	Universidade Estadual de Maringá
	Centro de Tecnologia
	Departamento de Informática
	Curso de Especialização em Desenvolvimento de Sistemas para Web

**Sistema de Informações e Solicitações Discentes via WEB
Uma Aplicação na Diretoria de Assuntos Acadêmicos
da Universidade Estadual de Maringá**

Ariovaldo Caldeira Bono

**Maringá - Paraná
Brasil**

Universidade Estadual de Maringá
Centro de Tecnologia
Departamento de Informática

**Sistema de Informações e Solicitações Discentes via WEB
Uma Aplicação na Diretoria de Assuntos Acadêmicos
da Universidade Estadual de Maringá**

Ariovaldo Caldeira Bono

Monografia apresentada ao curso de pós graduação em Desenvolvimento de Sistemas para Web do Centro de Tecnologia da Universidade Estadual de Maringá, como requisito parcial para obtenção do título de Especialista.

Orientador: Prof. Dr. Osvaldo Alves dos Santos

**Maringá- PR
Dezembro- 2003**

RESUMO

Com a crescente demanda de usuários que utilizam a *Internet*, as empresas públicas ou privadas têm investido cada vez mais em estratégias para oferecer produtos e serviços *on-line*. Assim, o objetivo deste trabalho é definir o modelo de um protótipo de Sistema de Informações para *Web* voltado aos alunos de graduação da Universidade Estadual de Maringá (UEM). Para isso, serão apresentados alguns conceitos sobre aspectos importantes da transposição dos dados internos da organização para o ambiente *Web*, principalmente relacionados à segurança e arquiteturas de desenvolvimento e integração. Finalizando, será apresentado um estudo de caso sobre o sistema proposto apresentado suas principais características e funcionalidades.

ABSTRACT

With the increasing demand of users who use the InterNet, the public companies or private they have invested each time more in strategies to offer to products and services on-line. Thus, the objective of this work is to define the model of an archetype of System of Information for Web directed to the graduation pupils of the State University of Maringá (UEM). For this, some concepts on important aspects of the transposition of the internal data of the organization for the Web environment will be presented, mainly related to the security and architectures of development and integration. Finishing, a study of case on the considered system will be presented, its main characteristics and functionalities.

Sumário

1. Introdução.....	2
1.1 Objetivos.....	2
1.1.1 Objetivos Gerais.....	2
1.1.2 Objetivos Específicos.....	2
1.2 Justificativas.....	2
2. Estrutura do Trabalho.....	4
3. Segurança na WEB	5
3.1 Segurança na rede.....	5
3.1.1 Mecanismos de Autenticação Básica.....	6
3.2 Encriptação.....	9
3.3 Firewalls e Proxies.....	17
3.4 Segurança e acesso ao Banco de Dados.....	18
4. Aplicações Multicamadas.....	21
4.1 Histórico das Aplicações baseadas em camadas.....	21
4.1.1 Aplicações baseadas em única camada.....	21
4.1.2 Aplicações baseadas em duas camadas (Cliente/Servidor).....	21
4.1.3 Aplicações Multicamadas.....	22
4.1.4 Modelo de sistema baseado em três camadas.....	25
4.1.5 Integração do modelo em camadas X WEB.....	27
4.2 Arquiteturas de EAI.....	29
4.2.1 IBM Component Broker.....	29
4.2.2 Microsoft Windows DNA 2000.....	29
4.2.3 Enterprise JavaBeans.....	29
5. Estudo de Caso	32
5.1 Definição do Problema.....	32
5.2 Objetivos.....	33
5.3 Arquitetura do Sistema.....	33
5.4 IBM DB2.....	34
5.5 O escopo do Sistema.....	35
5.6 Principais opções para os alunos.....	38
5.7 Aspectos de Segurança anexados ao sistema.....	39
5.8 Fluxo do Sistema.....	41
6. Conclusão.....	43
7. Considerações Finais	44
7.1 Trabalhos Futuros.....	44
7.2 Considerações sobre o estudo de caso.....	44

8. Referências Bibliográficas	46
9. Anexo A – Principais Telas do Sistema	47
9.1 Sistema de Consultas e Solicitações Web	47
9.1.1 Figura 4. Tela de Login.	47
9.1.2 Figura 5. Tela Principal do Sistema	47
9.1.3 Figura 6. Cadastro do Usuário Web	48
9.1.4 Figura 7. Tela de Consulta Notas e Faltas.	48
9.1.5 Figura 8. Tela de Escolha de Solicitação.....	49
9.1.6 Figura 9. Tela de Nova Solicitação.	49
9.1.7 Figura 10. Consulta Notas e Faltas.....	50
9.1.8 Figura 11. Comprovante de Solicitação.	50
9.2 Sistema de Administração Interna	51
9.2.1 Figura 12. Tela Principal do Sistema	51
9.2.2 Figura 13. Tela de Cadastro de Opções de Solicitação.	51
9.2.3 Figura 14. Tela de manutenção de Solicitações.	52
9.2.4 Figura 15. Tela de Pesquisa de Solicitações.....	52
9.2.5 Figura 16. Requerimento Acadêmico.....	53

Lista de Figuras

Figura 1. Representação genérica de um Firewall.....	17
Figura 2. Acesso a banco de dados em três camadas.	25
Figura 3. Modelo de implementação Enterprise JavaBeans.....	31
Figura 4. Fluxo do Sistema.....	41

1. Introdução

Os sistemas de informação para *Web* têm se tornado cada vez mais presentes na sociedade como suporte a serviços tais como transações bancárias, compras eletrônicas e ensino a distância. As empresas de todos os setores estão se reestruturando para atender um novo perfil de cliente, tentando manter a qualidade do atendimento e garantir a sua satisfação e fidelidade. Esta nova filosofia aplica-se tanto a empresa de comércio eletrônico quanto de prestação de serviços e o setor de educação, seja ele público ou privado, não é exceção e a cada dia vem buscando novas formas de atender às necessidades de seus clientes oferecendo serviços *on-line*.

1.1 Objetivos

1.1.1 Objetivos Gerais

O objetivo principal deste trabalho é estudar os principais conceitos envolvidos no desenvolvimento de sistema de informações para WEB. Neste propósito, serão analisados aspectos de interação com Sistemas Gerenciadores de Banco de Dados, arquitetura, segurança e outros conceitos que serão apresentados ao longo deste estudo.

1.1.2 Objetivos Específicos

Este estudo visa propiciar subsídios para o desenvolvimento de um protótipo de Sistema de Informações e Solicitações de Serviços destinados aos alunos de graduação da Universidade Estadual de Maringá (UEM).

O modelo proposto deverá atender a demanda de informações pertinentes à vida acadêmica do aluno e permitir a realização de solicitações de serviços junto à Diretoria de Assuntos Acadêmicos (DAA).

1.2 Justificativas

O Sistema de Controle Acadêmico hoje utilizado pela UEM, apesar de atender bem internamente, é restrito ao espaço físico da Diretoria de Assuntos Acadêmicos (DAA). Isto implica na obrigatoriedade de deslocamento dos alunos até o balcão de atendimento da DAA para solicitação e execução de qualquer tipo de serviço, desde

a confirmação de matrícula até a emissão de um simples horário de aula. Este processo, além de nada prático para os alunos ainda promove a ocorrência de longas filas junto a Secretaria do DAA principalmente em época de confirmação de matrícula.

O sistema proposto visa proporcionar aos alunos de graduação da Universidade Estadual de Maringá um meio via *Internet* para solicitação de serviços diversos junto à DAA, tais como, confirmação de matrícula, solicitação de documentos (histórico escolar, horário de aulas, currículo, etc), evitando assim que o mesmo tenha que se dirigir ao DAA para fazê-lo, além de proporcionar a realização de consultas *on-line* de notas, faltas e outras informações pertinentes ao curso.

Além da agilidade e comodidade nas solicitações, o sistema irá proporcionar uma qualidade de informação muito maior ao acadêmico de graduação.

2. Estrutura do Trabalho

Este trabalho está organizado conforme descrito a seguir:

O terceiro capítulo apresenta uma revisão da literatura com aspectos importantes na segurança em transações na WEB. Estes conceitos incluem Firewalls, Proxies, Protocolo HTTPS e autenticação por chaves públicas e privadas.

No quarto capítulo é dedicado aos aspectos de arquitetura e desenvolvimento em multi-camadas no ambiente de redes Intranet. Este estudo justifica-se pelo fato de que o projeto proposto será composto, além da interface de acesso WEB, de um modelo desenvolvido nesta arquitetura para o processamento e controle das solicitações dos alunos.

No quinto capítulo, será apresentado um estudo de caso do sistema proposto. Será apresentada uma visão geral do protótipo, a arquitetura utilizada e suas principais funcionalidades.

3. Segurança na WEB

Com a crescente utilização dos computadores e das redes de computadores, especialmente da *Internet*, a segurança tornou-se um requisito essencial. Apesar de uma boa parte das informações transferidas na *Internet* ser para acesso público, há operações que requerem algum nível de segurança, acertado entre as partes, como por exemplo transações com número de cartões de crédito, dados de contas bancárias, além de acesso a informações privadas.

Assim, aspectos como privacidade, autenticação, autorização e integridade da informação são todos elementos importantes da estratégia de segurança referente às transações via *Web*.

Aspectos acerca da segurança na transmissão de dados sob o protocolo HTTP são questionáveis, necessitando de mecanismos mais eficientes de segurança para gerenciar a comunicação cliente/servidor *Web*.

Outro problema importante é relativo à segurança nas transações entre o cliente *Web*, o servidor *web* e o SGBD, quando se discute a integração *Web* e Banco de Dados. Neste capítulo serão abordados alguns tópicos sobre estas questões.

3.1 *Segurança na rede*

Como se sabe, a transmissão de dados via protocolo padrão HTTP não é segura, podendo ser interceptada na rede. Em muitas aplicações *Web*, em particular na maioria das aplicações *Web* baseadas em SGBDs, é requerido que as entidades comunicantes, clientes e servidores *Web*, se autenticuem e certifiquem mutuamente de forma a garantir a privacidade das interações.

Face às limitações dos *softwares* correntemente mais utilizados e às limitações dos modelos e protocolos normalizados no tratamento das questões de segurança na *Internet*, tornou-se necessário definir, implementar e avaliar tecnologias alternativas que satisfaçam os requisitos de segurança na *Web*. Para este fim, têm sido criados vários grupos de trabalho, no contexto de empresas, essencialmente dos EUA, com claros interesses comerciais na *Web*, e outros sob a alçada do *Internet Engineering*

Task Force (IETF) [IETF97]. Várias soluções têm surgido, documentadas essencialmente na forma de *Internet-Drafts*, e muitas já implementadas, estando disponíveis comercialmente ou em domínio público. Um esforço adicional ainda precisa ser feito por parte da comunidade interessada na Web, para o estabelecimento e aceitação de padrões abertos que promovam a rápida expansão da integração de aplicações que exijam um mínimo de requisito de segurança na rede.

Afim de orientar o desenvolvimento de novos mecanismos de segurança para a Web, em particular, para o protocolo HTTP, um grupo de trabalho do IETF, estabeleceu de forma clara um conjunto de requisitos a serem satisfeitos pelas soluções eventualmente propostas. Estes requisitos são: privacidade através do uso de mecanismos de criptografia, autenticação de servidores e clientes Web e manutenção da integridade de transações de forma a se determinar se os dados recebidos foram corrompidos ou falseados.

Existem várias propostas, proprietárias ou não, para disponibilizar serviços de segurança na rede, que procuram atender parte destes requisitos. Pode-se citar mecanismos de autenticação básica, servidores com chave pública, *Secure HTTP* (S-HTTP) e *Secure Sockets Layer* (SSL).

3.1.1 Mecanismos de Autenticação Básica

Identificação do Usuário

A identificação do usuário, ou *userid*, deve ser única, isto é, cada usuário deve ter uma identificação própria. É necessário que todos os usuários autorizados tenham um *userid*, quer seja um código de caracteres, cartão inteligente ou outro meio de identificação. Essa unicidade de identificação permite um controle das ações praticadas pelos usuários por meio de logs. No caso de identificação a partir de caracteres, é comum estabelecer regras de composição.

Autenticação do Usuário

Após a identificação do usuário, ocorre sua autenticação, isto é, o sistema confirma se o usuário é ele mesmo. Os sistemas de autenticação são uma combinação de hardware, software e procedimentos que permitem o acesso de usuários aos recursos computacionais. Na autenticação, o usuário apresenta algo que ele sabe ou

possui, podendo até envolver a verificação de características físicas pessoais. A maioria dos sistemas atuais solicita uma senha (algo que só o usuário conhece), ou cartões inteligentes (algo que o usuário possui) ou ainda características físicas (algo intrínseco ao usuário), como o formato da mão, da retina ou do rosto, impressão digital e reconhecimento de voz.

Senhas

Para que os controles de senha funcionem, os usuários devem ter pleno conhecimento das políticas e senha da organização e devem ser orientados a segui-las. Ex-funcionários devem ter suas senhas canceladas.

È recomendável que os usuários sejam orientados a escolher senhas mais seguras, evitando o uso de senhas muito curtas ou muito longas, que os obriguem a escrevê-la em um pedaço de papel para lembrá-la. Utilizar a mesma senha em sistemas distintos é uma prática comum, porém vulnerável, pois quando um invasor descobre a senha pela primeira vez, sua atitude é testá-la em outros sistemas.

Identificação por cartões

O caso mais comum de autenticação "Algo que o usuário possui" é o da chave, que as pessoas usam para entrar em casa ou no carro. Outro dispositivo comum, além do cartão de banco, é o cartão de crédito. Se o cartão de crédito incluir ainda uma foto ou assinatura, ele combina um dispositivo físico de autenticação com um dispositivo biométrico primitivo (a pessoa que compara a assinatura ou atesta sua semelhança com a foto no cartão). Cartões deste tipo possuem ainda uma faixa magnética que contém informações. A vantagem desse mecanismo sobre senhas de acesso é que eles não são trivialmente reproduzíveis e podem conter senhas mais complicadas do que a maioria das pessoas estaria disposta a memorizar. Há, entretanto, algumas desvantagens, além das possibilidades de furto ou falsificação: o uso de dispositivos geralmente requer um hardware customizado em cada máquina, o que pode ser caro.

Um dispositivo análogo a cartões de banco, porém mais seguro, que tem sido crescentemente adotado em redes é o chamado *smart card*, ou cartão inteligente. Aproximadamente do mesmo tamanho dos cartões de crédito, os *smart cards* fazem processamento e têm embutidos memória, uma CPU e uma programação específica.

Quando inserido em um leitor de cartão inteligente, o cartão autentica o usuário através de um protocolo de comunicação com o leitor (em vez de simplesmente mostrar seu conteúdo, como no caso do cartão de crédito comum).

Os diferentes tipos de *smart cards* existentes se diferenciam pela sua forma de funcionamento, preço e nível de segurança que proporcionam. Há desde *smart cards* bem simples, efetivamente muito próximos de serem meros cartões magnéticos, até os mais complexos, de falsificação virtualmente impossível e com menor risco em caso de roubo.

Outros dispositivos mais complexos existem no mercado, mas a popularidade do *smart card*, especialmente do tipo que dispensa leitores, tem crescido rapidamente, pois implementam um nível de segurança bem mais alto do que simples senhas, a um custo bastante razoável (algo em torno de US\$50 por cartão ou menos, dependendo do fabricante).

Sistemas Biométricos

Nos últimos anos muito se tem pesquisado na área de sistemas automáticos de verificação de identidade baseados em características físicas do usuário. Esses estudos têm como objetivo suprir deficiências de segurança de senhas, que podem ser reveladas ou descobertas. Acredita-se que esses sistemas são mais difíceis de serem forjados, porém são infinitamente mais caros.

Os sistemas biométricos automáticos são uma evolução natural dos sistemas manuais de reconhecimento amplamente difundidos há muito tempo, como a análise grafológica de assinaturas, análise de impressões digitais e o reconhecimento da voz. Hoje já existem sistemas ainda mais sofisticados, como os sistemas de análise dos vasos sanguíneos da retina.

Teoricamente, qualquer característica de uma pessoa pode ser usada como base para sua identificação biométrica. Na prática, existem algumas limitações. A tecnologia deve ser capaz de medir determinada característica de tal forma que o indivíduo seja realmente único, distinguindo inclusive gêmeos. É recomendável que não seja invasiva e não comprometa os direitos dos indivíduos. Um dos problemas enfrentados por esses sistemas é sua alta taxa de erro, em função das mudanças das

características dos indivíduos com o passar dos anos, devido a problemas de saúde ou nervosismo, por exemplo.

- Impressões digitais – são características únicas e consistentes. Nos sistemas biométricos que utilizam essa opção, são armazenados de 40 a 60 pontos para verificar uma identidade. O sistema compara a impressão lida com sua base de dados de impressões digitais de pessoas autorizadas.
- Voz – os sistemas de reconhecimento de voz são usados para controle de acesso, porém não são tão confiáveis, em função dos erros causados por ruídos no ambiente e problemas na garganta ou cordas vocais das pessoas.
- Geometria da mão – também é usada em sistemas de controle de acesso, porém essa característica pode ser alterada por aumento ou diminuição do peso ou artrite.
- Configuração da íris e da retina – os sistemas que utilizam essas características se propõem a efetuar identificação mais confiável do que as impressões digitais. Entretanto são sistemas invasivos, pois direcionam feixes de luz aos olhos das pessoas.
- Reconhecimento facial por meio de termograma – o termograma facial é uma imagem tirada com uma câmera infravermelha que mostra os padrões térmicos de uma face. Essa imagem é única e, combinada com algoritmos sofisticados de comparação de diferentes níveis de temperatura distribuídos pela face, constitui-se de uma técnica não invasiva, altamente confiável, não sendo afetada por alterações de saúde, idade ou temperatura do corpo. São armazenados ao todo 19.000 pontos de identificação, podendo distinguir gêmeos idênticos, mesmo no escuro.

3.2 *Encriptação*

Quando se tem uma rede, com dados trafegando entre computadores e usuários, é fundamental que se possa estabelecer a privacidade das informações, para se permitir que a rede possa ser usada para a troca e armazenamento de documentos confidenciais. A criptografia compreende um conjunto de técnicas que, através da codificação de informações, permite que se tenha privacidade com elevados níveis de confiabilidade em uma rede. Além disso, a criptografia também pode ser utilizada em sistemas de autenticação (para a construção de assinaturas e certificados digitais) e para a verificação de integridade de documentos, sendo uma ciência fundamental para a segurança de redes.

A criptografia, evidentemente, não surgiu com a Internet, nem com os computadores. Desde a Antiguidade, técnicas de criptografia já são utilizadas, principalmente com fins militares (área onde o avanço dessas técnicas continua sendo muito grande, até hoje). Um dos mais simples sistemas de criptografia é o chamado Sistema de César, utilizado no Império Romano. Tal sistema consistia simplesmente na rotação de letras do alfabeto. O emissor da mensagem trocava cada uma das letras para, por exemplo, 5 letras adiante, no alfabeto, com isso gerando o texto cifrado. O receptor, de posse desse texto cifrado, fazia o processo inverso, rotacionando cada letra 5 letras para trás, no alfabeto, retornando à mensagem original. Assim, se um general quisesse enviar uma mensagem secreta para um capitão, dizendo "VAMOS INVADIR HOJE", o faria com privacidade enviando a mensagem cifrada "CFRTZ NSCFINX MTOJ". Seu capitão, conhecedor do sistema de criptografia, simplesmente retornaria à mensagem original trocando cada uma das letras pela correspondente, rotacionada cinco letras para trás no alfabeto. Um inimigo que interceptasse a mensagem cifrada, sem conhecer o sistema de encriptação sendo utilizado, não conseguiria entender a mensagem e o segredo do general estaria a salvo.

Embora seja simples, o Sistema de César ressalta os principais conceitos envolvidos em um processo de encriptação. Primeiramente, é interessante se separar a idéia de algoritmo de encriptação da idéia de "chave". No caso em questão, o algoritmo é simplesmente o rotacionamento das letras por letras correspondentes dentro do alfabeto. A chave é o número de letras rotacionadas. Assim como no exemplo foram 5 letras, poderiam ser 3 ou qualquer outro número. Se o capitão não soubesse que eram exatamente cinco letras a ser rotacionadas e que ele tinha que rotacionar as letras para trás, não conseguiria decifrar a mensagem. É o algoritmo de encriptação e a chave que identificam unicamente um processo de encriptação.

Um cenário típico do uso da criptografia, nas redes de computadores, é o de envio de uma mensagem. O emissor da mensagem usa um algoritmo de encriptação associado a uma chave. O produto dessa operação, o texto cifrado, é então enviado para o receptor, que a decifra utilizando também uma chave e um algoritmo de decifração. Algum impostor que tivesse acesso ao meio de comunicação entre o emissor e o receptor não teria como entender a mensagem, mesmo que tivesse

conhecimento do algoritmo utilizado, se não conhecesse a chave que o receptor deveria utilizar para decifrá-la.

Sistemas de criptografia normalmente envolvem o uso de um algoritmo e de pelo menos uma chave secreta. O uso de uma chave simplifica o processo de distribuição de mensagens, uma vez que todo o segredo fica sintetizado apenas nesse valor secreto e não no algoritmo (que é de distribuição mais complexa). O ideal é que os algoritmos sejam padronizados e que as pessoas possam utilizá-los em diferentes programas e ambientes de hardware e sistema operacional, para poderem se comunicar tranqüilamente, sem a necessidade de usar programas específicos, nem de estarem presas a um ou outro fabricante. Atualmente, os sistemas criptográficos mais utilizados (com diversos tamanhos de espaços das chaves associados) são os de chave secreta (simétricos) e os de chaves públicas (assimétricos).

Criptografia de Chaves Simétricas

A criptografia é dita simétrica quando apenas uma chave é utilizada para encriptar e decriptar mensagens. Um exemplo clássico de um tipo de criptografia simétrica é o próprio Sistema de César, onde tanto emissor quanto receptor utilizam a mesma chave ("5", no exemplo anterior) para cifrar e decifrar a mensagem.

Os sistemas de criptografia simétricos têm um problema sério: para que emissor e receptor possam usar uma mesma chave, ela deve ser combinada de alguma forma segura. Como se pode, utilizando um meio inseguro como a Internet, distribuir essa chave secreta para que se possa estabelecer uma sessão segura de comunicação? E se o objetivo for a comunicação com segurança para mais de uma pessoa? Teriam que ser criadas tantas chaves aleatórias quantos fossem os destinatários, o que provavelmente não é realizável, além de se ter que gerenciar uma enorme multiplicidade de chaves. O problema da distribuição e armazenamento de chaves é, infelizmente, comum a todos os algoritmos de encriptação baseados em chaves simétricas e essa é a razão principal pela qual algoritmos simétricos não são utilizados sozinhos nos sistemas atuais na Internet, atuando, na maioria das vezes, em conjunto com um algoritmo assimétrico, usado exatamente para a distribuição de chaves.

Os algoritmos de criptografia simétrica continuam sendo usados pois são bem mais rápidos do que os de criptografia assimétrica (abordados adiante). Atualmente, eles são utilizados em conjunto com algoritmos assimétricos, em protocolos como o SSL (*Secure Sockets Layer*), com tamanhos de chaves que tipicamente variam entre 40 e 128 bits. Alguns exemplos de algoritmos simétricos são o *Data Encryption Standard* (DES), o RC2, o RC5 e o IDEA.

Criptografia de Chaves Assimétricas

A criptografia baseada em sistemas de chaves públicas, ou assimétrica, recebe esse nome pelo fato de se utilizar de duas chaves distintas, uma para cifrar e outra para decifrar mensagens. Os algoritmos de chaves públicas solucionam o problema da distribuição de uma chave secreta, existente nos algoritmos simétricos. Todos os usuários possuem um par de chaves, unicamente relacionadas, sendo uma pública e outra privada. A chave pública é de conhecimento de todos os usuários do sistema, enquanto que a chave privada só é de conhecimento do próprio usuário.

Através de algoritmos adequados, o uso do par de chaves é combinado, de forma a se garantir a privacidade. A idéia é que uma mensagem cifrada com a chave pública de um usuário só pode ser decifrada pela chave privada deste usuário. Como somente o próprio conhece a sua chave privada, só ele consegue ler a mensagem. E, como todos conhecem sua chave pública, todos podem se comunicar com ele de maneira segura e usando a mesma chave, a sua chave pública.

No mesmo exemplo dado anteriormente de envio de mensagem, a criptografia assimétrica também pode ser utilizada. Se o emissor quiser mandar uma mensagem para o receptor, ele pode encriptar a mensagem utilizando a chave pública do receptor, que por sua vez decripta a mensagem com sua chave privada. Como a chave pública do receptor é de conhecimento de todos, não existe o clássico problema de distribuição de chaves: o procedimento é exatamente o mesmo para todas as mensagens enviadas ao receptor, por qualquer remetente. O segredo do algoritmo está todo concentrado na chave privada do receptor, que só ele deve conhecer para que haja realmente privacidade.

Alguns algoritmos de chaves públicas são:

- RSA, que recebe o nome de seus três criadores (Rivest, Shamir e Adleman). É atualmente o mais utilizado, inclusive para assinaturas digitais. Seu tamanho de chave tipicamente varia entre 512 bits (menos seguro e mais rápido) a 2048 bits (mais seguro e bem menos rápido).
- El Gamal e DSS, que somente fazem assinaturas digitais;
- Diffie-Hellman, que permite a distribuição de uma chave privativa para comunicação segura via criptografia simétrica, sendo um tipo de sistema de encriptação híbrido.

Os algoritmos de encriptação assimétrica têm grandes vantagens: possuem alto grau de segurança e podem ser utilizados para a criação de assinaturas digitais, como é examinado a seguir. Entretanto, também têm pelo menos dois grandes problemas, que acabam fazendo com que se continue usando tanto criptografia simétrica como assimétrica. O primeiro é ocasionado pela sua lentidão (tanto na geração do par de chaves, como principalmente no processo de cifragem/decifragem das mensagens) e pode ser resolvido com o uso de sistemas híbridos, que são baseados na encriptação assimétrica para a combinação de uma senha secreta a ser usada para a comunicação da maioria dos dados, através de encriptação simétrica.

O segundo grande problema da encriptação assimétrica é a possibilidade de se publicar chaves públicas falsas, atribuídas a pessoas que não são aquelas que verdadeiramente conhecem as chaves privadas correspondentes. Este tipo de problema é resolvido através do uso de certificados digitais, que são formas de se dar autenticidade a uma chave, como será visto a seguir.

Uma grande pergunta que se faz, em termos de criptografia assimétrica, é: "se o intruso conhece a chave pública, por que razão ele não consegue extrair a chave privada ou até mesmo a mensagem original a partir do texto cifrado?" Na verdade, os algoritmos como o RSA são baseados em um fato matemático interessante: não existe nenhum método fechado de se fatorar o produto de dois números primos (um número que só é divisível por ele mesmo ou por um). Assim, de posse do número 221, se uma pessoa não sabe de antemão que ele é divisível por 17 e 13 (dois números primos), não há maneira de se descobrir isso, a não ser fazendo inúmeras

tentativas. Caso os números escolhidos sejam realmente imensos, o número de tentativas cresce e o esforço computacional necessário para se fazer a fatoração é brutal.

Se um produto de dois números primos (221) fizer parte da chave pública e seus fatores (17 e 13) forem usados na composição da chave privada e, ainda, se os algoritmos de cifragem/decifragem forem tais que estes fatores precisem ser conhecidos para que a mensagem seja decifrada, tem-se um sistema de criptografia baseado em chaves públicas inteiramente funcional, que é baseado no uso de números primos muito grandes, para dificultar a fatoração e ataques de força bruta.

Assinatura Digital

Uma das grandes vantagens de algoritmos de chaves públicas, como o RSA , é que eles podem ser usados para resolver outro problema fundamental de segurança, a autenticação, permitindo a criação de assinaturas digitais. A partir da idéia de chaves públicas e privadas, sempre tendo-se em mente que elas são intrinsecamente relacionadas e que a pública é conhecida por todos, mas a privada somente pelo seu dono, têm-se maneiras de fazer a autenticação. A idéia é relativamente simples: aquele que quer assinar uma mensagem, a codifica com a sua chave privada, que só ele conhece. As outras pessoas, ao decifrarem a mensagem corretamente com a chave pública daquela pessoa, saberão que ela é realmente a autora da mensagem, já que só ela conhece a sua chave privada. Se só o emissor conhece sua chave privada e mais ninguém (e sua chave pública é conhecida de todos), para verificar se foi ele mesmo quem me mandou uma mensagem qualquer, basta que o receptor simplesmente utilize a chave pública do emissor na cifra recebida. Se ela for decifrada com sucesso, sabe-se que foi o emissor quem assinou (pois o par de chaves é único e não poderia existir outra chave privada que encriptasse a mesma mensagem).

Embora tudo possa parecer perfeito, uma vez que se tenha chaves secretas e chaves públicas, ainda há um problema crucial não resolvido: como se pode evitar que sejam distribuídas chaves falsas, isto é, uma chave ser publicada atribuída a uma pessoa, mas sendo, na verdade, de uma outra pessoa, de um impostor? Noutras palavras, de posse de uma chave pública, como um usuário pode se certificar de que o detentor dessa chave é realmente a pessoa (ou instituição) quem se diz a dona da

chave? Como saber se o site que diz ser de um determinado banco, na Internet, e que implementa encriptação através de um sistema que envolve chaves públicas, seguindo o SSL, realmente é o site daquele banco, e não de um impostor, querendo descobrir senhas de clientes do banco? Como saber que é seguro o envio do número de um cartão de crédito para o site de uma livraria, que não se está enviando o cartão para um hacker? Todas essas perguntas têm resposta no conceito de certificação.

O SSL é o principal padrão de segurança utilizado na Web, congregando o uso de diversos dos conceitos de encriptação, com técnicas de encriptação para a autenticação de sites e usuários, para a garantia de privacidade das informações e também para a verificação de integridade.

O uso mais típico do SSL é para a certificação de sites, normalmente para aplicações de comércio eletrônico e *Internet Banking*, nas quais a privacidade e a certeza de que os sites são legítimos é indispensável. O padrão do SSL congrega o uso de vários algoritmos e especificações de encriptação, tratando os problemas de autenticação, integridade e privacidade na Web.

Um certificado digital nada mais é do que um documento assinado digitalmente no qual uma entidade confiável e isenta confirma o dono de uma chave pública, assim permitindo que se confirme totalmente a identidade de uma pessoa ou de um site. O certificado pode ser facilmente obtido a partir do uso de criptografia assimétrica: trata-se de um documento, contendo uma chave pública de alguém, assinado por uma entidade idônea e confiável, através do uso da chave privada desta entidade. Essas entidades, que na *Internet* são empresas, como a Verisign <<http://www.verisign.com/>> e a Certsign <www.certisign.com.br> e em uma *intranet* podem existir representadas por um grupo de pessoas e um servidor, são as chamadas CA (do inglês *Certificate Authority*), ou autoridades certificadoras. A CA é quem emite certificados, que são espécies de carteiras de identidade para os indivíduos e sites na Internet e nas intranets. Falsificar um certificado é virtualmente impossível, pois somente a própria CA conhece sua chave privada e a chave pública das CAs é de amplo conhecimento.

Para se certificar um site, uma empresa deve se dirigir à uma CA e, seguindo as instruções apropriadas, provar, juridicamente (com papéis como seus contratos sociais, direitos de uso de um determinado domínio etc.), que ela é realmente a dona daquele site que deseja certificar. Do ponto de vista técnico, no momento do pedido do certificado, deve ser feita também a geração de um par de chaves pública e privada, para que a chave pública esteja presente dentro do certificado. Após a aprovação jurídica, a CA gerará um certificado (que nada mais é do que um documento assinado digitalmente por ela, contendo a chave pública da empresa que está se certificando), o qual deve ser instalado no servidor Web do site a ser certificado.

A partir daí, o site poderá ser acessado de maneira "segura", com a autenticação, privacidade e integridade. O mecanismo de encriptação utilizado pelo SSL para a privacidade é híbrido, sendo baseado inicialmente no uso da chave pública do servidor (que está presente no certificado) e da sua chave privada (criada no momento do pedido do certificado) para a combinação automática de uma chave secreta. A partir do estabelecimento desta chave secreta, a comunicação é toda encriptada com algoritmos simétricos, que garantem melhor performance.

Para que se possa acessar um site seguro, é necessário que se tenha um browser capaz de entender o padrão do SSL (até para que seja possível realizar todo o ritual da combinação de chave secreta). Quando uma conexão é feita através do uso do SSL, diz-se que o protocolo sendo utilizado é o HTTPS (HTTP seguro) e os *browsers* usam URLs iniciadas por "https://".

Embora o SSL tenha se tornado o padrão "de fato", sendo utilizado amplamente na Web, não foi a primeira iniciativa para o desenvolvimento de comunicação segura através da Web. Anteriormente, ainda em 1994, foi desenvolvido o S-HTTP, que também permitia autenticação, privacidade e verificação de integridade. A grande diferença é que o SSL é uma implementação dessa segurança na camada de transporte, enquanto que o S-HTTP é um protocolo na camada de aplicação que implementa a segurança. O modelo de segurança do SSL é de mais baixo nível e pode ser utilizado não só para o HTTP, mas também para FTP, NNTP e outros protocolos tradicionais de aplicação.

3.3 Firewalls e Proxies

Um dos mais importantes e conhecidos recursos de segurança das redes Internet/intranet é o que se chama de *firewall*. Um *firewall* é um computador que atua como um dispositivo de segurança entre uma rede e outra, tipicamente entre uma rede e o resto da Internet, tendo como função a filtragem de pacotes de dados que passam de uma rede para outra, como uma alfândega entre dois países, determinando o que pode e o que não pode passar entre duas ou mais redes.

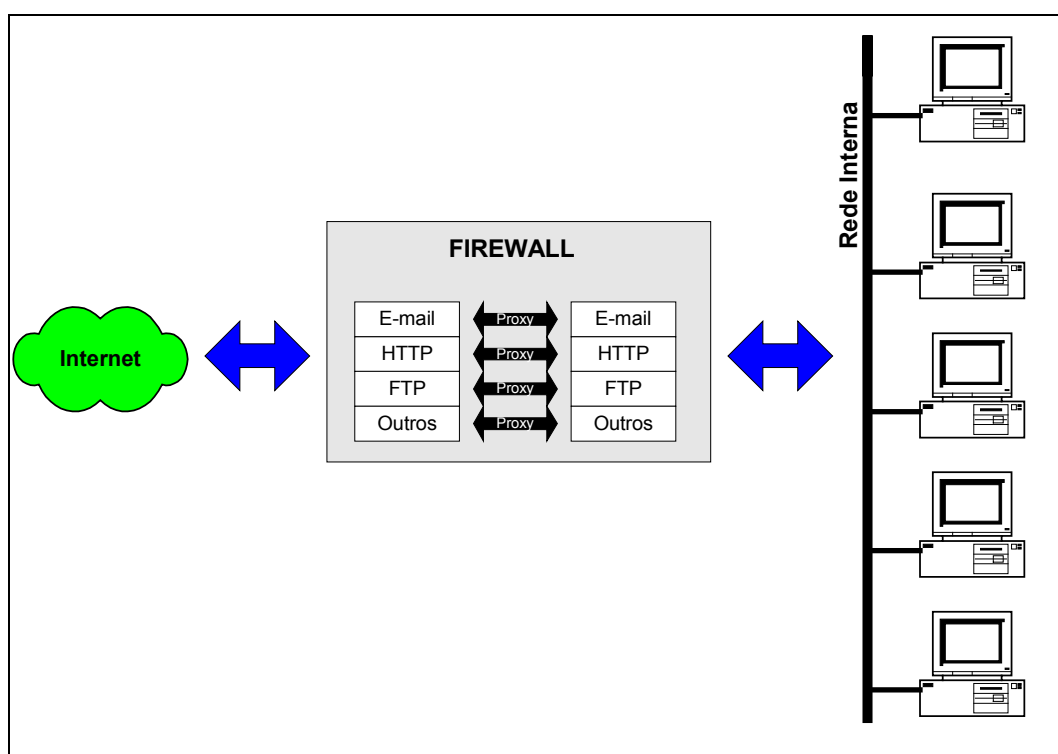


Figura 1. Representação genérica de um Firewall

Os servidores Web podem ser configurados para restringir o acesso aos dados da empresa. Caso se deseje que nenhum usuário externo acesse os dados no servidor Web interno e vice-versa, essa solução é eficiente. No entanto, é muito restritiva. Para garantir um dispositivo que forneça segurança ao acesso e à comunicação entre uma *Intranet* e o mundo exterior existem os chamados *firewalls* [LIMA]. Um *firewall* pode funcionar como servidor Web, permitindo que informações não confidenciais fiquem disponíveis ao público em geral. Os *firewalls* podem prestar com segurança serviços HTTP, FTP, SMTP (correio eletrônico), entre outros. Cada serviço oferecido dessa forma é chamado *proxy*.

Tecnicamente, um *proxy* é um programa que reside no *firewall* e tem acesso aos dois lados da interface, a *Intranet* e a *Internet*. A solicitação de serviço externo feita a partir da empresa como, por exemplo, um cliente Web interno (baseado na *Intranet*) que solicite um pedido a um servidor Web externo, é capturada pelo *proxy* de serviço HTTP e, se as normas do *firewall* permitirem, são transmitidas para a *Internet*. Por outro lado, o tráfego da *Internet* direcionado para a empresa, como no caso de correio eletrônico, é capturado pelo *proxy* de serviço SMTP e, se as normas do *firewall* permitirem, é transmitido para a *Intranet*. Dessa forma, os *firewalls* podem ser extremamente eficientes em barrar a entrada de usuários externos a aplicações e dados no ambiente da corporação.

Outra vantagem dos serviços *firewall/proxy* consiste no fato de que todos os serviços através do *firewall* podem ser registrados no *log* do sistema, se o servidor Web estiver configurado para essa finalidade, o que possibilita auditorias futuras referente às transações realizadas com o mundo externo.

3.4 Segurança e acesso ao Banco de Dados

Outro problema importante quando se discute a integração Web e Banco de Dados é relativo à segurança nas transações entre o cliente Web, o servidor Web e o SGBD. Três pontos se destacam: a segurança na comunicação entre cliente Web e servidor Web, a segurança no servidor Web e a segurança e acesso ao Banco de Dados. A segurança e acesso ao Banco de Dados diz respeito a uma das importantes funcionalidades dos atuais SGBDs. Pode-se ver a Web como um usuário do SGBD como qualquer outro. Nestes termos, os mecanismos de segurança e acesso ao Banco de Dados poderiam funcionar como os já existentes. Em particular, o uso de *visões* proporcionado pelos SGBDs no ambiente de integração pode ser bastante útil. Além disso, alguns fatores devem ser levados em consideração com relação à segurança e acesso ao Banco de Dados, como os destacados a seguir:

1. **Gerenciador de acesso:** a maioria dos SGBDs mantém permissões para acesso ao Banco de Dados (senhas de *login*) e aos objetos do Banco de Dados (*grants*). Caso o contexto da aplicação seja uma *Intranet* pressupõem-se que os usuários sejam limitados e assim o gerenciamento de senhas e *grants* pode ser o habitual.

Mas, se o SGBD puder ser acessado por um número ilimitado de usuários (*Internet*), pode ser inviável controlar o acesso ao Banco de Dados e aos seus objetos individualmente, devido a perda de desempenho dos mecanismos convencionais. É bom lembrar que os mecanismos de acesso aos Banco de Dados tradicionais e seus objetos foram projetados para gerenciar um número limitado de usuários. Para diminuir o problema, pode-se adotar a estratégia de classificar os usuários segundo grupos com as mesmas características de acesso ao Banco de Dados e seus objetos, e dar acesso a eles segundo o grupo em que estão. Perde-se no entanto, as características de acesso individuais de cada usuário.

2. **Acessos simultâneos:** o número de acessos simultâneos ao Banco de Dados é geralmente limitado. Dependendo dos requisitos da aplicação, um usuário pode não conseguir completar sua transação devido a sobrecarga no gerenciador de acessos do SGBD. Alguns *gateways* de integração podem oferecer um *cache* de acessos ao Banco de Dados e os compartilhar entre os usuários, se o SGBD permitir. Assim, um mesmo usuário com características de acesso ao Banco de Dados iguais a de um outro usuário já conectado ao SGBD pode usar este mesmo canal de comunicação já aberto para realizar seus pedidos. Já outros *gateways* requerem conexões dedicadas para cada usuário.
3. **Autorização:** em alguns *gateways* de integração Web e Banco de Dados, a autorização para execução de uma aplicação Web que consulta ou atualiza o Banco de Dados é dada ao *software* usuário que ativou o *gateway* (geralmente o próprio servidor Web). Assim, qualquer cliente Web poderia, por exemplo, disparar a aplicação via servidor Web. Mecanismos de configuração do servidor Web e do SGBD devem estar presentes de forma a contornar este problema.
4. **Recuperação:** um esquema de recuperação em Banco de Dados tradicionais é ativado como consequência de diversos tipos de falhas, como por exemplo, erros lógicos, paradas de sistema e falhas de disco. A extensão destes mecanismos para o ambiente Web não é imediata e os *gateways* devem implementar mecanismos adicionais para tratar de falhas dos componentes da aplicação para que possa, por exemplo, ser comunicado ao usuário o resultado

de uma transação solicitada. Hoje não existe nenhum mecanismo de recuperação de paradas e falhas de clientes e servidores Web. Só estão presentes os mecanismos tradicionais dos SGBDs, que podem ser insuficientes no contexto transacional Web.

4. Aplicações Multicamadas

4.1 *Histórico das Aplicações baseadas em camadas*

4.1.1 Aplicações baseadas em única camada

A arquitetura de aplicações baseadas em única camada é baseada em um ambiente compreensivo onde todos os componentes estão combinados em um simples programa integrado, sendo executado em uma única máquina. Isso é representado pelo ambiente de um *Mainframe* (Sistema computadorizado de grande porte) ou de um microcomputador.

A vantagem da abordagem baseada em única camada, para uma aplicação centralizada em um ambiente singular, é de que nesta aplicação é mais fácil de executarmos o gerenciamento, controle, e implementar segurança.

Há um número significativo de desvantagens associadas ao modelo baseado em única-camada. A medida que aplicações baseadas em única camada estão confinadas a um processador, a escalabilidade pode ser um algo inviável de ser processado. Se a atual máquina torna-se sobrecarregada durante seu uso, o único recurso a ser implementado é um *upgrade* da máquina. Entretanto, aplicações de única camada são extremamente dependente do ambiente operacional baseado no *Hardware*. Essencialmente, as companhias estão limitadas a uma plataforma de *hardware* homogênea. Isso dificulta criticamente a migração e disponibilidade de implantação de novas tecnologias na corporação.

4.1.2 Aplicações baseadas em duas camadas (Cliente/Servidor)

Com o advento dos computadores pessoais, redes locais, banco de dados relacionais, e aplicações *desktop* poderosas, a indústria da computação tem migrado para a arquitetura de sistemas abertos e baseados em cliente/servidor. Profissionais que constituem o corpo de formadores-de-decisão na organização, podem gerar seus próprios relatórios e manipular dados através do uso de ferramentas poderosas em suas estações de trabalho. A arquitetura baseada em duas camadas permite que o usuário realize funções que eles não seriam capazes de fazer antes.

A Arquitetura Cliente/Servidor baseada em duas camadas divide o processamento entre uma estação de trabalho *desktop* e uma máquina servidora da rede. O ambiente Cliente/Servidor mais popular usa um computador pessoal baseado em sistema operacional com ferramentas de desenvolvimento de GUI (Interface gráfica de usuário) e um servidor UNIX, LINUX ou Microsoft Windows 2000/NT Server com um banco de dados relacional.

A Arquitetura Cliente/Servidor baseada em duas camadas provê vantagens significativas sobre o modelo baseado em única camada. As ferramentas de desenvolvimento de GUI permitem um desenvolvimento mais rápido e implementação de aplicações. Durante o funcionamento, todo o processamento de uma aplicação concentra-se significativamente nas estações de trabalho, portanto, os sistemas servidores não precisam ser tão robustos como os clientes. Os sistemas servidores de pequeno porte hoje baseado em computadores Desktop são mais baratos que os sistemas robustos de grande porte. Os sistemas de banco de dados independentes de *hardware* viabilizam a fácil portabilidade entre sistemas aplicativos.

Apesar de todas vantagens, o modelo baseado em duas camadas perde em termos de segurança, escalabilidade, e gerenciamento. O modelo de duas camadas trabalha com extrema eficiência assim como é restrito ao desenvolvimento de aplicações menores.. Quando as aplicações tornam-se mais complexas, em termos de algoritmos de negócios procedidos, número de bancos de dados acessados, ou número de usuários suportados, o modelo baseados em duas camadas começa a perder no conceito. Sem uma segurança rígida provida pelo ambiente centralizado, cada aplicação cliente deve implementar seu próprio processo de segurança.

4.1.3 Aplicações Multicamadas

A arquitetura baseada em multi-camadas prove um ambiente o qual contem todos os benefícios, tanto de um modelo de única camada, assim como de um modelo de duas camadas e também suporta todos os objetivos de uma arquitetura flexível [MULTILAYER, 2000].

Cada camada é representada por um componente, parte lógica de uma aplicação, não ao número de máquina que é usada por essa aplicação. O modelo de aplicação multi-camadas divide uma aplicação em três ou mais partes lógicas. Atualmente o

modelo mais usado no mercado, é um modelo de desenvolvimento em três camadas, as quais são a camada de apresentação, lógica de negócios, e lógica de acesso a dados, sendo que não há limite para o número de componentes utilizados em suas respectivas camadas. Os componentes de aplicação comunicam-se entre si através de uma interface abstrata, a qual esconde a função executada pelo componente. Essa infraestrutura prove dimensão, segurança, e serviços de comunicação para os componentes da aplicação.

Estes componentes inclusive, não necessitam serem escritos na mesma linguagem de programação ou usar o mesmo sistema operacional

As vantagens do ambiente multi-camadas estendem-se além do ciclo de vida de uma simples aplicação. De fato, o que está sendo implementado, na realidade, não é uma aplicação: é uma coleção de módulos clientes e servidor que comunicam-se através de interfaces abstratas e padronizadas, e quando combinadas, elas comportam-se como um sistema de aplicação integrado. Cada modulo é realmente um objeto compartilhado e reutilizável que pode ser incluído em outros sistemas aplicativos.

Interface Abstrata

Os conceitos de orientação a objetos, como encapsulamento e abstração são fundamentais para a arquitetura multi-camadas e para a flexibilidade da aplicação. Cada componente da aplicação pode ser visto como um objeto encapsulado – uma estrutura de dados com um conjunto de operações ou métodos que podem ser usados para manipular dados. Os dados constantes em um objeto podem ser manipulados através do uso de operações definidas. As operações são invocadas através do uso de Interface Abstrata. A Interface Abstrata identifica a operação a ser executada e define os parâmetro de entrada e saída que são necessários para tal execução.

Uma Interface Abstrata esconde o real lógica desempenhada por um objeto de uma aplicação. A Interface Abstrata permite que um objeto seja visualizado como uma caixa preta por uma interface externa. Quanto for a duração da execução da operação pelo componente, os parâmetros de entrada e saída são preservados, nenhuma mudança precisa e ser feita a quaisquer componentes.

Manutenção facilitada do Sistema

A vantagem mais óbvia da arquitetura multi-camadas é a facilidade de manutenção. Como as funções da aplicação estão isoladas em pequenos módulos de objetos, a lógica da aplicação pode ser modificada muito mais facilmente do que anteriormente. Por exemplo, uma função que é ativada por uma aplicação financeira é usada para calcular salário anual médio dos empregados de uma instituição. Os algoritmos dessa função mudam periodicamente de acordo com o regulamento da taxa administrativa anual da corporação. Normalmente, mudanças na taxa administrativa anual, requerem mudanças em toda a aplicação financeira. Ao isolar essas regras de negócios em objetos de negócios autônomos, os algoritmos podem ser alterados para atender ao novo cálculo de taxa anual sem afetar o restante da aplicação.

Um uso mais efetivo de dados e Redes

Uma das vantagens significativas da arquitetura multi-camadas resulta do fato que a lógica da aplicação depende diretamente da estrutura do banco de dados. Objetos de aplicação individuais trabalham com suas próprias estruturas de dados encapsulados, o qual pode corresponder a uma estrutura de banco de dados, ou pode ser uma estrutura derivada de um número de diferentes fontes de dados. Quando os objetos da aplicação se comunicam, eles somente precisam enviar os parâmetros de dados como especificado em uma interface abstrata, reduzindo o tráfego da rede, já que no método convencional, como o conjunto de registro seria transportado na rede. Os objetos de acessos a dados são componentes aplicativos que fazem se comunicam diretamente com o banco de dados. Conceitualmente, uma base de dados poderia ser completamente migrado de um Sistema Gerenciador de Banco de Dados para um outro, sem afetar o desempenho da aplicação: somente a lógica de acesso a dados precisaria ser modificada.

A abstração da lógica de acesso a dados leva a outro benefício significativo. O conceito de dados que podem ser usados para ser inseridos em arquivos seqüenciais, arquivos indexados, banco de dados não-relacionais, e sistemas legados. Não há limitações baseadas nas capacidades do SQL. Um conjunto de módulos de acesso a dados pode ser desenvolvido para prover acesso a ambientes legados com um conjunto conveniente de interfaces abstratas que são acessíveis de qualquer lugar da corporação.

Outras vantagens [SONNINO, 2001]

- A aplicação-cliente fica muito mais leve, necessitando de máquinas menos potentes, reduzindo custos de atualização tecnológica.
- Não há necessidade de distribuir nenhuma *engine* de acesso a dados com a aplicação-cliente. Como não há acesso real aos dados, tudo que é necessário são ferramentas de conexão entre a aplicação-cliente e a intermediária.
- Redução no custo profissional e financeiro de manutenção de sistemas. Caso alguma regra de negócio seja alterada, não é necessário alterar a aplicação-cliente (muitas vezes o usuário nem fica sabendo) – apenas a aplicação intermediária é modificada.
- Redução do número de licenças do SGDB necessário. Como apenas a aplicação intermediária acessa o banco de dados, apenas uma licença é requerida – as aplicações-cliente utilizam de modo compartilhado a mesma conexão com o SGDB estabelecida pela aplicação intermediária.
- Balanceamento de carga. Caso haja mais de um servidor provendo dados, a aplicação intermediária pode distribuir as requisições para este ou aquele servidor, de maneira que as cargas estejam balanceadas.
- Trabalho ininterrupto. Caso haja dois servidores espelhados e um deles saia do ar, a aplicação intermediária pode desviar o fluxo das informações para o outro, sem que o usuário perceba qualquer interrupção.

4.1.4 Modelo de sistema baseado em três camadas

O acesso multicamadas inclui uma aplicação intermediária entre a aplicação – cliente e o servidor de banco de dados. A aplicação-cliente já não conversa mais diretamente com o SGDB, e sim com essa aplicação intermediária que, por sua vez, conversa com o SGDB. Isto reduz bastante as necessidades de processamento da aplicação-cliente: ela se encarrega apenas da interface com o usuário e recebe informações da aplicação intermediária, que conversa com o banco de dados.

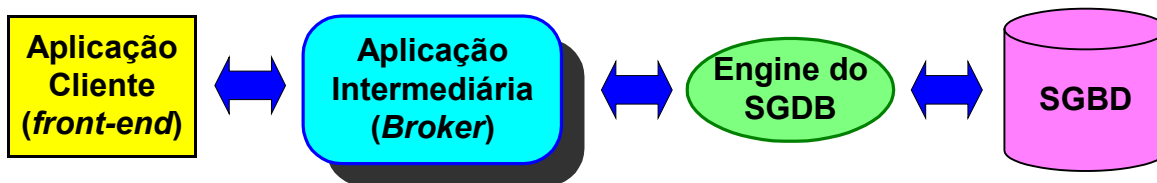


Figura 2. Acesso a banco de dados em três camadas.

Um modelo de implementação de sistema em 03 camadas é composto de:

- Clientes *front-end* (para interface ou apresentação),
- *Brokers* (também conhecidos como *Application Servers*),
- Um mecanismo de armazenamento ou um Servidor de Banco de dados.

Isso resulta no que também é chamado de Arquitetura de Distribuição em 3 camadas.

A camada de apresentação (*front-end*) consiste de componentes que são responsáveis unicamente para chamar os componentes da camada de negócios e mostra o resultado dessa interação ao usuário do aplicativo. Essa parte dos sistemas garante toda parte da interação do usuário e gerencia tarefas exclusivamente locais, gerenciadas pelos próprios usuários clientes, como uma validação de formulário antes da ativação de um componente lógico de negócios.

A camada do *broker*, suportada por um ambiente multi-usuário, manipula partições compartilhadas da aplicação e da camada lógica de negócios. Essa camada necessita de serviços como transações, controle concorrente e segurança. Como a maioria dos dados/informações requisitadas pelas aplicações são acessadas através dessa camada, operações de manipulação intensiva de dados deveriam ser executadas aqui sempre que possível. Os *Brokers* são responsáveis pela coordenação das transações e acesso à sistemas legados. Com essa camada, partes dela podem cooperar através de uma comunicação intra-camada.

A camada do servidor de banco de dados é responsável por gerenciar a persistência de certos dados e de executar transações no banco de dados. Um servidor de armazenamento opcional pode ser responsável por mapeamento especial, ou mapeamento entre objetos da aplicação e tabelas de um banco de dados relacional ou identificadores de objetos de um banco de dados orientado a objetos.

O processamento da aplicação, localizado nos *brokers*, é separado tanto da interface do usuário quanto do servidor da base de dados. Atividades como a validação ou pré-carga de dados pode ser realizada de forma local e privada nos clientes. Dados centralizados podem ser acessados sem ser movidos completamente para os clientes *front-end*. A possibilidade de pré-processamento de informação

nos *brokers* pode ajudar na redução do tráfego na rede e aumentar a performance da aplicação.

Toda a parte de negócios da aplicação, ou seja, a parte realmente inteligente da aplicação como um todo, está isolado do cliente. Toda essa parte encontra-se em um único lugar, o que torna a manutenção do código mais fácil e independentes de modificações dos clientes ou na base de dados.

A interface entre os *front-ends* e um *broker* pode ser desenvolvida por um programador assim que necessário pois o programador não está preso a uma API específica de um fabricante. *Brokers* provêm boas possibilidades de integração de sistemas legados, pois as interfaces e camada de negócios são independentes de novas implementações a nível de camadas externas.

4.1.5 Integração do modelo em camadas X WEB

Na década de 60 surgiram os primeiros sistemas informatizados, com aplicativos de finanças, folha de pagamento e manufatura, rodando em grandes máquinas, os *mainframes*. Em meados dos anos 80, a tecnologia da informação evoluiu, surgindo a arquitetura cliente/servidor e se ampliaram os sistemas e suas funcionalidades, surgindo os sistemas de ERP (*Enterprise Resource Planning*). Todos voltados para o controle dos processos internos da empresa. Nos últimos anos da década de 90, a atenção começou a se voltar para os processos externos, atingindo clientes, fornecedores e parceiros, utilizando o ambiente Web.

O conceito arquitetônico de ERM (*Enterprise Relationship Management*) evidencia que existirão dois grandes blocos [BARBIERI, 2001]: um de vitrine ou *front-end*, e outro de apoio ou *back-office*.

No bloco da frente se concentrarão os sistemas, aplicativos e as tecnologias voltadas ao tratamento com o cliente da empresa, os sistemas chamados de CRM (*Customer Relationship Management*). O objetivo será melhorar as ações de marketing, vendas e serviços.

No bloco de trás estão os sistemas ERP, conhecidos como sistemas legados contendo módulos de Finanças, Contabilidade, Investimento, Material, Pedidos, Planejamento, Recursos Humanos, etc.

Nesse ambiente de ERP os dados são armazenados nas formas transacionais, com ênfase na integração de processos.

Com a evolução dos negócios, da globalização e da competitividade cada vez mais acirrada, as empresas passaram a sentir a necessidade de integrar todos os sistemas e buscar soluções baseadas na Internet.

Nesse cenário, apresenta-se a seguinte questão. Como integrar os sistemas internos com soluções Web? A solução para isso é EAI (*Enterprise Application Integration*) - integração de aplicações corporativas .

EAI é a criação de uma nova estratégia de soluções de negócio por integrar a funcionalidade de aplicações existentes na empresa, aplicações de pacotes comerciais e novos códigos usando um *middleware* comum [PACHECO, 2001]. *Middleware* é software independente de aplicação que provê serviços que servem como mediadores entre as aplicações.

EAI pode ser usado para ajudar a eliminar passos manuais no processo de negócio e evitar a entrada redundante de dados. As aplicações de EAI usam uma ferramenta de automação de fluxo de trabalho como ponte entre as aplicações que estão sendo integradas.

4.2 Arquiteturas de EAI

Existem atualmente no mercados diversas soluções de software para implementação de EAI. Podemos citar:

4.2.1 IBM Component Broker

IBM Component Broker - O *Component Broker* da IBM [IBM] é um ORB¹ (*Object Request Broker*), que é uma implementação CORBA². Isto é, o *component broker* é um ORB com um conjunto de serviços e facilidades CORBA.

4.2.2 Microsoft Windows DNA 2000

Microsoft Windows DNA 2000 – A solução Windows DNA 2000 [MICROSOFT] é uma arquitetura da Microsoft que usa a tecnologia de objetos distribuídos para integração de aplicações empresariais. Baseado no modelo de programação COM+ e serviços do servidor Windows 2000.

4.2.3 Enterprise JavaBeans

Enterprise JavaBeans – Implementação de EAI baseada em componentes Java. Esta solução, bastante utilizada pelos principais Sistemas Gerenciadores de Banco de Dados, como Oracle, IBM DB2, SQL Server, entre outros, possivelmente será a opção futura para implementação do objeto desse estudo e por esta razão, será apresentada em mais detalhes.

A arquitetura *Enterprise JavaBeans*, conhecida como EJB da Sun Microsystems [SUN] define um modelo para o desenvolvimento e disponibilização de componentes Java reusáveis. A tecnologia EJB tem objetivo de ser multicamada e possibilitar o desenvolvimento de aplicação distribuída.

EJB provê um conjunto de interfaces componentes para a padronização dos componentes na plataforma Java:

¹ ORB (*Object Request Broker*) é o conceito central da arquitetura que permite que CORBA seja implementada. E provê o mecanismo para a comunicação dos cliente de forma transparente.

² CORBA é uma arquitetura de objetos distribuídos padronizada pelo OMG (*Object Management Group*) [OMG]. CORBA provê padrão de interface orientada a objeto, mas permite a implementação de vários tipos de serviços serem de qualquer natureza.

- API Enterprise JavaBeans: define um modelo de servidor de componentes que provê portabilidade entre servidores de aplicação e implementa serviços automáticos de interesse de componentes de aplicação;
- Nomeação Java e API de interface de diretório: provêem acesso para nomeação e serviços de diretórios, tais como, DNS, NDS, NIS+, LDAP e Cos Naming;
- Chamada de método remoto: cria interfaces remotas para computação distribuída na plataforma Java;
- API da linguagem de definição de interface Java: cria interfaces remotas para suportar comunicação CORBA na plataforma Java. IDL Java inclui um compilador IDL e ORB;
- Java Servlets e APIs de páginas de servidores Java: suportam geração de HTML dinâmico e gerenciamento de seção para clientes baseados em *browser*;
- API de serviço de mensagem Java: suporta comunicação assíncrona usando sistema de mensagem, tais como filas confiáveis e serviços de publicar/subscrever;
- API de transação Java: provê uma API de demarcação de transação;
- API de serviço de transação Java: define um serviço de gerenciamento de transação distribuída baseado no serviço de transação de objetos da CORBA;
- API de acesso a banco de dados usando JDBC: provê acesso uniforme a bancos de dados relacionais, tais como DB2, Informix, Oracle, SQL Server e Sybase;
- Integração com serviço de mensagem Java (JMS – *Java Message Service*): permite que partes da empresa participem de operações de mensagens assíncronas;
- Persistência gerenciada por recipiente (CMP – *Container-Message Persistence*): torna o desenvolvimento da aplicação mais simples e mais rápido por permitir desenvolvedores implementarem aplicações portáteis, independentes de banco de dados;
- Interoperabilidade entre servidores: com o objetivo de suportar aplicações B2B (*Business-to-business*) e integração entre ambientes heterogêneos.

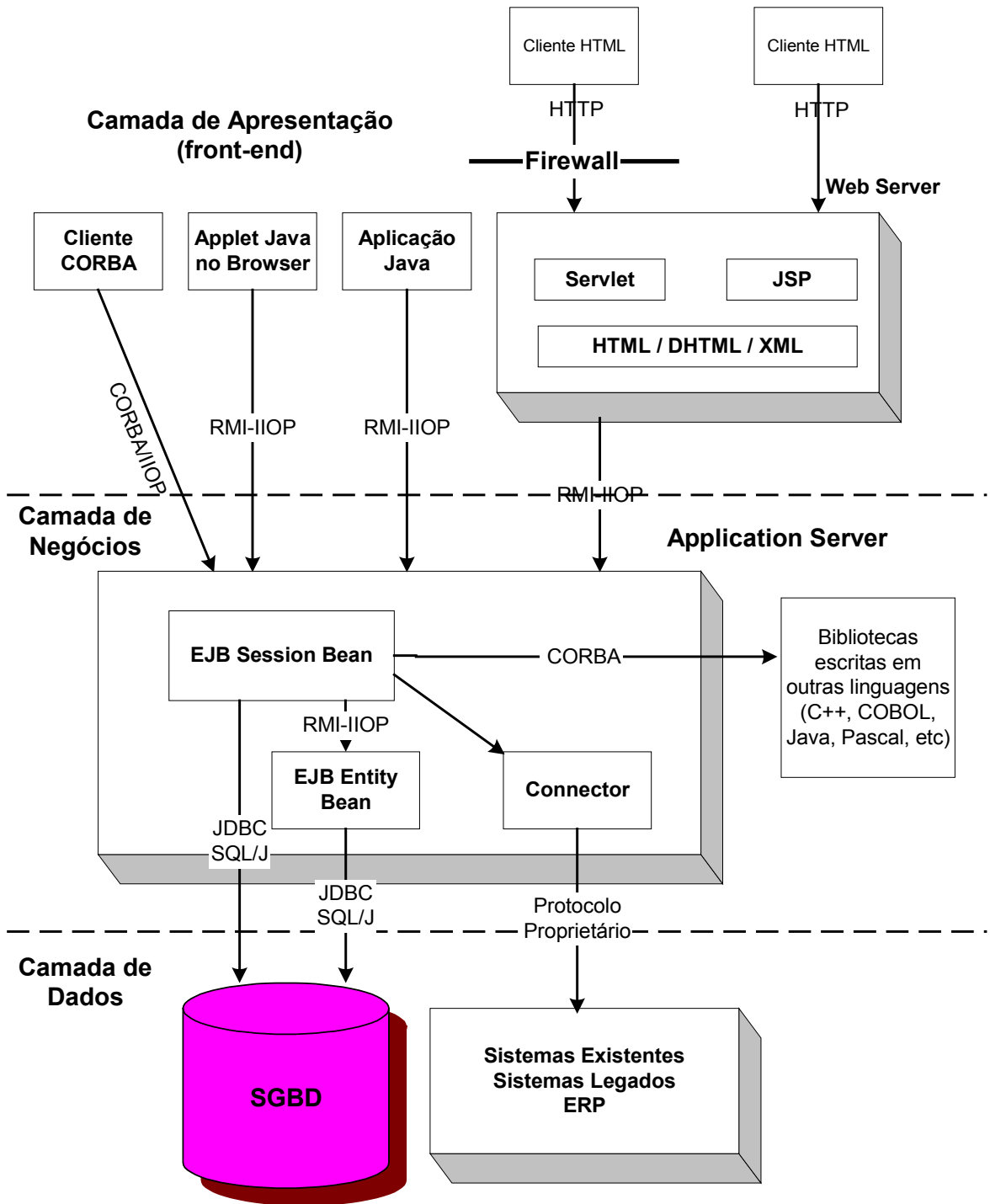


Figura 3. Modelo de implementação Enterprise JavaBeans.

5. Estudo de Caso

5.1 Definição do Problema

O propósito central deste estudo é definir um projeto de implementação de um sistema Web para ser utilizado pelos alunos dos cursos de graduação da Universidade Estadual de Maringá.

O controle acadêmico correspondente envolve um significativo conjunto de procedimentos que, pelo volume de informações envolvidas, por sua extensão e complexidade, exigiram a implantação de um sistema informatizado.

Desenvolvido de forma colaborativa entre os técnicos da Diretoria de Assuntos Acadêmicos e do Núcleo de Processamento de Dados, o sistema informatizado de controle acadêmico dos cursos de graduação exibe uma estrutura geral centralizada e constituída de subsistemas indissociáveis, tornando-se tarefa relativamente complexa sintetizá-lo.

No decorrer dos anos e com a implantação do regime seriado anual, as mudanças (interna e externa) ocorreram em ritmo acelerado. Acompanhá-las e até antecipar-se às mudanças, foi uma forma de garantir seu ótimo desempenho. Tal procedimento consistiu na introdução de ajustes ou alterações racionais e práticas na estrutura deste sistema, procurando buscar um controle acadêmico efetivo, seguro e eficiente, possibilitando maximizar seus processos e um acompanhamento acadêmico moderno e eficaz. Pode-se dizer que é praticamente impossível controlar a vida acadêmica dos alunos de graduação na Universidade Estadual de Maringá por antigos processos administrativos manuais.

Alcançado seu pleno funcionamento, este sistema encontra-se maduro e consolidado, pode-se afirmar que o sistema hoje existente atende satisfatoriamente às necessidades da instituição como um todo.

Porém, a tendência atual da disseminação da informação provocada principalmente pela utilização em massa da Internet, cria a necessidade da abertura deste sistema

para um número maior de usuários que, acostumados com a praticidade dos serviços oferecidos via Web nas mais diversas áreas, exigem também do controle acadêmico uma gama de serviços e informações on-line hoje inexistentes.

5.2 Objetivos

O objetivo do sistema proposto é possibilitar à área discente da Universidade Estadual de Maringá o acesso a informações da vida acadêmica do aluno, inicialmente voltado para os cursos de graduação.

Aliado a isso, permitir ao aluno a solicitação de serviços on-line hoje disponíveis apenas no balcão da Diretoria de Assuntos Acadêmicos. Assim, buscando atender aos seguintes objetivos:

- Permitir acesso controlado do discente à informações pertinentes à sua turma, tais como notas, faltas, horário de aula.
- Possibilitar a solicitação de serviços que hoje necessitam de um requerimento feito presencialmente junto à DAA, através de um meio eletrônico on-line.
- Melhorar o atendimento acabando com filas provocadas por processos administrativos realizados pela Instituição para manter um controle sobre os cursos como por exemplo a confirmação de matrícula que deve ser feita a cada ano letivo.
- Minimizar o atendimento direto no balcão da DAA, permitindo o deslocamento de funcionários do setor para a realização de atividades internas.
- Com as solicitações cadastradas diretamente no sistema agilizar o processo de avaliação, processamento, divulgação e auditoria das solicitações.

5.3 Arquitetura do Sistema

A arquitetura utilizada na implementação do sistema de avaliação de alunos é composta de dois servidores principais e de uma interface Web. Os servidores são: o servidor de banco de dados e o servidor de páginas HTML com *scripts*. De um lado estão estes servidores, e do outro está o cliente com um navegador (*browser*) que suporta HTML.

Como podemos observar, a arquitetura se divide em três níveis distintos: o nível em que se encontra o servidor de páginas HTML com *script*, o nível do servidor de banco de dados, que fica somente encarregado do tratamento das consultas e atualizações nas tabelas, e o nível do cliente, que interpreta somente código HTML. Através da linguagem *script* e de expressões em linguagem SQL utilizadas nas páginas, o servidor de páginas consegue a interação necessária com o servidor de banco de dados para a realização da funcionalidade do sistema. Todo *script* de acesso ao banco de dados é, portanto, executado no servidor de páginas. O cliente não tem qualquer conhecimento deste código *script* e nem da existência do servidor de banco de dados.

Os programas que fazem parte do sistema de avaliação construído estão instalados no servidor de aplicações Web da UEM, alocado no Núcleo de Processamento de Dados. Trata-se de um servidor de arquitetura AMD Athlon 1.7 Ghz, com sistema operacional Linux Red Hat 6.3, servidor Apache 2.0.3 e interpretador PHP versão 4.3.3. O servidor de Banco de Dados utilizado é o IBM DB2 versão 7.2, instalado em um servidor Risc IBM R50, processamento duplo em paralelo, com sistema operacional AIX versão 4.3.3.

5.4 IBM DB2

O ambiente de desenvolvimento de Banco de Dados utilizado no projeto foi o IBM DB2. Neste banco, instalado em um servidor de arquitetura Risc, com sistema operacional AIX, alocado no núcleo de operações, subsolo da Biblioteca Central, encontram-se os Dados de todo o controle acadêmico da Universidade bem como de outros departamentos administrativos que se utilizam de sistemas desenvolvidos por técnicos do NPD.

O DB2 Universal Database é um SGBD relacional da IBM Corporation, projetado para ser um banco de dados capaz de oferecer um suporte relacional de gerenciamento de dados rápido, seguro e eficiente. Sua tecnologia é derivada do conceito de Banco de Dados Universal, que visa ampliar as capacidades da tecnologia de banco de dados, permitindo manipular tipos de dados não-convencionais, tais como: informações armazenadas em documentos, planilhas e

objetos multimídia. Hoje as versões mais recentes oferecem suporte a vários tipos de aplicações cliente/servidor disponíveis, incluindo a internet.

Como um banco de dados relacional que segue o padrão ANSI, o IBM DB2 suporta os mesmos comandos e sintaxe largamente usados por outros bancos de dados relacionais. Isto permite que código escrito em outras plataformas em SQL (*Structured Query Language*), sejam executados no IBM DB2 com pouca modificação. Analogamente, como a gramática SQL está em conformidade com o padrão 1992, programadores aprendem rapidamente o uso do IBM DB2.

5.5 O escopo do Sistema

O controle acadêmico de qualquer Universidade é uma das atividades vitais e mais onerosas de se cumprir, demandando grande volume de material e trabalho por parte da Instituição; tanto no que diz respeito à manutenção quanto na própria documentação dos fatos importantes neste contexto. A vida do corpo discente e docente da Universidade, o desempenho dos cursos mantidos por ela, das suas turmas e de si mesma dependem diretamente do efetivo armazenamento e controle dos dados diariamente produzidos. O mínimo descaso poderia comprometer a avaliação da própria Universidade.

Com a finalidade de reduzir a ocorrência de erros e ausência de dados condizentes com a realidade, a Universidade Estadual de Maringá conta com um sistema automatizado, o Sistema de Controle Acadêmico, que é o software responsável por cumprir as seis funções principais da realidade acadêmica: a matrícula; o encerramento do período; o controle da vida acadêmica dos alunos; a oferta de turmas e o auxílio ao planejamento dessas; e, finalmente, a organização curricular, que nada mais é do que o acompanhamento da evolução dos cursos oferecidos pela Universidade, com a atualização de seus currículos e habilitações.

As responsabilidades do sistema caracterizam-se por: executar os processos de matrícula em geral, com ênfase na justa distribuição de vagas entre os alunos; integrar-se aos resultados do vestibular, fazendo admissão de novos alunos diretamente do sistema de inscritos no vestibular; registrar e controlar as

solicitações e ofertas de vagas em turmas, os lançamentos de notas e frequências, e o fechamento de turmas; registrar e controlar os cursos, seus currículos, além de disciplinas locais e externas; controlar as diversas situações de um aluno em sua vida acadêmica, incluindo emissão de históricos, aproveitamentos de disciplinas e complementos de habilitação.

O Sistema foi desenvolvido primeiramente sob a plataforma *mainframe*, e projetado para ter suas informações mantidas em arquivos VISAN e CSP (a linguagem de desenvolvimento). Porém, com o surgimento de outros paradigmas e o lançamento de computadores menores e robustos levaram a Instituição empresas a optar pelo *downsizing*.

Com esta nova evolução tecnológica, surgiu a arquitetura cliente-servidor, que prometia uma melhor performance a custos menores. Assim, o sistema foi migrado para a plataforma atual utilizando VisualAge como ferramenta de desenvolvimento buscando o reaproveitamento do código escrito em CSP e IBM DB2 para armazenamento das informações.

Porém, o sistema acadêmico enfrenta um novo desafio: atender a demanda por aplicativos acessíveis e manipuláveis via Web.

Diversas tarefas do controle acadêmico podem e devem ser transferidas para o mundo WWW. O que trará as tantas vantagens inerentes à própria abordagem de aplicações Web: interface amigável, portabilidade, custos menores com a aquisição de softwares proprietários, entre outros. A própria evolução tecnológica e a comodidade e transparência proporcionada pela Internet faz com que o sistema não se desvie deste novo horizonte.

O desafio de formular um sistema para solicitações e informações direcionados aos alunos de graduação da UEM, que possua uma complexidade conceitual baixa e riscos também reduzidos levaram a escolha de opções que, talvez não sejam as melhores porém dadas as características de infraestrutura, pessoal e financeira da Instituição são as mais viáveis.

Assim, o sistema foi definido em duas partes: A primeira, de administração interna, desenvolvida em Delphi 5.0, com tecnologia Midas, arquitetura em três camadas com conexão via *Sockets* e a segunda, desenvolvida para Web utilizando linguagem PHP.

Esta escolha se justifica, no primeiro caso, porque o Delphi foi adotado como linguagem opcional para o desenvolvimento de sistemas pelo Núcleo de Processamento de Dados quando a linguagem padrão adotada, o VisualAge não permitisse ou dificultasse sobremaneira o desenvolvimento.

Para o caso específico, optou-se pelo Delphi visto que o sistema deverá conter acesso à comunicação serial, com a instalação de uma impressora fiscal para emissão de comprovantes de solicitações e recibos de pagamento. Este recurso entretanto não está disponível na solução VisualAge, onde o processamento ocorre no servidor de aplicações e não permite acesso à portas da máquina local.

A implementação do sistema em 03 camadas traz duas importantes vantagens que reforçaram a opção por este tipo de arquitetura:

- Segurança – Como os dados não são acessados diretamente na máquina cliente, não é necessária a instalação de cliente para o DB2, evitando acessos de usuários não autorizados que são facilitados por aplicações que se utilizam de acesso ODBC, como por exemplo Microsoft Access.
- Praticidade – Nada é instalado no cliente e portanto basta permitir acesso ao servidor da aplicação para que o cliente possa utilizar-se do sistema, reduzindo a necessidade de apoio e serviços de suporte ao usuário.

A opção da linguagem PHP para o desenvolvimento da aplicação na WEB deu-se pelo fato de que esta é a linguagem já utilizada por outros aplicativos, como o lançamento de notas via internet, para o desenvolvimento de aplicações Web, além de ser software livre, uma tendência entre as Instituições Públicas do Estado do Paraná. Aliado a isto, temos como vantagem o conhecimento mais aprofundado da linguagem por parte dos profissionais da área de desenvolvimento do NPD e a boa performance em ambientes de arquitetura de rede não tão rápidas, quando comparadas com outras opções como por exemplo Java.

5.6 Principais opções para os alunos

O sistema disponibilizará, via Web, em diversas fases, os seguintes serviços:

Antecipação de Diploma
Atestado de Confirmação de Matrícula
Atestado de Matrícula
Auxílio Financeiro
Colaço de Grau Especial
Confirmação de Matrícula
Consultas de Notas e Faltas
Devolução de Documentos
Devolução de Taxas
Estágio Curricular
Estágio Extra-Curricular
Exame de Proficiência
Fotocópias de Documentos
Fotocópias Diversas
Histórico Escolar
Participação em Projeto
Prorrogação de Prazo
Reanálise de AAC
Reanálise de Aproveitamento de Estudos
Reconsideração
Relação de Acadêmicos Matriculados
Reopção de Vestibular
Solicitação de Disciplinas de outro Curso
Solicitação de Disciplinas do mesmo Curso
Subseqüente de Transferência
Transferência Interna de Turno
Vistas à Prova

Muitos destes serviços, entretanto, possuem taxas que devem ser pagas ou documentos que devam ser anexados. As taxas deverão ser pagas no ato da retirada

da solicitação e os documentos, deverão ser entregues na DAA nos prazos estipulados para cada opção que serão devidamente informadas no ato da solicitação.

Apesar destes entraves administrativos, o aluno terá facilidades para a solicitação *on-line* principalmente no nível de informação recebida e no fato de que como estas solicitações geralmente têm prazos de início e término e a solicitação via Web permite entrar com o pedido no prazo com mais comodidade.

5.7 Aspectos de Segurança anexados ao sistema

A rede de computadores utilizada pela Universidade Estadual de Maringá já implementa requisitos de segurança tradicionais como *firewall*, *proxy*, controle de usuários e senhas e triagem de emails.

Ainda é adicionado a estes mecanismos, o controle de senhas e usuários no SGBD, com adoção de *Schemas* de tabelas e permissões controladas para cada usuário baseado totalmente em *Views*.

Apesar do sistema proposto não oferecer tecnicamente grandes riscos a integridade dos dados gerais da UEM ou mesmos aos específicos do Sistema de Controle Acadêmico, foram previstas a adoção de três medidas adicionais de segurança:

Criptografia nas tabelas de usuário, senhas e direitos

As tabelas de usuários do sistema, senhas e direitos são armazenados no Banco de Dados utilizando criptografia MD5. Além dessa encriptação dos dados, que não permite que mesmo usuários internos, com acesso direto ao banco de dados possam identificar qualquer registro na tabela, os dados, antes de serem gravados são montados e embaralhados por um algoritmo desenvolvido internamente e de conhecimento restrito aos analista diretamente ligados ao Sistema de Controle Acadêmico.

Chave de identificação única para comprovantes de solicitação

Para alguns tipos de solicitações, aqui podemos exemplificar a confirmação de matrícula, é muito importante que a Diretoria de Assuntos Acadêmicos possa identificar de forma inequívoca a validade do comprovante de solicitação apresentado pelo aluno com o intuito de evitar fraudes.

Vamos continuar no exemplo da Confirmação de Matrícula e imaginar o seguinte cenário:

O aluno, devidamente matriculado na Universidade, perde os prazos para Confirmação da Matrícula no curso e tem a sua matrícula cancelada por parte da instituição, procedimento este bastante comum na UEM. Este aluno, após constatado o fato resolve entrar com processo contra a DAA apresentando um comprovante fraudado de um outro colega de curso onde informa o seu requerimento de matrícula no prazo. Como garantir que se trata de uma fraude e não de um erro de atualização do sistema?

Para resolver esta questão foi adotado a geração de uma chave de validação para cada comprovante de solicitação emitido pelo sistema. Trata-se de um número de 16 caracteres fornecido por meio de um algoritmo próprio de compilação e que é gerado através das informações de Registro Acadêmico, ano de ingresso, tipo de solicitação, data e hora da solicitação.

Este número de confirmação, quando submetido a um outro algoritmo descompilação, retorna as informações utilizadas para sua geração que podem ser facilmente validadas com o Banco de Dados.

Log de acessos

Um terceiro mecanismo de segurança foi projetado para o sistema visando o rastreamento de tentativas de acesso ou acesso indevido. Este mecanismo gera um log no banco de dados com informações do usuário, endereço IP da conexão, data e hora de entrada no sistema. Estas informações em conjunto com outros mecanismos de segurança e rastreamento podem acrescentar importantes subsídios na investigação de ações suspeitas cometidas pelos usuários do sistemas, sejam eles internos ou externos.

5.8 Fluxo do Sistema

O processo de funcionamento do sistema pode visto ser pela figura abaixo:

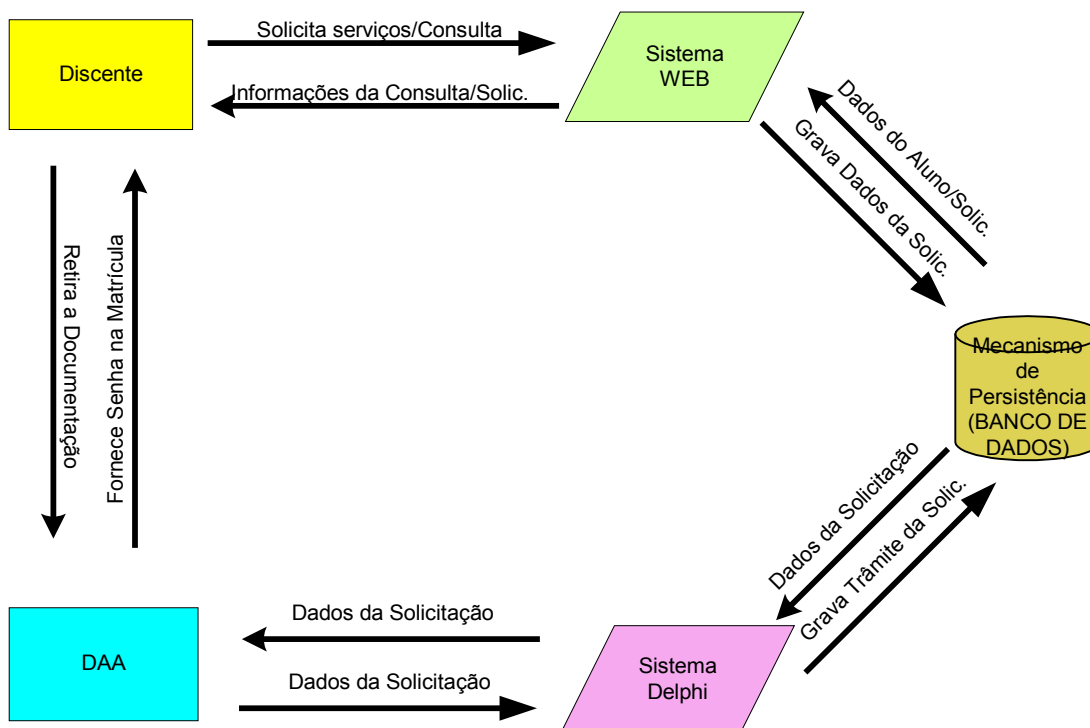


Figura 4. Fluxo do Sistema.

O aluno receberá uma senha de acesso quando for efetuar a matrícula definitiva no curso ou quando for fazer a reconfirmação de matrícula se já aluno da Instituição.

De posse desta senha, o aluno dará entrada no sistema informando o número do seu Registro Acadêmico a senha.

A partir daí, estarão disponíveis os serviços de solicitação e consultas. Estes serviços porém podem estar fora do prazo, caso em que a opção aparecerá desabilitada e os prazos serão comunicados ao aluno. Também, alguns serviços permitem um número de solicitações/ano sem cobrança de taxa e caso o aluno tenha extrapolado este número, será notificado que deverá fazer o pagamento na retirada do documento.

Em se tratando de serviços que podem ser acessados e impressos *on-line*, a solicitação será atendida de imediato e registrada para futuras consultas.

Serviços que necessitam intervenção da DAA e processamento ou avaliação interna, serão automaticamente encaminhados para os setores responsáveis. Neste caso, o aluno receberá um comprovante de solicitação e uma chave para acompanhamento da mesma.

Executada a solicitação, cada setor responsável fará o trâmite interno da solicitação utilizando o software de uso restrito, realizando o processamento e emissão de documentos necessários.

Este fluxo interno é inerente ao documento e após o trâmite completo, será encaminhado, se for o caso para a Secretaria da DAA.

Uma vez na Secretaria, fica à disposição do aluno para retirada e conseqüente pagamento, se for o caso.

6. Conclusão

A proposta inicial da realização de estudos para o desenvolvimento de um sistema de informações para WEB para solicitações e consultas de alunos foi alcançada.

Foi verificada a total viabilidade da implantação do sistema, considerando as características técnicas e implementação e a estrutura básica disponível. Neste aspecto a necessidade urgente do usuário principal, o aluno, de ter informações precisas e rápidas de sua vida acadêmica aliada à agilização dos processos administrativos da DAA justificam a implementação do sistema.

O sistema de desenvolvimento em camadas mostrou-se eficiente e adequado ao propósito previsto considerando a estrutura atual e as perspectivas futuras de implantação de novas tecnologias baseadas na linguagem Java.

Para que os aspectos da tecnologia de acesso e de segurança sejam efetivamente comprovados, será necessário a implantação de um projeto piloto para a certificação direta no ambiente de produção, já que os testes preliminares foram executados apenas em ambiente de desenvolvimento.

7. Considerações Finais

7.1 Trabalhos Futuros

É importante ressaltar que o trabalho concluído possui vários aspectos que não foram tratados. Este fato se dá devido à complexidade do projeto e a fatores técnicos e administrativos internos da Instituição.

A médio prazo, pretende-se ampliar o alcance do projeto para atender também aos alunos dos cursos de Pós Graduação. Estes alunos ainda não serão atendidos pois o sistema de controle de pós-graduação foi desenvolvido por terceiros e não seguem os padrões adotados pelo NPD, não permitindo a integração total e coexistência com o sistema de graduação. Para isso, tanto o sistema da graduação quanto o de pós graduação deverão ser reformulados para possibilitar que os dados de ambos possam ser vistos como um todo na Instituição.

7.2 Considerações sobre o estudo de caso

O modelo sugerido e implementado também servirá como apoio para futuros trabalhos da UEM na área acadêmica, que entrará brevemente em uma nova fase, em que será feito um estudo detalhado de toda sua estrutura, analisando problemas, levantando críticas e propondo soluções aos trabalhos desenvolvidos. Todo o projeto de sistema de avaliação criado será reformulado nesta fase e as aplicações desenvolvidas serão migradas para uma única linguagem, que integrará todo o ambiente do controle acadêmico.

Este novo ambiente será baseada na solução IBM WebSphere. *WebSphere* é software de infra-estrutura que permite desenvolver e integrar aplicativos utilizando tecnologia Java com grandes vantagens nos quesitos de integração, escalabilidade, flexibilidade e compatibilidade. A solução *WebSphere* faz parte do pacote de desenvolvimento da IBM, com total integração com o VisualAge e banco de dados DB2, provendo um portfólio de software aprovado, seguro e confiável.

Nessa nova fase do projeto, as aplicações de uso interno da DAA e na Web serão integradas utilizando-se tecnologia de EAI (*Enterprise Application Integration*)

através da camada de negócios implementada através de recursos de EJB (*Enterprise JavaBeans*) implementados pelo IBM WebSphere.

8. Referências Bibliográficas

[BARBIERI, 2001] Barbieri, Carlos – BI-Business Intelligence – Modelagem & Tecnologia – Axcel Books do Brasil Editora, 2001.

[COMPUTERWORLD, 1999] Correspondentes locais – Integração do legado é essencial no pós-ERP, Computerworld, 1999.

<http://www.sit.com.br/SeparataGTI019.htm>

[FIGUEIREDO, 2001] Figueiredo, Leandro Soares – Segurança da Tecnologia da Informação, Módulo Security System,, Rio de Janeiro, 2002.

<http://www.modulo.com.br>

[IBM] <http://www.ibm.com>

[IETF97] Internet Engineering Task Force: “IETF home page”, 1997.

<http://www.ietf.org/>

[LIMA, 1997] Lima, Iremar Nunes de – O Ambiente WEB Banco de Dados: Funcionalidades e Arquiteturas de Integração, Dissertação de Mestrado Apresentada à PUC- RJ , Rio de janeiro, Agosto, 1997.

[MARTINS, 2001] Martins, Alessandro - Autoridade Certificadora para Acesso Seguro, Laboratório RAVEL / COPPE / UFRJ, Rio de Janeiro, Fevereiro, 2001.

http://www.ravel.ufrj.br/~martins/publicacoes/ Origem_do_Certificacao_Digital.pdf

[MICROSOFT] <http://www.microsoft.com>

[MULTILAYER, 2000] Grando, André Lopes Caribe; Bomfim, David Ricardo Damasceno do; Leda, Marcel Ianuck – Multilayer – A Arquitetura Multi-Camadas, Monografia de Graduação Apresentada ao Centro Universitário de Brasília - UNICEUB, Brasília, Outubro, 2000.

[OMG] <http://www.omg.org>

[PACHECO, 2000] Pacheco, Viviane – EAI: a sigla da integração dos sistemas internos e externos, Surftrade Brasil, 2000.

<http://www.dialdata.net.br/artigos/artigos.asp?m=520>

[SONNINO, 2001] Sonnino, Bruno – Desenvolvendo Aplicações com Delphi 6 – Makron Books, 2001.

9. Anexo A – Principais Telas do Sistema

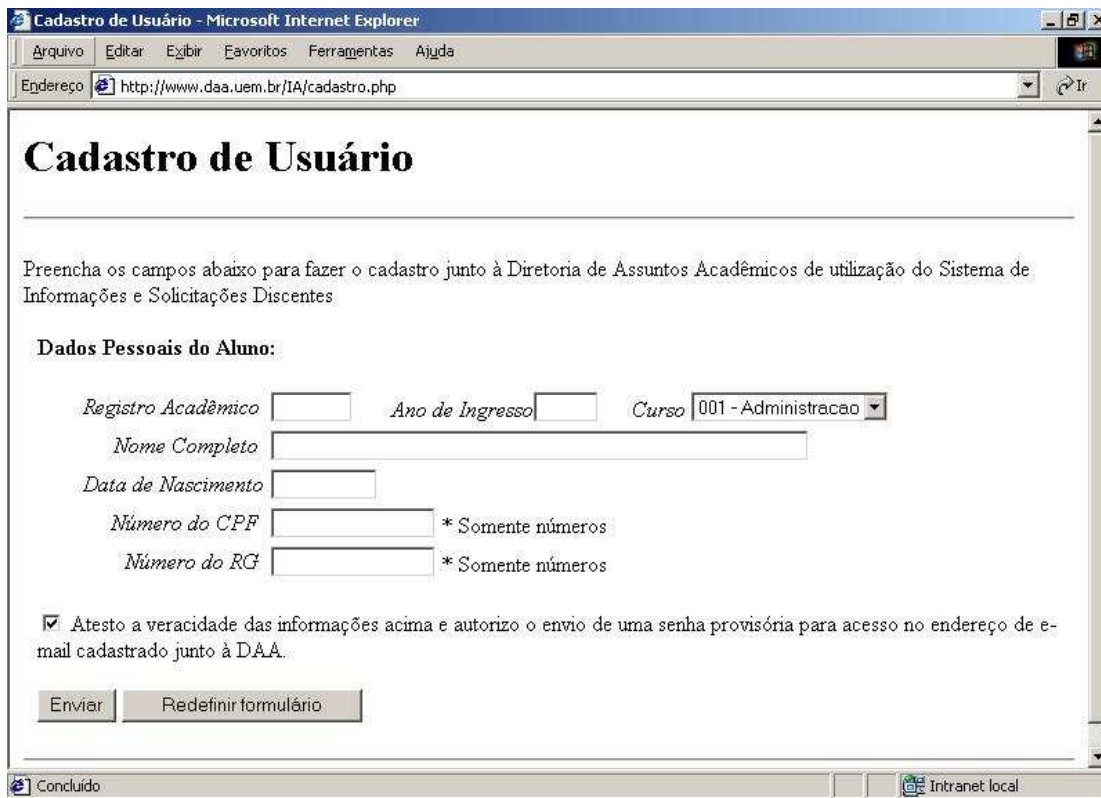
9.1 Sistema de Consultas e Solicitações Web



9.1.1 Figura 4. Tela de Login.



9.1.2 Figura 5. Tela Principal do Sistema



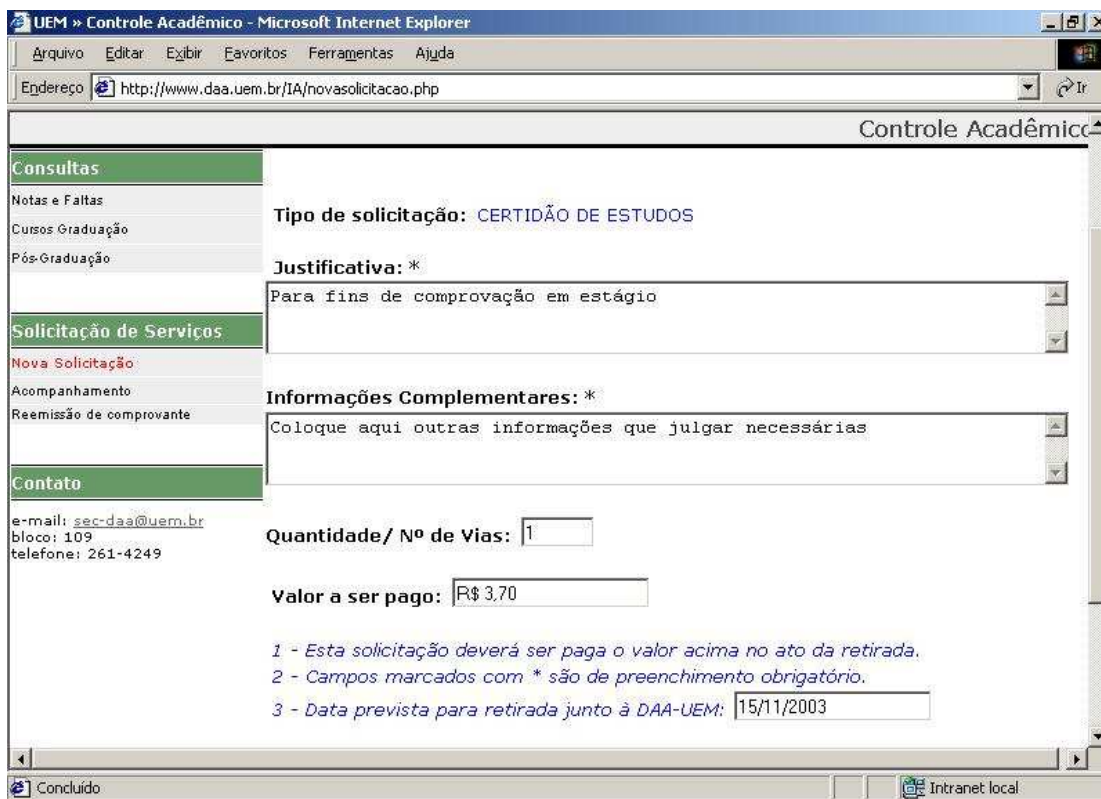
9.1.3 **Figura 6.** Cadastro do Usuário Web



9.1.4 **Figura 7.** Tela de Consulta Notas e Faltas.



9.1.5 Figura 8. Tela de Escolha de Solicitação.



9.1.6 Figura 9. Tela de Nova Solicitação.

UEM - Universidade Estadual de Maringá DAA - Diretoria de Assuntos Acadêmicos - Microsoft Internet Explorer

Arquivo Editar Exibir Favoritos Ferramentas Ajuda

Endereço http://www.daa.uem.br/IA/notas.php

Universidade Estadual de Maringá - Diretoria de Assuntos Acadêmicos

Edital de Avaliação Avaliação: 1 - Peso: 1

Departamento - 11 - DES - ESTATISTICA Max. Faltas : 17

Disc./Turma : 1991/51 - ESTATISTICA CH Total: 68

Professor: AGNALDO PASSALONGO PRETI

R.A.	Ord.	Nome Aluno	Nota	Faltas	Observações
30586	1	ADEMIR JOSE BOIS	10.0	1	
30561	2	ADILSON CELERINO DA SILVA	N/C	15	
30578	3	ADRIANO ALBERTO LAVERDE MISTRO	8.2	-	
30535	4	AIRTON LIMA	5.0	-	
30533	5	ALEXANDRE SPRANGER	5.0	-	
30552	6	ALIXON ORLANDO RIBEIRO	6.5	-	
30568	7	ALTAIR PEREIRA LIRA	5.0	-	
30576	8	ANDRE NAKASHIMA	N/C	-	TRANCADO
30549	9	AZOR PEREIRA DA SILVA	7.8	-	
30538	10	CELIA ELLANE SANCHES POLASTRO	10.0	1	
30536	11	CESAR ANTONIO LAVAGNINI	7.0	2	
30579	12	CIBELE CRISTINA FRASSON	4.0	-	
30581	13	CLEBERSON LUIZ ALVES	4.5	-	
30559	14	CLEBERSON MONTEIRO SOLIN	5.0	-	

Concluído Intranet local

9.1.7 Figura 10. Consulta Notas e Faltas.


Bankline - Microsoft Internet Explorer

Arquivo Editar Exibir Favoritos Ferramentas Ajuda

Voltar Pesquisar Favoritos Histórico

Endereço http://www.daa.uem.br/IA/comprovante.php

Google Buscar Web Buscar neste Site Info sobre a página Nível superior Destacar

 UNIVERSIDADE ESTADUAL DE MARINGÁ

Diretoria de Assuntos Acadêmicos

Comprovante de Solicitação de Serviços

ANTECIPAÇÃO DE DIPLOMA

Dados do Solicitante:
 RA : 12962
 Nome : ARIIVALDO CALDEIRA BONO

Dados da Solicitação:
 Número do Protocolo: 2345/2000 - DAA:
 Código de Controle: 1049000308019862-17
 Valor a Pagar : R\$ 3,90
 Data Provável de Retirada : 17/11/2003

Solicitação efetuada em 11.11.2003 às 19:52:11 h.

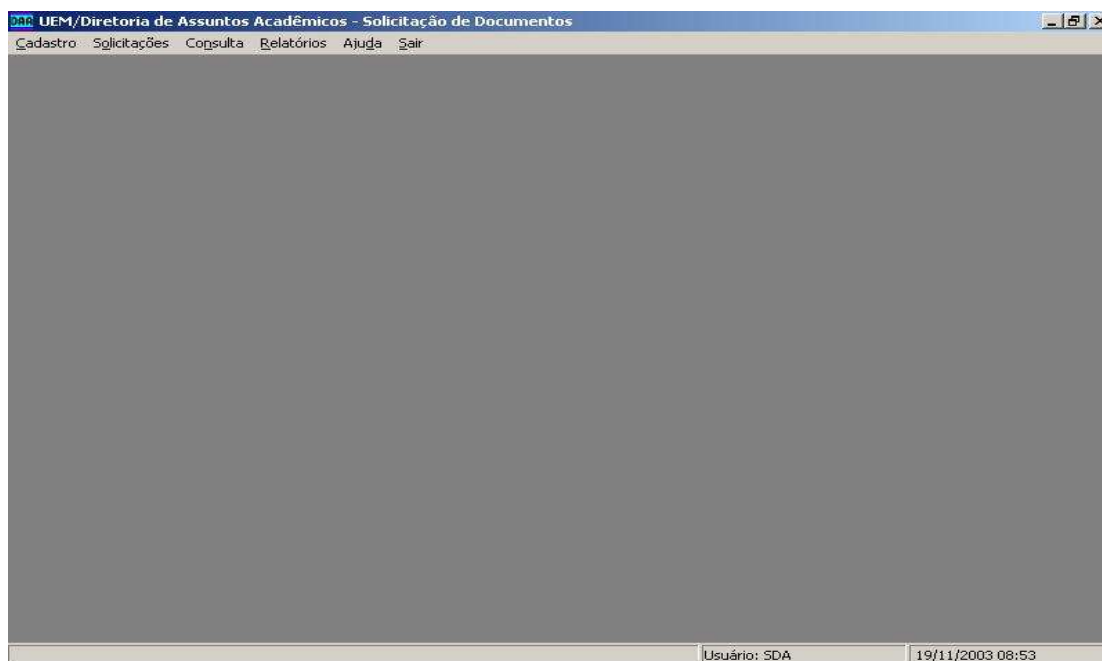
Estou Ciente de que o valor acima será cobrado no ato da retirada do documento junto à DAA/UEM.

Imprimir o Comprovante Voltar para o Controle Acadêmico

5194 Intranet local

9.1.8 Figura 11. Comprovante de Solicitação.

9.2 Sistema de Administração Interna



9.2.1 **Figura 12.** Tela Principal do Sistema

UEM / DAA - Cadastro de Opções

Código: Tipo:

Descrição:

Início de solicitação: Fim de solicitação:

Setor Responsável: Aplicação geradora:

Custo do documento: Nº de solicitações/ano sem custo:

Justificativa obrigatória: Permitido via Internet:

Protocolo

Gera: Assunto:

Informações sobre a opção:

9.2.2 **Figura 13.** Tela de Cadastro de Opções de Solicitação.

DAR Requerimentos

Dados do Requerimento

CÓDIGO.: PROTOCOLO.:

R.A.: Ano de Ingresso:

Nome:

Tipo

Graduação
 Pós-Graduação

Opção

Código:

Descrição:

Sector Resp.: Custo: Qtde/Nº de Vias:

Data Pgto: Data Execução: Data Entrega:

Justificativa:

Observações da Solicitação:

Localizar F3 Novo F2 Procolo Fechar Excluir Recibo

9.2.3 Figura 14. Tela de manutenção de Solicitações.

DAR Pesquisar Solicitações

Pesquisar por

R.A. Procurar F3 Ano de Ingresso:

Opção Procurar F3

Setor

Período a

Situação

OBS: Somente os campos checados serão considerados na pesquisa

Fechar OK

9.2.4 Figura 15. Tela de Pesquisa de Solicitações.

Requerimento Acadêmico

Universidade Estadual de Maringá
DIRETORIA DE ASSUNTOS ACADÊMICOS
Protocolo Acadêmico

Processo Nº: F6
Rubrica:

PROTOCOLO

REQUERIMENTO ACADÊMICO

1. IDENTIFICAÇÃO

Nome do Aluno: ARIIVALDO CALDEIRA BOND
Registro Acadêmico: 12962 Ano de Ingresso: 1996
Curso: FORM.TECNOLEM PROC.DADOS
Situação Acadêmica: FORMADO EM 1999

2. REQUERIMENTO

Tipo de Requerimento: ATESTADO DE CONCLUSÃO DE CURSO

Justificativa:

Observações:

Informações Complementares: _____

0% Page 1 of 1

9.2.5 Figura 16. Requerimento Acadêmico.