Sistemas de Anotações em Biossequências

Melissa Lemos
e-mail: melissa@inf.puc-rio.br

Luiz Fernando Bessa Seibel
e-mail: seibel@inf.puc-rio.br

Marco Antônio Casanova
e-mail: casanova@inf.puc-rio.br

PUC-RioInf.MCC04/03 Fevereiro, 2003

Abstract

The human genome project is a program to map and sequence the entire human genome. A number of model organisms were selected for complete sequencing, partly in order to develop new technology for mapping, sequencing and sequence analysis. In addition, the sequences from these genomes were expected to facilitate the elucidation of the functions of genes and sequences in the human genome.

One of the most important tasks of genome projects is the interpretation of the experimental data in order to obtain biological knowledge of this data. The researchers execute this task searching external data sources, executing analysis programs on its sequences and generating manual annotations in accordance with its interpretation of the data.

This monograph makes a survey of the existing annotation systems and presents the project and the implementation of 'BioNotes', which is a new system proposal that aims at bringing advantages to the researchers in this area.

Keywords: Annotation Systems, bioinformatic, database, sequence analysis algorithms, BioNotes.

Resumo

O Projeto Genoma Humano tem como objetivo o mapeamento e sequenciamento do genoma humano inteiro. Além do genoma humano, alguns organismos modelo foram selecionados para serem sequenciados e ajudarem a desenvolver novas tecnologias e a elucidar o genoma humano.

Uma das tarefas mais importantes dos projetos genoma é a interpretação dos dados experimentais para se obter conhecimento biológico destes dados. Os pesquisadores executam esta tarefa consultando as fontes de dados externas, executando programas de análise em suas biossequências e gerando anotações manuais de acordo com sua interpretação dos dados.

Este trabalho faz um levantamento sobre os sistemas de anotações existentes e apresenta o projeto e a implementação do sistema BioNotes, uma nova proposta que visa trazer vantagens aos pesquisadores da área.

Palavras-chave: Sistema de anotações, bioinformática, bancos de dados, algoritmos de análise de sequências, BioNotes.

Índice

1 Introdução

O Projeto Genoma Humano iniciou-se oficialmente em 1990 como um programa para mapear e sequenciar o genoma humano inteiro, obtendo as 3 bilhões de bases da cadeia que representam o DNA humano, configurando-se como um esforço que revolucionaria a biologia e a medicina. Além do genoma humano, alguns organismos-modelo foram selecionados para serem sequenciados. Isto foi feito para que novas tecnologias de mapeamento, sequenciamento e análise de biossequências (ou, simplesmente, sequências) fossem desenvolvidas e, além disso, porque as biossequências destes organismos facilitariam a elucidação de funções de genes e sequências do genoma humano (Sterky, F., Lundeberg, J, 2000).

Um dos principais problemas do sequenciamento em larga escala é que seus métodos (Maxam-Gilbert ou Sanger (Cooper, N., 1994, Watson, J. et. al., 1987) sequenciam somente uma pequena parte do DNA. Isto é, mesmo que o sequenciamento seja feito por uma máquina automatizada ou por um método manual, a maior subcadeia de DNA que pode ser determinada com qualidade, em um procedimento em laboratório, possui cerca de 500 bases. Desta forma, para sequenciar cadeias mais longas, ou um genoma inteiro, o DNA precisa ser dividido em vários fragmentos pequenos ou *reads* que são sequenciados individualmente e depois montados para obter a cadeia inicial completa.

Infelizmente este processo não é simples. Por exemplo, podem acontecer erros no sequenciamento ou falhas durante a quebra e cópia de *reads*, o que faz com que várias partes do genoma não sejam sequenciadas, dificultando a montagem completa do genoma. Normalmente consegue-se juntar partes dos *reads*. Cada conjunto de *reads* que se junta é chamado de contig. O objetivo é unir todos os contigs e formar um único contig, que é o genoma.

Em certos casos, o genoma é tão grande (geralmente isto acontece para algumas plantas, como a cana de açúcar (SUCEST, 2003)) que não vale a pena criar um projeto para o seu sequenciamento completo. Nestes casos, somente as partes do genoma que se traduzem em proteínas, que são consideradas mais importantes, são sequenciadas. Estas partes do genoma são chamadas de EST (*Expressed Sequence Tags*).

Uma das tarefas mais importantes dos projetos genoma é a interpretação dos dados experimentais para se obter conhecimento biológico destes dados. Os pesquisadores executam esta tarefa consultando as fontes de dados externas, executando programas de análise em suas sequências (contigs e *reads*) e gerando anotações manuais de acordo com suas interpretações obtidas na bancada do seu experimento e na análise através das comparações com as fontes de dados externas e da execução dos programas aplicativos.

Existem atualmente várias fontes de dados que armazenam as sequências e vários programas de análise destes dados. Como o avanço destes projetos, muitos dados estão sendo obtidos e armazenados em bancos de dados diferentes e vários programas estão sendo executados para analisar estes dados.

Ao sequenciar uma pequena sequência, o pesquisador pode submetê-la para uma fonte de dados pública, como o Genbank (Genbank, 2002), ou armazená-la em sua própria fonte de dados. O pesquisador pode fazer algumas anotações sobre a sequência, como, por exemplo, o laboratório que a sequenciou, a placa e o gel utilizado para o sequenciamento. Da mesma forma, o pesquisador também pode submeter os contigs e fazer anotações sobre eles nas fontes de dados. Estas anotações são manuais, isto é, escritas manualmente por um pesquisador.

Existem diversos programas que analisam os contigs. A comparação de sequências compara o contig com as sequências de uma fonte de dados para descobrir quais sequências se parecem mais com o contig. Ao descobrir sequências similares, os pesquisadores analisam suas anotações para descobrir as funções desconhecidas do contig. Além disso, existem programas que descobrem onde estão padrões importantes, como as ORF's (possíveis genes) nos contigs, os tRNA's (RNA transportador), as repetições, etc. Todos estes programas geram resultados que são anotações automáticas.

Dependendo se o projeto do genoma é completo ou de ESTs, as análises feitas e, consequentemente, as anotações variam.

É possível classificar as anotações em duas classes: automática e manual.

As anotações automáticas são obtidas das fontes de dados externas e dos resultados da execução dos aplicativos. As anotações manuais são feitas pelas comunidades de pesquisadores.

Um dos grandes desafios da bioinformática é ajudar os cientistas a minerar os dados, converter os dados experimentais em informações biológicas relevantes, perceber, explorar e testar suas idéias rapidamente, acessar rapidamente as anotações armazenadas nas fontes de dados externas, executar os programas e obter rapidamente as anotações geradas por eles, analisar as anotações existentes até então (facilitada por uma interface apropriada) e gerar novas anotações manualmente.

Diante deste desafio, a bioinformática precisa lidar com inúmeras fontes de dados heterogêneas, inúmeros aplicativos e uma enorme quantidade de dados.

Este trabalho trata de um estudo sobre os sistemas de anotações disponíveis e a apresentação do projeto e implementação de uma nova proposta que visa trazer vantagens aos pesquisadores da área. Essa nova proposta foi construída a partir da infra-estrutura do *framework* de integração de fontes de dados e de aplicativos da biologia molecular denominado BioAXS.

Este trabalho está estruturado da seguinte forma. O capítulo 2 apresenta um estudo dos sistemas de anotação existentes. O capítulo 3 descreve um novo sistema de anotações, cuja implementação é chamada de BioNotes. A proposta contempla a modelagem do banco de dados e o projeto da aplicação (usando os diagramas da UML). O capítulo 4 compara entre os sistemas de anotações, buscando avaliar e enquadrar o sistema que implementado, apresentando os comentários finais do estudo.

2 Trabalhos Relacionados

Existem diversas ferramentas de anotação em biologia que se diferem em vários aspectos, entre elas estão:

- Artemis (Artemis, 2002)(Rutherford,K. et.al., 2000),
- DAS (DAS, 2002)(Stein, L., Dowell, R., 2002),
- CeleraBrowser (CeleraBrowser, 2002),
- EDITtoTrEMBL (Möller, S. et. Al.),
- GASP (Reese, M.G., 2000),
- GenDB (GenDB, 2000)(Goesman, A. et.al., 2002),
- GeneMine (GeneMine, 2002)(Lee, C., Irizarry, K., 2000),
- GeneQuiz (GeneQuiz, 2002)(Hoersch, S. et.al., 2000) (Andrade, M.A. et. al., 1999),
- Apollo (Apollo, 2002),
- GGB (GGB, 2002),
- Imagene (Imagene, 2002)(Médigue, C. et.al. 1999)(Bisson, G., Garreau, A.,1995),
- MAGPIE (MAGPIE, 2002)(Gaasterland,T., Sensen, C.W., 1996)
- Manatee (Manatee, 2002),
- Pedant (Pedant, 2002)(Frishman, D. et.al., 1998) (Frishman, D. et.al., 2001) (Frishman, D.,Mewes, H.W, 1997a, 1997b),
- VisualGenome (VisualGenome, 2002)
- Pseudomonas aeruginosa community annotation project (PseudoCAP, 2003),
- Community Annotation Project (CAP, 2003),
- Alternative Splicing Annotation Project (ASAP, 2003),
- Genestream (GeneStream, 2003),
- Cancer Annotation Project (Cancer Annotation Project, 2003),
- Ensembl Genome Annotation Project (Ensembl, 2002),
- NCBI's Genome Annotation Project (NCBI, 2003),
- *Vibrio vulnificus YJ016* annotation project (*Vibrio vulnificus,* 2003).

Este capítulo descreve brevemente algumas das ferramentas listadas acima.

Artemis

O Artemis permite a visualização de dados extraídos do Genbank (GENBANK, 2002) e do EMBL (EMBL, 2002) e a utilização de programas externos como BLAST (BLAST, 2002) (Altschul, S.F. et.al., 1990) e FAST (Pearson, W. R, 1991). Os programas e os dados não foram agregados à arquitetura para manter sua portabilidade. Para que o sistema seja usado é necessário que o usuário tenha os dados e os programas em sua máquina.

As anotações que são tratadas neste sistema estão baseadas na *feature table* definida pelo NCBI.

Os usuários conseguem fazer anotações sobre os dados.

Este sistema foi escrito em Java, tem interface Web implementada através de um applet.

DAS

O DAS é um sistema em que o usuário (cliente) seleciona um único servidor de referência (genoma) e vários servidores de anotações. O cliente DAS combina a informação retornada dos servidores em uma única tela gráfica. A Figura 1 mostra um exemplo de uma arquitetura básica de um DAS. O servidor *Washington University Genome Sequencing Center* foi projetado para ser o servidor de referência. Outros servidores de anotação como Ensembl, Whitehead e Sean Eddy Laboratory provêem informações das sequências que estão no servidor de referência. O cliente, no caso o Cold Spring Harbor Laboratory, obtém os dados de vários servidores, gera automaticamente uma visão integrada destes dados e os representa graficamente em uma única tela.

As anotações no DAS estão associadas à posições (marcadas por início e fim) das sequências de acordo com o organismo. Em alguns projetos as sequências podem ser de cromossomos inteiros, enquanto em outros elas podem ser contigs ou reads. Por exemplo no caso do projeto do *C.elegans* há seis cromossomos. Cada cromossomo é formado por vários contigs, sendo que cada contig possui uma ou mais sequências. Uma anotação pode estar relacionada a posições marcadas em um cromossomo, em um contig ou em um read.

As anotações possuem significado biológico definidos de acordo com a *feature table* desenvolvida pelo NCBI e possuem uma descrição de como a anotação foi obtida (que pode ser uma referência para um programa de análise). Caso o usuário queira anotar algo diferente

do definido pela *feature table* é permitido adicionar novos tipos de anotações sem restrição de vocabulário.

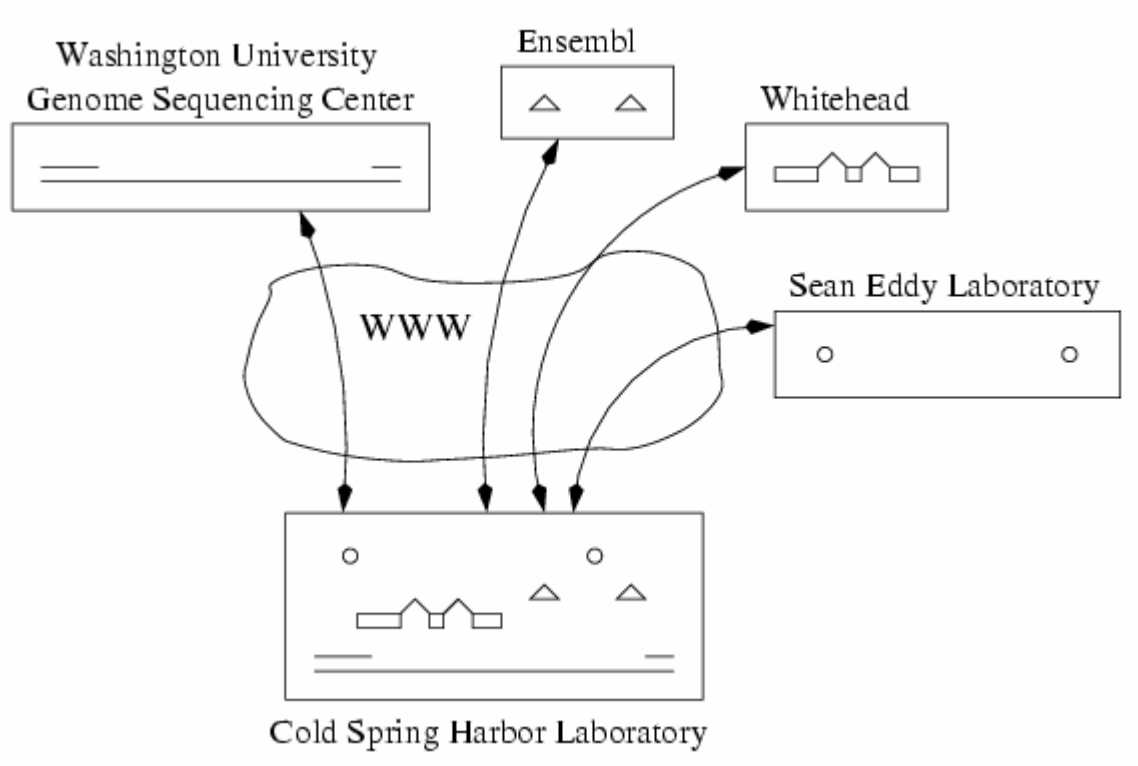


Figura 1. Arquitetura do DAS.

Celera Browser

O sistema Celera Genome Browser permite que os pesquisadores visualizem os dados públicos da Celera, os seus dados (proprietários) armazenados na Celera, os dados do genoma humano e de rato da Celera. Além disso os pesquisadores podem fazer suas anotações, armazená-las em sua máquina, visualizar o alinhamento entre sua sequência e a sequência consenso, executar e visualizar alinhamentos com BLAST, ajustar sítios de *splicing*, remover, modificar ou criar exons e transcriptomas.

O Genome Browser consiste em três módulos: uma interface gráfica de usuário, uma aplicação servidora e um banco de dados relacional. As anotações podem ser salvas no formato Celera Exchange (GAME XML).

Não foi possível encontrar referências que descrevessem melhor o sistema. Assim, muitas questões ficaram em aberto, como por exemplo, se a interface é Web, qual é o sistema de gerência de banco de dados utilizado, se o sistema permite anotação distribuída, entre outras.

EDITtoTrEMBL

O EBI desenvolveu um sistema de anotação, chamado de EDITtoTrEMBL, para os dados da fonte de dados TrEMBL (TrEMBL, 2002). Este sistema permite executar aplicativos em cada sequência armazenada no TrEMBL, sendo que a saída destes programas pode ser selecionada e submetida como entrada para novos aplicativos, permitindo derivar novas e melhores anotações.

Foram acoplados ao sistema várias fontes de dados externas como ENZYME (ENZYME, 2002), PROSITE (PROSITE, 2002), PRINTS (PRINTS, 2002) e aplicativos como TMHMM (TMHMM, 2002) e NNPSL (NNPSL, 2002).

O EDITtoTrEMBL trata a automação das anotações como um problema de workflow. De acordo com este workflow, para cada registro/sequência do TrEMBL vários aplicativos são executados. Diferente de outras ferramentas de anotação (como GeneQuiz, MAGPIE e Pedant) que pré-calculam resultados de vários aplicativos, no EDITtoTrEMBL só são executados os aplicativos que o usuário considere mais adequados.

Além disso, o EDITtoTrEMBL não está fortemente focado na apresentação das anotações, como outras ferramentas de anotação, por exemplo GAIA (GAIA, 2002) e Genotator (Genotator, 2002)(Harris, N.L.,1997).

O EDITtoTrEMBL foi programado em Java e usa o mecanismo RMI para a comunicação e distribuição de processos.

GenDB

O GENDB é uma ferramenta de anotação de genoma *open-source* que possui seu projeto orientado a objeto, sendo que os objetos são mapeados para um banco de dados relacional MySQL utilizando O2DBI (O2DBI, 2002). Os objetos são descritos em Perl e toda comunicação com o banco de dados é feita pela camada O2DBI.

No GENDB são feitas anotações automáticas pré-calculando resultados de aplicativos como BLAST, TMHMM e HMMPfam (HMMPfam, 2003). Destes resultados, apenas um resumo é armazenado. Para um maior detalhamento, o usuário conta com o sistema Sequence Retrieval System (SRS, 2002).

São executados outros aplicativos, como GLIMMER (GLIMMER, 2002), tRNAScan (tRNAScan, 2002) e Orpheus (Orpheus, 2002) e acessadas fontes de dados externas como PFam (Pfam, 2002).

É possível armazenar anotações manuais dos pesquisadores.

O sistema atual pode ser executado em máquina Unix ou Linux do usuário. Para isso, é necessária a instalação do SGBD MySQL ou Postgres, acesso ao sistema SRS, e acesso às fontes de dados e aplicativos externos, como GLIMMER e BLAST.

A Figura 2 exemplifica o modelo de dados utilizado no sistema. É possível notar que são armazenados resultados de aplicativos como Glimmer (tabela que armazena dados de ORF's) e que as anotações feitas referem-se às ORF's (atributo *orf_id* na tabela *annotation*).

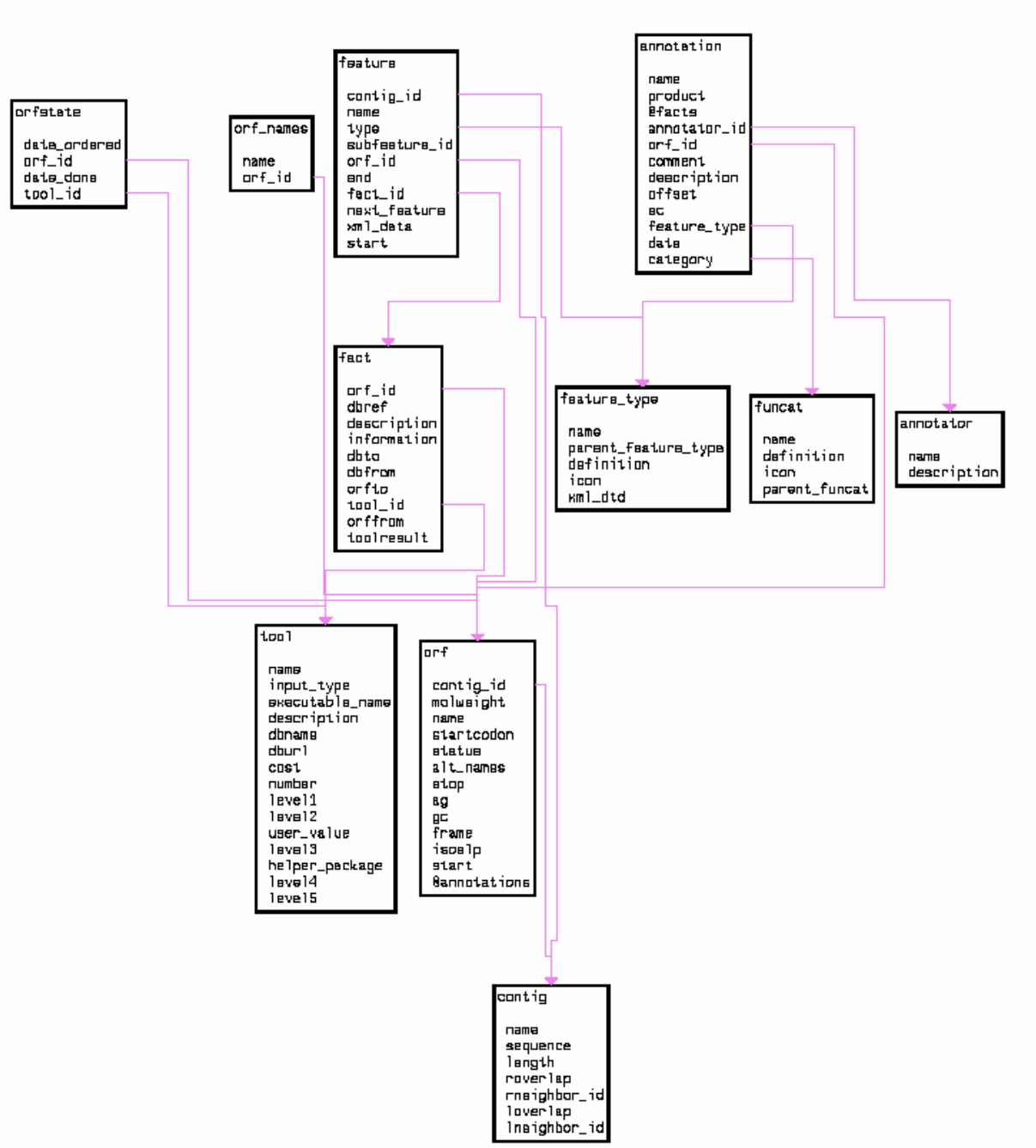


Figura 2. Modelo de dados do GenDB.

GeneMine

O GeneMine é um sistema que faz consultas às fontes de dados externas, executa os aplicativos externos via Web e apresenta as informações obtidas em uma interface gráfica

(que não é Web) para o usuário. O objetivo do sistema é apresentar ao usuário informações com links para detalhes destas informações, para que ele possa interpretá-las.

Como exemplo de fontes de dados externas, destacam-se SWISSPROT (SWISSPROT, 2002), PIR (PIR, 2002), PROSITE, PDB (PDB, 2002). A busca nestas bases não é simples, já que são heterogêneas e é necessário conversão entre os formatos existentes nas bases de dados para o formato que se deseja apresentar ao usuário.

Como exemplo de aplicativo externo, tem-se o BLAST. As sequências de consulta são enviadas aos servidores que executam a busca contra seus bancos de dados e retornam os alinhamentos com as sequências homólogas. O GeneMine lê os alinhamentos e apresenta-os em uma janela. Este tipo de anotação é chamada de *BLASTHomology*.

Diferente de outros sistemas, o GeneMine não possui um *data warehouse* e, consequentemente, não armazena as informações obtidas das fontes de dados e dos aplicativos. Mas, ele possui um sistema de cache para evitar o acesso na Web desnecessário.

O sistema foi escrito em Perl e atualmente pode ser executado em estações Silicon Graphics com IRIX 5.3 ou posterior.

GeneQuiz

GeneQuiz é um sistema de análise de sequências biológicas, que trata desde a sequência de proteína até sua função bioquímica, usando uma variedade de aplicativos e de fontes de dados de proteínas e DNA.

O GeneQuiz consiste em quatro módulos: (1) atualização do banco de dados; GQupdate, (2) busca; GQsearch, (3) interpretação; GQreason e (4) e visualização em um browser; GQbrowse.

GQupdate

O módulo GQupdate é responsável por: (1) transferir os dados, (2) refomatar os dados quando necessário, (3) atualizar os bancos de dados locais.

PDB, SWISSPROT, Ensembl (Ensembl, 2002), PIR, TrEMBL, EMBL, BLOCKS (BLOCKS, 2002) e Prosite são exemplos de fontes de dados tratadas pelo GeneQuiz.

Após a transferência dos dados atualizados, o GQupdate executa automaticamente todos os procedimentos de reformatação para fazer com que os dados fiquem no formato adequado para execução de vários programas de busca, como o BLAST.

Este módulo usa o SRS para ter referência cruzada às fontes de dados externas para que o usuário possa examinar com mais detalhe as informações de cada dado.

GQsearch

Este módulo é responsável pela análise das sequências. Para cada nova sequência de consulta, um diretório novo é criado onde todos os arquivos com resultados de programas de análises são armazenados. Resultado de programas como BLAST e FAST são colocados nestes diretórios.

Outros programas de análise de sequência também são executados como para descoberta de ORF´s e descoberta de outros padrões como de regiões *coiled-coil* e segmentos *transmembrane*.

Para que fosse possível a execução destes programas vários *parsers* foram construídos.

GQreason

Este módulo é responsável pela atribuição funcional às proteínas.

Após a busca nas fontes de dados externas e a execução dos aplicativos, os dados obtidos são armazenados em uma tabela "feature" no banco de dados relacional. Esta tabela possui um resumo dos resultados dos programas para uma determinada sequência.

Através da análise destes dados, o GeneQuiz gera um dicionário que associa palavras-chave de uma sequência com um conjunto de classes funcionais. As palavras-chave utilizadas são as definidas pelo SWISSPROT.

Além disso, este módulo avalia os resultados dos aplicativos e os dados obtidos nas fontes de dados externas. Por exemplo, é feito um controle da qualidade dos dados das fontes de dados, mostrando através de uma classificação quais dados de quais fontes de dados são mais confiáveis. Analogamente, resultados de busca feito pelo BLAST e FAST são avaliados.

Finalmente, no último passo, as tabelas "feature" são resumidas com resultados compreensíveis.

GQbrowse

Este módulo é reponsável pela apresentação dos resultados armazenados no banco de dados e pelo acesso a outras fontes de dados externas. Atualmente a interface para os usuários é feita através de páginas dinâmicas na Web. Estas páginas executam chamadas a scripts CGI e apresentam resultados de consulta ao banco de dados e links para o SRS que mantém muitas fontes de dados relevantes indexadas e fornece rápido acesso à esta informação. O banco de dados armazena informações sobre estas fontes de dados e o Gqbrowse gera os links para estas fontes. Desta forma, para obter informação detalhada de cada sequência é necessário que o usuário apenas clique no link.

Apollo

Apollo é um sistema de anotações *open-source* que permite que os pesquisadores explorem anotações em vários níveis de detalhe e criem anotações, em um ambiente gráfico. Está sendo usado pelos biólogos do FlyBase para fazer as últimas anotações do genoma da Drosophila, e também será utilizado para compartilhar estas anotações na comunidade.

Apollo foi projetado para ser flexível e extensível de forma que possa atender diferentes organismos. O projeto Generic Model Organism Database (GMOD, 2002) adotou o modelo do Apollo para o módulo de anotação.

Este sistema pode ler anotações em vários formatos incluindo GAME XML(GAME XML, 2002), GFF (GFF, 2002) (usado pelo Sanger Center para armazenar dados do genoma humano) e GenBank via CGI, CORBA, e *flat files* e pode ser adaptado para ler novos formatos.

Além disso são executadas vários programas de análise nos dados como GENSCAN (GenScan, 2002) e BLAST.

Muitas anotações foram obtidas de outras fontes de dados como SWISSPROT, EMBL, Ensembl e Genbank e os pesquisadores podem obter informações extras através de links para

estas fontes de dados. Para facilitar as anotações possuem um link para a URL que pode ser vista se o usuário clicar no link.

O Apollo é uma aplicação escrita em Java e possui uma interface Swing.

GGB

O sistema GGB (Generic Genome Browser) é uma combinação de banco de dados e de páginas na Web para manipular e apresentar anotações de genoma de diversas formas diferentes. Por exemplo, é possível dar *scroll* e *zoom* no genoma, anexar URL´s às anotações e fazer consultas.

O GGB utiliza o banco de dados relacional MySQL para executar diversas consultas sobre as anotações. Um módulo Perl provê consultas como a recuperação de uma anotação dada sua identificação, nome ou palavra-chave.

O banco de dados é carregado a partir de arquivos texto delimitados por *tabs* e pode ser atualizado de forma incremental.

Imagene

O Imagene é um sistema que permite que os biólogos executem aplicativos em seus dados para que possam analisá-los e interpretá-los. O pesquisador cria uma estratégia onde indica os aplicativos que deseja executar. Durante a execução, as sub-tarefas são apresentadas graficamente para o usuário. Além disso, quando uma tarefa é concluída, o usuário pode reiniciar outra em qualquer ponto; o sistema apenas recalcula as subtarefas necessárias e mantém as diversas versões. Isto permite manter diversas hipóteses durante uma análise. Com esta flexibilidade, os usuários podem criar novas estratégias combinando sub-tarefas existentes, sem ter que escrever nenhum tipo de código.

Diversos programas podem ser executados sobre os dados, como Glimmer, GeneMark, busca de ORF´s etc, mas não há o pré-processamento destes programas e nem o armazenamento dos resultados.

O modelo de classes utilizado pelo Imagene está apresentado na Figura 3. A classe BIO-OBJECT descreve propriedades comuns de um objeto biológico pertencente ao genoma

(genes, sinais regulatórios, etc.). Esta classe tem duas sub-classes dependendo se o objeto se refere a um objeto de nucleotídeos (BIO-NUCLEIC) ou de aminoácidos (BIO-PROTEIN). A classe BIO-NUCLEIC é especializada em outras duas sub-classes: a SIMPLE, que descreve unidades de informação elementares como genes (GENE) e sinais (SIGNAL) envolvidos na regulação da tradução e transcrição, e a COMPOSITE que descreve estruturas biológicas que são compostas de vários objetos biológicos elementares. Por exemplo, uma unidade de transcrição é composta de gene(s) e sinais regulatórios.

A classe SEQ-OBJECT representa as classes de sequências (SEQ-DATA) de nucleotídeos e aminoácidos e as características destas sequências (SEQ-FEATURE). As sequências de DNA são representadas pela sub-classe NUCLEIC-DATA e correspondem aos registros obtidos de fontes de dados externas (DB-ENTRY) e de laboratórios individuais (FILE-ENTRY). A sub-classe ASSEMBLY representa sequências que foram obtidas a partir da montagem de fragmentos e a classe FRAGMENT representa os fragmentos.

As características do DNA e da proteína são modeladas pela classe SEQ-FEATURE. Estas características são extraídas de anotações em fontes de dados externas ou de resultados da execução de programas. Por exemplo, os sítios de ligação de ribossomos (RBS's) são modelados pela classe RBS.

A arquitetura do Imagene é baseada no modelo de dados biológico, orientado a objetos, na linguagem IlogTalk (Ilog-Talk, 1995), na biblioteca Ilog-Views (Ilog-Views, 1995), IlogPowerClasses (Ilog-Power-Classes, 1996) e IlogControl (Ilog-Control, 1997). O sistema representa os objetos biológicos através de uma interface conhecida como Cartographic Interface (Bisson, G., Garreau, A., 1995).

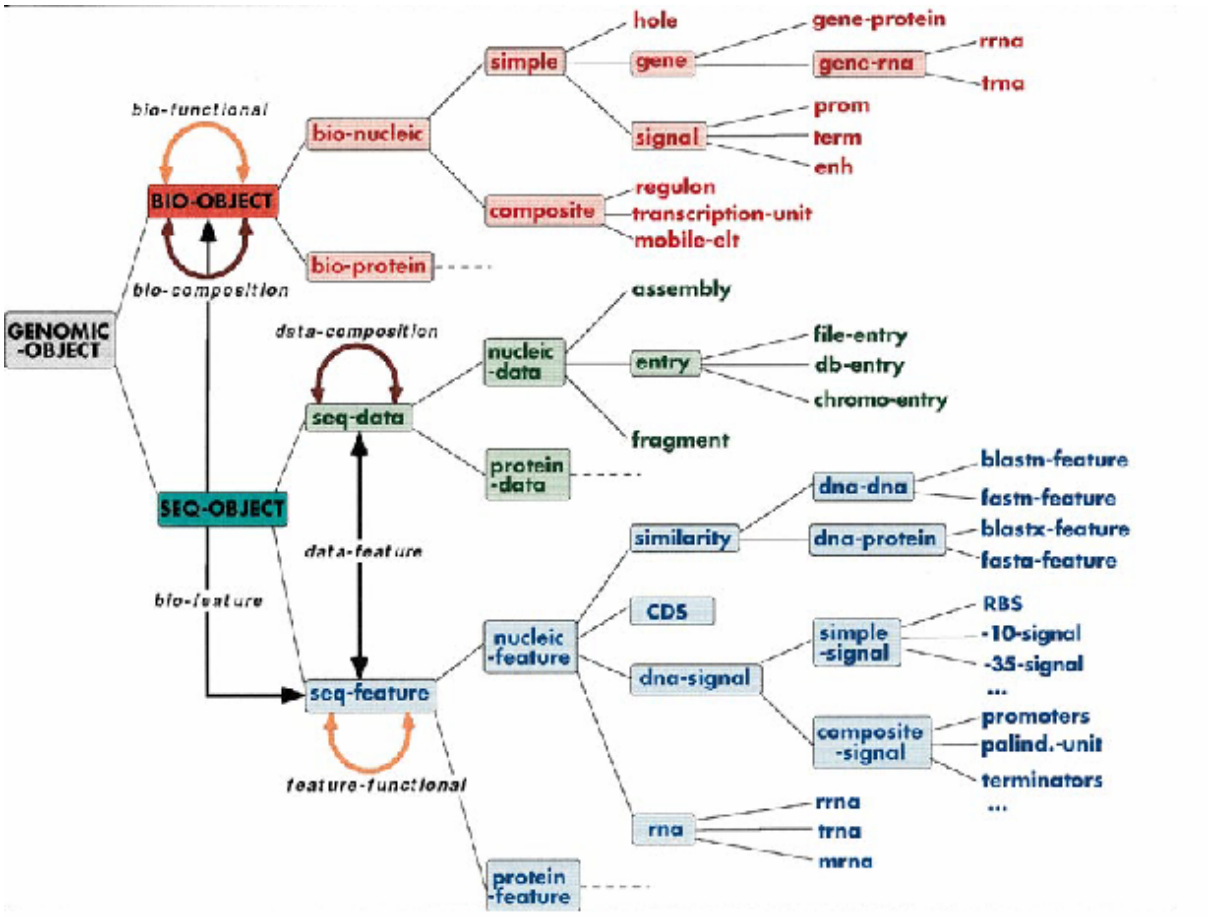


Figura 3. Visão parcial do esquema do Imagene.

MAGPIE

O sistema MAGPIE (Multipurpose Automated Genome Project Investigation Environment) aceita uma sequência de entrada, identifica características genômicas dentro de regiões da sequência e constrói modelos lógicos estáticos de acordo com estas características.

Este sistema está escrito em Perl e em C, e utiliza a linguagem Prolog para interpretar os dados biológicos.

Os usuários podem adicionar novos aplicativos e modificar os parâmetros destes aplicativos em arquivos de configuração. Há também diretórios que armazenam sequências novas e sequências de trabalho.

Vários programas são executados sobre os dados como, por exemplo, obtenção de contigs e ORFs, busca de similaridade, predição de estrutura secundária e busca de padrões.

Além disso, o usuário conta com uma interpretação lógica de seus dados através de regras lógicas definidas em Prolog. Seja, por exemplo, a seguinte regra "Existe uma região codificante em um contig se existir similaridade global e similaridade local contra a mesma sequência de um determinado banco de dados, e o valor da qualidade desta região codificante é igual ao maior valor de qualidade das similaridades". Esta regra é representada em Prolog da seguinte maneira:

```
coding region(Contig,From,To,Confidence) ←

       global similarity(Contig,From1,To1,DBSequence,Confidence1),

       local similarity(Contig,From2,To2,DBSequence,Confidence2),

       overlaps(From1,To1,From,To),

       overlaps(From2,To2,From,To),

       Confidence = maximum(Confidence1, Confidence2).
```

Atualmente o sistema consiste em um conjunto de arquivos interconectados. Um projeto tem uma homepage que apresenta seu estado, isto é, o número de contigs e de ORFs em cada grupo. A página de um grupo apresenta os seus contigs e links para as sequências de DNA, e se aplicável, para as sequências de aminoácidos traduzidas, para informações mais detalhadas sobre a sequência e para um relatório resumido sobre as funções das proteínas envolvidas. Estes links estão associados a resultados lógicos e a registros públicos, como SRS e o Expasy.

Manatee

O sistema Manatee (Manual Annotation Tool Etc, Etc...) é uma ferramenta de anotação de genoma que possui uma interface na Web para os biólogos identificarem rapidamente os genes e atribuir funções a eles.

Este projeto foi desenvolvido no departamento de bioinformática do TIGR (The Institute for Genomic Research)(TIGR, 2002) e tem sido utilizado para anotar manualmente vários genomas de procariontes e eucariontes.

Manatee é escrito em Perl e executa em um servidor que tenha CGI, como o Apache. Manatee requer pelo menos um banco de dados MySQL (ou Sybase) e arquivos de busca

associados. O banco de dados utiliza um modelo de dados desenvolvido no TIGR para representar os dados de procariontes e eucariontes. Este modelo pode ser visto em (MANATEEa,MANATEEb, MANATEEc, 2002).

Pedant

O PEDANT (Protein Extraction, Description and ANalysis Tool) é um sistema de anotações para sequências individuais e genomas completos.

A ferramenta possui um banco de dados relacional (atualmente MySQL) com um conjunto de tabelas que modelam as sequências e suas anotações. Programas escritos em Perl capturam os dados do banco de dados, executam os aplicativos e armazenam as respostas destes aplicativos (BLAST, montagem de fragmentos, etc) no banco de dados, como é possível notar no esquema simplificado do sistema, apresentado na Figura 4. O sistema conta com uma interface Web.

O pesquisador, via Pedant, pode fazer consultas e pode anotar, de forma manual, seus próprios dados.

Outro aspecto interessante da ferramenta Pedant é que ele trata versões das anotações. Como o sistema pode ser utilizado para genomas que ainda não estão totalmente sequenciados, com o passar do tempo outras sequências vão sendo disponibilizadas, o que faz com que os programas de análises dos dados gerem resultados diferentes (por exemplo, modificação nos contigs, diminuição do número de contigs, descobrimento de novas ORFs). Desta forma, as anotações geradas automaticamente se modificam e as anotações manuais que estavam relacionadas às anotações automáticas antigas também podem se modificar. O Pedant trata desta questão atribuindo versões às anotações armazenadas no banco de dados.

VisualGenome

O Visual Genome é uma ferramenta *stand alone* de comparação de genoma e de anotação. A interface gráfica “click and zoom” permite que o usuário navegue pelo genoma de acordo com as áreas de seus interesses e faça comparações com os genomas presentes no banco de dados. A comparação pode ser visualizada em diversos formatos. Um exemplo é uma representação gráfica que mostra o alinhamento entre um determinado gene de consulta e todos os que são mais similares a ele.

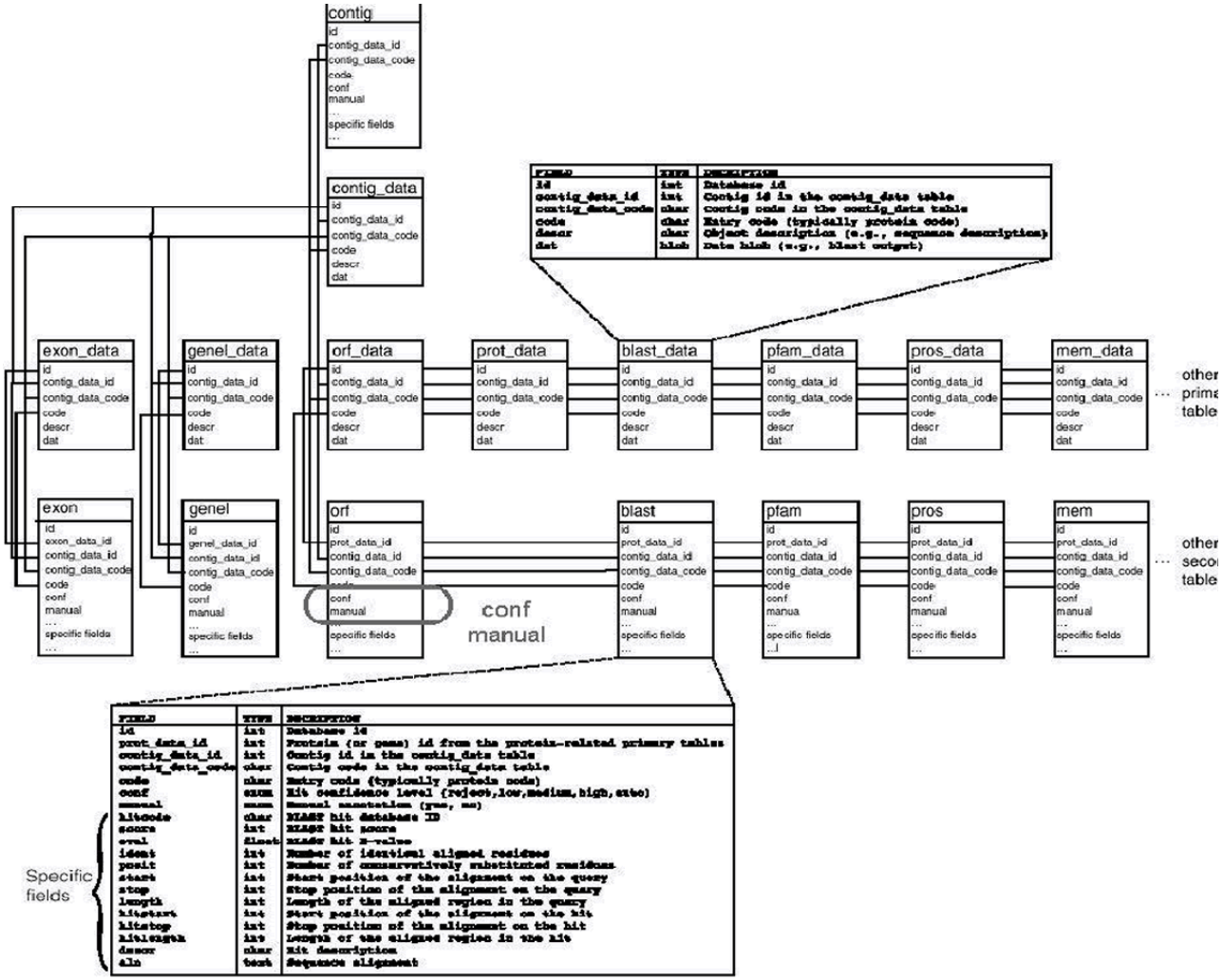


Figura 4. Esquema simplificado do Pedant.

Clicando em um gene qualquer, é possível visualizar os melhores alinhamentos relacionados a ele e uma anotação textual sobre o gene. Com isso, o sistema pretende que o usuário seja mais preciso e rápido na interpretação dos seus dados.

Antes do usuário começar a trabalhar, o Visual Genome pré-calcula todas as comparações possíveis utilizando o BLAST e armazena o resultado em tabelas no banco de dados. Este sistema possui uma representação dos genes conhecidos no genoma humano, *levedura,*, *Pseudomona*, *E. coli*, *Arabidopsis Thaliana* e outros.

Além disso, ao fornecer arquivo do Genbank contendo as sequências e suas anotações, o Visual Genome lê automaticamente o arquivo e extrai todos os dados relevantes para sua execução.

O Visual Genome é executado localmente em um computador PC onde deve ser instalado com um banco de dados.

O Visual Genome pretende ser utilizado para anotações sobre qualquer organismo. Ele vem pré-instalado com um banco de dados com procariontes, eucariontes ou uma combinação de ambos (existem três versões do Visual Genome: Visual Genome Prokaryote, Visual Genome Eukaryote, and Visual Genome Gold) e com os resultados do BLAST pré-calculados e um significativo número de anotações relacionada com cada organismo obtidas de fontes de dados públicas externas e proprietárias.

Além disso, é possível que o pesquisador trabalhe com o Visual Genome e seus dados proprietários. Para tanto ele precisa apresentar ao sistema um arquivo simples delimitado por tabs de acordo com um padrão definido pelo Visual Genome. As sequências serão varridas e será executado e o BLAST de cada uma contra o banco de dados.

Visual Genome está escrito em C e pode ser executado em Windows 95, 98, NT 4.0 e Windows 2000.

3 BioNotes

BioNotes é a nossa proposta de um sistema distribuído de anotações biológicas que possa ser utilizado pela comunidade da área de biologia molecular para facilitar a interpretação de seus dados.

Cada comunidade possui um grupo de usuários, a serem cadastrados, que podem pertencer a diferentes instituições. Não é permitido que um usuário não cadastrado, acesse os dados armazenados no BioNotes.

As anotações são obtidas de diversas fontes de dados externas (como, por exemplo, o Genbank, PIR, SWISSPROT e Prosite) e de diversos aplicativos externos (como, por exemplo, BLAST, CAP3 (CAP3, 2002), Phred, Phrap (Phred/Phrap/Consed System Homepage, 2002), Crossmatch (Crossmatch, 2002), GLIMMER (GLIMMER, 2002) e tRNAScan (tRNAScan, 2002)). As anotações geradas pela execução destes aplicativos são analisadas, transformadas a partir do formato específico de saída do aplicativo para o formato de dado semi-estruturado XML, e armazenadas em um *data warehouse*.

As anotações armazenadas no *data warehouse* podem ser consultadas de diversas maneiras. Por exemplo, dado um identificador de uma sequência é possível saber todas as anotações relacionadas a ela que estão nas fontes de dados externas, como, por exemplo, as *keywords* e a *feature table* do Genbank, e também os resultados da execução dos aplicativos externos que tem a ver com tal sequência, como, por exemplo, quais sequências homólogas a ela, obtidas pela execução do BLAST.

Além da consulta, o usuário pode cadastrar, atualizar e remover anotações. Desta forma, além de poder analisar as anotações geradas automaticamente, os usuários podem criar as suas próprias anotações e disponibilizá-las para toda a sua comunidade. Isto permite um maior controle, organização, compartilhamento e distribuição das anotações.

A atualização de anotações manuais é feita de maneira cuidadosa porque não remove a anotação existente e a substitui por outra. Ao pedir para atualizar uma anotação o usuário, na verdade, estará criando uma nova versão para a sua anotação. Com isso, a anotação pode ter várias versões, sendo possível consultar cada uma delas e perceber as modificações que foram feitas e os usuários que as fizeram.

Enquanto um projeto genoma estiver em andamento, o sequenciamento continua gerando novos reads. Isto faz com que as fontes de dados externas estejam sempre com novos dados e os programas de análise geralmente gerando resultados diferentes. Para que as informações armazenadas no BioNotes estejam sempre atualizadas, de tempos em tempos os novos dados são obtidos, a partir das fontes externas, e inseridos no data warehouse. Os programas de análise são re-executados sobre os dados atualizados e os resultados gerados são também armazenados no BioNotes, sendo marcados como uma nova versão.

A remoção de uma anotação também só pode ser feita pelo usuário que a criou.

Existe ainda vários perfis de usuários que estão relacionados aos comandos que podem executar. Alguns usuários podem executar todos os comandos, enquanto outros podem apenas consultar.

O sistema está escrito em Java 1.4.0, o que permite portabilidade. A interface é Web e foi desenvolvida em JSP para Internet Explorer versão 5 ou superior. O banco de dados utilizado é o Oracle 9i.

3.1 A Adoção do Modelo de Dados Semi-estruturado

A pesquisa em dados semi-estruturados começou com a observação de que muitos dos dados não estavam de acordo com os modelos relacional e orientado a objeto. Várias aplicações armazenam seus dados em formatos não padronizados, documentos estruturados, como HTML e SGML. Além disso, a integração de fontes de dados heterogêneas frequentemente trata de fontes de dados que pertencem a organizações externas que não são controladas e que sua estrutura é parcialmente conhecida e que pode se modificar. Os dados destas aplicações podem ser modelados como dados orientados a objeto, mas sua estrutura é irregular: alguns objetos podem não ter alguns atributos, outros podem ter várias ocorrências do mesmo atributo, o mesmo atributo pode ter diferentes tipos em diferentes objetos, informações semanticamente relacionadas podem estar representadas diferentemente em vários objetos. Os dados com estas características são chamados de semi-estruturados (Suciu, D., 1998).

O modelo de dados semi-estruturado representa um novo paradigma em banco de dados. Diferente dos paradigmas relacional e orientado a objeto, o esquema dos dados semi-

estruturado está embutido com o dado, tornando-o fácil para troca de dados entre aplicações, negócios e organizações (Suciu, D., 1998).

A World Wide Web Consortium (W3C, 2002) aprovou o padrão eXtensible Markup Language, XML (XML, 2002). A aplicação central do XML é a troca de dados eletrônica, na qual XML é usado como o formato de troca dos dados. O dado representado em XML é similar ao dado semi-estruturado porque consiste de objetos e atributos (chamados de elementos e tags respectivamente) e tem o seu esquema embutido (Widom, J., 1999).

As idéias básicas de XML são muito simples: tags de elementos de dados indicam o significado do dado, em vez de, por exemplo, especificar como o dado deve ser apresentado (como em HTML) e os relacionamentos entre os dados são obtidos através de simples referências e aninhamentos. Com isso, os servidores Web e as aplicações que codificam seus dados em XML podem rapidamente tornar sua informação disponível em um formato simples e útil, e os clientes de tais informações podem utilizá-la facilmente. O conteúdo do dado fica separado da sua apresentação, facilitando prover múltiplas visões do mesmo dado. Como XML foi projetado para representação de dados, ele é simples, fácil de ser percorrido por programas e se auto-descreve (Widom, J., 1999).

Muitos avanços têm ocorrido desde a criação de XML. Atualmente, já existem várias tecnologias para o tratamento de XML, como, por exemplo:

- XSL (eXtensible Stylesheet Language) (XSL,2002), uma linguagem para expressar stylesheets, que permite a apresentação de arquivos em XML via determinadas especificações,
- XML Query (XML Query, 2002), que tem como objetivo prover maior flexibilidade em consultas e extração de dados XML
- Xpath (Xpath, 2002), uma linguagem que provê facilidades em manipulação de strings, números e booleans com o objetivo de tratar partes de um documento XML facilitando os parsers que analisam o formato XML.
- XML Schema (XML Schema, 2002), define a estrutura, o conteúdo e a semântica de documentos XML,
- XSLT (XSL Transformations) (XSLT, 2002), linguagem para transformação de documentos XML em outros documentos XML,

- XLink (XML Linking Language) (Xlink, 2002), que permite que elementos inseridos em documentos XML sejam links para outros recursos,

além disso, vários vendedores de software já estão adotando e criando ferramentas para lidar com XML.

Acreditamos que XML será tão importante na Web como HTML foi em seu início e que será a ferramenta mais usada para manipulação e transmissão de dados.

Como muitos, a comunidade de biólogos vem usando XML para representação de dados. Até mesmo aqueles que já tinham um formato próprio de representação de dados, estão migrando os dados para XML. Como exemplo é possível destacar as fontes de dados externas EMBL, Genbank e DDBJ (XML at NCBI, 2002)(XEMBL, 2002), PIR (PIR-XML, 2002), SWISSPROT (SPTr-XML Documentation, 2002) e, até mesmo, os resultados de programas de análise de dados biológicos, como o BLAST (NCBI BLAST Homepage, http://www.ncbi.nlm.nih.gov/BLAST/, 2002), também estão sendo formatados em XML.

Além disso existem inúmeros grupos em bioinformática trabalhando com XML, como descrito em (XML Bioinformatics, 2002) e (XML Examples for Biology, 2002).

Diante destes fatos, o sistema BioNotes possui um modelo de dados híbrido onde as entidades estão armazenadas em tabelas, que possuem colunas que representam informações biológicas de acordo com o modelo semi-estruturado. Nestas são armazenados os documentos (em XML) e os esquemas destes documentos (em XML Schema). O banco de dados utilizado é o Oracle 9i, que armazena dados em XML em registros do tipo XMLType.

Atualmente é possível:

- fazer consultas em dados XMLType utilizando funções pré-definidas pela Oracle,
- melhorar o desempenho das consultas utilizando índices especiais criados para dados em XML,
- checar se o formato do dado está sintaticamente de acordo com o padrão XML e com o esquema definido por um XML Schema, etc.

Acreditamos também que, em futuro próximo, existirão outros recursos que irão permitir simplificar e agilizar a consulta a dados semi-estruturados.

3.1.1 O Modelo de Dados da Aplicação

A Figura 5 apresenta o modelo de dados principal do sistema BioNotes. Estão representados os dados, as fontes de dados externas onde os dados foram obtidos, as anotações que são atribuídas aos dados, os esquemas que representam os dados e as próprias anotações. Além disso estão representados os usuários e as comunidades, bem como as instituições a que pertencem.

A Figura 6 apresenta os tipos de anotações. Estão representadas as anotações feitas manualmente e as anotações obtidas pela execução dos programas (como BLAST, CAP3 e Phrap).

Na Figura 7 são representadas as anotações obtidas nas fontes de dados externas (como PIR, Genbank, SWISSPROT, Prosite).

Todas as anotações (manuais e automáticas) estão registradas nas tabelas *Dado* e *Anotação*, no atributo *conteudo* de tipo XMLType.

3.1.2 Exemplo de XML SCHEMA

Como exemplo de esquema, a seguir está o XML Schema para o resultado do BLAST. O resultado é representado pelos seguintes elementos:

- ✓ blast_result: resultado do BLAST entre uma sequência de consulta e as sequências de um banco de dados
  - ✓ input_sequence: informações da sequência de consulta
  - ✓ database: informações do banco de dados
  - ✓ alignment_db_sequences: informações sobre os alinhamentos entre a sequência de consulta e as sequências do banco de dados
    - alignment_db_sequence: alinhamentos entre a sequência de consulta e uma determinada sequência do banco de dados
      - db_sequence: sequência do banco de dados
      - alignments: alinhamentos

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">

	<xs:element name="blast_result">
		<xs:complexType>
			<xs:sequence>
				<xs:element name="reference" type="xs:string"/>
				<xs:element name="input_sequence">
					<xs:complexType>
						<xs:sequence>
							<xs:element ref="accession"/>
							<xs:element ref="comment"/>
							<xs:element ref="length"/>
						</xs:sequence>
					</xs:complexType>
				</xs:element>
				<xs:element name="database">
					<xs:complexType>
						<xs:sequence>
							<xs:element name="path" type="xs:string"/>
							<xs:element name="number_sequences" type="xs:byte"/>
							<xs:element name="total_length" type="xs:string"/>
						</xs:sequence>
					</xs:complexType>
				</xs:element>
				<xs:element name="alignment_db_sequences">
					<xs:complexType>
						<xs:sequence>
							<xs:element name="alignment_db_sequence" maxOccurs="unbounded">
								<xs:complexType>
									<xs:sequence>
										<xs:element name="db_sequence">
											<xs:complexType>
												<xs:sequence>
													<xs:element ref="accession"/>
													<xs:element ref="comment"/>
													<xs:element ref="length"/>
												</xs:sequence>
											</xs:complexType>
										</xs:element>
										<xs:element name="alignments">
											<xs:complexType>
												<xs:sequence>
													<xs:element name="score" type="xs:string"/>
													<xs:element name="expect" type="xs:string"/>
													<xs:element name="p" type="xs:string"/>
```

```
                                                                                                    <xs:element name="identities"
type="xs:string"/>
                                                                                                    <xs:element name="positives"
type="xs:string"/>
                                                                                                    <xs:element name="alignment"
type="xs:string"/>
                                                                                            </xs:sequence>
                                                                                    </xs:complexType>
                                                                            </xs:element>
                                                                    </xs:sequence>
                                                            </xs:complexType>
                                                    </xs:element>
                                            </xs:sequence>
                                    </xs:complexType>
                            </xs:element>
                    </xs:sequence>
            </xs:complexType>
    </xs:element>
    <xs:element name="accession" type="xs:string"/>
    <xs:element name="comment" type="xs:string"/>
    <xs:element name="length" type="xs:string"/>
</xs:schema>
```

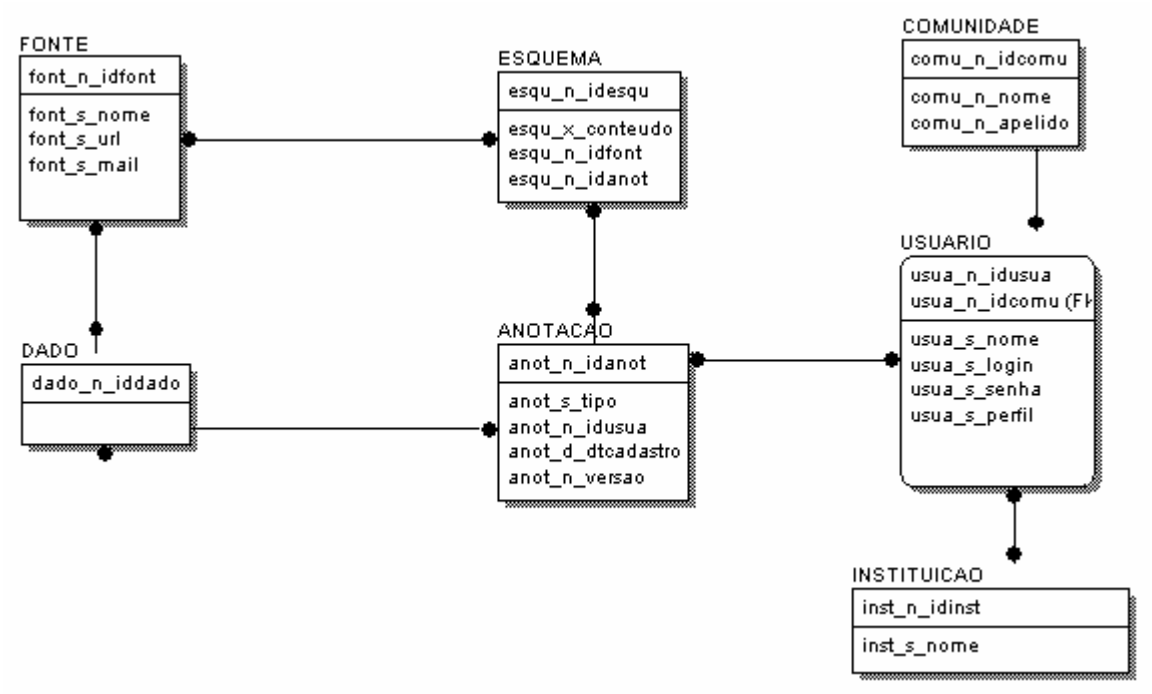


Figura 5. Modelo de Dados Resumido da Parte Principal da Aplicação.

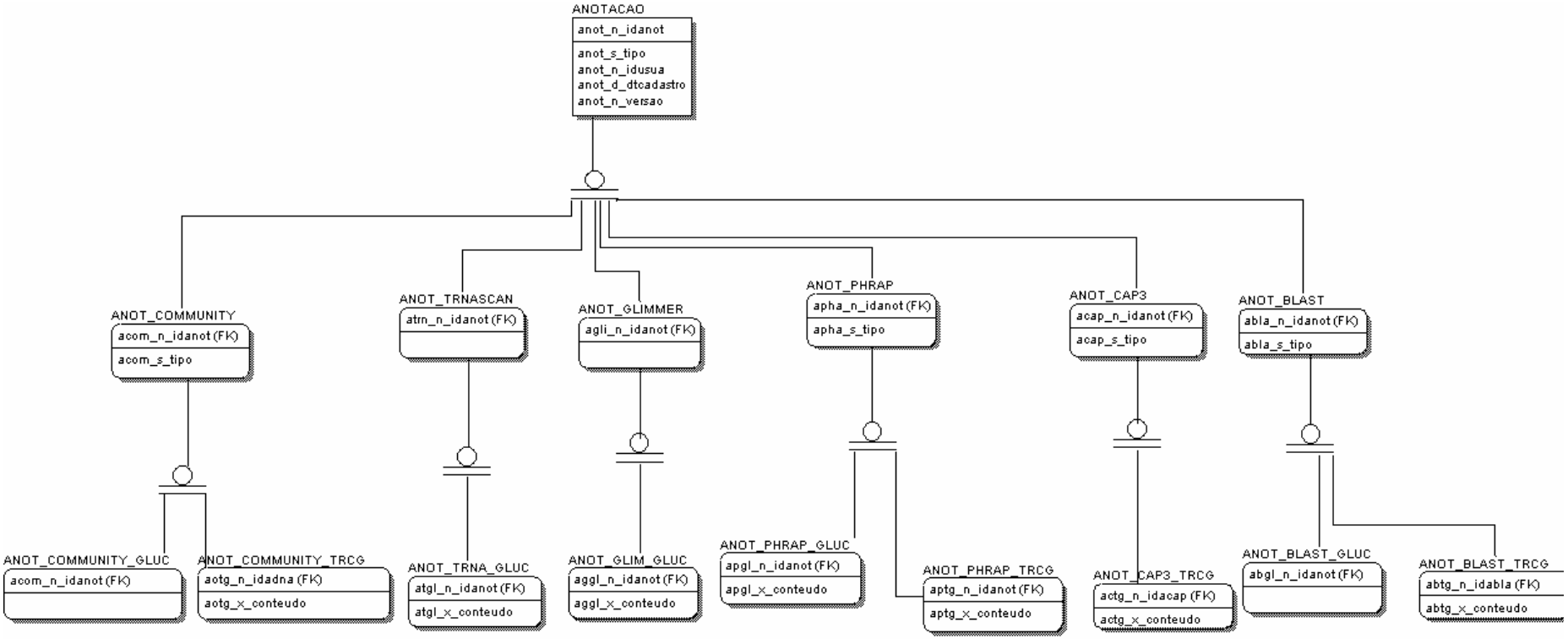


Figura 6. Modelo de Dados Resumido da Parte de Anotações da Comunidade e dos Resultados das Aplicações.

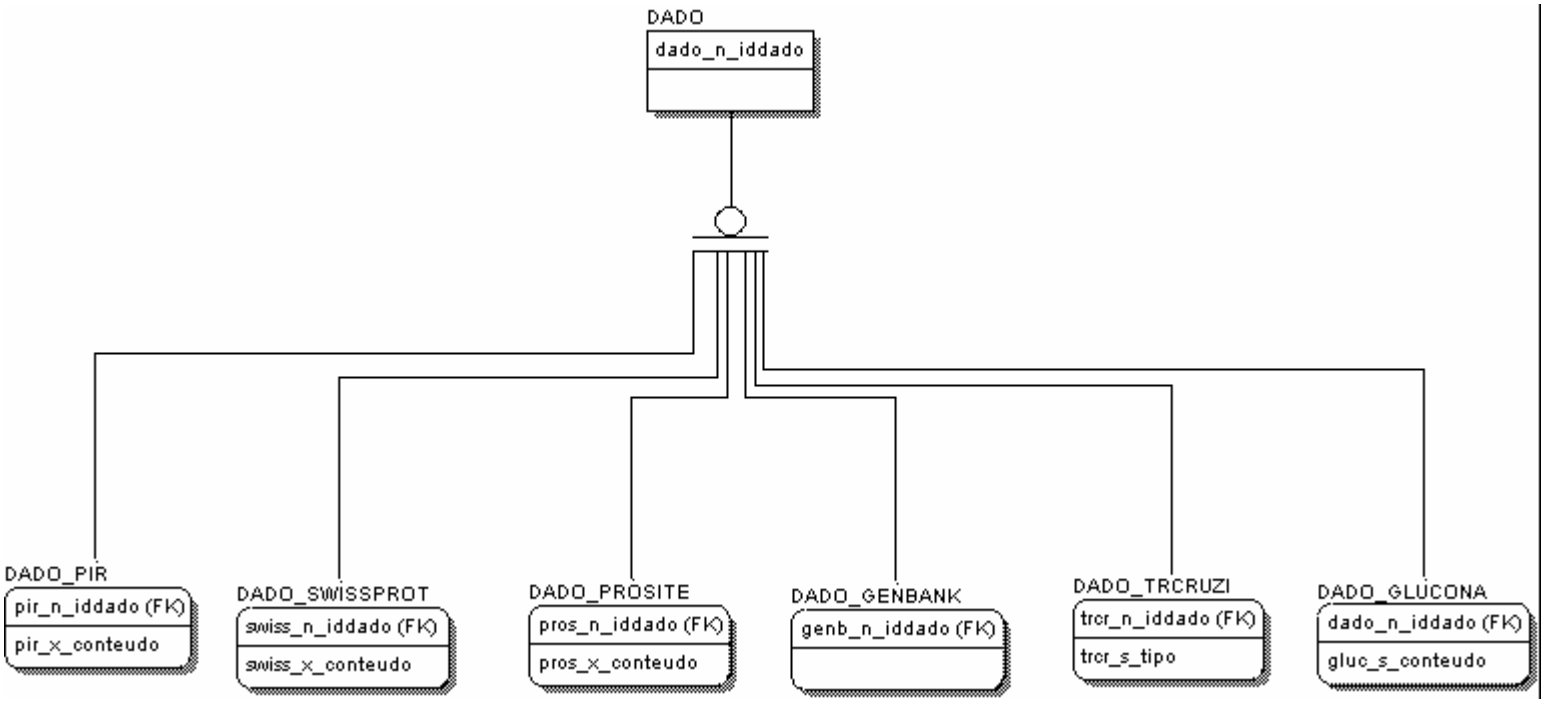


Figura 7. Modelo de Dados Resumido das Anotações das Fontes de Dados Externas.

3.2 Diagrama de Casos de Uso

A Figura 8 apresenta o diagrama de caso de uso que descreve as principais funcionalidades do sistema, que são:

1. Criar, atualizar e remover instituições
2. Criar, atualizar e remover comunidades
3. Criar, atualizar e remover usuários em determinadas comunidades
4. Criar, atualizar, remover e consultar anotações manuais de determinadas comunidades
5. Obter e consultar anotações automáticas de fontes de dados e aplicativos externos

Neste diagrama são apresentados os quatro tipos de usuários existentes no sistema:

- System_Administrator: que possui direito de executar as tarefas 1, 2 e 5.
- Community_Administrator: que possui direito de executar as tarefas 3 e 4.
- Top_Community_User: que possui direito de executar a tarefa 4.
- Simple_Community_User: que possui direito de executar o subítem consultas da tarefa 4.

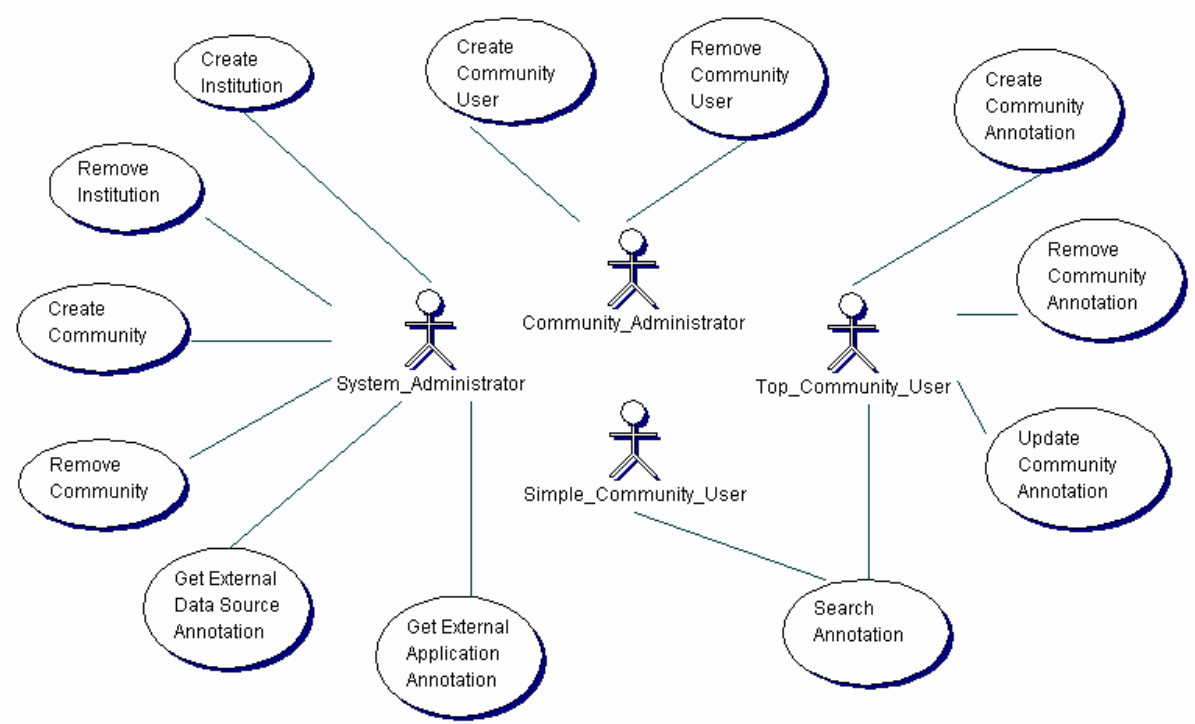


Figura 8. Diagrama de Caso de Uso.

3.3 Organismos em estudo

Atualmente o sistema está sendo construído para anotar o genoma dos seguintes organismos:

- *Trypanosoma cruzi*, em estudo na Fundação Instituto Osvaldo Cruz
- *Gluconacetobacter diazotrophicus* ou simplesmente *Glucona,* em estudo na Universidade Federal do Rio de Janeiro, dentro do Projeto RioGene

Inicialmente, foram obtidas as sequências (reads) para estes organismos. A partir daí, foram executados, por exemplo, os seguintes aplicativos, de forma a registrar os resultados dessa execução como anotações automáticas.

- Phrap ou CAP3, para determinar e registrar os contigs,
- Glimmer, para determinar e registrar as ORF's,
- tRNAScan, para determinar e registrar os tRNA's,
- TRANSTERM, para determinar e registrar os terminadores das ORF's,
- RBSFinder, para determinar e registrar os sítio de ligação de ribossomas,
- BLAST, para registrar as sequências similares.

Em seguida, os pesquisadores cadastrados podem visualizar as anotações de forma organizada e registrar a sua interpretação como uma nova anotação (manual).

Como exemplo de telas do sistema serão apresentadas aquelas relativas ao *Trypanosoma cruzi*, que tratam de:

- Formulário de consulta na fonte de dados externa Genbank (Figura 9)
    - Neste formulário o usuário escolhe sobre qual atributo deseja realizar a consulta, informa a palavra-chave de consulta e outros parâmetros como, por exemplo, se deseja ou não a ordenação do resultado.
- Índice de resposta da consulta na fonte de dados externa Genbank (Figura 10)
    - Esta página informa de maneira resumida e paginada os resultados da consulta ao Genbank.
- Detalhe da resposta da consulta na fonte de dados externa Genbank (Figura 11)

    - Esta página apresenta todos os detalhes de um registro de *Trypanosoma cruzi* no Genbank.
- Índice de resposta da consulta na aplicação externa BLAST (Figura 12)
    - Esta página exibe, de maneira resumida e paginada, informações sobre os alinhamentos obtidos através da execução do programa BLAST para uma determinada sequência de *Trypanosoma cruzi*.
- Detalhe da resposta da consulta na aplicação externa BLAST (Figura 13)
    - Esta página apresenta todos os detalhes de um determinado alinhamento encontrado entre uma sequência de *Trypanosoma cruzi* e outra armazenada no Genbank.
- Cadastro de anotação da comunidade (Figura 14)
    - Formulário onde o usuário cadastra uma anotação manual.
- Atualização, remoção e consulta a versões de uma anotação da comunidade (Figura 15).
    - Página onde o usuário consulta uma determinada anotação manual e tem links para alterá-la, removê-la e criar uma nova versão.

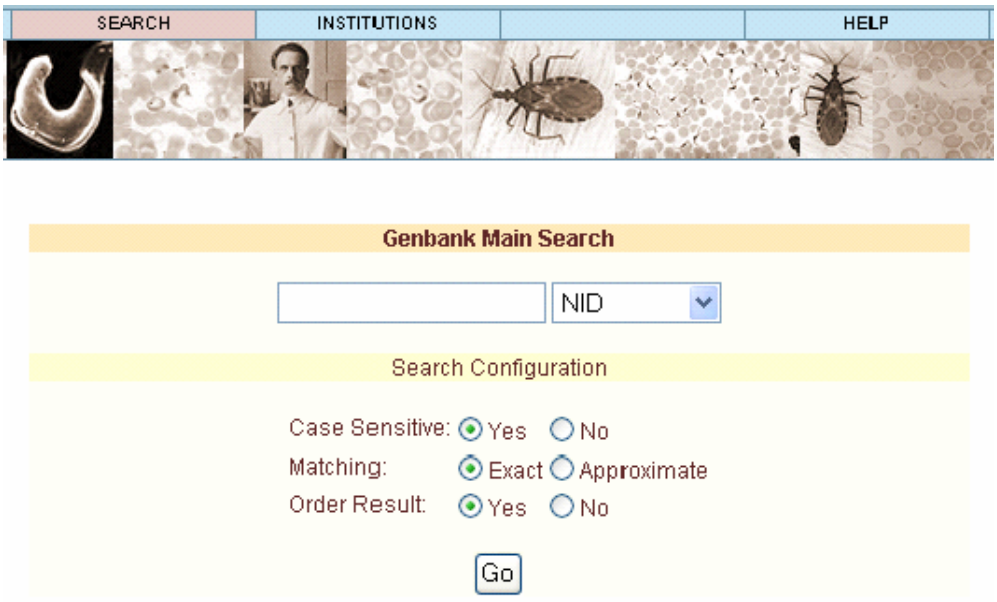


Figura 9. Formulário de Consulta no Genbank.

Genbank Search Result - Index

Total: 5 registers Page 1/1

Register	NID	Accession	Keywords
1	AA399708	AA399708	EST.
2	AA399707	AA399707	EST.
3	AA399706	AA399706	EST.
4	AA399705	AA399705	EST.
5	AA399704	AA399704	EST.

Figura 10. Resposta resumida de consulta do Genbank.

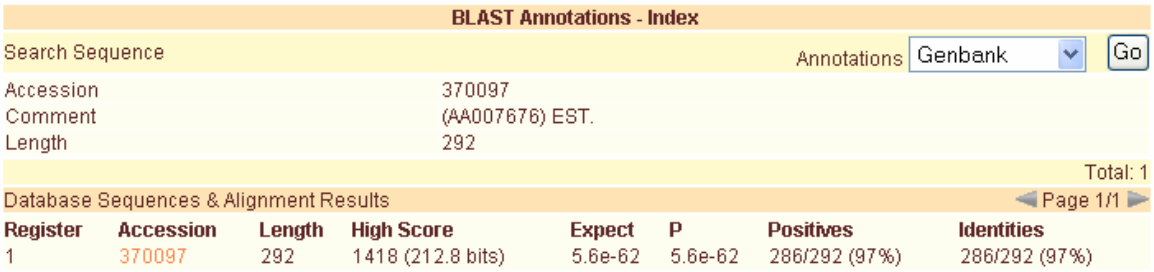
Genbank Search Result - Detail

Annotations: BLAST — Go

Search Result - Index — Annotation 31/634

Locus Name	AA433351
Length of Sequence	203 b
Type of Molecule	p mRNA
Topology of Molecule	linear
Date	EST 03-MAR-2000
Definition/Brief Description	TEUF209 T.cruzi epimastigote non-normalized cDNA Library Trypanosoma cruzi cDNA clone 209 5', mRNA sequence.
Accession Number(s)	AA433351
Version (Accession Number/NCBI GI:Identifier)	AA433351.1 GI:2130828
Keywords	EST.
Abbreviated Form of the Organism Name	Trypanosoma cruzi.

Figura 11. Resposta detalhada de registro do Genbank.



BLAST Annotations - Index

Search Sequence — Annotations: Genbank — Go

Accession	370097
Comment	(AA007676) EST.
Length	292

Total: 1

Database Sequences & Alignment Results — Page 1/1

Register	Accession	Length	High Score	Expect	P	Positives	Identities
1	370097	292	1418 (212.8 bits)	5.6e-62	5.6e-62	286/292 (97%)	286/292 (97%)

Figura 12. Resposta resumida de consulta do BLAST para uma sequência de Trypanosoma cruzi.

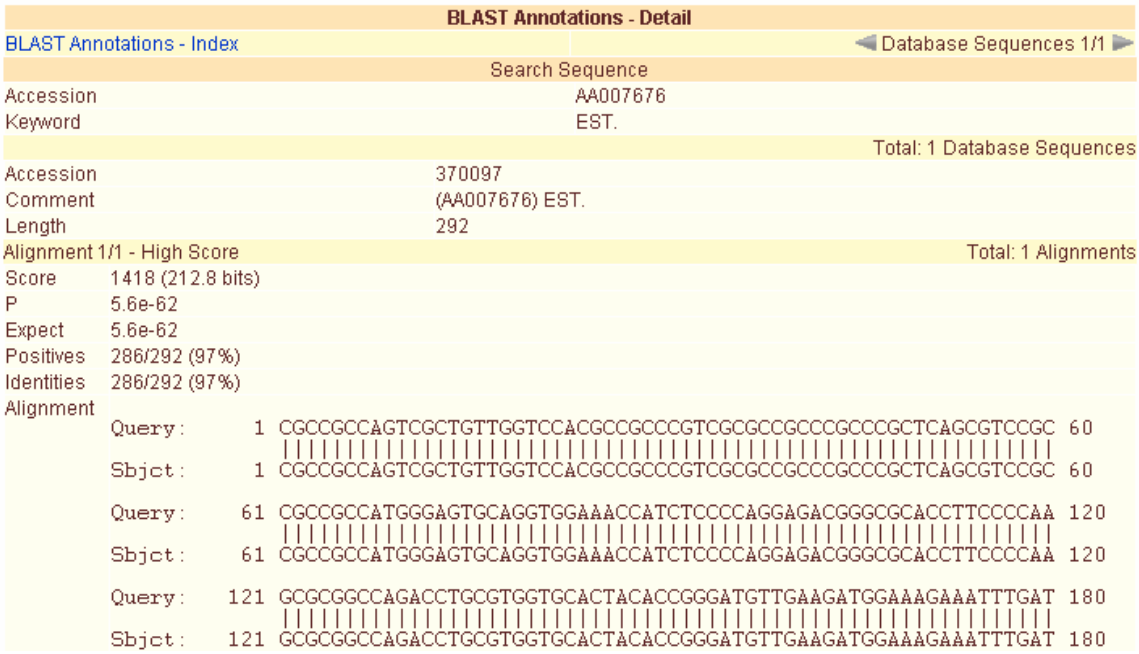


Figura 13. Resposta detalhada de consulta de resultado do BLAST.

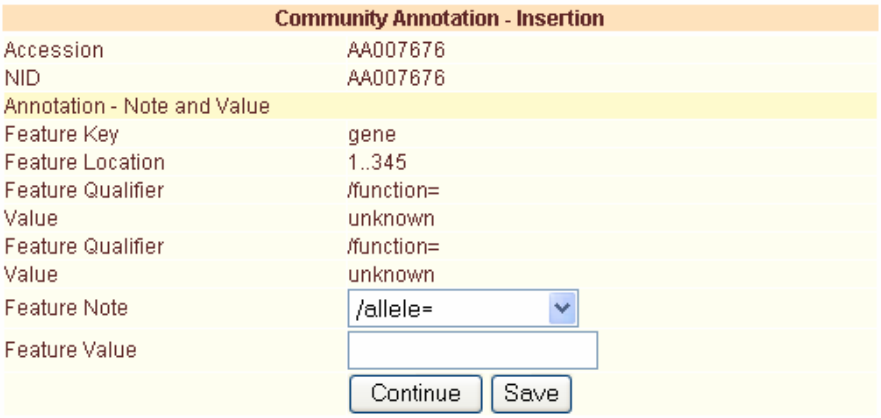


Figura 14. Cadastro de anotação da comunidade.

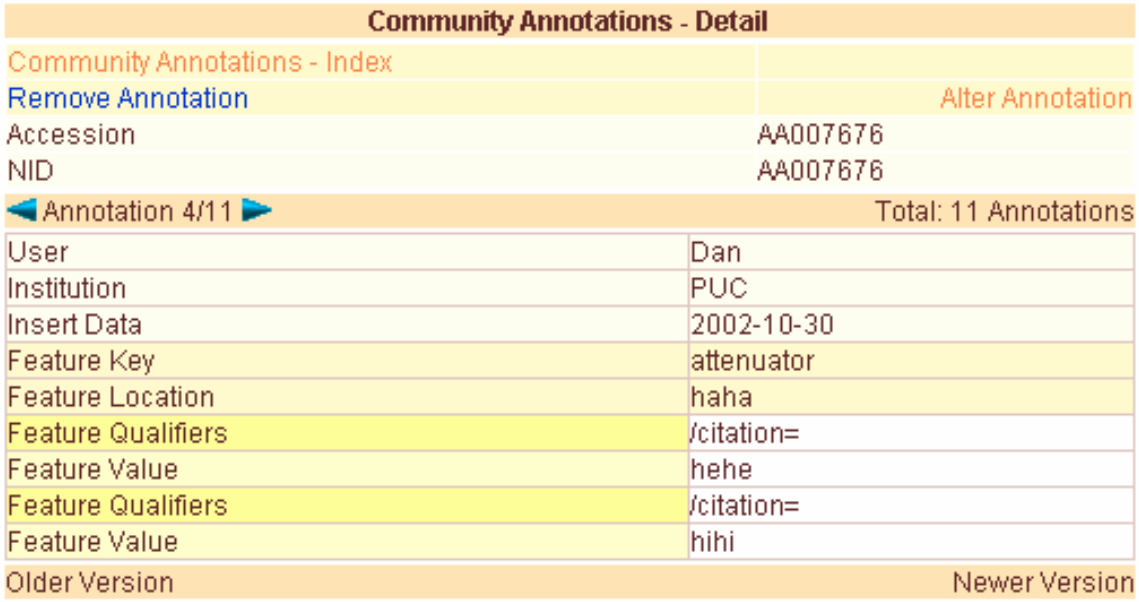


Figura 15. Consulta, remoção e atualização de anotação da comunidade.

3.4 Projeto da Aplicação

Um dos objetivos do projeto da aplicação é que as mudanças em um módulo provoquem o mínimo ou nenhum impacto em outros módulos. Isto permite:

- Adicionar características sem reprojetar a aplicação
- Adicinar novos tipos de clientes
- Mudar interfaces de clientes com mínimo impacto para a lógica do negócio
- Mudar a lógica do negócio sem mudar a apresentação do dado
- Mudar o esquema do banco de dados ou a fonte de dados com o mínimo impacto na aplicação

A seguir são descritas as principais características do projeto da aplicação.

3.4.1 Páginas Encadeadas

No projeto de páginas encadeadas, as páginas da aplicação que são *linkadas* sequencialmente, i.e. a página 1 tem um link que chama a página 2, a página 2 tem um link para a página 3, e assim por diante.

Cada página pode ser gerada diferentemente. Por exemplo, a página 1 pode ser um arquivo HTML, a página 2 um servlet enquanto a 3 uma JSP. As páginas contém links ou elementos de formulários (se o usuário precisar entrar com algum valor) para permitir que o usuário acesse a próxima página. Em qualquer caso, o link para a próxima página é escrito em cada página.

As vantagens deste projeto são que as páginas são diretas e fáceis de se entender. Este projeto é gerenciável para pequenas aplicações que dificilmente se tornarão grandes ou que as páginas dificilmente serão alteradas.

As desvantagens deste projeto é que não existe um ponto central para tratamento de pedidos de clientes e é difícil fazer manutenção das páginas. Se as páginas forem ser modificadas, adicionadas ou removidas da aplicação, a aplicação torna-se menos organizada porque será necessário analisar o código das páginas em busca dos links modificados, ou mudar as dependências para que uma página seja acessada por uma página diferente.

3.4.2 Uso do padrão de projeto *Model-View-Controller* (MVC)

Uma forma melhor de projetar uma aplicação é usar o padrão de projeto MVC (*model-view-controller*). O padrão MVC permite que a aplicação seja extensível e modular separando a aplicação em três partes:

- parte lógica do negócio (*Model*), que implementa a recuperação e a manipulação dos dados
- parte interface de usuário (*View*), que é o que os usuários da aplicação vêem
- parte controladora (*Controller*), que encaminha pedidos aos objetos adequados.

Separando a aplicação nestas três partes, o padrão de projeto MVC permite modificar uma parte da aplicação sem atrapalhar as outras. Isto significa que é possível ter vários desenvolvedores trabalhando em diferentes partes da aplicação na mesma hora sem que seja necessário que um entre no domínio do outro. Cada desenvolvedor sabe seu papel na

aplicação. Por exemplo, a interface de usuário não deve conter nenhum código que tenha a ver com a lógica do negócio, e vice-versa.

Para detalhes do MVC consulte (JAVATM BLUEPRINTS – MVC, 2002)(JAVATM BLUEPRINTS, 2002)(JAVA TUTORIAL – DESIGN PATTERNS, 2002).

3.4.2.1 *Controller*

O controlador é o primeiro objeto da aplicação a receber requisições dos clientes. Todas as requisições para qualquer página da aplicação devem passar primeiro pelo controlador.

O mapeamento de cada tipo de requisição para a classe que trata o pedido é feito no controlador. A ação é um parâmetro com uma string passada para o controlador. O controlador recebe o valor do parâmetro para determinar o tipo de ação que deve tomar.

Quando o controlador recebe uma requisição, ele procura o valor do parâmetro da ação, determina qual classe é responsável por resolver tal requisição, cria uma instância da classe e envia o pedido à tal instância.

Tendo um controlador como o primeiro ponto de contato da aplicação, é possível adicionar novas funcionalidades à aplicação com relativa facilidade. Basta adicionar o novo mapeamento e escrever novas classes que implementem a funcionalidade.

No BioNotes, o controlador é um servlet chamado *ControllerServlet*. As páginas da aplicação tem links para este servlet. Ao receber o pedido, o servlet repassa-o para uma sub-classe da classe Command, que é responsável por executar o comando específico que atende o pedido do usuário.

A Figura 16 apresenta o resumo do diagrama de classes. Na figura há um exemplo de comando que trata do *login* (classe *UserLoginCommand*) e do *logout* (classe *UserLogoutCommand*) de um usuário no sistema, e de comandos que são executados por duas comunidades que estão atualmente utilizando o sistema: a que estuda o organismo *Trypanosoma cruzi* (classe *TRCRCommand*) e a que estuda a *Glucona* (classe *GLUCCommand*). Há ainda dois comandos que tratam de consulta do resultado do BLAST para determinada sequência de *Trypanosoma cruzi* (classes *BLASTAnnotationPagingSearchCommand* e *BLASTAnnotationDetailSearchCommand*).

Usando o objeto controlador, a aplicação fica livre das páginas encadeadas, onde é necessário verificar todos os links quando páginas forem adicionadas ou removidas da aplicação para que todos continuem funcionando.

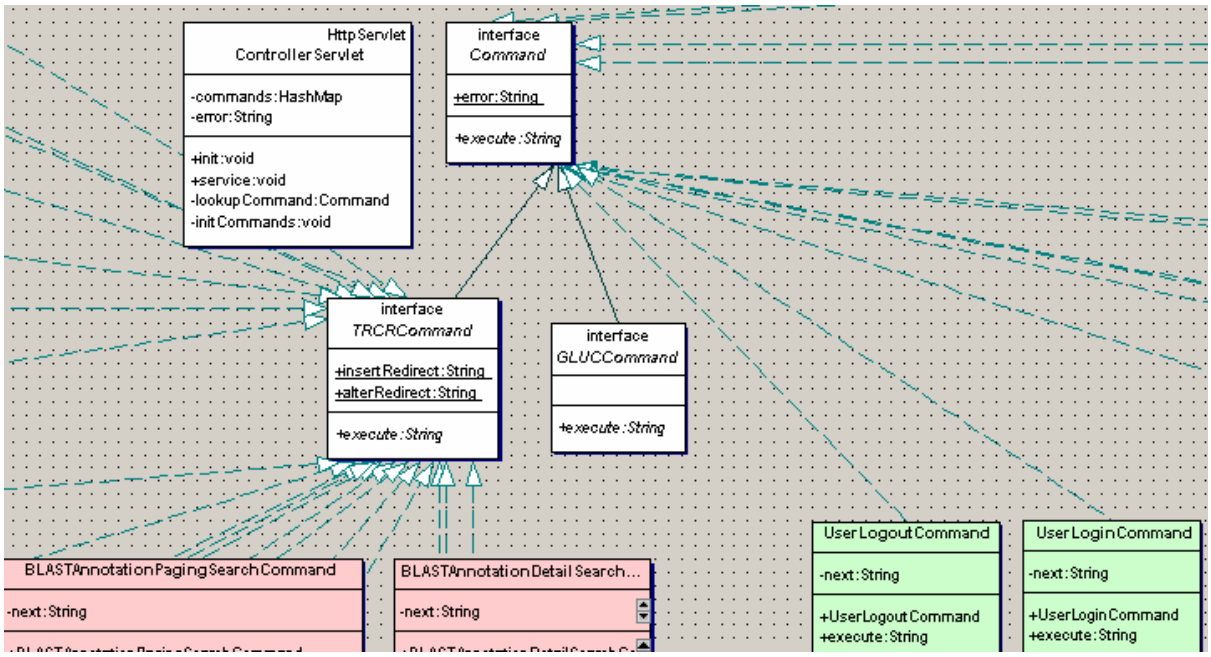


Figura 16. Resumo do Diagrama de Classes do Controlador.

3.4.2.2 Lógica do Negócio

A camada da lógica do negócio representa os objetos que processam os dados do cliente e retornam uma resposta. Esta camada também inclui os dados do banco de dados. Os objetos desta camada podem incluir Enterprise JavaBeans, JavaBeans, e classes Java. O controlador invoca os objetos da camada da lógica do negócio.

Depois que o controlador reconhece um pedido, os objetos do modelo executam o trabalho para responder tal pedido. Por exemplo, os objetos podem receber parâmetros, validar o pedido, autenticar o cliente, iniciar a transação e acessar o banco de dados.

3.4.2.2.1 Tipos de Anotações

Um das classes que mais importantes no nosso sistema são as que representam as anotações. O diagrama de classes resumido da Figura 17 apresenta algumas destas anotações. Estão representadas a classe *AnnotationObject* que possui especializações para as anotações das comunidades que o sistema está atendendo. Como atualmente existem duas comunidades que estão utilizando o sistema, tem-se duas sub-classes: a do organismo *Trypanosoma cruzi* (*TRCR AnnotationObject*) e a do organismo *Glucona* (*GLUCAnnotationObject*). As classes que tratam das anotações das comunidades também se especializam. Por exemplo, no caso do *Trypanosoma cruzi* existem classes para tratar anotações de aplicativos externos (como BLAST (*BLASTAnnotationObject*) e CAP3 (classe *CAP3AnnotationObject*)), fontes de dados externas (como Genbank (classe *AuthorAnnotationObject*)) e anotações da comunidade (como *CommunityAnnotationObject*).

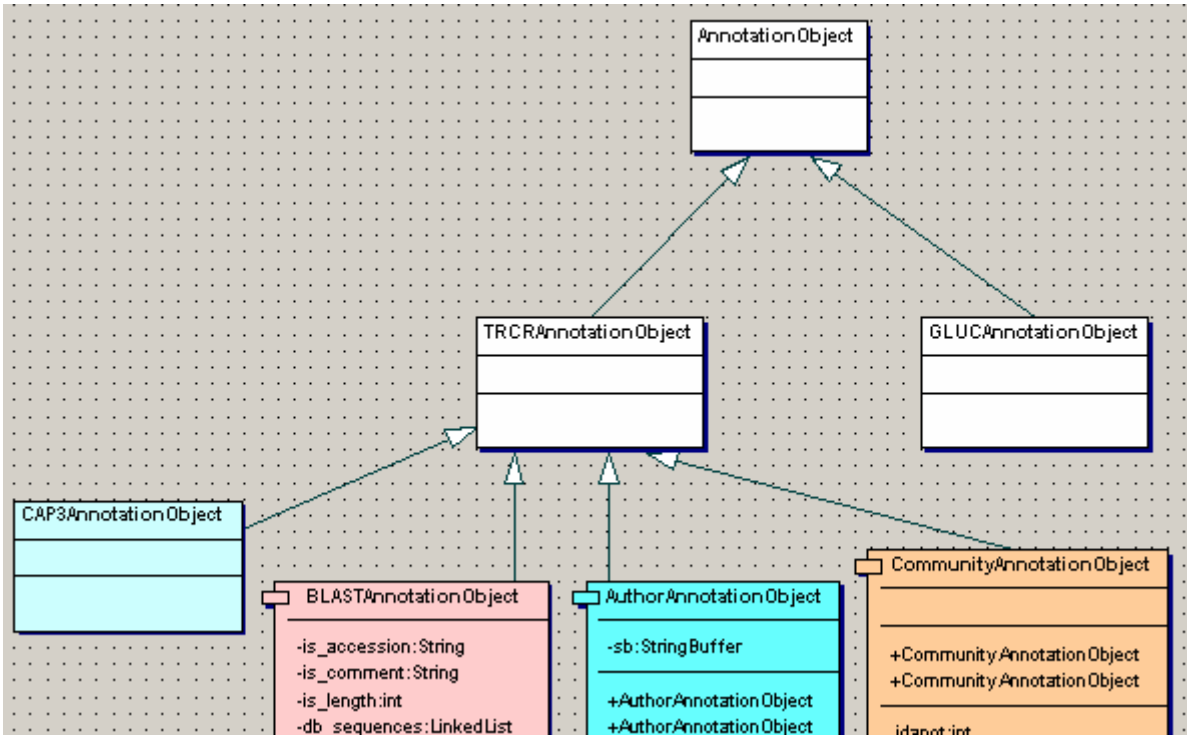


Figura 17. Resumo do Diagrama de Classes de Anotações.

3.4.2.2.2 Captura das Anotações das Fontes de Dados e das Aplicações Externas

Como descrito antes, o sistema BioNotes captura jnformações sobre anotações de fontes de dados públicas e de aplicações externas.

As fontes de dados públicas como Genbank, EMBL, PIR, SWISSPROT e Prosite disponibilizam seus dados em arquivos texto, em formatos específicos e documentados. Após a obtenção dos dados é necessário transformar o formato dos mesmos para aquele esperado pelo BioNotes, que adota o modelo de dados semi-estruturado, no padrão XML.

Como XML está cada vez mais sendo utilizado e divulgado, atualmente várias fontes de dados já estão distribuindo seus dados em seus formatos específicos e também em XML. Em conjunto com os dados em XML, há o esquema em XMLSchema ou DTD. Este é o caso atualmente do EMBL, Genbank e DDBJ (XML at NCBI, 2002)(WANG, L., RIETHOVEN, J.J.M., ROBINSON, A.J., 2002), PIR (PIR-XML, 2002) e SWISSPROT (SPTr-XML Documentation, 2002).

Para que o BioNotes represente os dados das fontes que não divulgam suas informações no formato XML, estão sendo construídos programas que transformam os dados para XML, de acordo com o esquema (XMLSchema) definido no projeto da aplicação.

Com relação aos aplicativos externos, estes precisam ser instalados em uma máquina que o sistema tenha acesso. Os aplicativos são executados sobre os dados contidos no banco de dados do BioNotes e seus resultados são também armazenados no banco de dados (em XML). Por exemplo, para executar o BLAST das sequências do *Trypanosoma cruzi* contra as sequências deste organismo existentes no Genbank, é feita uma consulta no banco de dados do BioNotes que retorna todos os dados de Trypanosoma cruzi relevantes para o BLAST. Estes dados são transformados para um arquivo texto no formato FASTA (formato de entrada do BLAST). Para cada sequência de *Trypanosoma cruzi* existente no Genbank, o BLAST é executado e o seu resultado é descrito em XML (de acordo com o XMLSchema definido no projeto) e é também armazenado no banco de dados do BioNotes.

A execução dos aplicativos e o armazenamento das anotações resultantes dessa execução é uma tarefa que consome muito tempo, mas que é necessária porque os usuários têm acesso imediato às anotações.

Outra questão que deve ser tratada é de quanto em quanto tempo um determinado aplicativo deve ser novamente executado sobre os dados. Isto dependerá dos objetivos de cada comunidade, mas o sistema está projetado para que o aplicativo possa ser executado quantas vezes e quando se queira. Geralmente os resultados dos aplicativos se alteram à medida que novos dados são armazenados no banco de dados. Por exemplo, quando um genoma ainda não está completo, enquanto existirem novos *reads* sendo sequenciados, os resultados de programas como Phrap e CAP3 formarão provavelmente novos contigs. Assim, programas que têm como entrada os contigs, como Glimmer e tRNAScan, também terão resultados diferentes. Com isto, as análises e as anotações geradas pela comunidade que dependem destes resultados também podem se modificar. Desta forma, é necessário tratar as versões das anotações. Isso ocorre, por exemplo, para se saber quais são os contigs, reads e anotações de cada etapa de sequenciamento de um genoma.

Sendo assim, é possível fazer anotações, mesmo sobre genomas ainda incompletos, que estejam em fase de sequenciamento, pois o procedimento de controle de versões permite um acompanhamento das anotações ao longo do sequenciamento do genoma, liberando dados para análise dos pesquisadores, sem que eles precisem esperar que o genoma esteja totalmente sequenciado para começar a interpretá-lo.

Os diagramas de classes resumidos apresentados na Figura 18, Figura 19 e Figura 20 mostram as classes responsáveis pela

- obtenção dos dados em XML no banco de dados dados do BioNotes (em XML) e a transformação para o formato esperado pelos aplicativos (classe *DataRetrieve*);
- execução dos aplicativos (classe *ApplicationRun*);
- obtenção do resultado dos aplicativos, transformação para XML e armazenamento no banco de dados do BioNotes (classe *DataLoad*).

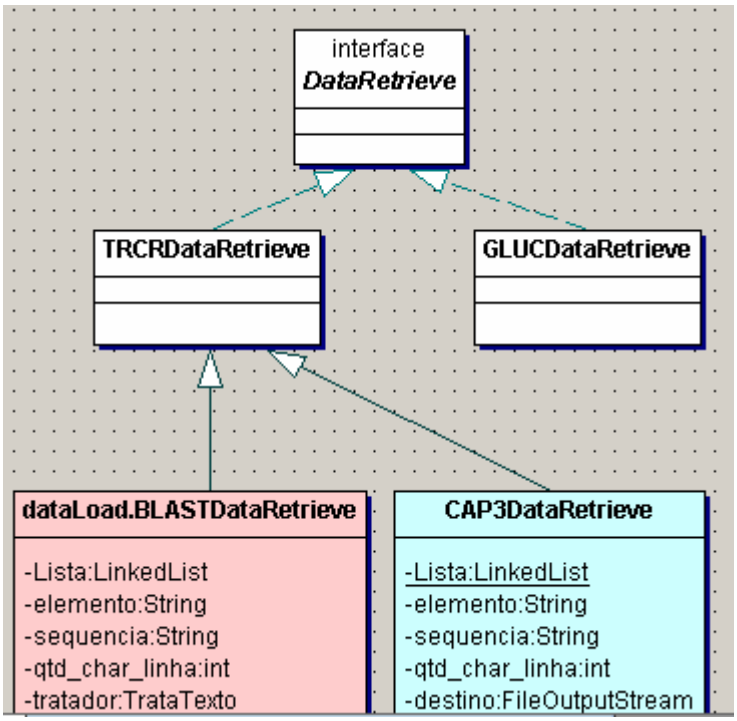


Figura 18. Resumo do Diagrama de Classes de Recuperação dos Dados no Banco de Dados para Execução de Aplicativos.

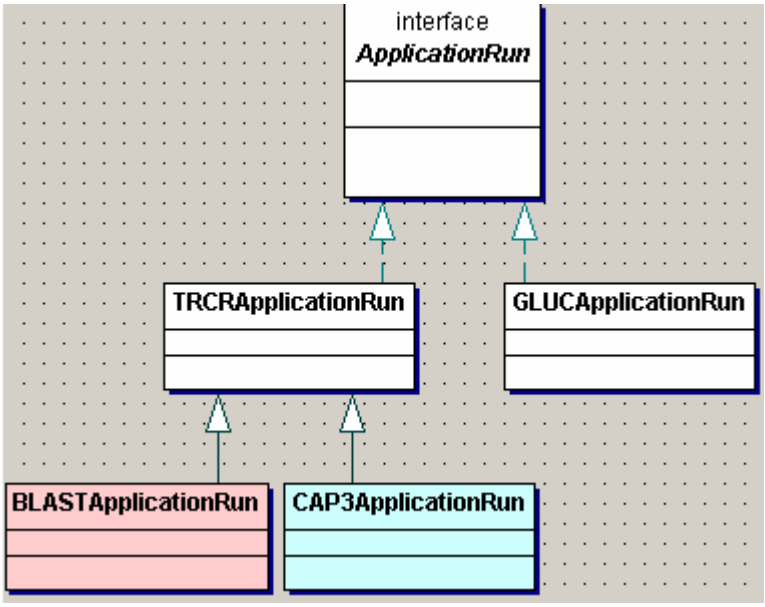


Figura 19. Resumo do Diagrama de Classes de Execução de Aplicativos.

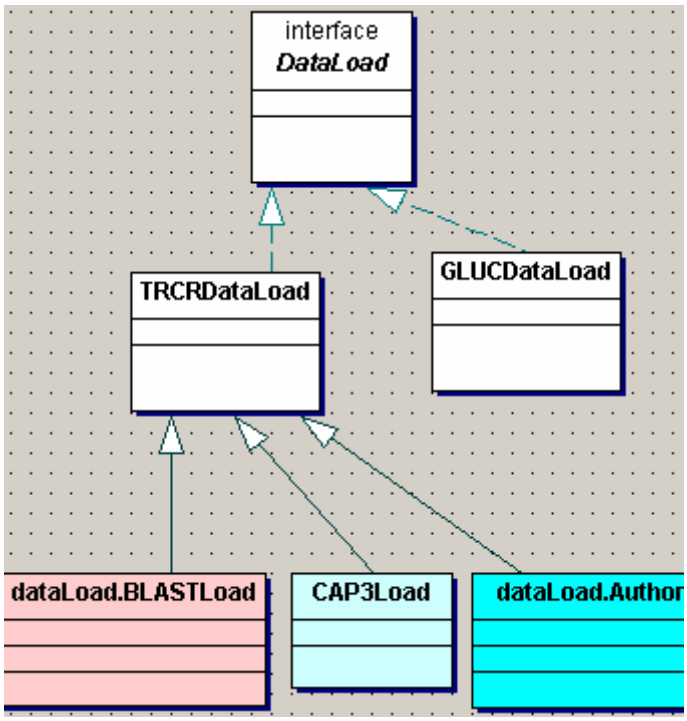


Figura 20. Resumo do Diagrama de Classes de Armazenamento de Resultados de Aplicativos no Banco de Dados.

3.4.2.2.3 Acesso ao Banco de Dados

Tipicamente, os objetos da camada da lógica do negócio lêem e atualizam dados no banco de dados através do uso do padrão de projeto DAO (*data access objects*) que separa o acesso ao banco de dados do restante da camada da lógica do negócio. Isto permite isolar os comandos SQL que são enviados para o banco de dados e dar flexibilidade às mudanças na fonte de dados. Se alguma mudança for feita no banco de dados (por exemplo, renomear ou mudar a estrutura dos dados), é necessário atualizar os comandos SQL nos objetos da camada do DAO sem se alterar o restante da aplicação. Além disso, o padrão DAO pode se tornar altamente flexível se for adotado o padrão *Abstract Factory* (Gamma, E. et.al., 1994). O padrão *Factory* retorna uma instância de uma ou várias classes, dependendo do dado apresentado (Java Tutorial - Design Patterns, 2002).

Outro aspecto importante de uma aplicação é a conexão com o banco de dados. Ao invés de se criar uma nova conexão a cada comunicação com o banco de dados, é interessante utilizar um *pool* de conexões, que consiste de uma coleção de objetos instanciados do tipo *Connection*.

A criação, a abertura e o fechamento de uma conexão com o banco de dados envolve um pequeno, mas significante *overhead* de memória e tempo. Usar um *pooling* de conexões provê várias vantagens. Primeiro, os objetos Connection já estão criados esperando para serem usados. Isto reduz tempo e memória de criação de uma nova conexão a cada acesso ao banco de dados. Depois, o sistema trata do ciclo de vida da conexão, o que inclui criação, fechamento e geração de novas conexões. Isto significa que mais tempo será dedicado à tarefas específicas e menos tempo à administração das conexões com o banco de dados.

A nossa proposta usa um DAOFactory para obter um DAO. O DAO configura a conexão com o banco de dados através de um *pooling* de conexões, executa os comandos SQL e retorna os dados. Veja o diagrama de classes resumido da Figura 21. Como o BioNotes utiliza o Sistema de Gerência de Banco de Dados Oracle, temos especializações para as classes *DAOFactory* e *DAO*, que são *OracleDAOFactory* e *OracleDAO*. A classe *OracleDAO* também possui várias especializações: uma para os usuários (classe *UserOracleDAO*), outra para as instituições (classe *InstitutionOracleDAO*), uma para cada comunidade existente, no caso *Trypanosoma cruzi* (*TRCROracleDAO*) e *Glucona* (*GLUCOracleDAO*). As classes que tratam das comunidades também se especializam. Como exemplo, note que a comunidade de *Trypanosoma cruzi* tem classes para tratar anotações de aplicativos externos (como BLAST (*BLASTAnnotationOracleDAO*) e CAP3 (classe *CAP3AnnotationOracleDAO*)), fontes de dados externas (como Genbank (classe *AuthorAnnotationOracleDAO*)) e anotações da comunidade (como *CommunityAnnotationOracleDAO*).

3.4.2.2.4 Consultas que Retornam Lista com Inúmeros Ítens

Outro padrão de projeto importante é o *Iterator*. Ele é útil quando um cliente pede uma lista de ítens, sendo que o número de ítens é desconhecido podendo ser muito grande. Isto acontece muito quando o cliente faz uma consulta ao banco de dados. Este padrão possui uma classe chamada *ValueListHandler* que controla a execução da consulta e o cache do resultado. O *ValueListHandler* armazena o resultado obtido através do DAO como uma coleção de objetos. O cliente requisita o *ValueListHandler* para obter os resultados da consulta quando necessário. O *ValueListHandler* implementa o padrão *Iterator* para prover esta solução. Em especial, as consultas SQL feitas pelo BioNotes são tratadas com o padrão *Iterator*. Vide o diagrama de classes resumido da Figura 22. Existem especializações para a

classe *ValueListHandler*. No diagrama, há uma especialização que trata de consultas em usuários do sistema (classe *UserListHandler*) e outras que tratam das duas comunidades existentes (classes *TRCRListHandler* e *GLUCListHandler*). A classe *TRCRListHandler* possui especializações para consultas realizadas em anotações obtidas pelo BLAST (*BLASTAnnotationListHandler*), CAP3 (*CAP3AnnotationListHandler*) e Genbank (*AuthorAnnotationListHandler*) em sequências de *Trypanosoma cruzi*.

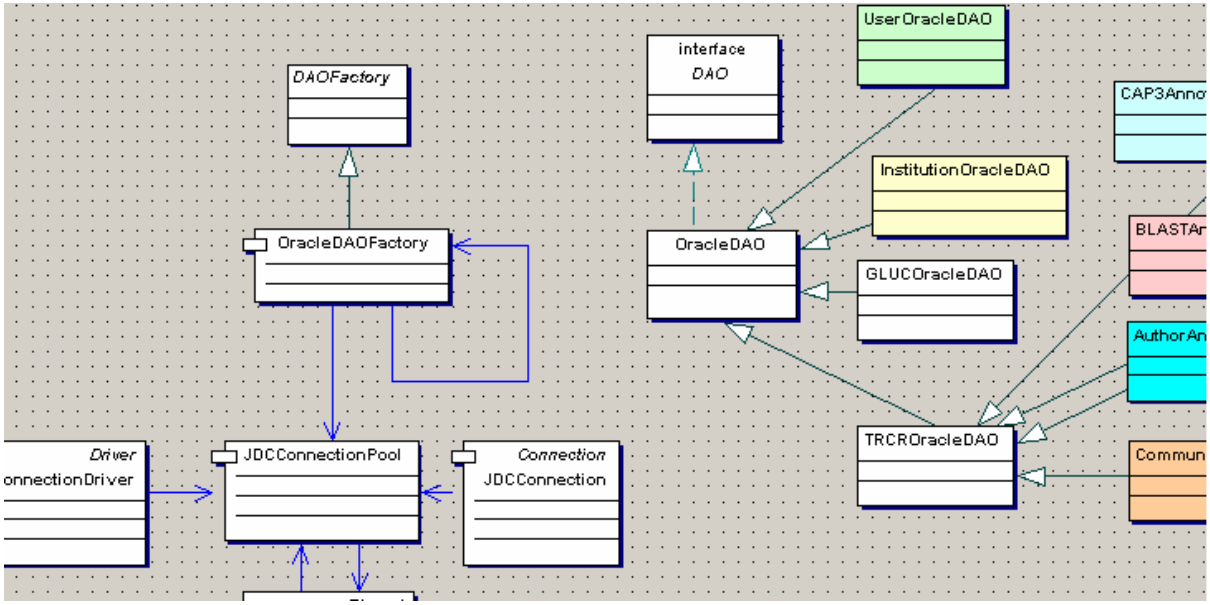


Figura 21. Resumo do Diagrama de Classes do Repositório.

3.4.2.2.5 Tratamento de Erro e de Log

O tratamento de erro é essencial em qualquer sistema. No BioNotes, existe uma classe chamada *FailHandler* e várias subclasses desta que tratam erros de quaisquer comandos executados no sistema. Sempre que um erro é gerado (como, por exemplo, quando o sistema não consegue conexão com o banco de dados, quando o sistema não consegue acessar algum arquivo, quando um usuário fornece o nome ou a senha incorreta, e quando um usuário tenta cadastrar uma instituição que já foi cadastrada, etc.), alguma subclasse específica que trata aquele erro é criada. Uma instância desta classe envia uma mensagem explicativa para o usuário indicando que erro ocorreu.

Além disso, existe a classe *FailLog* que armazena o erro ocorrido em um arquivo texto específico para um usuário. Desta forma, existirá um arquivo texto para cada usuário do

sistema contendo informações de erros que ocorreram com ele durante a execução do sistema. Isto permite atender melhor os usuários e facilita a manutenção do sistema.

O diagrama de classes resumido na Figura 23 exemplifica este tratamento.

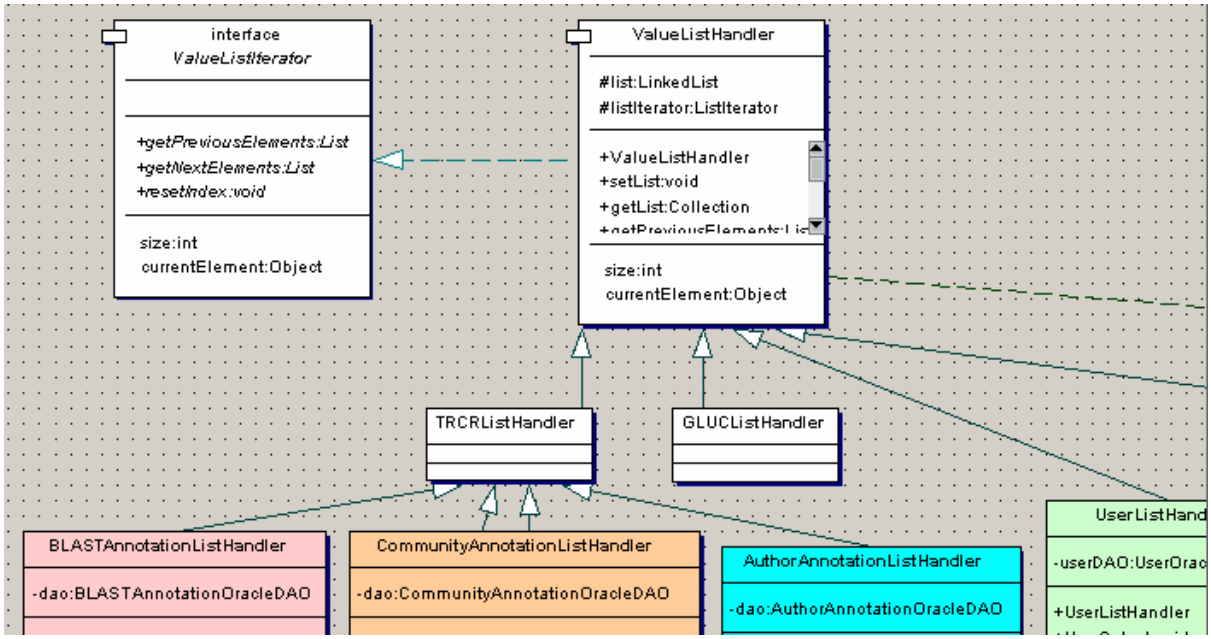


Figura 22. Resumo do Diagrama de Classes do Iterador.

3.4.2.2.6 Controle de Usuário

Como o sistema está dedicado para atender usuários com perfis diferentes, ele está preparado para perceber o perfil do usuário e deixar que ele execute somente os comandos que tem direito.

O sistema permite o cadastro, a remoção, a consulta e a atualização de anotações biológicas. Existem usuários que têm direito de executar todos estes comandos, enquanto outros podem apenas consultar.

Além disso o sistema atende a diferentes comunidades de usuários. Cada comunidade tem o seu grupo de usuários e não é permitido que um usuário que não está cadastrado em uma comunidade, acesse os dados daquela comunidade. Para efetivar este controle, existe o papel

de usuário administrador da comunidade, que tem o direito de cadastrar, atualizar, remover e consultar os usuários de sua comunidade.

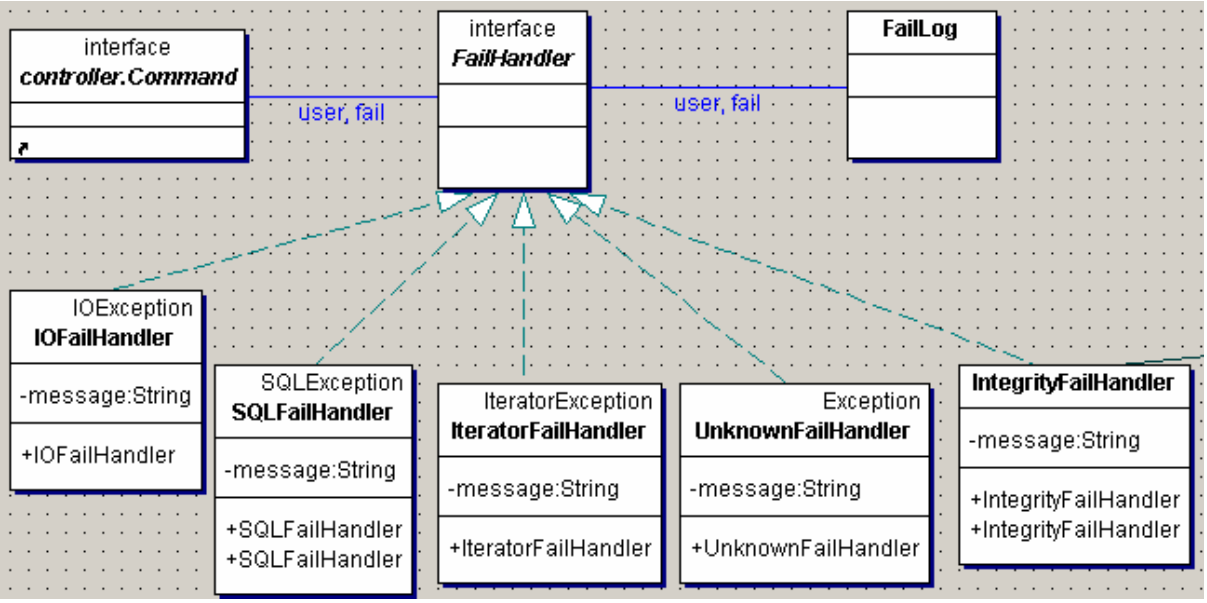


Figura 23. Resumo do Diagrama de Classes de Tratamento de Falhas e de Log do Sistema.

3.4.2.3 *View*

O módulo visualizador ou camada de apresentação trata da apresentação dos dados como tags HTML com os dados retornados da camada da lógica do negócio. Os dados das camadas de apresentação e da lógica do negócio são enviados para o cliente como resposta à sua requisição. As tags HTML geralmente têm dados e elementos em formulários (como caixas de textos e botões) que permitem ao usuário interagir com a aplicação, bem como outros elementos da camada de apresentação. Os dados da lógica do negócio vêm da camada da lógica do negócio, enquanto os dados da apresentação são arquivos JSP. Desta forma, há uma separação entre os dados da camada apresentação e da camada lógica do negócio.

Um benefício de codificar separadamente o negócio e a apresentação é que facilita a extensão da aplicação para se adequar a diferentes clientes. Novos clientes da aplicação podem não querer ter uma visão gráfica, e sim, apenas as tags apresentadas. Eles podem estar interessados apenas no resultado, que podem tratar de forma conveniente. Separando estes

módulos, é mais fácil reusar os objetos da camada da lógica do negócio com novas formas de apresentação dos dados.

3.4.2.3.1 Session Tracking

Na Web existem vários métodos de associar uma sessão a um usuário, todas envolvendo a passagem de um identificador entre o cliente e o servidor. Os métodos principais exigem que o cliente aceite cookies ou que o componente Web reescreva qualquer URL que é retornada para o cliente.

Como o BioNotes usa objetos do tipo sessão (por exemplo, para armazenar dados do usuário que está acessando o sistema naquela sessão), foi implementado o *session tracking* permitindo que a aplicação reescreva a URL sempre que o cliente desabilitar seus cookies. Este método inclui o identificador da sessão na URL quando o cookie está desabilitado, caso contrário, a URL continua a mesma. (Java Tutorial-Web, 2002)

A Figura 24 apresenta a arquitetura do sistema.

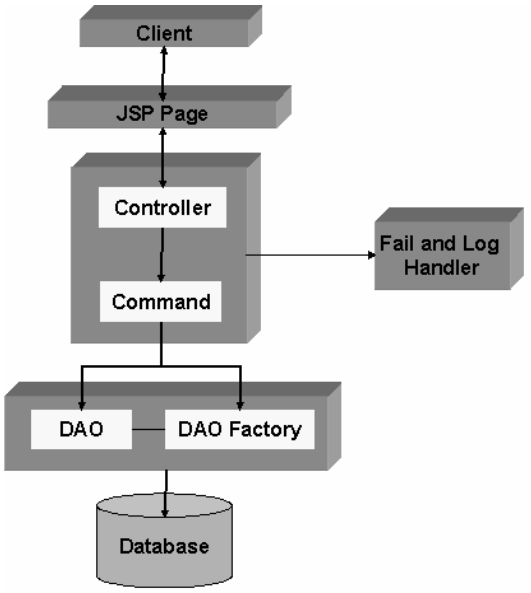


Figura 24. Esquema da Arquitetura da Aplicação.

4 Comparação entre os Sistemas de Anotações de Biossequências

Foram apresentados diversos sistemas de anotações que possuem aspectos diferentes, sejam eles tecnológicos ou funcionais,como:

- o modelo de dados,
- a linguagem de programação utilizada,
- a interface adotada,
- o sistema de gerência de bancos de dados utilizado,
- a forma de armazenamento forma das anotações das fontes de dados externas,
- processamento dos programas de análise dos dados,
- o controle de versão das anotações, e
- tratamento das anotações manuais.

A **Error! Reference source not found.**, apresentada a seguir, permite que seja feita uma comparação entre os sistemas estudados.

Quanto ao modelo de dados, alguns sistemas utilizam o modelo relacional, outros o modelo orientado a objeto, e outros o modelo relacional em conjunto com o modelo de dados semi-estruturado. Os sistemas que adotam um modelo de dados estruturado, quer seja relacional ou orientado a objeto, têm como desvantagem a necessidade do entendimento de todas as fontes de dados externas e aplicativos externos para a criação de um modelo global único centralizador. Caso contrário, qualquer nova fonte de dados que deva ser incorporada pode provocar alterações no modelo que podem ser de difícil implementação. Geralmente estas fontes de dados e aplicativos são de difícil entendimento e, consequentemente, a criação do modelo único também. O modelo global único pode conter um enorme conjunto de tabelas ou classes de objetos de difícil compreensão e, normalmente, possui várias colunas ou atributos com campos nulos. Este modelo é geralmente difícil de manter e não permite tratar a evolução de esquemas.

Já o modelo semi-estruturado traz vantagens na alteração do esquema adotado, uma vez que é relativamente simples a remoção e remoção de tags. Além disso, este modelo facilita a compreensão das informações pois os dados são armazenados em formato texto (podem ser lidos por qualquer editor) e incluem descrições dos mesmos. Outra vantagem é que não é

necessário a existência de atributos nulos. A principal desvantagem da adoção do modelo de dads semi-estruturado é que a consulta em texto é normalmente mais lenta, mas vários estudos estão sendo feitos e, hoje, já é possível criar índices em tags do XML, melhorando o desempenho das consultas.

Várias linguagens de programação são utilizadas na bioinformática, destacando-se Perl, C e Java. Normalmente a linguagem Perl é escolhida porque ela é muito utilizada pelos biólogos, já que eles constróem muitos parsers para fazer transformações e análises nos dados e esta linguagem é adequada para tratar strings. A linguagem Java, adotada por alguns sistemas, é uma escolha vantajosa porque permite a portabilidade do código, é orientada a objeto, o que permite o reuso de código e, caso haja um bom modelo orientado a objeto projetado, há uma grande facilidade no entendimento, manutenção e extensão do código. A desvantagem da utilização de uma linguagem como Java é que requer profissionais altamente qualificados que possuem custo alto.

As interfaces escolhidas podem ser locais ou em HTML. Uma das principais vantagens de se utilizar HTML (em conjunto com JSP, por exemplo) é a distribuição das anotações e a facilidade de acesso a estas informações de qualquer computador. Se cada usuário tiver seus dados em seu próprio computador e fizer sua própria análise, as informações não serão compartilhadas e o processo de descoberta de novas informações biológicas se tornará mais lento.

Quanto ao banco de dados, vários sistemas utilizam o SGBD MySQL porque não tem custo. A vantagem de se utilizar um banco de dados comercial, como Oracle, está na quantidade de facilidades e recursos que possui, como por exemplo o tratamento de dados em XML, fundamental para o BioNotes.

Sobre a forma de armazenamento das anotações das fontes de dados externas, alguns sistemas possuem os dados armazenados em um *data warehouse* e outros possuem links para os registros das fontes de dados externas. A existência de um *data warehouse* é vantajosa porque a consulta se torna mais rápida em uma base local do que em uma base externa, mas é necessário controlar a atualização dos dados porque é constante a inserção e atualização de dados nas fontes externas. Já a estratégia de links para as fontes externas não precisa tratar da atualização de dados, mas está sensível às mudanças dos esquemas das fontes de dados.

Quanto ao processamento dos programas de análise dos dados, alguns sistemas de anotações existentes realizam esta tarefa a priori e armazenam os resultados em um data warehouse. Outros sistemas permitem que o usuário executem os programas. A desvantagem desta escolha é que normalmente a execução destes programas é demorada, o que faz com que o usuário tenha que aguardar muito tempo para obter o resultado. A complicação de se ter um data warehouse que armazene resultados dos programas é que deve haver espaço em disco suficiente e deve ser realizada a execução dos programas de tempos em tempos para que os resultados sejam atualizados.

Outra diferença que existe entre estes sistemas é o tratamento de anotações manuais feitas pelos usuários do sistema. Alguns sistemas permitem que o usuário analise os dados e armazene anotações, que são resultados de sua pesquisa, no data warehouse. Isto é interessante porque permite o controle, organização, compartilhamento e distribuição das anotações para a comunidade de pesquisadores. Geralmente estas anotações manuais estão baseadas em anotações automáticas (oriundas de fontes de dados externas e da execução de programas) e, portanto, elas podem se modificar conforme outras anotações se modifiquem. Por exemplo, em um projeto de genoma em andamento, a quantidade de reads sequenciados aumenta e, consequentemente, a montagem dos reads se modifica e o número de contigs tende a diminuir. Com isso, os possíveis genes encontrados em um contig passam a ser encontrados em outro contig. Neste caso deve existir uma maneira de relacionar as anotações feitas para este gene que estava localizado no contig antigo para o contig novo. A maneira mais simples e confiável de se fazer isso é permitir que o próprio pesquisador compare os resultados das montagens dos reads e repasse as anotações manuais feitas da montagem antiga para a nova. É claro que o sistema pode ajudar a fazer isso, dando pistas de quais contigs se uniram e quais se separaram e investigando os reads que os contigs possuem. Esta tarefa consiste de um controle de versão das anotações, já que, em um projeto em andamento, elas tendem a se modificar.

Aspecto	Funcional			Tecnológico				
Sistema/ Tópico	Acessa Fontes de Dados Externas	Executa Aplicativos Externos	Armazena Anotações de Usuários do Sistema	Armazena dados de Fontes de Dados Externas	Armazena Resultados Pré-calculados de	Modelo de Dados	BD	Linguagem
Artemis	S	S	S	N	N	-	-	Java
DAS	S	S	N	N	N	-	-	Java
Celera Browser	1	S	N, 2	1	N	R	?	Java
EDITtoTrEMBL	S	S	N	N	N	-	-	Java
GenDB	S	S	S	N	N	R	MySQL	Perl
GeneMine	S	S	N	N	N	-	-	Perl
GeneQuiz	S	S	N	S, 3	S	R	RDB	Perl
Apollo	S	S	S	S?	S?	R?	?	Java
GGB	S?	?	S	?	?	R	MySQL	Perl
Imagene	-	S	-	-	-	-	-	Comp. Ilog.
MAGPIE	S	S	N	S	S	-	-	Perl,C,Prolog
Manatee	S?	S?	S	S?	S?	R	MySQL	Perl
PEDANT	S	S	S	S	S	R	MySQL	Perl,C++
Visual Genome	S	S, 4	N?	S	S	?	?	C
BioNotes	S	S	S	S	S	R, SM	Oracle	Java

Tabela 1. Comparação dos sistemas de anotações existentes.

Legenda
S = Sim, N = Não, - = Não se aplica, ? = Não é informado em nenhum paper, R = relacional, O = orientado a objeto, SM = semi estruturado
1(Somente da própria Celera), 2 permite que o usuário crie e armazene em sua máquina, 3 não armazena tudo, mas um resumo das fontes de dados externas , 4 somente o BLAST

5 Conclusão

Este trabalho apresentou um estudo dos diversos sistemas de anotações existentes, de forma a se ter parâmetros de comparação entre elas e a ferramenta desenvolvida na Pontifícia Universidade Católica do Rio de Janeiro, denominada BioNotes.

Esta ferramenta vem sendo utilizada para anotação do genoma do *Trypanossoma cruzi* (projeto de anotações que é liderado pela Fundação Oswaldo Cruz e vem sendo realizado em conjunto com laboratórios europeus e sul-americanos) e para anotação do genoma da *Gluconacetobacter diazotrophicus* (que vem sendo realizado no âmbito do projeto RioGene, que agrega diversos laboratórios localizados no Rio de Janeiro).

O projeto e a implementação do BioNotes, também documentados nesta monografia, trazem uma nova proposta de sistema de anotações, que traz vantagens aos pesquisadores da área, não só em termos de facilidades de uso, como também de desempenho computacional.

A comparação entre os sistemas estudados, incluindo-se o BioNotes, foi uma consequência natural e foi também inserida neste trabalho.

Os sistemas de anotação tem como objetivo principal ajudar os pesquisadores a anotar informações biológicas consideradas relevantes sobre os dados em estudo. Para que este objetivo seja alcançado, vários desafios precisam ser tratados, como, por exemplo, o entendimento das inúmeras fontes de dados externas, heterogêneas, com enorme volume de dados e em constante crescimento, e lidar com diversos programas de análise dos dados, que geralmente não são documentados e que apresentam problemas em sua execução, dificultando sobremaneira o seu uso. Geralmente estas fontes de dados externas não possuem uma documentação detalhada de seu esquema. Também os programas de análise não disponibilizam informações detalhadas sobre parâmetros de execução, formato dos dados de entrada e dados de saída, dificultando a definição de um workflow para o projeto.

Em termos funcionais, os principais requisitos de um sistema de anotação referem-se a:

- Execução prévia dos aplicativos e anotação das saídas de cada um;
- Manutenção de versões de anotações, especialmente útil para anotar dados de genoma com sequenciamento ainda incompleto;

- Visualização das informações de forma gráfica (visualização dos elementos que compõem o genoma, como: cromossomo, read, contig, ORF, entre outros) e tabular, de forma a apresentar anotações que estão atribuídas aos elementos nas fontes de dados, possibilitando o estudo e verificação por comparação visual;
- Possibilitar anotações de forma distribuída;
- Acessar fontes de dados externas via links para validação de anotações vindas destas fontes;
- Etc...

Em termos tecnológicos, os sistemas diferem quanto a:

- Disponibilizar interface web para os pesquisadores;
- Manter as anotações em um datawarehouse;
- Possibilitar o uso de linguagem de consulta aos dados;
- Construção do workflow de execução dos aplicativos;
- Etc...

O quadro comparativo apresentado no capítulo 4 permite concluir que o Sistema BioNotes traz uma nova abordagem para tratar o problema, uma vez que trata de forma conveniente quase todos os requisitos necessários.

Os requisitos ainda não contemplados pelo BioNotes estão em estudo e serão objeto de futuras versões do sistema. Entre estes podemos citar a possibilidade de construção do workflow do projeto, a partir do uso dos aplicativos registrados no sistema.

6 Referências

ANDRADE, M.A. ET. AL. Automated genome sequence analysis and annotation. Bioinformatics, 1999. 15 391-412.

Altschul, S.F., et.al.. "A basic local alignment search tool". J. of Molecular Biology 215, pp. 403-410, 1990.

ARTEMIS. Disponível em http://www.sanger.ac.uk/Software/Artemis/. Acesso em Novembro de 2002.

APOLLO. Disponível em http://www.fruitfly.org/annot/apollo/ e http://gmod.sourceforge.net/. Acesso em Novembro de 2002.

ASAP. Alternative Splicing Annotation Project. Disponível em http://www.bioinformatics.ucla.edu/HASDB/generic.php3. Acesso em Janeiro de 2003.

BISSON, G. , GARREAU, A.. APIC – a generic interface for sequencing projects. In Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology. 1995. AAAI Press, Menlo Park, CA, pp. 57-65.

BLAST. Basic Local Alignment Search Tool. Disponível em http://www.ncbi.nlm.nih.gov/BLAST/. Acesso em Dezembro de 2002.

BLOCKS. BLOCKS WWW Server. Disponível em http://www.blocks.fhcrc.org/. Acesso em Dezembro de 2002.

CANCER ANNOTATION PROJECT. Disponível em http://cancer.lbi.ic.unicamp.br/. Acesso em Janeiro de 2003.

CAP. Community Annotation Project. Disponível em http://www-genome.wi.mit.edu/annotation/microbes/methanosarcina/sarcinaCAP/. Acesso em Janeiro de 2003.

CAP3. CAP3 Sequence Assembly Program. Disponível em http://genome.cs.mtu.edu/cap/cap3.html. Acesso em Dezembro de 2002.

CELERABROWSER. Disponível em http://www.celera.com/genomics/commercial/home.cfm?ppage=genomebrowser e em http://www.celera.com/genomics/commercial/home.cfm?ppage=literature. Acesso em Novembro de 2002.

Cooper, N.. The Human Genome Project. Univ. Science Books, Mill Valey, CA, 1994.

CROSSMATCH. Disponível em http://www.phrap.org/phrap.docs/general.html. Acesso em Dezembro de 2002.

DAS. Distributed Annotation System. Disponível em http://www.biodas.org/. Acesso em Novembro de 2002.

EMBL. European Molecular Biology Laboratory. Disponível em http://www1.embl-heidelberg.de/. Acesso em Dezembro de 2002.

ENSEMBL. Ensembl Genome Browser. Disponível em http://www.ensembl.org/. Acesso em Dezembro de 2002.

ENZYME. Enzyme nomenclature database. Disponível em http://us.expasy.org/enzyme/. Acesso em Dezembro de 2002.

FRISHMAN D. ET. AL. Combining weak diverse evidence for gene recognition in completely sequenced bacterial genomes. Nucl. Acids Res. 1998, in press.

FRISHMAN, D. ET. AL. Functional and structural genomics using PEDANT. Bioinformatics 2001 17, p. 44-57.

FRISHMAN, D., MEWES, H.W.. Protein structural classes in five complete genomes. Nature Struct. Biol.. 1997. 4, p. 626-628.

FRISHMAN, D., MEWES, H.W.. PEDANTic genome analysis. Trends in Genetics 1997 , 13, p. 415-416.

GAASTERLAND, T, SENSEN, C.W.. "MAGPIE: Automated Genome Interpretation," Trends in Genetics 12, 76–78 (1996).

GAIA. GAIA Web Site. Disponível em http://www.cbil.upenn.edu/gaia2/gaia/. Acesso em Dezembro de 2002.

GAME XML Genome Annotation Markup Elements XML. Disponível em http://www.discoverylogic.com/logic2/xmllinks.htm, http://www.xml.org/xml/industrySectorList.jsp?CATEGORY=50 e http://www.bioxml.org/Projects/game/. Acesso em 18 de Fevereiro de 2003.

GAMMA, E. ET. AL. Design Patterns - Elements of Reusable Object-Oriented Software, Addison Wesley Professional Computing Series. 1994. ISBN 0-201-63361-2.

GBROWSER. Disponível em http://gmod.sourceforge.net/ e http://www.gmod.org/ggb/index.shtml. Acesso em Novembro de 2002.

GENBANK. Disponível em http://www.ncbi.nlm.nih.gov/Genbank/index.html. Acesso em Dezembro de 2002.

GENDB. Disponível em http://gendb.Genetik.Uni-Bielefeld.DE/. Acesso em Novembro de 2002.

GENEMINE. Disponível em http://www.bioinformatics.ucla.edu/genemine/. Acesso em Novembro de 2002.

GENEQUIZ. Disponível em http://jura.ebi.ac.uk:8765/ext-genequiz/. Acesso em Novembro de 2002.

GENESTREAM. Disponível em http://xylian.igh.cnrs.fr/getseq/genbank_sequence_finder.html. Acesso em Janeiro de 2003.

GENOTATOR. Genotator Web Site. Disponível em http://www.fruitfly.org/~nomi/genotator/. Acesso em Dezembro de 2002.

GENSCAN. The New GENSCAN Web Server at MIT. Disponível em http://genes.mit.edu/GENSCAN.html. Acesso em Dezembro de 2002.

GFF. GFF: an Exchange Format for Feature Description. Disponível em http://www.sanger.ac.uk/Software/formats/GFF/. Acesso em Dezembro de 2002.

GLIMMER. The GLIMMER Homepage. Disponível em http://www.tigr.org/software/glimmer/. Acesso em Dezembro de 2002.

GMOD. Generic Model Organism Database Construction Set. Disponível em http://www.gmod.org/. Acesso em Dezembro de 2002.

GOESMANN, A. ET.AL. PathFinder: reconstruction and dynamic visualization of metabolic pathways. Bioinformatics 2002 18: 124-129.

HARRIS, N.L. (1997). Genotator: A workbench for sequence annotation. Genome Research 7(7):754-762.

HMMPfam. HMMPfam - search sequences against an HMM database. Disponível em http://bioweb.pasteur.fr/seqanal/interfaces/hmmpfam.html.

HOERSCH, S ET.AL.. The GeneQuiz Web server: protein functional analysis through the Web. Trends in Biochemical Sciences, 2000. 25, p. 33-35.

IMAGENE. Disponível em http://wwwabi.snv.jussieu.fr/research/thema/imagene/index.html. Acesso em Novembro de 2002.

ILOG-CONTROL, 1997. ILOG CONTROL Version 1.4 – Reference Manual. *ILOG*. Disponível em http://www.ilog.com. Acesso em Novembro de 2002.

ILOG-POWER-CLASSES, 1996. ILOG POWER CLASSES Version 1.3 – Reference Manual. *ILOG*. Disponível em http://www.ilog.com. Acesso em Novembro de 2002.

ILOG-TALK, 1995. ILOG TALK Version 3.13 – Reference Manual. *ILOG*. Disponível em http://www.ilog.com. Acesso em Novembro de 2002.

ILOG-VIEWS, 1995. ILOG VIEWS Version 2.3 – Reference Manual. *ILOG*. Disponível em http://www.ilog.com. Acesso em Novembro de 2002.

JAVATM BLUEPRINTS - MVC. Guidelines, Patterns, and code for end-to-end Java applications. Model View Controller. Disponível em http://java.sun.com/j2ee/blueprints/design_patterns/model_view_controller/index.html. Acesso em Novembro de 2002.

JAVATM BLUEPRINTS. Guidelines, Patterns, and code for end-to-end Java applications. Disponível em http://java.sun.com/j2ee/blueprints. Acesso em Novembro de 2002.

JAVA TUTORIAL – DESIGN PATTERNS. A Tutorial on Java Platform Design Patterns. Disponível em http://developer.java.sun.com/developer/technicalArticles/javaone00/developer/page4.html. Acesso em Novembro de 2002.

JAVA TUTORIAL – WEB. The JavaTM Web Services Tutorial Disponível em http://java.sun.com/webservices/docs/ea1/tutorial/index.html. Acesso em Novembro de 2002.

LEE, C, IRIZARRY, K.. The GeneMine System for genome/proteome annotation and collaborative data mining. IBM Systems Journal 40, 200, p. 592-603.

MAGPIE. Disponível em http://genomes.rockefeller.edu/research.shtml#magpie. Acesso em Novembro de 2002.

MANATEE. Disponível em http://manatee.sourceforge.net/. Acesso em Novembro de 2002.

MANATTE. Esquema dos Eucariontes. Disponível em http://manatee.sourceforge.net/ifx/devel/manatee/sf/manatee_docs/schemas/Euk/Euk1.htm. Acesso em Novembro de 2002.

MANATTE. Esquema dos Procariontes. Disponível em http://manatee.sourceforge.net/ifx/devel/manatee/sf/manatee_docs/schemas/Prok/Prok1.htm. Acesso em Novembro de 2002.

MÉDIGUE, C. ET.AL. Imagene: an integrated computer environment for sequence annotation and analysis. BioInformatics,1999. 15, p. 2-15.

MÖLLER, S. ET.AL. EDITtoTrEMBL: A distributed approach to high-quality automated protein sequence annotation. Bioinformatics 15: 219-227, 1999.

NCBI. NCBI's Genome Annotation Project. Disponível em http://hgm2001.hgu.mrc.ac.uk/Abstracts/Publish/Workshops/Workshop09/hgm0074.htm. Acesso em Janeiro de 2003.

NCBI BLAST Homepage. Disponível em http://www.ncbi.nlm.nih.gov/BLAST/. Acesso em Novembro de 2002.

NNPSL. Prediction of the Subcellular Location of Proteins by Neural Networks. Disponível em http://predict.sanger.ac.uk/nnpsl/. Acesso em Dezembro de 2002.

O2DBI. Persistent Objects with O2DBI. Disponível em http://www.techfak.uni-bielefeld.de/~joern/dev/perl/o2dbi/o2dbi.html. Acesso em Dezembro de 2002.

ORPHEUS. Orpheus Home Page. Disponível em http://pedant.gsf.de/orpheus/. Acesso em Dezembro de 2002.

PDB. Protein Data Bank. Disponível em http://www.rcsb.org/pdb/. Acesso em Dezembro de 2002.

Pearson, W. R.. "Searching Protein Sequence Libraries: Comparison of the Sensitivity and Selectivity of the Smith-Waterman and FASTA algorithms." Genomics 11, pp.635-650, 1991.

PEDANT. Disponível em http://pedant.gsf.de/. Acesso em Novembro de 2002.

PFAM. The Pfam database of protein families and HMMs . Disponível em http://pfam.wustl.edu/. Acesso em Dezembro de 2002.

Phred/Phrap/Consed System Homepage. Disponível em http://www.phrap.org/. Acesso em Dezembro de 2002.

PIR. Protein Information Resource. Disponível em http://pir.georgetown.edu/. Acesso em Dezembro de 2002.

PIR-XML Disponível em ftp://nbrfa.georgetown.edu/pir_databases/psd/xml. Acesso em Novembro de 2002.

PRINTS. Protein Fingerprint Database. Disponível em http://www.bioinf.man.ac.uk/dbbrowser/PRINTS/. Acesso em Dezembro de 2002.

PROSITE. PROSITE Database of protein families and domains. Disponível em http://us.expasy.org/prosite/. Acesso em Dezembro de 2002.

PSEUDOCAP . Disponível em http://www.cmdr.ubc.ca/bobh/PAAP.html. Acesso em Janeiro de 2003.

REESE, M.G. ET.AL.. Genome Annotation Assessment in *Drosophila melanogaster*. Genome Res. 2000 10: 483-501.

RUTHERFORD, K. et. al.. M-A. Artemis: sequence visualisation and annotation. Bioinformatics, 2000, 16 (10), p. 944-945.

SPTR-XML. SPTr-XML Documentation. Disponível em http://www.ebi.ac.uk/swissprot/SP-ML/. Acesso em Novembro de 2002.

SRS. Sequence Retrieval System. Disponível em http://srs.ebi.ac.uk/. Acesso em Dezembro de 2002.

STEIN, L., DOWELL, R.. Manual do DAS. Disponível em http://biodas.org/documents/spec.html. Acesso em Novembro de 2002.

STERKY, F., LUNDEBERG, J.. Sequence analisys of genes and genomes. Journal of Biotechnology. 76, 1 - 31, 2000. Disponível em www.elsevier.com/locate/jbiotec.

SUCEST. Sugar Cane EST Genome Project. Disponível em http://sucest.lad.dcc.unicamp.br/en/. Acesso em Janeiro de 2003.

SUCIU, D.. An Overview of Semistructured Data, SIGACTN: SIGACT News (ACM Special Interest Group on Automata and Computability Theory) ,vol 29, 1998.

SWISSPROT. SWISSPROT Protein knowledgebase. Disponível em http://us.expasy.org/sprot/. Acesso em Dezembro de 2002.

TIGR. The Institute For Genomic Research. Disponível em http://www.tigr.org/. Acesso em Dezembro de 2002.

TMHMM. Prediction of transmembrane helices in proteins. Disponível em http://www.cbs.dtu.dk/services/TMHMM/. Acesso em Dezembro de 2002.

TREMBL. TrEMBL - Computer-annotated supplement to SWISSPROT. Disponível em http://us.expasy.org/sprot/. Acesso em Dezembro de 2002.

TRNASCAN. Search for tRNA genes in genomic sequence. Disponível em http://www.genetics.wustl.edu/eddy/tRNAscan-SE/. Acesso em Dezembro de 2002.

VIBRIO VULNIFICUS. *Vibrio vulnificus YJ016* annotation project. Disponível em http://binfo.ym.edu.tw/yuchung/Vibrio_project/protocol.htm. Acesso em Janeiro de 2003.

VISUALGENOME. Disponível em http://www.rationalgenomics.com/visualgenome.html. Acesso em Novembro de 2002.

W3C. World Wide Web Consortium. Disponível em http://www.w3.org/XML/. Acesso em Dezembro de 2002.

WANG, L., RIETHOVEN, J.J.M., ROBINSON, A.J. XEMBL - distributing EMBL data in XML format", Bioinformatics 2002. 18(8), p. 1147-1148. Disponível em http://www.ebi.ac.uk/xembl/. Acesso em Novembro de 2002.

WATSON, J., HOPKINS, N., ROBERTS, J., STEITZ, J., WEINER, A.. Molecular Biology of the Gene, 4th ed. Benjamin Cummings, Menlo Park, CA, 1987.

WIDOM, J.. Data Management for XML: Research Directions. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, Special Issue on XML 3, 1999. p. 44-52, vol 22. Disponível em ftp://db.stanford.edu/pub/papers/xml-whitepaper.ps. Acesso em Novembro de 2002.

XML. Extensible Markup Language. Disponível em http://www.w3.org/xml. Acesso em Dezembro de 2002.

XML AT NCBI. Disponível em http://www.ncbi.nlm.nih.gov/IEB/ToolBox/XML/, Acesso em Novembro de 2002.

XML BIOINFORMATICS. Disponível em http://www.rdcormia.com/COIN78/files/XML_Finals/BIOML/bio.html. Acesso em Novembro de 2002.

XML EXAMPLES FOR BIOLOGY. Disponível em http://www-alt.pasteur.fr/~letondal/XML/. Acesso em Novembro de 2002.

XSL. The Extensible Stylesheet Language. Disponível em http://www.w3.org/Style/XSL/. Acesso em Dezembro de 2002.

XML Query. Disponível em http://www.w3.org/XML/Query. Acesso em Dezembro de 2002.

Xpath. XML Path Language.Disponível em http://www.w3.org/TR/xpath. Acesso em Dezembro de 2002.

XML Schema. Disponível em http://www.w3.org/XML/Schema. Acesso em Dezembro de 2002.

XSLT. XSL Transformations. Disponível em http://www.w3.org/TR/xslt. Acesso em Dezembro de 2002.

Xlink. W3C XML Pointer, XML Base and XML Linking. Disponível em http://www.w3.org/XML/Linking. Acesso em Dezembro de 2002.