serem reintroduzidos, ou seja, carga direta de informações.

Além disso, Turban et al. (1999) considera que o uso primário do EDI é transferir transações repetitivas de negócios, tais como: encomendas, faturas, aprovações de crédito e

notificações de envio. Isto significa que o EDI hoje, contrariamente ao que muitos acreditam, não implica comunicação em tempo real.

O DISA (*Data Interchange Standards Association*) aponta os seguintes pontos fortes do EDI:

- É um padrão aberto e com fluxos de dados formalizados;
- Garante a troca segura de dados;
- Segura na perspectiva de que diferentes "*checksums*" garantem que os dados enviados sejam confiáveis.

O EDI geralmente fornece três serviços "chave" para trocas de dados aplicação-aaplicação:

- Contexto: através do uso de documentos de negócio identificáveis;
- Semântica: um método para perceber o significado dos dados, usando dicionários de dados, segmentos e descrições dos conjuntos de transações;
- Sintaxe: através dos tipos de dados e regras padronizáveis, permite que os dados sejam empacotados em mensagens.

A EDI tradicional, do ponto de vista de uma empresa conectada à *Internet*, sofre de algumas limitações, o que reduz sua utilidade e eficiência em tais soluções. A seguir, serão apresentados os mais relevantes destas limitações:

- Implementação complexa - um problema associado a soluções EDI tradicional é que cada novo contato é único;
- Falta de flexibilidade - as relações baseadas em EDI ainda dominantes sofrem de fraca adaptação a novas tendências e desenvolvimentos. Isto é uma conseqüência que advém do fato do EDIFACT, que é o processo de reconciliar diferenças entre os diferentes dialetos de EDI e ter-se proposto como o padrão para todas as áreas de negócio;

- Penetração limitada - o preço para estabelecer uma ligação EDI de acordo com um padrão, como o EDIFACT, tem sido elevado;
- Padronização difusa - o esforço do EDIFACT para se fazer um padrão válido para todos os negócios, deixou-o com muitas variantes / subconjuntos;
- Uso limitado nas pequenas e médias empresas - as pequenas e médias empresas não são capazes de suportar os custos do EDI.

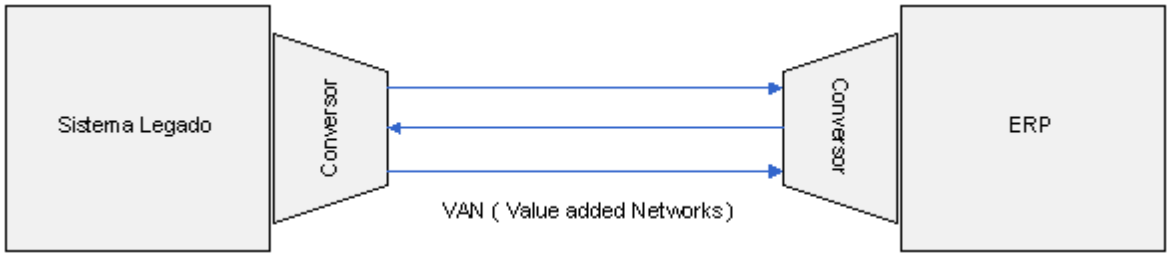

Figura 12 – Diagrama de comunicação usando EDI

2.3.8. XML

XML é a abreviação de *Extensible Markup Language* (Linguagem extensível de formatação). Trata-se de uma linguagem que é considerada uma grande evolução na internet.

O XML é uma especificação técnica, desenvolvida pela W3C (*World Wide Web Consortium* - entidade responsável pela definição da área gráfica da internet), para superar as limitações do HTML, que é o padrão das páginas da *web*.

A linguagem XML é definida como o formato universal para dados estruturados na *web*. Esses dados consistem em tabelas, desenhos, parâmetros de configuração, etc. A linguagem, então, trata de definir regras que permitem escrever esses documentos de forma que sejam adequadamente visíveis ao computador.

A seguinte definição é apresentada pra XML (*Working Group W3C*):

> Um módulo de software solicita serviços de um processador de XML que é usado para ler documentos XML e possibilitar acesso a seu conteúdo e estrutura. Supõe-se que um processador de XML está fazendo seu trabalho em nome de outro módulo, chamado aplicação. Esta especificação descreve o comportamento requerido de um processador de XML nos termos de como os dados de XML devem ser lidos e a informação deve ser fornecida à aplicação.

> Cada documento XML possui uma estrutura lógica e física. Fisicamente, os documentos são compostos de unidades chamadas entidades. Uma entidade pode referenciar outras entidades na criação dos documentos. Logicamente, o documento é composto de declarações, elementos, comentários, caracteres de referência e instruções de processamento, os quais são indicados explicitamente no documento.

Abaixo está um exemplo de XML:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
      <INVCON01>
            <IDOC BEGIN="1">
                  <EDI_DC40 SEGMENT="1">
                        <TABNAM>TBL_JY97</TABNAM>
                              <MANDT>100</MANDT>
                                    <DOCNUM>000000006736722</DOCNUM>
                              <DOCREL>764</DOCREL>
                              <STATUS>21</STATUS>
                              <DIRECT>1</DIRECT>
                              <OUTMOD>2</OUTMOD>
                              <IDOCTYP>INVCTB56</IDOCTYP>
                              <MESTYP>INVCTB</MESTYP>
                              <SNDPOR>SAPPC7</SNDPOR>
                              <SNDPRT>LS</SNDPRT>
                              <SNDPRN>PCTB_250</SNDPRN>
                              <RCVPOR>MAX_PRT</RCVPOR>
                              <RCVPRT>LS</RCVPRT>
                              <RCVPRN>MAXVLE</RCVPRN>
                              <CREDAT>2011-10-15</CREDAT>
                              <CRETIM>14:58:20</CRETIM>
                  </EDI_DC40>
                        <E1ICSL0 SEGMENT="1">
                              <MATNR>M15673452</MATNR>
                              <WERKS>105</WERKS>
                              <DISPO>109</DISPO>
                              <MTART>COMM</MTART>
                              <MATKL>M</MATKL>
                              <DISMM>V1</DISMM>
                              <WAERS>EUR</WAERS>
                              <MEINS>PCI</MEINS>
                              <LABST>0.000</LABST>
                              <LBKUM>15.000</LBKUM>
                        </E1ICSL0>
                        </IDOC>
      </INVCON01>
```

2.3.9. XML para EDI

As possibilidades do XML como uma substituição ou um complemento para o EDI tradicional são significativas ilustradas na figura 13. Algumas das características que fazem do XML uma alternativa atraente para comunicação empresarial eletrônica são:

- O XML é uma tecnologia capaz de executar comunicação empresarial pela *Internet*. E, ao mesmo tempo, as técnicas de comunicação anteriores não são ignoradas pelo XML, por causa da compatibilidade reversa "*backward compatibility*" para transformação de informação estruturada;
- O XML deixa seus utilizadores definirem o seu próprio dicionário usado para executar a troca de dados. Este é um grande passo a frente, comparado com a técnica de EDI tradicional, e impõe também grande exigência no desenvolvimento de padrões empresariais uniformizados. Se não for estabelecido nenhum padrão, o efeito poderá ser caótico;
- Como resposta ao perigo da não padronização, isto é, quando todos desenvolvem o seu próprio "padrão", surgiram fortes iniciativas para desenvolver padrões que satisfizessem as necessidades empresariais. Exemplos são BizTalk abordadas no Microsoft Miztalk Server (2004) e RosettaNet encontrada no site da internet da organização (2006);
- A tendência de XML é um passo promissor em direção à integração da aplicação-para-aplicação de um modo mais barato, mais fácil e mais flexível, comparado com o EDI tradicional. Este é um grande passo para um maior uso do comércio eletrônico *business-to-business* (B2B);
- Como conseqüência da possibilidade de definir os elementos na transação e doesforço de projetos como RosettaNet e BizTalk, o XML tem a possibilidade de ser interpretável por humanos e máquinas. XML pode transferir todos os tipos de dados, por exemplo, uma fatura, um pedido de cuidados médicos, estado de projeto, etc.

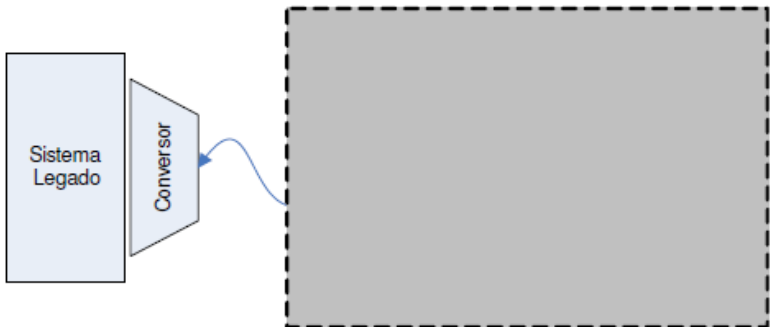

Figura 13 – EDI com XML baseado na WEB

2.3.10. *Web Services*

Uma Web Services é uma aplicação de software identificada por um URI "*Uniform Resource Identifier*", que é a Tecnologia de endereçamento pela qual URLs são criadas. Tecnicamente, http:// e ftp:// são subconjuntos específicos de uma URI., onde as interfaces e relacionamentos são capazes de serem definidos, descritos e localizados por artefatos XML e suportem interações diretas com outras aplicações de software utilizando mensagens baseadas em XML através de protocolos baseados em internet segundo Kaj Van de Loo (2005).

Estes serviços fornecem padrões para comunicação entre diferentes softwares com o objetivo de apresentar a informação dinamicamente ao usuário. Uma arquitetura padrão é necessária quando existe a necessidade de promover interoperabilidade e maior extensão entre as aplicações, assim como permitir que fossem combinadas a fim executar operações mais complexas.

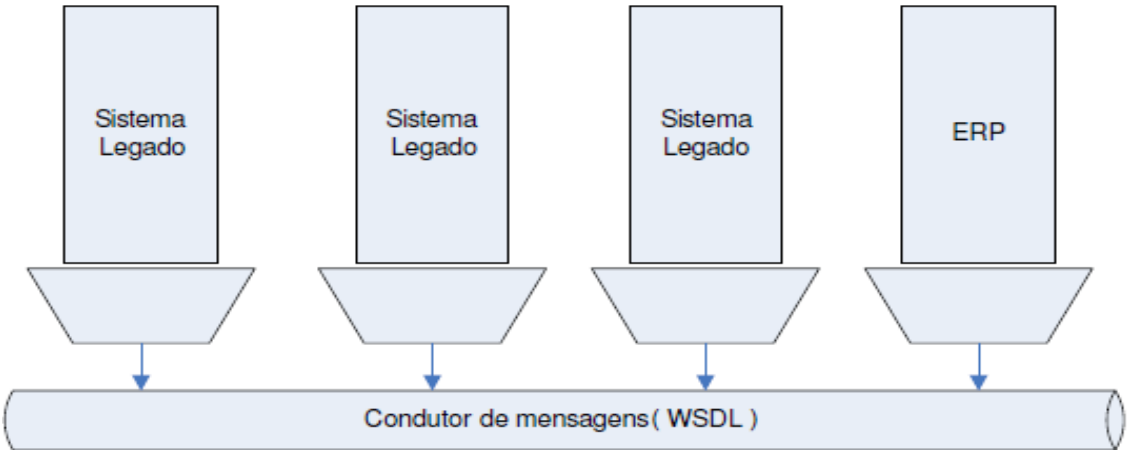

Figura 14 – *Web Services*

Para descrever os serviços de rede como um conjunto de pontos de comunicação operando com mensagens que possuem informações sobre documentos ou procedimentos no condutor, é definido segundo *Working Group W3C*, o WSDL.

WSDL "*Web Services Description Language*" é um formato de XML usado para descrever o que um *Web-services* pode fazer, onde está localizado no condutor de mensagens, e como invocar seus serviços. É baseado em XML e suporta de simples a complexas transações definidas pelas mensagens trocadas. Da perspectiva de interoperabilidade, WSDL define a

interface para o *Web-service*. Se estas interfaces forem bem definidas, as possibilidades de interoperabilidade aumentam.

2.3.11. *Service Oriented Architecture* (SOA)

As aplicações podem ser desenvolvidas como um conjunto de serviços, oferecendo relacionamentos bem definidos a seus usuários potenciais Brown et al. (2002). A descrição de um serviço na arquitetura SOA é essencialmente a descrição de mensagens que são trocadas através de softwares que solicitam e fornecem serviços, permitindo o fluxo de mensagens. Um serviço pode ser considerado como um conjunto de tarefas relacionadas, possuindo descrições que podem ser expressas formalmente em uma ou mais linguagens para descrição destes definidos pela *Working Group W3C*.

SOA permite que os sistemas legados publiquem suas funcionalidades, algumas vezes sem fazer mudanças significativas nos mesmos. As características do SOA, tornam a interoperabilidade mais eficaz e fácil, com o baixo acoplamento, interfaces publicadas, e modelo padrão de comunicação. Construir serviços para os sistemas existentes a fim de obter os benefícios da arquitetura SOA, entretanto, não é automático. De fato, tal migração pode representar uma tarefa complexa da engenharia.

O SOAP (*Simple Object Access Protocol*), segundo *Working Group W3C*, é um protocolo para troca de informações em ambientes descentralizados e distribuídos.

O SOAP fornece um mecanismo simples para a troca de informações estruturadas em um ambiente descentralizado e distribuído utilizando XML. O próprio SOAP não define nenhuma semântica da aplicação tal como um modelo ou uma semântica de programação específica para a execução; em vez disto, define um mecanismo simples para expressar a semântica da aplicação, fornecendo modelos de pacotes modulares e mecanismos para codificar os dados dentro dos módulos. Isto permite que o protocolo SOAP seja utilizado em uma grande variedade de sistemas.

De forma resumida, o protocolo SOAP é dividido em três partes:

- O envelope, que define uma estrutura total para expressar o que está em uma mensagem;

- As regras de codificação, que definem um mecanismo de serialização;
- A representação de RPC do SOAP: define uma convenção que pode ser usada para representar as chamadas e respostas remotas.

Para que haja significado na utilização do *Web-services*, é necessário fornecer informações, além de especificações técnicas do próprio serviço. Partindo dessa premissa, surge o *Universal Description Discovery & Integration* (UDDI). O foco do UDDI, ilustrado na figura 15, é a definição de um conjunto de serviços que suportam a descrição e a localização (1) de negócios, de organizações e outros *Web-services*, (2) se os *Web-services* estão disponíveis e (3) das interfaces pelas quais estes serviços podem ser acessados. Baseado em um conjunto comum de padrões da indústria, que inclui HTTP, XML, e o SOAP, o UDDI fornece uma infraestrutura interoperável para um ambiente de software baseado em *Web-services* para serviços publicados e disponíveis externamente ou para serviços expostos somente dentro da organização.

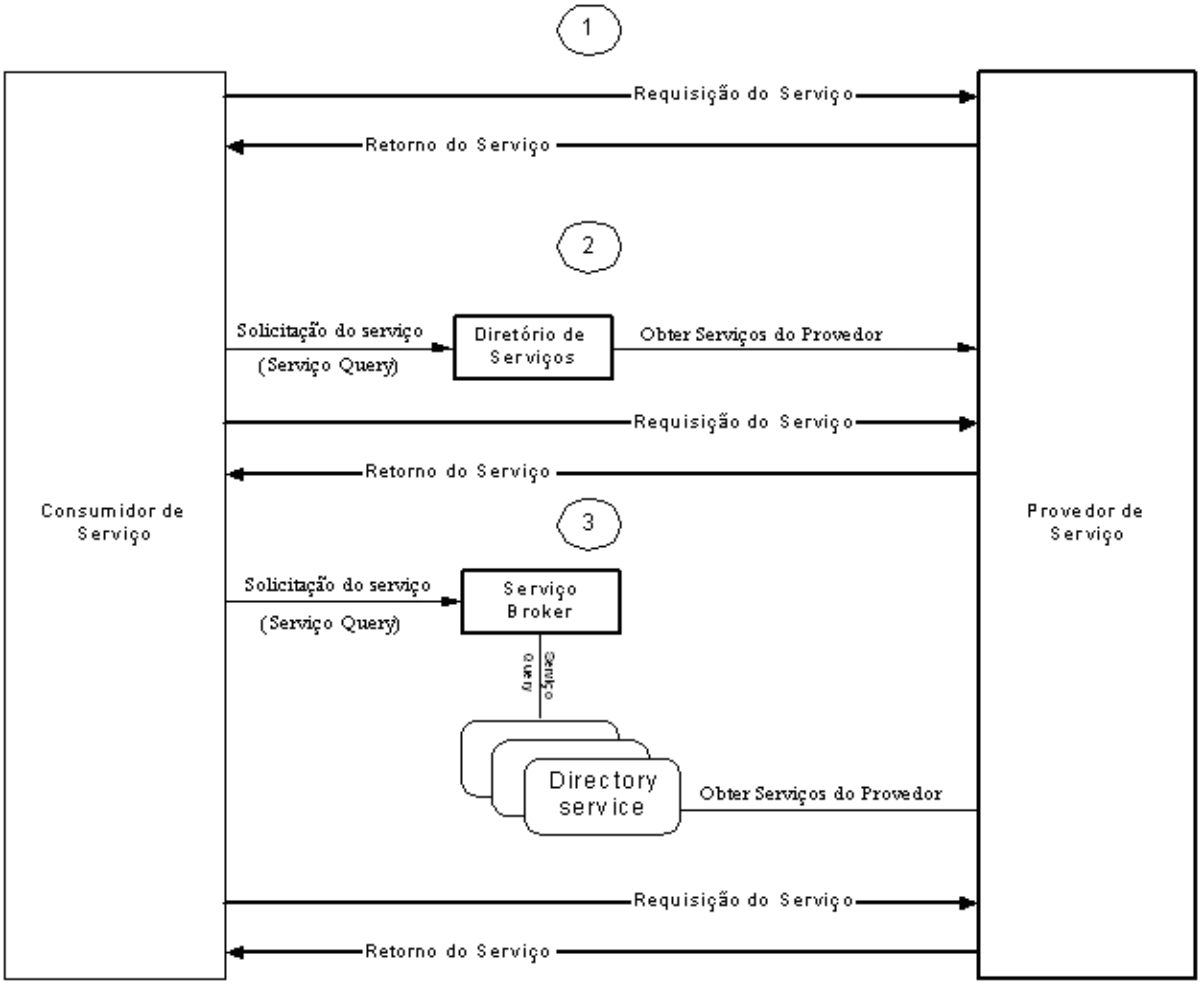

Figura 15 – Diagrama entre consumidor e provedor de serviços

2.3.12. *Enterprise Service Architecture* (ESA)

Em particular, o conceito do ESA ou Arquitetura de serviços empresarial da SAP segundo Kaj Van de Loo (2005) habilita desenvolvedores a fazer integrações entre aplicações heterogêneas e permitir seu uso simples, flexível, e independente da localização dos usuários finais.

O ESA proporciona um modelo para a concepção de soluções de negócio baseadas em serviços que sejam adaptáveis, flexíveis e abertas com um custo mínimo de propriedade ou TCO ("*Total Cost of Ownership*"). Através do ESA as aplicações podem ser desenvolvidas sobre funcionalidades já existentes com o objetivo de aumentar o valor dos sistemas existentes e aumentar a automatização de novos processos de negócio.

O ESA estende a utilização dos *Web-Services* para o ambiente empresarial.

Os serviços empresariais utilizarão a sintaxe e os modelos dos Web Services no sentido de implementarem vários requisitos de negócio como escalabilidade, robustez, segurança e facilidade de uso.

Na arquitetura orientada a serviço (via *Web-services*) do ESA todas as aplicações são tratadas como serviços. A implementação técnica do SAP R/3 é baseada em uma combinação de tecnologias testadas, inovadoras, abertas e padronizadas.

3. COMPARATIVO ENTRE AS INTERFACES

3.1. Considerações para a comparação

Como demonstrado, a partir do momento que a empresa precisa decidir qual dos *Middleware* será utilizado, encontrará no mercado tecnologias com características, siglas e nomes diferentes, mas se considerar algumas observações essenciais, este processo de escolha será mais fácil. Cada uma das tecnologias tem suas características próprias e tem um determinado propósito. Um comparativo será feito agora, comparando as tecnologias abordadas neste trabalho.

Para efetuar esse comparativo serão considerados os fatores abaixo devido a serem os mais importantes no impacto do projeto de desenvolvimento e implementação do *Middleware*:

- Custos e necessidades de recursos (*hardware*, *software* e humanos);
- Tempo de total implementação;
- Risco de adaptações das aplicações atuais para suportar a interface;
- Complexidade para as atividades de Sustaim (suporte ao ambiente produtivo);
- Desempenho;
- Tempo ao qual a interface ficara ativa;
- Segurança;
- Risco de impacto com as interfaces existentes.

A importância desta comparação é fazer com que a equipe do projeto leve em conta as tecnologias apresentadas neste documento devido aos diversos fatores já discutidos. Os elementos aqui apresentados não possuem caráter científico ou comprovadamente eficaz para todas as situações, mas, entretanto, tem-se mostrado útil como mecanismos para a seleção de um *Middleware* em detrimento a outros. Neste ponto é importante esclarecer que se acredita não existirem metodologias científicas consolidadas para tal processo, dado a diversidade de situações que envolvem a implantação de um sistema ERP. Todavia o objetivo deste comparativo é demonstrar que tal procedimento é sempre útil e, sem dúvida, é uma estratégia ou um mecanismo efetivo para a seleção do melhor *Middleware* de ERP.

3.2. Comparativo

A tabela abaixo foi montada levando-se como base à implementação de uma interface entre um ERP da SAP/R3, que deverá disponibilizar os pedidos de compra que foram gerados para um sistema legado. O sistema legado deverá, por sua vez, ler essa informação e retornar para o ERP quais pedidos foram atendidos/efetuados. Este levantamento foi gentilmente cedido pela Empresa EDS (*Electronic Data System*) em novembro de 2006 e é utilizado mundialmente para tomada de decisões nos projetos que efetua em conjunto com a General Motors.

Atividade	TxtFiles	RPC	DCOM	CORBA	RMI	RFC
Custos e Recursos	Baixo	Baixo	Médio	Médio	Médio	Baixo
Tempo Total da Implementação	Baixo	Médio	Médio	Médio	Médio	Baixo
Risco de adaptações das aplicações atuais para suportar interface	Nenhum	Médio	Baixo	Baixo	Médio	Médio
Complexidade de Sustaim	Baixa	Baixa	Média	Média	Baixa	Baixa
Desempenho	Baixa	Alta	Média	Média	Alta	Alta
Segurança	Baixa	Média	Alta	Alta	Média	Média
Risco de impactar interfaces existentes	Baixo	Baixo	Baixo	Baixo	Baixo	Nenhum

Tabela I – Esforço da implementação de um *middleware*

3.3. Prós e contras de cada solução

As tecnologias como RPC, RMI, RFC são similares por trabalharem de forma parecida e são modelos de componentes aceitos pelo mercado para interoperabilidade do tipo "*plug-and-play*", sendo estas a escolha para pequenas interfaces. Seu desenvolvimento e manutenção são relativamente simples e os projetos são de pequenos a médios prazos.

Os arquivos Textos ("*Text Files*") são amplamente utilizados por projetos em que a necessidade não é on-line, ou seja, os processamentos da informação são feitos em horários pré-definidos através de processamentos em batch. Existem diversos formatos de arquivos, por exemplo, CSV, TXT e DAT. Utilização de aquivos textos pode ser considerada como a forma

mais simples de *Middleware* já que envolve baixo custo e esforço para a sua implementação. Normalmente o controle da segurança é feito nos programas que geram os arquivos e não no tráfego dos arquivos em si.

A tecnologia CORBA foi projetada desde o início como uma infraestrutura multiusuário de objetos distribuídos, prevendo interoperabilidade entre diferentes plataformas e linguagens oferecidas por uma variedade de fornecedores. Mas CORBA não especifica detalhes de implementação, não é uma aplicação, um ambiente de programação ou um mecanismo para implementação de interfaces de usuário. Em contraste, OLE/COM foi projetado como uma tecnologia monousuário para gerenciamento de documentos compostos, enfatizando a ligação e combinação de objetos e gerenciamento de eventos, sendo definido para propagar mudanças de estado para múltiplas visualizações de um documento. Para isto o OLE/COM suporta funções para apresentação visual, negociação para apresentação do estado real entre aplicações e mecanismos para entrada de dados do usuário como "*drag-and-drop*" (Arrastar e soltar) e "*cut-and-paste*" (Copiar e colar) (Rosen, Michael; Curtis, David (1998)).

CORBA é um padrão aberto da indústria que facilita a invocação remota de métodos através de interface públicas. No entanto, ainda não é um modelo de componentes aceito pelo mercado para interoperabilidade do tipo *plug-and-play*. No mundo Microsoft, o CORBA seria equivalente ao DCOM como via de comunicação.

Já as tecnologias como SOA, ESA, EDI, XML e *Web-services* são iniciativas independentes, algumas lideradas por grandes empresas como Microsoft, IBM, Oracle e SAP e outras por adeptos da padronização. É muito cedo ainda dizer qual o formato final. Tudo indica, pela experiência anterior, que o padrão a ser estabelecido deve ser um padrão de fato e, portanto, terminar por obrigar empresas a abandonarem quaisquer tentativas de apropriação destes.

Estes padrões tendem a convergir rapidamente e, com isto, o estabelecimento de um padrão de fato. Conseqüentemente, uma massificação deste tipo de *Middleware* irá levar a uma redução dos seus custos de implementação. Esta redução sensível do custo aliado à simplicidade deste *Middleware* tende a fazê-lo de maior aceitação e, portanto, acelerando a sua adoção pelo mercado. Em se confirmando esta tese, poderá num futuro não muito distante ser comprovadamente a maneira mais eficiente e eficaz de se integrar ERP com sistemas legados. O importante, a saber, é que essas tecnologias são as tecnologias em foco hoje, com exceção do EDI, e que ainda estão surgindo novidades. Se por um lado, as implementações normalmente

dependem de um prazo maior, de recursos e de treinamentos, devido ainda serem novas no mercado, por outro, elas vêm se mostrando cada vez mais sólidas e mostrando-se muito eficiente na sua implementação e utilização.

CONCLUSÃO

As implementações de sistemas ERP são complexas e requerem consideráveis recursos de tempo e dinheiro – o que exige uma série de cuidados para que não sejam ultrapassados prazos e orçamentos. Se a implantação não for planejada corretamente, trará mais problemas que benefícios.

Dentre os problemas encontrados em relação à implementação de ERP, incluem-se a resistência à mudança das pessoas que fazem parte da organização, a necessidade de modificação de processos existentes nas organizações e o alto custo associado à implementação de um produto ERP.

Em relação a esses problemas, qualquer que seja a solução indicada, deve-se considerar as características comportamentais da organização ao implementar um ERP adequado.

Tendo em vista que as conseqüências das falhas num projeto de implementação de um ERP são prejudiciais, devido ao grande custo financeiro do projeto e ao grau de esforço exigido durante o projeto, deve-se fazer um bom planejamento do projeto antes de iniciar o processo de implementação.

Se por um lado, entende-se, com o apoio na literatura, que muitos benefícios podem ser obtidos na adoção de um sistema ERP, mesmo que os esforços e custos sejam altos e enfrentem-se grandes desafios. Por outro lado, o constante amadurecimento das soluções ERP, a constante qualificação dos usuários e a constante evolução da tecnologia, tendem cada vez mais a reduzir seus riscos, desafios e custos e aumentar seus benefícios, tornando os sistemas ERP fundamentais para a gestão das empresas modernas.

Dentre as diversas dificuldades na implantação de um ERP, processuais, comportamentais, recursos, organizacionais, culturais, etc., quase todos de difícil mensuração, as tecnológicas são as mais simples e uma boa pratica é a escolha do *Middleware* adequado. Integrar sistemas de grande porte, através de um conjunto de serviços independentes utilizando a arquitetura SOA, mostra-se entre os mais estáveis e com perspectiva de crescimento, como pode ser visto no caso do ESA da SAP, oferecendo relacionamentos bem definidos.

REFERÊNCIAS BIBLIOGRÁFICAS E ACESSOS

SOUZA, Cesar. **Sistemas ERP no Brasil: (Enterprise Resource Planning) : Teoria e Casos.** 1 ed. Ed.Atlas, 2003. 368p.

CORRÊA, H.L. **Planejamento, Programação e Controle da Produção – MRP II / ERP: Conceitos, Uso e Implantação**. 2 ed. Ed.Atlas, 1999. 412p.

CLIFFE, S. **ERP Implementation**. Boston, Harvard Business Review, 1999. DAVENPORT, Thomas. Living with ERP. CIO Magazine. 01/12/1998.

NAH, F.F.H.; LAU, J.L.S.; KUANG, J., **"Critical factors for successful implementation of enterprise systems"**. Business Process Management Journal, Vol. 7 No. 3, p 285-296, 2001.

DAVENPORT, Thomas. **Putting the Enterprise into the Enterprise System**. Boston, Harvard Business Review, Jul/Ago de 1998.

ROBEY, Daniel. **Learning to Implement Enterprise Systems: An Exploratory Study of the Dialects of Change.** MIT Center for Information Systems Research, Georgia, 2000.

SOUZA, Cesar; ZWICKER, Ronaldo. **Ciclo de Vida de Sistemas ERP**. Caderno de Pesquisas em Administração. São Paulo, FEA/USP, v.1, nº 11, p.46-57, 1º trimestre 2000.

SCHNEIDER, Neureither ; SCHLINDWEIN, Schungel: **The ABAP Developer's Guide to Java 2005** – SAP Press

F. Heinemann; C.Rau: **Web Programming with SAP Web Application Server 2004** - SAP Press

DAVENPORT, Thomas. **Mission Critical: realizing the promisse of enterprise system.** Boston, MA: Havard Business School Press, 2000.

SMITH, H. Jeff, KEIL, Mark e DEPLEDGE, Gordon. **"Keeping num as the project goes number"**, Journal of management information System 19, n.2, outono de 2001. KEIL, Mark e ROBEY, Daniel "Blowing the whistle on troubled software projects", Communications of the ACM 44, n.4 abr. 2001.

ALTER, Steve e GINZBERG, Michael. **"Managing uncertainty in MIS implementation",** Sloan Management Review 20, n.1, outono 1978.

FRANZ, Charles e ROBEY, Daniel. **"An investigation of user-led system design: rational and political perspectives"**, Communications of the ACM 27, dez. 1984.

JOSHI, Kailash. **"A Model of users perspective on change: the case of information system technology implementation",** MIS Quarterly 15, n.2, jun. 1991.

KEEN, Peter W. **"Information system and organizational change"**, Communications of the ACM 24, jan. 1981.

Bancroft, Nancy H., Seip, Henning e Sprengel, Andrea (1998), **Implementing SAP R/3: How to introduce a large system into a large organization** (2. edição). Greenwich: Manning.

Slater, Derek (1999). **"An ERP package for you… and you… and even you"**. CIO Magazine, 15/02/99.

Bingi, Prasad, Sharma, Maneesh K. e Godla, Jayanth K. (1999). **"Critical issues affecting na ERP implementation"**. Information Systems Management, 1999, vol 16, no. 13, pp 7-14.

W. Noffsinger, R. Niedbalski, M. Blanks, N. Emmart, **Legacy object modeling speeds software integration**, CACM 41 (12) (1998) 80 -- 89. ACM Press

Keil, Mark, Mann, Joan e Rai, Arun. "**Why software projects escalate: an empirical analysis and test of four theoretical models"**, MIS quarterly 24, n. 4, dez. 2000.

Kenneth C. Laudon, Jane P. Laudon. **"Sistemas de informações Gerenciais"** Administração da empresa digital. 5 Edição, 2005.

Irani, Zahir e Love, Peter E. D. **"The propagation of technology management taxonomies for evaluating investments in information systems"** Journal of Management information Systems 17, n.3, inverno 2000-2001.

L. Roberts. The ARPANET and computer networks. In A. Goldberg, editor, **A History of Personal Workstations**, pages 141-171. Addison-Wesley, Reading, MA, 1988.

IPM - Impacta Pesquisa Periódica de Mercado - http://www.impacta.com.br/ipm/ipm.asp (acesso em 30/10/2011).

Turban E., Mclean, W., Wetherbe, J., **Information Technology for Management: Making Connections for Strategic Advantage**, John Wiley & Sons, Inc., 1999.

W3C Extensible Markup Language (XML) 1.0 (Third Edition); http://www.w3.org/TR/2004/REC-xml-20040204

Kaj van de Loo; Enterprise Services Architecture - SAP UK; 26 April 2005; https://www.sdn.sap.com/irj/sdn/weblogs?blog=/weblogs/

W3C Web Services Description Language (WSDL) 1.1; http://www.w3.org/TR/2001/NOTE-wsdl-20010315

Brown, A; Johnston, S.; & Kelly, K. Using Service-Oriented Architecture and Component-Based Development to Build Web Service Applications. Rational Software Corporation, 2002. http://www-106.ibm.com/developerWorks/rational/library/4860.html.

W3C Web Services Architecture; August,2003; http://www.w3.org/TR/2003/WD-ws-arch-20030808/

W3C Simple Object Access Protocol (SOAP) 1.1; May 2000; http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

CLEVERLEY, James. COM Plus Watch: COM Developer's Eagle Eye on the future of COM and DCOM. [S.l.]: COMDeveloper, 1999. Disponível por WWW em: http://www.comdeveloper.com/complus/ (12 out. 2011).

KRUGLINSKI, David. J. **Inside Visual C++**. 4.ed. Redmond, Washington: Microsoft Press, 1997. 986p. p.555-597. ISBN 1-57231-565-2.

ROY, Mark; EWALD, Alan. Inside DCOM. [S.l.]: Miller Freeman, Inc., Apr. 1997. Disponível por WWW em: http://www.dbmsmag.com/9704d13.html (12 out. 2011).

SZYPERSKI, Clemens. Component **Software: Beyond Object-Oriented Programming**. New York: Addison-Wesley, 1997. 411p. p.132-144; 178-217. ISBN 0-201-17888-5.

HORSTMANN, M.; KIRTLAND, M. DCOM Architecture, 1997. Disponível por WWW em: http://www.microsoft.com/library/backgrnd/html/msdn_dcomarch.htm (13 out. 2011).

SUN MICROSYSTEMS, http://java.sun.com/j2se/1.5.0/docs/guide/rmi/relnotes.html

ORFALI, Robert; HARKEY, Dan; EDWARDS, Jeri. Instant CORBA. New York: Wiley & Sons, 1997. 313p. p.3-28. ISBN 0-471-18333-4.

SAMPAIO, Cleuton. **Soa e Web Services em Java,** Rio de Janeiro: Brasport, 2006. p.01. ISBN 85-7452-267-8.

STEPHEN Mohr, SCOTT Woodgate. Microsoft Biztalk Server 2004 Unleashed. SAMS 2004. Microsoft Corporation.

EDS – Electronic Data System do Brasil Ltda. www.eds.com.