4
Ambiente de Implementação

A implementação de um sistema para gestão de conteúdo de aprendizado envolve aspectos complexos como, por exemplo, a preparação do material de aprendizado, recuperação de objetos de aprendizado complexos, distribuição de responsabilidades, controle de acessos, controle de direitos autorais. Estas atividades, ainda que essenciais, estão fora do escopo deste trabalho, focado especificamente na atividade de armazenamento.

Ao dispor de um conjunto de sítios com capacidade de prover objetos de aprendizado a proposição desta dissertação é implementar uma arquitetura para compartilhar estes objetos de aprendizado utilizando a WEB. Para implementar esta arquitetura é proposto que cada sítio disponha de uma base de dados local e o conjunto seja administrado por uma base de dados federada propiciando assim uma visão uniforme para exploração via navegador (*browser*).

Neste capítulo, ao descrever as funcionalidades mínimas que deverão estar disponíveis para atender a arquitetura proposta, será feita uma consolidação dos assuntos já tratados. Para gerir a federação de banco de dados e para tratar de dados multimídia e XML serão descritas as características do sistema gerenciador de banco de dados IBM DB2. Para explorar dados utilizando navegador serão apresentadas características do produto IBM Net.Data. E, por último será apresentada a especificação da aplicação, denominada no esboço de arquitetura como LO_DB, desenvolvida para atualizar objetos de aprendizado em um banco de dados.

4.1.
Funcionalidades Mínimas

Neste tópico são relacionados os requisitos mínimos que devem estar disponíveis em cada sítio que irá prover recursos a serem compartilhados com os demais.

PUC-Rio - Certificação Digital Nº 0115653/CA

Para a tarefa de armazenar objetos de aprendizado em um ambiente de banco de dados deverá estar disponível uma aplicação que atenda as seguintes funcionalidades:

- Carregar objetos de aprendizado no banco de dados;
- Carregar a composição de objetos de aprendizado no banco de dados;
- Visualizar os dados de um objeto de aprendizado armazenado no banco de dados.

Para atender as necessidades de armazenamento local dos objetos de aprendizado, a relação de funcionalidades mínimas a serem providas pelo gerenciador de banco de dados é:

- Carga de arquivo XML em atributo de relação;
- Pesquisa e leitura de dados em cadeias XML por rótulos;
- Carga de arquivo de multimídia em atributo de relação;
- Pesquisa e leitura de atributos de dados multimídia (tempo, tamanho, tipo, etc);
- Pesquisa e leitura de dados multimídia.

Neste trabalho estamos considerando um ambiente distribuído, sobre o qual o usuário final deverá dispor de uma visão global, abstraindo-se da distribuição física dos dados. Das alternativas analisadas para atender esta funcionalidade foi escolhida a opção de banco de dados federado, e para atender esta opção deverão estar disponíveis no ambiente de implementação:

- Recurso para banco de dados federado;
- *Wrappers* para acesso aos gerenciadores de bancos de dados componentes.

A opção por utilizar produtos de banco de dados disponíveis comercialmente tornou necessário identificar pelo menos um gerenciador cujas funcionalidades atendessem as necessidades acima citadas. Como os principais gerenciadores de banco de dados disponíveis comercialmente têm funcionalidades bastante parecidas e, não sendo objetivo deste trabalho realizar comparação de desempenho, optou-se nesta dissertação por utilizar o IBM DB2.

Este gerenciador disponibiliza extensores para tratamento de dados multimídia (áudio, vídeo e imagem), de dados XML e dispõe de recursos para

PUC-Rio - Certificação Digital Nº 0115653/CA

gerenciamento de banco de dados federado, incluindo *wrappers* para acesso a diversos tipos de fontes de dados.

Além dos tópicos ligados ao armazenamento e distribuição dos dados, esta dissertação pretende abordar a exploração dos dados por navegador (*browser*). Para esta função são necessárias ferramentas para permitir a comunicação entre o servidor de banco de dados e o servidor WEB. Para implementar esta funcionalidade existem diversas alternativas como, por exemplo:

- Construir páginas utilizando HTML;
- Construir aplicações utilizando JAVA/Servlets
- Utilizar um produto disponível comercialmente como, por exemplo, IBM Net.Data

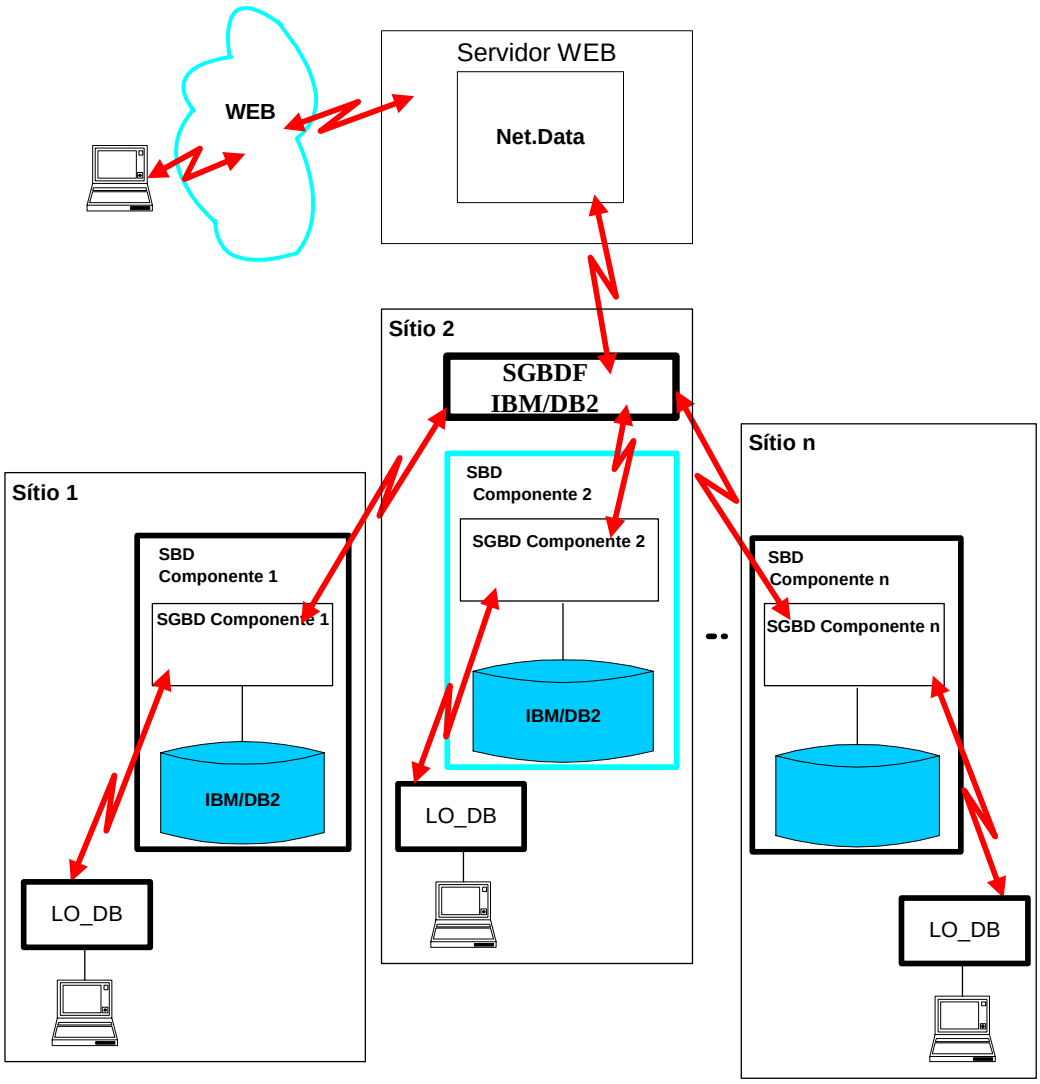

Figura 19. Arquitetura do ambiente de implementação

PUC-Rio - Certificação Digital Nº 0115653/CA

A alternativa oferecida pelo produto IBM Net.Data será adotada tendo em vista que atende aos requisitos mínimos necessários, oferece bons recursos para exploração de dados em diversas fontes, inclusive do IBM DB2.

Um exemplo da arquitetura proposta é apresentado na **Figura 19**, onde estão exemplificados dois sítios utilizando IBM DB2 como sistema de gerenciamento de banco de dados local e outros sítios com outros gerenciadores de banco de dados. Além dos sistemas de gerenciamento de banco de dados local, está indicado um sistema gerenciador de banco de dados federado e um servidor WEB ao qual está adicionada a camada IBM Net.Data para interface com o servidor federado. Em cada sítio está representada a existência da aplicação LO_DB responsável pela atualização local dos objetos de aprendizado.

As características mais relevantes deste ambiente de implementação serão detalhadas nos próximos itens.

PUC-Rio - Certificação Digital Nº 0115653/CA

4.2. Sistema de Banco de Dados Federado IBM DB2

Um *sistema federado* IBM DB2 consiste de um servidor central, chamado servidor federado, de um gerenciador de banco de dados IBM DB2 e de um conjunto de diversas fontes de dados para as quais são enviadas consultas. As fontes de dados são consultadas utilizando *wrappers*. Diversos *wrappers* são disponibilizados para acesso a gerenciadores relacionais e a fontes de dados não relacionais.

Um sistema federado do DB2, conforme esboço de arquitetura apresentado na **Figura 20** [Ibm03a], disponibiliza os seguintes recursos:

- Acesso a dados relacionais como, por exemplo, Oracle, Sybase, MS SQL Server e a dados não relacionais como servidor de pesquisa BLAST, arquivo XML, ou planilha EXCEL;
- Acesso homogêneo aos dados locais e aos dados remotos, como se todos estivessem armazenados localmente na base de dados federada;
- Transparência de localização que se estende a eventuais migrações de objetos de banco de dados. Se um objeto remoto mudar de localização,

as referências a este objeto poderão ser atualizadas sem que seja preciso alterar as aplicações que o utilizarem;

- Compensação para limitações de processamento nas fontes de dados remotas. A escolha da alternativa para a compensação é feita durante o processo de otimizar consultas pelo IBM DB2. Esta escolha pode ser pela execução local no servidor federado, ou pela execução de processamento alternativo remoto. Por exemplo, uma fonte de dados não suporta SQL recursiva, o DB2 pode executar localmente SQL recursiva contra os dados da fonte de dados, ou a falta de uma função matemática pode ser atendida por um outro recurso remoto que esteja disponível;

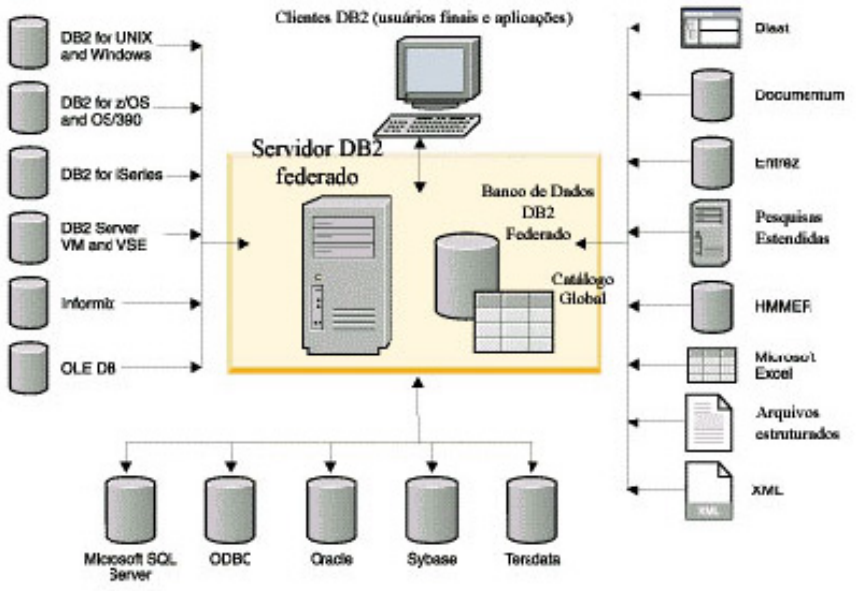

Figura 20. Arquitetura de um DB2 federado [Ibm03a]

- A distribuição de requisições às fontes de dados utiliza estatísticas e outras informações que estejam disponíveis sobre as fontes de dados para melhoria do desempenho;
- Autonomia para as fontes de dados. O servidor federado pode enviar uma consulta a uma fonte de dados, enquanto outras aplicações estejam acessando os dados. O sistema não monopoliza ou restringe acessos às fontes de dados, além das restrições de integridade e exclusividade que forem explicitadas.

PUC-Rio - Certificação Digital Nº 0115653/CA

Maiores detalhes sobre *wrappers* e catálogo global serão apresentados em seguida.

4.2.1. *Wrappers*

Wrappers são rotinas através das quais os comandos submetidos ao gerenciador federado são repassados para as fontes de dados.

Um invólucro executa diversas tarefas, entre as quais:

a. **Conexão à fonte de dados:** Normalmente é utilizada uma interface padrão da fonte de dados para estabelecer a conexão;
b. **Submissão de requisições para as fontes de dados:** Se a fonte de dados possui suporte para linguagem SQL a requisição é submetida em SQL. Caso contrário, a requisição será convertida para a linguagem nativa da fonte de dados ou então para uma série de chamadas de API;
c. **Recepção dos resultados da fonte de dados**: utilizando uma interface padrão;
d. **Informar ao servidor federativo sobre tipos de dados locais**: Os mapeamentos de tipos de dados de usuário são armazenados no catálogo global;
e. **Informar ao servidor federativo sobre funções locais**: Os mapeamentos de funções de usuário são armazenados no catálogo global.

4.2.2. Catálogo Global do IBM DB2 Federado

Quando uma fonte de dados é acrescentada a uma base de dados federada, as informações a respeito de índices, que esta fonte de dados possui no servidor local, são acrescentadas a um catálogo global. A otimização de consultas do gerenciador federado irá considerar esta informação para contabilizar custos de consultas e despachar requisições distribuídas.

O catálogo de informações globais é um conjunto de metadados, e é denominado de especificação de índices. Um servidor federado irá criar a especificação de índices para uma fonte de dados nos seguintes casos:

PUC-Rio - Certificação Digital Nº 0115653/CA

- A tabela possui índices locais;
- A visão tem índices associados no catálogo remoto;
- A fonte de dados possui catálogo remoto para fornecer informações de índices.

4.3. Tratamento de Dados XML e Multimídia

Para tratamento dos dados XML e multimídia nos sítios que tiverem o gerenciador de banco de dados IBM DB2 devem ser instaladas as seguintes extensões:

- AVI[19]: disponibiliza recursos do tipo UDT[20] e UDF[21] para tratamento de dados de áudio, vídeo e imagens;
- XML: disponibiliza recursos do tipo UDT e UDF para tratamento de dados de dados XML;

Após a instalação destas extensões é necessário que sejam executados comandos associando a base de dados e as tabelas que irão conter os dados aos extensores.

No caso dos dados XML é necessário que sejam indicadas eventuais tabelas paralelas a serem geridas automaticamente pelo gerenciador para melhoraria do desempenho das consultas.

O acesso às funcionalidades providas pelas extensões do IBM DB2 permite a execução de comandos SQL como no exemplo abaixo:

```
SELECT db2xml.extractVarchar (LO_XML,'/lom/general/description/langstring') ,
        db2xml.extractVarchar (LO_XML,'/lom/rights/description/langstring') ,
        cast(mmdbsys.thumbnail (lo_image) as blob(10000)),
       cast(mmdbsys.thumbnail (lo_video) as blob(10000)),
       mmdbsys.comment(lo_audio), id
FROM db2admin.lo_tab
WHERE id = '$(id)'
```

[19] AVI – *Audio, Video and Image*

[20] UDT - *User Defined Type*

[21] UDF *User Defined Function*

PUC-Rio - Certificação Digital Nº 0115653/CA

Neste comando estão sendo recuperadas:

- Informações identificadas por *rótulos* na cadeia de dados XML por meio da utilização de UDF apropriada (*extractVarchar*), que recebe como parâmetros um atributo de tabela (*LO_XML*) e um rótulo XML, por exemplo, '/lom/rights/description/langstring'.
- Informações sobre dados de áudio, imagem e vídeo, por meio da utilização de UDF apropriadas (*thumbnail*, *comment*), que recebem como parâmetros atributos de tabela (*lo_image, lo_vídeo, lo_audio*).

Os prefixos *db2xml*, *mmdbsys*, *db2admin* são para indicar o esquema a que pertencem os objetos.

Para armazenar os dados XML o extensor oferece duas opções:

a) Os dados serão armazenados em um único atributo de uma tabela;
b) Os dados serão armazenados em várias tabelas.

Estas opções são definidas em um arquivo de parametrização denominado DAD[22], conforme apresentado esquematicamente na **Figura 21**.

As seguintes UDF são disponibilizadas para armazenamento de dados XML:

- XMLVarcharFromFile (): utilizada para armazenar um campo tipo *varchar* a partir de um arquivo;
- XMLCLOBFromFile (): utilizada para armazenar um campo tipo *clob* a partir de um arquivo;
- XMLFileFromVarchar (): utilizada para armazenar um arquivo a partir de um campo tipo *varchar*;
- XMLFileFromCLOB (): utilizada para armazenar um arquivo a partir de um campo tipo *clob*;

[22] DAD – Document Access Definition

PUC-Rio - Certificação Digital Nº 0115653/CA

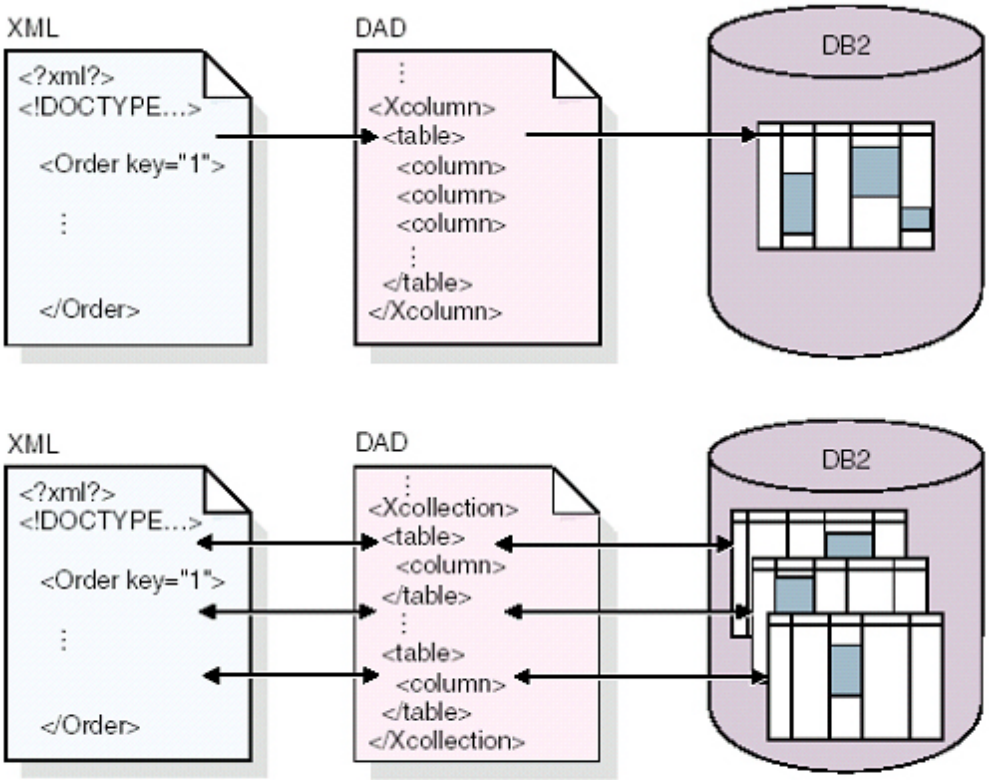

Figura 21. Opções de armazenamento de dados XML.

As seguintes UDF são disponibilizadas para recuperar dados XML:

- Content (): utilizada para recuperar um conteúdo a partir de um *XMLFil*e para um *clob;*
- Content (): utilizada para recuperar um conteúdo a partir de um *XMLVarchar* para um arquivo externo;
- Content (): utilizada para recuperar um conteúdo a partir de um *XMLCLOB* para um arquivo externo.

Para extrair dados associados a rótulos as seguintes UDF são disponibilizadas:

- extractInteger ()
- extractSmallInt ()
- extractDouble ()
- extractReal ()
- extractVarchar ()
- extractChar ()
- extractCLOB ()

PUC-Rio - Certificação Digital Nº 0115653/CA

- extractTime ()

Ao armazenar dados de áudio, vídeo e imagem no banco de dados, na tabela de usuário são armazenados apenas ponteiros (*handles*). O objeto AVI é armazenado em tabelas administrativas de suporte, a menos que a opção para armazenar o objeto AVI seja em arquivo externo, sendo neste caso armazenado o identificador deste arquivo nas tabelas administrativas. Esta arquitetura está esboçada na **Figura 22.** Existem várias tabelas administrativas implementadas para apoio aos extensores. As primeiras são utilizadas para armazenar as informações sobre as tabelas de usuário e as colunas utilizadas para armazenar dados AVI. Estas tabelas podem referenciar outras tabelas que armazenam atributos específicos a cada tipo de dados. Por exemplo, para uma imagem o extensor terá dados específicos como largura, altura e número de cores, assim como, outros dados gerais.

PUC-Rio - Certificação Digital Nº 0115653/CA

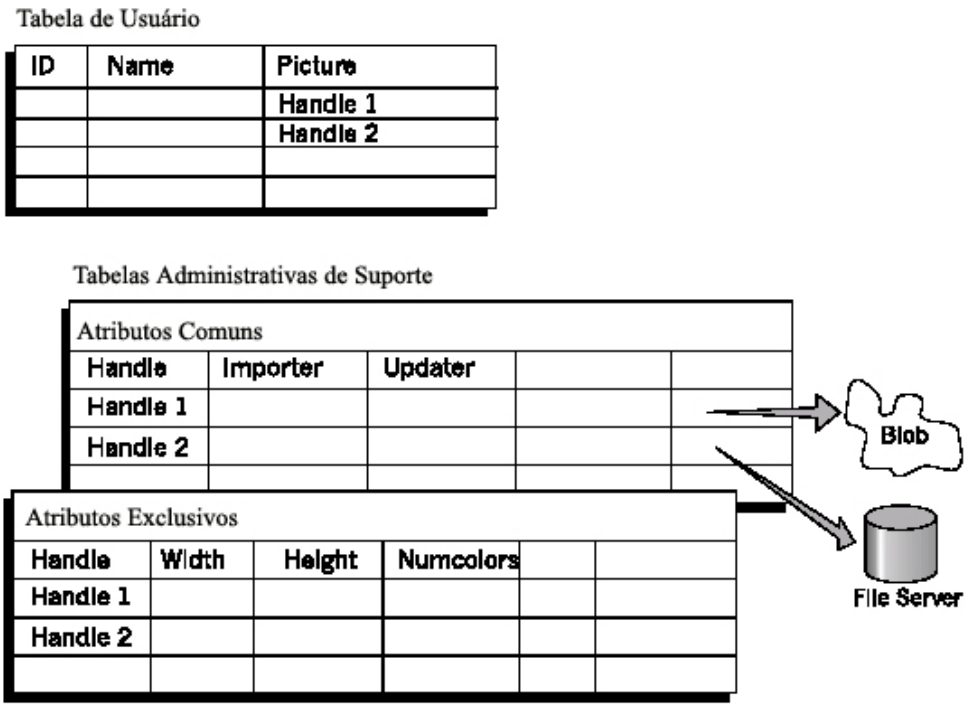

Figura 22. Armazenamento de dados AVI.

A título de exemplo, as UDF disponibilizadas para recuperar ou atualizar dados pelo extensor imagem são:

- Comment (): recupera ou atualiza comentários do usuário;
- Content (): recupera ou atualiza o conteúdo de uma imagem;

- DB2Image (): armazena o conteúdo de uma imagem;
- Filename (): recupera o nome do arquivo que contem a imagem;
- Format (): recupera o formato da imagem (GIF, JPEG,...);
- Height (): recupera a altura da imagem em *pixels*;
- NumColors (): recupera o número de cores usadas em uma imagem;
- Size (): recupera o tamanho da imagem em octetos;
- Width (): recupera a largura da imagem em *pixels*;

4.4. Disponibilidade de Dados via WEB

Para disponibilizar os dados via WEB diversas opções estão consagradas:

- Construção de telas HTML;
- Construção de aplicações com JAVA/Servlets;
- Utilização de camadas intermediárias providas por produtos comerciais.

Como os dados dos objetos de aprendizado possuem características complexas, a sua exploração a partir do armazenamento em banco de dados também iria exigir algum esforço de desenvolvimento.

Por sua vez, o foco desta dissertação é o armazenamento de objetos de aprendizado utilizando tecnologia de banco de dados, buscando mostrar como o uso desta tecnologia pode melhorar o compartilhamento dos objetos de aprendizado.

Dentro deste enfoque para minimizar o esforço de desenvolvimento e ao mesmo tempo prover a funcionalidade optou-se nesta proposta pela utilização do produto IBM Net.Data.

O produto IBM Net.Data é uma camada que permite o desenvolvimento de telas a serem exploradas por meio do navegador, sem que sejam necessárias profundas imersões em HTML, facilitando a exploração de dados via WEB.

Na **Figura 23** a seguir é fornecido o esquema de funcionamento desta camada em relação aos demais componentes.

PUC-Rio - Certificação Digital Nº 0115653/CA

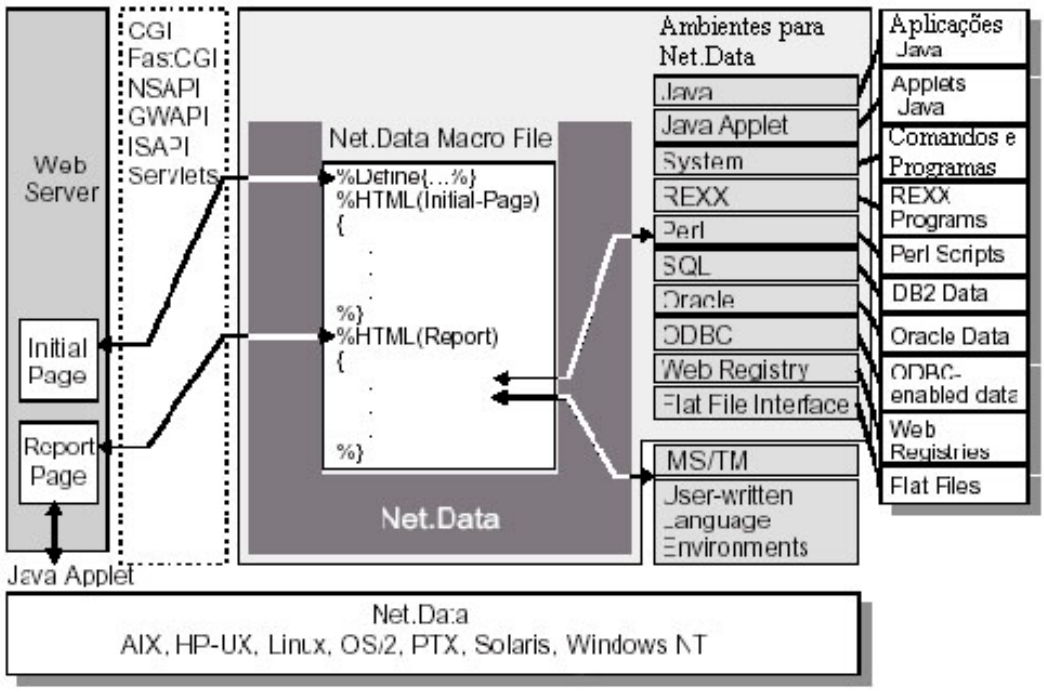

Figura 23. Arquitetura do Net.Data

Em termos de arquitetura o produto IBM Net.Data funciona abaixo do servidor WEB fazendo a interface com diversas fontes de dados. Sua codificação permite a montagem de páginas HTML com dados oriundos de diversas fontes como, por exemplo, programas JAVA, bancos de dados e arquivos, entre outras. Sua codificação é relativamente simples dispondo de diversos recursos de linguagem como, por exemplo, utilização de comandos SQL, sub-rotinas, comandos lógicos, comandos compostos, etc. Um módulo Net.Data é composto por duas partes: declarações e apresentações, conforme apresentado na **Figura 24.**

PUC-Rio - Certificação Digital Nº 0115653/CA

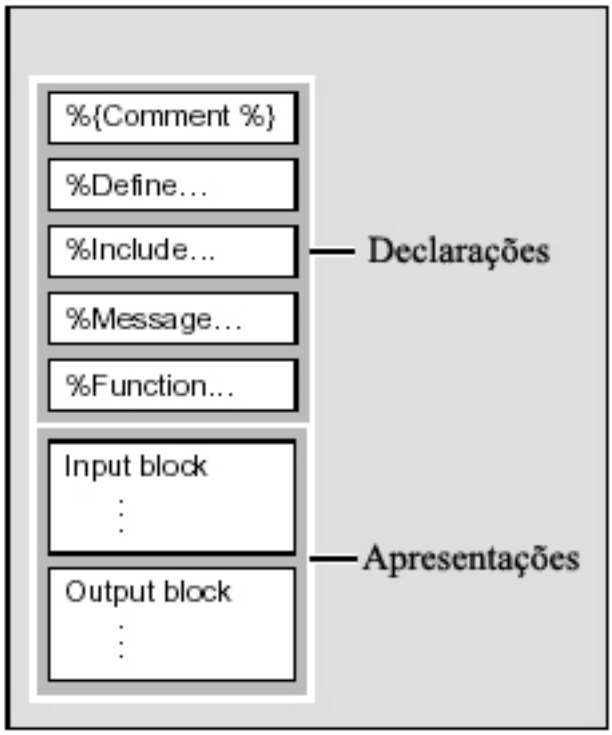

Figura 24. Estrutura de um módulo Net.Data.

Segue um breve sumário dos comandos disponíveis:

- Bloco **DEFINE**: tem por objetivo definir as variáveis que serão utilizadas posteriormente nos blocos HTML;
- Bloco **FUNCTION:** contem as declarações de funções que serão utilizadas por blocos HTML. As funções são executadas com os recursos do ambiente e podem executar programas, consultas SQL ou procedimentos armazenados (*stored procedures*);
- Bloco **HTML**: definem o aspecto da página WEB, referencia as variáveis e chama funções. São utilizados como pontos de entrada e de conclusão da execução do arquivo Net.Data;
- Bloco **XML**: utilizado para transferir conteúdo em XML para uma outra aplicação ou navegador;
- Blocos **IF** e **WHILE**: são blocos utilizados para criar processamento condicional e processamento repetitivo.

Um exemplo de codificação de módulo em Net.Data é apresentado no item **Apêndices - 10.4. Exemplo de Macro Net.Data.** .

PUC-Rio - Certificação Digital Nº 0115653/CA

4.5. Aplicação LO_DB

Esta aplicação, a ser instalada em cada sítio participante da federação, tem por objetivo prover uma interface para atualizar, em banco de dados, informações referentes a objetos de aprendizado.

Como definição de escopo para esta aplicação foi considerado que a base de dados será alimentada com informações que serão produzidas por outros módulos externos a esta aplicação. Ou seja, não cabe a esta aplicação produzir o material dos objetos de aprendizado.

A aplicação deverá prover as seguintes funcionalidades mínimas:

- Uma interface gráfica para entrada e consulta de dados;
- Consulta ao catálogo local de objetos de aprendizado;
- Deverão ser armazenados dados não estruturados como áudio, vídeo, imagens e XML;
- As informações descritivas de um objeto de aprendizado seguirão a padronização LOM em formato XML.

Na **Figura 25** é apresentado um exemplo do projeto físico da aplicação LO_DB, onde são destacadas as presenças de um servidor de banco de dados e de micros conectados para a interface de usuário.

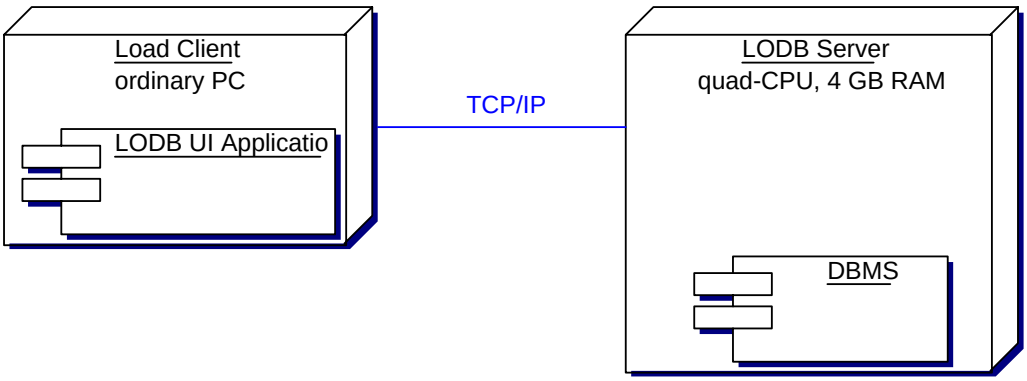

Figura 25. Arquitetura física da aplicação LO_DB

A descrição detalhada desta aplicação está apresentada no item **Apêndices - 10.3. Documentação da Aplicação LO_DB.**

PUC-Rio - Certificação Digital Nº 0115653/CA