ARQUITETURA DE SISTEMAS CRÍTICOS: UMA DIFÍCIL ESCOLHA[1]

Frederico Medina Vargas[2]
Roberto Werneck do Carmo[3]
Alexander Cramer von Clausbruch[4]
Marco Túlio Duarte Rodriguez[5]

A escolha da arquitetura ideal de um sistema crítico não é uma tarefa simples. Tecnologias, prazos e custos precisam ser conciliados a fim de se obter o melhor resultado. O objetivo deste trabalho é apresentar alguns aspectos inerentes às diferentes tecnologias existentes no mercado para desenvolvimento de sistemas, lançando mão de casos reais vividos pela Chemtech, visando estabelecer alguns critérios que orientem os profissionais de TI nessa escolha. Ao definir a arquitetura mais adequada deve-se levar em conta uma série de aspectos envolvidos no projeto. Parâmetros do sistema tais como criticidade (disponibilidade), número de usuários, distância entre usuários, e outros como o parque de máquinas (servidor e estações) e o orçamento disponível para custear o desenvolvimento devem ser considerados. Um ponto nem sempre lembrado no início do desenvolvimento são os custos envolvidos em manutenção preventiva e evolutiva. Essa avaliação permitirá a definição de uma arquitetura de duas ou mais camadas, da apresentação e ainda do banco de dados que será utilizado. A experiência da Chemtech no desenvolvimento de sistemas críticos em clientes de diferentes áreas mostra que não existe uma fórmula simples para a definição da arquitetura de hardware ou software ideal. Essa definição tem que ser feita necessariamente de acordo com a necessidade de cada cliente.

Palavras-chave

Arquitetura de sistemas; MES; sistemas críticos; manutenção.

---

[1] VIII SEMINÁRIO DE AUTOMAÇÃO DE PROCESSOS
6 a 8 de outubro de 2004 – Belo Horizonte – MG

[2] *Engenheiro de Desenvolvimento, Chemtech*
[3] *Gerente Sênior, Chemtech*
[4] *Gerente de Projetos, Chemtech*
[5] *Gerente Sênior, Chemtech*

A EVOLUÇÃO DOS SISTEMAS DE INFORMAÇÃO

Há pouco tempo atrás, a única tecnologia disponível para desenvolvimento de sistemas industriais críticos era o Mainframe. Tratava-se de uma arquitetura centralizada e com muitas restrições. O avanço da informática permitiu que esses sistemas fossem substituídos por outros bem mais flexíveis que trouxeram consigo mais confiabilidade, menor custo e maior produtividade.

Essas novas tecnologias possibilitaram tratar processos de negócio jamais antes imaginados na era do Mainframe, o que permitiu aos usuários utilizarem seu tempo na análise e não mais na manipulação dos dados.

Esse novo mundo de possibilidades trouxe novos desafios aos profissionais de TI. A complexidade dos sistemas aumentou e o que antes se resumia em gerenciar um único computador projetado para ser confiável (o Mainframe) se tornou um trabalho altamente especializado, complexo e essencial para as grandes corporações, mudando inclusive a visão que algumas empresas tinham da área de TI: de simples geradores de custo a uma equipe estratégica para o crescimento de uma companhia.

A complexidade aparece mais nítida quando se está criando um novo sistema. O número de plataformas e as várias estratégias dentro de cada uma dessas plataformas criam um grande número de arquiteturas possíveis para novos sistemas. A escolha da melhor arquitetura não é um processo trivial e está exigindo cada vez mais conhecimento dos profissionais de TI.

AS APLICAÇÕES INDUSTRIAIS NA ERA DA *WEB*

Os profissionais de TI das grandes corporações devem estar se perguntando: Será que as aplicações *web* vão substituir totalmente as aplicações *desktop*? Ou ainda: Será que esse é o melhor momento para migrar os sistemas atuais para a plataforma *web*?

Essas perguntas não são nada fáceis de serem respondidas, mas já se notam no mercado alguns movimentos nesse sentido. Várias indústrias que já possuíam sistemas críticos na arquitetura cliente-servidor estão migrando seus sistemas ou parte deles para essa nova tecnologia. Algumas outras, que não possuíam sistemas integrados, estão construindo novos sistemas já na plataforma *web*.

Aplicações industriais desenvolvidas em *web* nada mais são do que sistemas de informação que utilizam como interface gráfica páginas da *web* (como os *sites* da internet). Para acessar essas aplicações, basta ter um computador com *browser* interligado à rede corporativa (via intranet ou internet).

Mas para decidir pela adoção ou não desta nova tecnologia, é imprescindível conhecê-la bem, identificando todas as suas restrições e benefícios que ela poderá trazer para a empresa. Como acontece com qualquer outra tecnologia, ela só fará de seu sistema um sucesso se ela for adequada a ele.

A precariedade das ferramentas de desenvolvimento de aplicações *web* sempre foi um problema. Até pouco tempo atrás, as ferramentas existentes no mercado ofereciam poucos recursos, exigindo grande esforço dos programadores para

executarem tarefas simples. Esse problema foi resolvido com o surgimento do Visual Studio .NET e das IDEs de desenvolvimento J2EE. Apesar das facilidades trazidas por essas novas ferramentas, o resultado final obtido não sofreu grandes evoluções, principalmente no que diz respeito ao controle da aplicação. Em aplicações *desktop*, o controle que o programador tem sobre as ações do usuário é bem maior, o que torna o aplicativo menos suscetível a erros de operação. Validações de dados e controle de navegação são bem mais simples de serem feitos em aplicações *desktop* (GROH). Essas dificuldades obviamente impactam os custos de desenvolvimento de aplicações *web*.

Não se deve criar expectativas imaginando que o sistema *web* da sua empresa terá a mesma qualidade gráfica de uma página da internet. Muitas páginas da internet funcionam com boa velocidade somente em computadores com hardware atualizado, o que contraria um dos mais poderosos argumentos utilizados pelos defensores desta tecnologia: a não necessidade de atualização de hardware das máquinas cliente. Os *sites* da internet utilizam *applets* e *plug-ins*, que nada mais são do que programas que rodam localmente nas máquinas clientes. Esses programas necessitam máquinas com boa performance e conexão de alta velocidade. Muitos desses *applets* e *plug-ins* só funcionam com a instalação local de componentes, o que também não é desejável do ponto de vista da manutenção.

Um problema grave, mas que ainda persiste, apesar de já ter sido amenizado, é que aplicações *web* ainda não conseguem competir com o nível de funcionalidades oferecidas pela interface gráfica das aplicações *desktop* tradicionais (GROH). A quantidade de componentes disponíveis no mercado ainda é restrita e os poucos que existem apresentam altos custos.

Um bom exemplo sobre as diferenças funcionais entre uma aplicação *web* e uma aplicação tradicional pode ser obtida comparando-se versões de gerenciadores de correio eletrônico para *desktop* com os *webmails* (apesar dos *webmails* terem evoluído sensivelmente nos últimos anos).

É bom ter em mente que o maior custo trazido por uma aplicação é o tempo que o usuário gasta utilizando-a. Essa situação se agrava ainda mais quando o usuário não sabe como usa-la ou quando o mau uso tem grande impacto sobre o resultado final. Uma aplicação *desktop*, devido à sua qualidade gráfica e controle sobre as ações dos usuários, pode ser mais segura no caso de aplicações críticas (GROH)

Apesar dos problemas relacionados acima, essas aplicações trazem consigo uma grande vantagem: os custos de manutenção, escalabilidade e flexibilidade. Aplicações *web* são ideais para serem utilizadas por grande quantidade de usuários distantes entre si ou onde os custos de distribuição e instalação são altos. Atualizações nesses ambientes são bem mais simples, já que não dependem de intervenção direta na estação cliente (GROH).

No momento da escolha é bom avaliar aquilo que se espera da nova aplicação. Se você está lidando com dados complexos ou onde uma interface amigável traria benefícios para o seu sistema, utilize uma aplicação *desktop* (GROH). Se o seu problema é a distância física entre os usuários ou quantidade, uma aplicação *web* pode ser o melhor caminho.

Provavelmente ainda vai passar algum tempo até que as aplicações *web* substituam completamente as aplicações *desktop* (GROH).

AS NOVAS TECNOLOGIAS

Até pouco tempo atrás, as tecnologias disponíveis para desenvolvimento de páginas em *web* era muito limitada. As ferramentas dispunham de poucos recursos e a construção de simples funcionalidades demandavam tempo ou simplesmente não eram possíveis. Para completar, o resultado final era de baixa qualidade. No intuito reduzir esses problemas, duas tecnologias surgiram no mercado: o J2EE e o .NET.

Antes de optar pela tecnologia a ser utilizada em um projeto, é necessário definir o que se espera dele. Fatores como prazo e custo de desenvolvimento, nível de criticidade e expectativas relativas ao tempo de vida da solução implementada precisam ser considerados. A tendência de se utilizar a melhor tecnologia no menor prazo pode contribuir enormemente para o fracasso do seu projeto. É fundamental compreender os motivos envolvidos nessa escolha antes da decisão por J2EE ou .NET (STOECKERT)

Definidas as expectativas em torno do projeto, o próximo passo é compreender alguns aspectos básicos de cada uma das tecnologias. A primeira coisa que se precisa ter em mente é o que está sendo comparado. O J2EE não é um produto, mas sim uma especificação (STOECKERT). Vários fornecedores implementaram a especificação J2EE em servidores de aplicação criando produtos diferentes. Oracle, Sun, IBM e BEA possuem implementações da especificação J2EE altamente competitivas. Já o .NET é composto por um conjunto de diferentes tecnologias desenvolvidas por um único fornecedor, a Microsoft.

A existência de várias implementações J2EE faz com que exista grande concorrência no fornecimento de produtos que atendem a esta especificação. Existem no mercado ferramentas de vários preços, com vários níveis de recursos, incluindo algumas gratuitas. Para o .NET, o único fornecedor é a Microsoft. O fato de o J2EE possuir vários fornecedores talvez seja o maior argumento em seu favor.

Com relação à performance ambos os produtos têm atendido às expectativas do mercado. Microsoft e fornecedores de implementações J2EE se esforçam para provar a superioridade de seus produtos realizando constantes comparações. Mas o que se observa na prática é que, quando bem implementadas, ambas as soluções apresentam níveis de performance satisfatórios.

Do ponto de vista de arquitetura, J2EE e .NET são semelhantes. Ambos estão baseados no conceito multi-camadas. O objetivo desta estrutura é construir a camada de interface com o usuário separadamente das regras de negócio e do acesso a dados. O .Net tem a vantagem do suporte a múltiplas linguagens. Visual Basic .NET e C# são as duas mais usadas, enquanto que a única linguagem suportada pelo J2EE é o Java. O número de linhas de código gerado por sistemas semelhantes chega a ser 5 vezes maior no J2EE, mas boa parte deste código é gerada automaticamente pelas ferramentas de desenvolvimento (YOUNIS).

Os projetos J2EE, de uma maneira geral, apresentam maiores custos do que os .NET. Os custos com licenciamento de software J2EE são maiores, mas podem ser

amenizado com o uso de ferramentas free (YOUNIS). A pequena quantidade de recursos das IDEs J2EE para o desenvolvimento de interfaces gráficas também é um ponto crítico. Neste quesito, o .NET é bastante superior. Por fim, a disponibilidade de programadores .NET no Brasil é bem maior, o que reduz o custo deste tipo de profissional.

ARQUITETURA EM CAMADAS

A mesma evolução observada na plataforma (*desktop* ou *web*) e na tecnologia de desenvolvimento (J2EE ou .NET) também pode ser observada na arquitetura dos sistemas. Mainframes e arquiteturas cliente-servidor estão sendo rapidamente substituídas pela arquitetura multi-camadas.

O desenvolvimento utilizando arquitetura multi-camadas permite que as aplicações se tornem menos dependentes da arquitetura geral de um sistema e torna a manutenção mais rápida e menos complicada. A arquitetura de sistemas multi-camadas possibilita que a camada de apresentação do software seja independente da camada de controle de regras de negócios e que essa seja independente da camada de armazenamento dos dados. Dessa forma, é possível construir um software inteiro sem se preocupar se o armazenamento de dados será em arquivos simples, xml ou bancos de dados relacionais. Além disso, softwares construídos dessa forma têm manutenção mais simples, pois impedem que uma mudança em uma parte do software seja propagada para todo ele.

Separar a aplicação em várias camadas é bastante vantajoso, mas também tem seu lado ruim: ela aumenta o *overhead* de comunicação entre as camadas. Caso a implementação desses sistemas não seja feita de acordo com os padrões disponíveis na literatura (*design patterns*) eles perdem eficiência, o que, dependo do grau, pode levar o projeto ao fracasso (STOECKERT) A divisão de tarefas entre camadas é um ponto crítico em que os *design patterns* podem ser muito útieis. Esse problema é mais evidente em aplicações J2EE.

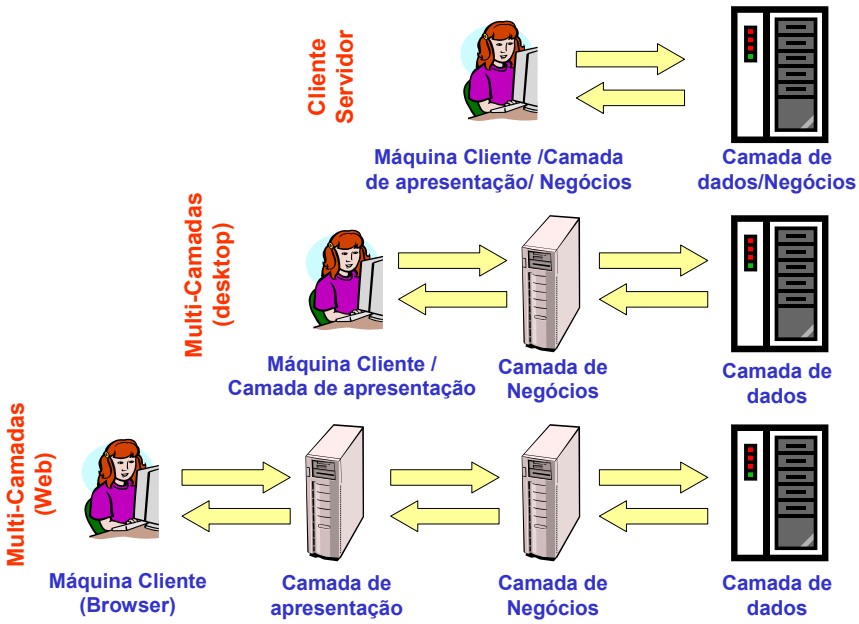

Figura 1: Exemplo de arquiteturas de sistemas multi-camadas

A Figura 1 apresenta um diagrama simplificado dos três tipos de arquiteturas mais comuns: cliente-servidor, três camadas em desktop e três camadas em web. O diagrama representa uma situação extrema onde cada uma das camadas roda em um servidor dedicado, mas nada impede todas as camadas rodem em um único servidor.

ESTUDOS DE CASO

A Chemtech atua o mercado de soluções integradas há alguns anos. Atualmente existem no país vários sistemas críticos por ela desenvolvidos em diferentes épocas e, por esse motivo, utilizando tecnologias distintas. A seguir serão apresentados breves estudos de casos de alguns sistemas críticos desenvolvidos pela Chemtech no segmento de metais e mineração, com foco na arquitetura de software.

Sistema MES da CSN

Com desenvolvimento iniciado em setembro de 1999, o sistema MES da CSN veio para complementar o modelo integrado de negócios que estava sendo implementado na companhia. O sistema foi colocado em operação em julho de 2001.

A arquitetura utilizada para este sistema foi a cliente-servidor, a mais difundida na época. Como ferramentas de desenvolvimento, optou-se pela tecnologia Microsoft, com Visual Basic 6.0 para a aplicação cliente e SQL-Server 7.0 como banco de dados. Mais tarde, o banco de dados foi migrado para SQL-Server 2000 (MIELE et al).

Apesar de atender às expectativas no que diz respeito à performance e confiabilidade, a arquitetura cliente-servidor apresenta algumas restrições, principalmente nos aspectos relacionados à manutenção e escalabilidade (VARGAS et al). Para instalação de novas versões ou correção de erros, freqüentemente é necessário tirar o sistema do ar, o que acaba impactando toda a usina. Em um sistema com mais de cem usuários simultâneos, a operação logística envolvida nestas manutenções é complexa.

Por outro lado, o uso de uma tecnologia consolidada no mercado reduz os custos de desenvolvimento e traz mais agilidade ao processo de evolução do sistema.

O sistema MES Redução da CSN

O sistema MES da CSN incorporava várias funcionalidades da área da redução, mas foi recentemente complementado com funções de logística interna, permitindo desativar alguns sistemas legados baseados em Mainframe, que precisavam ser substituídos (SOTTOMANO et al). Com esse intuito, surgiu o projeto MES da redução. Desenvolvido entre Maio e Dezembro de 2003 ele incorporou, em um só sistema, toda a área da redução, contemplando os sistemas baseados em Mainframe e as funcionalidades da redução já existentes no MES da CSN. Sua concepção foi feita em um cenário tecnológico completamente diferente daquele de 1999. O mercado de sistemas industriais apontava para uma nova tendência: sistemas multi-camadas utilizando plataforma *web*.

Desenvolvido em plataforma *web*, sua arquitetura foi dividia em três camadas: apresentação, negócio e dados, esta última dividida em duas partes: acesso a dados e banco de dados. Para a camada de negócios e apresentação, foi utilizado como ferramenta de desenvolvimento o Visual Studio .NET da Microsoft. Na camada de dados, foi utilizado banco de dados Oracle.

O desenvolvimento de sistemas em *web* mostrou-se bastante desafiador. A baixa oferta de *design patterns* no mercado para a tecnologia utilizada trouxe algumas dificuldades no momento da escolha da melhores estratégias a serem aplicadas. Não só questões relacionadas à arquitetura, mas uma grande fonte de problemas foi o desenvolvimento da interface gráfica, que constantemente esbarrava nas limitações da *web*.

O resultado final foi um sistema de alta confiabilidade, flexível e com manutenção bastante simples.

O sistema MES da SAMARCO

Desenvolvido entre Setembro e Dezembro de 2003, o MES da SAMARCO trouxe um desafio para os engenheiros da Chemtech: o curto prazo de desenvolvimento. Em cerca de quatro meses, um sistema complexo precisava ser especificado, desenvolvido e testado. Diante deste quadro foi consensado que não havia tempo para riscos tecnológicos.

Sendo assim, optou-se por desenvolver um sistema do tipo *desktop*, o que afastaria os riscos da plataforma *web*. A tecnologia escolhida foi o .NET com arquitetura multi-camadas e banco de dados Oracle 9i. De acordo com essa concepção, o sistema teria um executável rodando localmente nas máquinas cliente e um componente de negócios rodando em um servidor remoto (RODRIGUEZ et al).

A opção por um sistema desktop possibilitou o desenvolvimento de uma interface gráfica com muitos recursos visuais que tornaram a navegação e a interpretação dos dados fornecidos pelo sistema mais simples e intuitiva.

CONSIDERAÇÕES FINAIS

Ao se iniciar um projeto de software é de suma importância estabelecer os atributos de qualidade que se deseja do produto final. Custo de manutenção, performance e disponibilidade são três quesitos que sempre devem ser considerados em sistemas críticos para indústria de processo.

Porém, este artigo deixa claro que optar por essa ou aquela tecnologia não é tão simples quanto possa parecer. As Tabelas 1 e 2 sumarizam os tópicos abordados com os dados mais relevantes.

Tabela 1: Sumário geral – Tecnologia (YOUNIS)

Quesito	J2EE	.NET
Definição	Especificação – Várias implementações	Produto Microsoft
Portabilidade	Alta. Suporta vários sistemas operacionais	Suporta apenas Windows
Maturidade da tecnologia	Alta. Chegou ao mercado em 1999.	Baixa. Chegou ao mercado em 2002.
Performance	Artigos pró J2EE concluíram que este é mais rápido	Artigos pró .NET concluíram que este é mais rápido
Suporte/ Produtividade	Baixa.	Reconhecidamente melhor do a melhor IDE J2EE.
Linguagem de programação	JAVA	Suporte multi-linguagens
Desenvolvedores	Baixa disponibilidade no Brasil. Necessita grande capacitação.	Maior disponibilidade de profissionais. Programadores de outras tecnologias Microsoft se adaptam facilmente.
Custos de licenciamento de software	Alto, mas existem muitas ferramentas free	Baixo quando comparado ao J2EE

Tabela 2: Sumário geral – plataforma e arquitetura

Quesito	Cliente/Servidor	Multi-Camadas (Desktop)	Multi-Camadas (Web)
Overhead de comunicação entre camadas	Baixo	Alto, podendo ser amenizado pelo uso de *design patterns*	Alto, podendo ser amenizado pelo uso de *design patterns*
Interface com usuário	Rica	Rica	Limitada
Características do cliente	Boa parte da regra de negócio está implementada no cliente (“cliente gordo”)	Processamento distribuído entre camadas, reduzindo a carga nos clientes (“clientes menos gordos”)	Processamento distribuído entre camadas, reduzindo a carga nos clientes (“clientes magros”)
Controle sobre os usuários	Alto	Alto	Baixo
Complexidade da Administração dos ambientes	Baixo (fica tudo centralizado em um servidor)	Alto (vários servidores se comunicando entre si)	Alto (vários servidores se comunicando entre si)
Características da máquina cliente	Estações robustas (muito processamento no cliente)	Estações robustas (muito processamento no cliente)	As restrições se limitam a uma versão atualizada do browser

Finalmente, é importante dizer que a escolha da plataforma, tecnologia e arquitetura devem ser feitas considerando-se a estratégia da empresa como um todo e não apenas como uma decisão tecnológica. Existem muitos argumentos técnicos que precisam ser analisados, mas o fator preponderante para a decisão deve ser os aspectos envolvendo o negócio. Este artigo deixa claro que custo, flexibilidade e riscos são fatores que podem variar bastante de acordo com a tecnologia utilizada. Deve ser dada atenção ao investimento inicial, que também pode ser elevado.

REFERÊNCIAS BIBLIOGRÁFICAS

GROH, M. Are Windows Apps Dead?. Disponível em: http://advisor.com/doc/09317 Acesso em 11 de Junho de 2004.

LANDGRAVE, T. Web front-ends versus Windows. Disponível em: http://insight.zdnet.co.uk/software/applications/0,39020466,2131357,00.htm Acesso em 24 de Maio de 2004.

STOECKERT, M. J2EE vs. NET – A guide do finding the best Fit for your institution. Disponível em: http://www.press1.de/upload/files/1084966866lwcinfo.pdf Acesso em 24 de Maio de 2004.

YOUNIS, S. J2EE vs. .NET – An Executive look. Disponível em: http://elearning4guruscom.ntitemp.com/only4gurus/techlib/miscellaneous/j2ee_vs_net.pdf Acesso em 24 de Maio de 2004.

RODRIGUEZ, M. T. D., COSTA, L. B. L., PEREIRA, C. S., SILVA, K. C., AMARAL, C. C., SILVA, A. M. MES – Solução para integração entre sistemas de chão-de-fábrica e corporativos. **Anais do VIII Seminário de Automação de Processos**, Associação Brasileira de Metalurgia e Materiais, out. 2004.

RODRIGUEZ, M. T. D., CLAUSBRUCH, A. C., SILVA, A. M., AMARAL, C. C. Avaliando a disponibilidade de sistemas e equipamentos em uma aplicação MES. **Anais do VIII Seminário de Automação de Processos**, Associação Brasileira de Metalurgia e Materiais, out. 2004.

SOTTOMANO, S. A., ABREU JR, M. P., TAVARES, C. A. C., OLIVEIRA, A. S. M., TAVARES R. L., CRUZ, R. S. Tecnologia web multi-camadas em soluções integradas na indústria siderúrgica. **Anais do VIII Seminário de Automação de Processos**, Associação Brasileira de Metalurgia e Materiais, out. 2004.

VARGAS, F. M., ALMEIDA, A. E. R., CARMO, R. W., MANSUR, E., LIMA, M. J. R., RUBIÃO, L. E. G. Mantendo um sistema de missão crítica: A experiência da CSN. **Anais do VII Seminário de Automação de Processos**, Associação Brasileira de Metalurgia e Materiais, out. 2003.

MIELE, M. S. M., FONSECA, R. P, SOTTOMANO, S. A., LIMA, M. J. R, CARMO, R. W., CARDOSO JR, HELCIO e FERREIRA, C. C. N. Caso de sucesso na migração de plataforma de um sistema crítico 24x7 na planta de produção industrial da CSN. **Anais do VII Seminário de Automação de Processos**, Associação Brasileira de Metalurgia e Materiais, out. 2003.

ARCHITECTURE OF MISSION-CRITICAL SYSTEMS: A TOUGH SELECTION[1]

Frederico Medina Vargas[2]
Roberto Werneck do Carmo[3]
Alexander Cramer von Clausbruch[4]
Marco Túlio Duarte Rodriguez[5]

The selection of the ideal architecture for a mission-critical system is not a simple task. Technology, schedule and costs need to be reconciled in order to achieve the best results. The goal of this paper is to present some aspects related to the different technologies that exist today in the software development market, using examples from real case studies. The idea is to discuss some guidelines that can help IT professionals in this task. A series of project issues must be taken into account when defining the architecture of the system. This includes system parameters such as criticality (availability), number of users, distance between users, and others such as the hardware (server and workstations) and the available budget for the development. A point that is often forgotten at the beginning of the development is the cost involved in preventive and adaptative maintenance. The selection process will decide between applications based on two or more layers, and will select the database to be used. Hands-on experience in the development of mission-critical systems for clients in different industries show that there is no simple formula for an ideal hardware or software platform. This definition needs to take into account the requirements of each individual client.

Keywords

maintenance; MES; mission-critical;architecture

---

[1] VII SEMINÁRIO DE AUTOMAÇÃO DE PROCESSOS
October, 6 to 8, 2004 – Belo Horizonte – MG – Brazil
[2] *Engenheiro de Desenvolvimento, Chemtech*
[3] *Gerente Sênior, Chemtech*
[4] *Gerente de Projetos, Chemtech*
[5] *Gerente Sênior, Chemtech*